
NLSY97 CODEBOOK SUPPLEMENT

MAIN FILE ROUND 2

Prepared for the
U.S. Department of Labor by

Center for Human Resource Research
The Ohio State University

Under contract with
National Opinion Research Center
University of Chicago

2000

NLSY97 CODEBOOK SUPPLEMENT

MAIN FILE ROUND 2

**Prepared for the
U.S. Department of Labor by**

**Center for Human Resource Research
The Ohio State University**

**Under contract with
National Opinion Research Center
University of Chicago**

2000

This publication is prepared in conjunction with contract #J-9-J-9-0007 with the Bureau of Labor Statistics, U.S. Department of Labor. None of its contents is to be construed as necessarily representing the official position or policy of the Department of Labor.

TABLE OF CONTENTS

| | |
|---|-----|
| Introduction to the CAPI Questionnaire and Codebook..... | i |
| NLSY97 Attachment 1: 1990 Census Industrial & Occupational Classification Codes | 1 |
| Introduction to the Created Variable Appendices..... | 23 |
| NLSY97 Appendix 1: Education Variable Creation | 27 |
| NLSY97 Appendix 2: Employment Variable Creation | 65 |
| NLSY97 Appendix 3: Family Background and Formation Variable Creation..... | 171 |
| NLSY97 Appendix 4: Geographic Variable Creation | 187 |
| NLSY97 Appendix 5: Income and Assets Variable Creation..... | 191 |
| NLSY97 Appendix 6: Event History Creation and Documentation | 243 |
| NLSY97 Appendix 7: Continuous Month Scheme and Crosswalk | 253 |
| NLSY97 Appendix 8: Instrument Rosters | 265 |

Also considered part of the *Codebook Supplement* is Appendix 9, “Family Process and Adolescent Outcome Measures.” This appendix, which describes a number of variables created by Child Trends, Inc., is provided in a separate printed document.

For more information about any aspect of the NLS program, contact:

NLS User Services
Center for Human Resource Research
921 Chatham Lane, Suite 100
Columbus, OH 43221-2418
(614) 442-7366
usersvc@postoffice.chrr.ohio-state.edu

Introduction to the CAPI Questionnaire and Codebook

The newest survey in the NLS program, the National Longitudinal Survey of Youth 1997 (NLSY97) is designed to be representative of the U.S. population born during the years 1980 through 1984. Through the NLSY97, the Bureau of Labor Statistics (BLS) will be able to identify characteristics that define the transition today's youths make from school to the labor market and into adulthood. The NLSY97 cohort includes 8,984 respondents ages 12–16 as of December 31, 1996; the sample contains a cross-sectional subsample and an oversample of black and Hispanic respondents. More information on the selection of respondents for the cohort is available in the *NLSY97 User's Guide*.

This survey was conducted as a computer-assisted personal interview (CAPI). The Round 1 survey was conducted using three different questionnaires: the *Screener, Household Roster, and Non-Resident Roster Questionnaire*; the *Youth Questionnaire*; and the *Parent Questionnaire*. The *Screener, Household Roster, and Non-Resident Roster Questionnaire* was administered to a household resident over the age of 18. The *Youth Questionnaire* was administered to the youth respondent living in the household. The *Parent Questionnaire* was administered to a parent or parent-like figure of the youth residing in the household. For information about how the responding parent was selected, researchers should refer to the *NLSY97 User's Guide*.

The Round 2 survey used a *Youth Questionnaire* similar to Round 1. The interview also included a new instrument, a brief *Household Income Update* administered to one of the respondent's parents. No screener or parent interviews were conducted. The content of these survey instruments is presented in separate questionnaire documents, available from NLS User Services. More information about the administration of the various instruments is provided in the *NLSY97 User's Guide*.

Frequencies from the data collected during the NLSY97 interviews are provided for researchers in the form of a codebook contained on the NLSY97 CD-ROM. For each variable taken directly from the interview, the codebook provides information about the question or check item, the variable reference number, the universe of respondents to whom the question applied, and the distribution of responses to the question. The codebook also contains a number of variables created by survey personnel after the interview, providing information about the content of the variable and the distribution of responses.

Some information relevant to the data contained in the codebook cannot be included on the CD-ROM due to practical constraints. This document, the *NLSY97 Codebook Supplement*, presents additional information which will help researchers to use the data more effectively. The first attachment lists industry and occupation codes and their associated descriptors; this list is too lengthy to be incorporated into the codebook. A number of appendices then provide programs for the created variables; researchers may want to use this information to identify the raw survey data used in a created variable and to understand the rules applied during the creation process. This document also describes the creation of the event history arrays and helps researchers to understand their structure and use.

This introduction to the *Codebook Supplement* is intended to assist researchers in understanding some of the terms and survey methods associated with a CAPI interview. The way in which a CAPI instrument is constructed has important implications for the presentation of the data, and so this discussion will aid in interpretation of the codebook and accompanying documentation.

I. TERMS

Discussions of CAPI surveys include a number of terms that may be unfamiliar to many researchers. The following terms are used throughout the survey documentation; we provide definitions here for easy reference.

Instrument Rosters

A “roster” is a list of one or more items of information pertaining to a specific set of subjects, such as the biological children of the respondent or members of the respondent’s household. For instance, the BIOCHILD roster contains a variety of items (e.g., name, gender, birthdate, etc.) pertaining to each biological child of the respondent. By using the roster format, the CAPI program can gather an inventory of information, tag the data to a specific subject, carry the data along through the interview, and access them when necessary. This format also allows for these items of information to be presented to the interviewer at any time. A listing of rosters, such as the Household Roster, is included in this *Codebook Supplement* as appendix 8. This listing includes the contents of the rosters, the applicable names attached to the rosters that might be encountered in the codebook and/or the questionnaire, and the variable reference number assigned to each piece of data in the codebook.

Interviewer’s Reference Manual

The *Interviewer’s Reference Manual* reproduces the electronic “help screens” that were available to the interviewers for specific questions. These help screens contain definitions, instructions, and other information which interviewers used during the interview to obtain consistent information from respondents, as well as listings of the questions for which they were used during the interview. The *Interviewer’s Reference Manual* should be used in conjunction with the codebook or the questionnaire so that researchers can fully understand the intent of each question in the survey.

Symbols and “Text Fills”

Symbols are reserved fields into which data can be placed, stored and accessed throughout the questionnaire. Each symbol field is assigned a distinct name, often with an index number appended when the same piece of information is stored for a set of subjects. For instance, the symbol “lintdate” contains the date of last interview for the respondent. The symbols “emp.name(1)” through “emp.name(7)” contain the names of the first through the seventh employers of the respondent. Throughout the survey, the last interview date and/or the name of the respondent’s first employer can be accessed by invoking the symbol names “lintdate” and “emp.name(1)” respectively.

Users will encounter these symbol names both in the codebook and in the questionnaire. Sometimes data in a symbol is accessed and used to govern a skip or perform a calculation. At other times, a symbol name may be part of a question text. In this case, the content of the specific symbol content becomes part of the text of the question, and is read as such. This is referred to as a “text substitution” or “text fill.” For instance, a question text reading “When you started working for [emp.name(1)], what kind of work did you do? That is, what was your occupation?” would appear to the interviewer with the appropriate name replacing the symbol “emp.name(1).”

Other types of text fills are also present in the questionnaire. For example, the responding parent may be asked the following question: “What was the reason [he/she] did not live with [his/her] [mother/father/parents]?” In this question, “he/she” and “his/her” refer to the parent’s spouse or partner. The computer automatically fills in the correct gender, and the interviewer reads the question appropriately. “Mother/father/parents” in the question text indicates a set of possible responses to the previous question. The computer automatically fills in the appropriate choice from the three words for the given interview. These automated text fills reduce error and remove the burden of asking the question using the appropriate phrasing from the interviewer.

Loops

Certain sequences of questions in the CAPI instrument are repeated a number of times. For instance, some sets of questions are repeated up to 7 times in the Employment Section of the youth questionnaire, for each of up to 7 jobs. Question names that include a “.01”, “.02”, “.03”, etc. at the end, belong to these repeating sequences of questions. Each repetition of the sequence of questions is referred to as a “loop.” Taking the Employment Section as an example, the sequence of questions asking about the first

job is referred to as the first “loop.” The sequence asking about the second employer is referred to as the second “loop,” and so on. While the codebook generally includes more than one loop in a series (because more than one may contain valid data), the questionnaire includes only the **first loop** in each set of loops.

Hard and Soft Range Restrictions

The “Hard Minimum,” “Hard Maximum,” “Soft Minimum,” and “Soft Maximum” specifications control the allowable range of values that can be entered for a given question. These fields are only active when a question calls for the entry of a time date or amount. Questions that require the interviewer to select one response or to select all that apply do not contain this field, as the range limits are implicit in the distribution code block and are thereby enforced.

Hard minima and maxima are absolute limits that an interviewer- or respondent-generated answer must obey. Entry of values outside the hard range is not allowed. In such cases the interviewer is instructed to enter the maximum or minimum allowable value, as appropriate, enter the actual response in the comment field, and “flag” the case for central office checking. Soft minima and maxima are nested within the hard range. When a response falls outside the soft range but inside the hard range, the computer beeps and asks the interviewer to either confirm or change the response.

In some cases, the hard ranges are themselves determined by a variable. For example, a question may use the respondent’s birth date as a minimum and the current interview date as a maximum, preventing the interviewer from entering a date earlier than the hard minimum birth date or later than the hard maximum interview date. This is a powerful tool for the collection of event histories (among other types of data sequences) and is used extensively in the instrument.

II. APPEARANCE AND PRESENTATION OF DATA

Some CAPI questions generate more than one variable. For example, some questions collect information about the date an event happened and generate three variables: month, day and year. Similarly, questions that ask the respondent to indicate which of several responses are appropriate, and to pick all responses that apply, can generate multiple variables. When a question record generates multiple variables, those variables have decimals in the reference numbers. These two types of questions (date-entry and “code-all-that-apply”) and the examples of the resulting variables, are discussed further below.

Date-Entry Questions

All date questions, whether full dates (month, day and year) or just month and year, are represented in the CAPI codebook with, respectively, three or two variables. Each variable contains the same codeblock displaying the ranges and missing values for all elements of the date. A base reference number ending in “.00”, is assigned to the first variable in the set. The same base reference number, with endings of “.01” and “.02” if necessary, is assigned to the other elements of the date. Thus, if the day is assigned a reference number of R10851.00 for the variable “Date of Birth of HH Member 01 (Scr Ros Item),” the month and year would be assigned reference numbers of R10851.01 and R10851.02 respectively.

Code-All-That-Apply Questions

The NLSY97 includes questions that allow respondents to give multiple responses or code all responses that apply. In the CAPI codebook, each possible response constitutes its own variable. In each codeblock for a given code-all-that-apply question, frequencies for all possible responses are represented. However, each specific variable contains **only** the valid data for one specific possible response, as researchers will note when extracting data. Reference numbers are assigned in the same manner as described for date-entry questions. A base reference number is assigned to the first possible response, with a decimal value being appended to that base number for each following possible response. For example, variables R02461.00–R02461.11 (Activities to Find a Job Emp 01) provide responses (coded yes/no) for 12 different methods

of finding a job. Although the codeblock for R02461.01 represents the frequencies for all possible responses to the question, accessing the data for that specific variable will produce only the data for the response “Contacted employment agency.”

Machine-Generated Check Items

Many questions in the NLSY97 data are “machine checks.” In these questions, previously reported information is checked by the computer and computations are made automatically, causing the appropriate skip pattern to be executed without intervention by the interviewer. For example, the survey program may check the respondent’s gender before asking women whether they have ever been pregnant and men whether they have ever fathered a child. In an effort to clarify the skip patterns present in the instrument, a large number of these machine checks appear in the codebook. The text of machine checks is generally in machine language or equation form; the codeblock also includes a note clarifying the purpose of the check. For example, R02678., “Chk R Female (Pregnancy Leave) Emp 01,” includes the following code:

([male/female] = 2);

The codeblock also contains a translation of the code which is more accessible to users:

/* Is respondent a female? This determines whether the paid pregnancy leave questions which follow are asked. */

NLSY97 Attachment 1:
1990 Census Industrial & Occupational
Classification Codes

U.S. DEPARTMENT OF COMMERCE
Bureau of the Census
Washington, DC 20233

1990 CENSUS OF POPULATION INDUSTRIAL CLASSIFICATION SYSTEM

“N.e.c.” means not elsewhere classified.

| 1990 Census Code | Industry |
|--|---|
| AGRICULTURE, FORESTRY, AND FISHERIES | |
| 010 | Agricultural production, crops |
| 011 | Agricultural production livestock |
| 012 | Veterinary services |
| 020 | Landscape and horticultural services |
| 030 | Agricultural services, n.e.c. |
| 031 | Forestry |
| 032 | Fishing, hunting, and trapping |
| MINING | |
| 040 | Metal mining |
| 041 | Coal mining |
| 042 | Oil and gas extraction |
| 050 | Nonmetallic mining and quarrying, except fuel |
| 060 | CONSTRUCTION |
| MANUFACTURING | |
| Nondurable Goods | |
| <i>Food and kindred products</i> | |
| 100 | Meat products |
| 101 | Dairy products |
| 102 | Canned, frozen, and preserved fruits and vegetables |
| 110 | Grain mill products |
| 111 | Bakery products |
| 112 | Sugar and confectionery products |
| 120 | Beverage industries |
| 121 | Miscellaneous food preparations and kindred products |
| 122 | Not specified food industries |
| 130 | Tobacco manufactures |
| <i>Textile mill products</i> | |
| 132 | Knitting mills |
| 140 | Dyeing and finishing textiles, except wool and knit goods |
| 141 | Carpets and rugs |
| 142 | Yarn, thread, and fabric mills |
| 150 | Miscellaneous textile mill products |
| <i>Apparel and other finished textile products</i> | |
| 151 | Apparel and accessories, except knit |
| 152 | Miscellaneous fabricated textile products |
| <i>Paper and allied products</i> | |
| 160 | Pulp, paper, and paperboard mills |
| 161 | Miscellaneous paper and pulp products |
| 162 | Paperboard containers and boxes |

| 1990 Census Code | Industry |
|------------------|---|
| | <i>Printing, publishing, and allied industries</i> |
| 171 | Newspaper publishing and printing |
| 172 | Printing, publishing and allied industries, except newspapers |
| | <i>Chemicals and allied products</i> |
| 180 | Plastics, synthetics, and resins |
| 181 | Drugs |
| 182 | Soaps and cosmetics |
| 190 | Paints, varnishes, and related products |
| 191 | Agricultural chemicals |
| 192 | Industrial and miscellaneous chemicals |
| | <i>Petroleum and coal products</i> |
| 200 | Petroleum refining |
| 201 | Miscellaneous petroleum and coal products |
| | <i>Rubber and miscellaneous plastics products</i> |
| 210 | Tires and inner tubes |
| 211 | Other rubber products, and plastics footwear and belting |
| 212 | Miscellaneous plastics products |
| | <i>Leather and leather products</i> |
| 220 | Leather tanning and finishing |
| 221 | Footwear, except rubber and plastic |
| 222 | Leather products, except footwear |
| | Durable Goods |
| | <i>Lumber and wood products, except furniture</i> |
| 230 | Logging |
| 231 | Sawmills, planing mills, and millwork |
| 232 | Wood buildings and mobile homes |
| 241 | Miscellaneous wood products |
| 242 | Furniture and fixtures |
| | <i>Stone, clay, glass, and concrete products</i> |
| 250 | Glass and glass products |
| 251 | Cement, concrete, gypsum, and planter products |
| 252 | Structural clay products |
| 261 | Pottery and related products |
| 262 | Miscellaneous nonmetallic mineral and stone products |
| | <i>Metal industries</i> |
| 270 | Blast furnaces, steelworks, rolling and finishing mills |
| 271 | Iron and steel foundries |
| 272 | Primary aluminum industries |
| 280 | Other primary metal industries |
| 281 | Cutlery, handtools, and general hardware |
| 282 | Fabricated structural metal products |
| 292 | Ordnance |
| 300 | Miscellaneous fabricated metal products |
| 301 | Not specified metal industries |

| 1990 Census Code | Industry |
|--|---|
| | <i>Machinery and computing equipment</i> |
| 310 | Engines and turbines |
| 311 | Farm machinery and equipment |
| 312 | Construction and material handling machines |
| 320 | Metalworking machinery |
| 321 | Office and accounting machines |
| 322 | Computers and related equipment |
| 331 | Machine, except electrical, n.e.c. |
| 332 | Not specified machinery |
| | <i>Electrical machinery, equipment, and supplies</i> |
| 340 | Household appliances |
| 341 | Radio, TV, and communication equipment |
| 342 | Electrical machinery, equipment, and supplies, n.e.c. |
| 350 | Not specified electrical machinery, equipment, and supplies |
| | <i>Transportation equipment</i> |
| 351 | Motor vehicles and motor vehicle equipment |
| 352 | Aircraft and parts |
| 360 | Ship and boat building and repairing |
| 361 | Railroad locomotives and equipment |
| 362 | Guided missiles, space vehicles, and parts |
| 370 | Cycles and miscellaneous transportation equipment |
| | <i>Professional and photographic equipment, and watches</i> |
| 371 | Scientific and controlling instruments |
| 372 | Medical, dental, and optical instruments and supplies |
| 380 | Photographic equipment and supplies |
| 381 | Watches, clocks, and clockwork operated devices |
| 390 | Toys, amusement, and sporting goods |
| 391 | Miscellaneous manufacturing industries |
| 392 | Not specified manufacturing industries |
| TRANSPORTATION, COMMUNICATIONS AND OTHER PUBLIC UTILITIES | |
| | Transportation |
| 400 | Railroads |
| 401 | Bus service and urban transit |
| 402 | Taxicab service |
| 410 | Trucking service |
| 411 | Warehousing and storage |
| 412 | U.S. Postal Service |
| 420 | Water transportation |
| 421 | Air transportation |
| 422 | Pipelines, except natural gas |
| 432 | Services incidental to transportation |
| | Communications |
| 440 | Radio and television broadcasting and cable |
| 441 | Telephone communications |
| 442 | Telegraph and miscellaneous communications services |

| 1990 Census Code | Industry |
|-------------------------|--|
| | Utilities and sanitary services |
| 450 | Electric light and power |
| 451 | Gas and steam supply systems |
| 452 | Electric and gas, and other combinations |
| 470 | Water supply and irrigation |
| 471 | Sanitary services |
| 472 | Not specified utilities |
| | WHOLESALE TRADE |
| | Durable Goods |
| 500 | Motor vehicles and equipment |
| 501 | Furniture and home furnishings |
| 502 | Lumber and construction materials |
| 510 | Professional and commercial equipment and supplies |
| 511 | Metals and minerals, except petroleum |
| 512 | Electrical goods |
| 521 | Hardware, plumbing and heating supplies |
| 530 | Machinery, equipment, and supplies |
| 531 | Scrap and waste materials |
| 532 | Miscellaneous wholesale, durable goods |
| | Nondurable Goods |
| 540 | Paper and paper products |
| 541 | Drugs, chemicals and allied products |
| 542 | Apparel, fabrics, and notions |
| 550 | Groceries and related products |
| 551 | Farm-product raw materials |
| 552 | Petroleum products |
| 560 | Alcoholic beverages |
| 561 | Farm supplies |
| 562 | Miscellaneous wholesale, nondurable goods |
| 571 | Not specified wholesale trade |
| | RETAIL TRADE |
| 580 | Lumber and building material retailing |
| 581 | Hardware stores |
| 582 | Retail nurseries and garden stores |
| 590 | Mobile home dealers |
| 591 | Department stores |
| 592 | Variety stores |
| 600 | Miscellaneous general merchandise stores |
| 601 | Grocery stores |
| 602 | Dairy products stores |
| 610 | Retail bakeries |
| 611 | Food stores, n.e.c. |
| 612 | Motor vehicle dealers |
| 620 | Auto and home supply stores |
| 621 | Gasoline service stations |
| 622 | Miscellaneous vehicle dealers |
| 623 | Apparel and accessory stores, except shoe |
| 630 | Shoe stores |
| 631 | Furniture and home furnishings stores |

| 1990 Census Code | Industry |
|--|---|
| 632 | Household appliance stores |
| 633 | Radio, TV, and computer stores |
| 640 | Music stores |
| 641 | Eating and drinking places |
| 642 | Drug stores |
| 650 | Liquor stores |
| 651 | Sporting goods, bicycles, and hobby stores |
| 652 | Book and stationary stores |
| 660 | Jewelry stores |
| 661 | Gift, novelty, and souvenir shops |
| 662 | Sewing, needlework and piece goods stores |
| 663 | Catalog and mail order houses |
| 670 | Vending machine operators |
| 671 | Direct selling establishments |
| 672 | Fuel dealers |
| 681 | Retail florists |
| 682 | Miscellaneous retail stores |
| 691 | Not specified retail trade |
| FINANCE, INSURANCE, AND REAL ESTATE | |
| 700 | Banking |
| 701 | Savings institutions, including credit unions |
| 702 | Credit agencies, n.e.c. |
| 710 | Security, commodity brokerage, and investment companies |
| 711 | Insurance |
| 712 | Real estate, including real estate-insurance offices |
| BUSINESS AND REPAIR SERVICES | |
| 721 | Advertising |
| 722 | Services to dwellings and other buildings |
| 731 | Personnel supply services |
| 732 | Computer and data processing services |
| 740 | Detective and protective services |
| 741 | Business services, n.e.c. |
| 742 | Automotive rental and leasing, without drivers |
| 750 | Automobile parking and carwashes |
| 751 | Automotive repair and related services |
| 752 | Electrical repair shops |
| 760 | Miscellaneous repair services |
| PERSONAL SERVICES | |
| 761 | Private households |
| 762 | Hotels and motels |
| 770 | Lodging places, except hotels and motels |
| 771 | Laundry, cleaning, and garment services |
| 772 | Beauty shops |
| 780 | Barber shops |
| 781 | Funeral service and crematories |
| 782 | Shoe repair shops |
| 790 | Dressmaking shops |
| 791 | Miscellaneous personal services |

1990 Census Code

Industry

ENTERTAINMENT AND RECREATION SERVICES

- | | |
|-----|---|
| 800 | Theaters and motion pictures |
| 801 | Video tape rental |
| 802 | Bowling centers |
| 810 | Miscellaneous entertainment and recreation services |

PROFESSIONAL AND RELATED SERVICES

- | | |
|-----|---|
| 812 | Offices and clinics of physicians |
| 820 | Offices and clinics of dentists |
| 821 | Offices and clinics of chiropractors |
| 822 | Offices and clinics of optometrists |
| 830 | Offices and clinics of health practitioners, n.e.c. |
| 831 | Hospitals |
| 832 | Nursing and personal care facilities |
| 840 | Health services, n.e.c. |
| 841 | Legal services |
| 842 | Elementary and secondary schools |
| 850 | Colleges and universities |
| 851 | Vocational schools |
| 852 | Libraries |
| 860 | Educational services, n.e.c. |
| 861 | Job training and vocational rehabilitation services |
| 862 | Child day care services |
| 863 | Family child care homes |
| 870 | Residential care facilities, without nursing |
| 871 | Social services, n.e.c. |
| 872 | Museums, art galleries and zoos |
| 873 | Labor unions |
| 880 | Religious organizations |
| 881 | Membership organizations, n.e.c. |
| 882 | Engineering architectural and surveying services |
| 890 | Accounting, auditing, and bookkeeping services |
| 891 | Research, development, and testing services |
| 892 | Management and public relations services |
| 893 | Miscellaneous professional and related services |

PUBLIC ADMINISTRATION

- | | |
|-----|--|
| 900 | Executive and legislative offices |
| 901 | General government, n.e.c. |
| 910 | Justice, public order, and safety |
| 921 | Public finance, taxation, and monetary policy |
| 922 | Administration of human resources programs |
| 930 | Administration of environmental quality and housing programs |
| 931 | Administration of economic programs |
| 932 | National security and international affairs |
| 991 | Military |

U.S. DEPARTMENT OF COMMERCE
Bureau of the Census
Washington, DC 20233

1990 CENSUS OF POPULATION OCCUPATIONAL CLASSIFICATION SYSTEM

"N.e.c." means not elsewhere classified.

| 1990 Census Code | Occupation |
|--|--|
| MANAGERIAL AND PROFESSIONAL SPECIALTY OCCUPATIONS | |
| Executive, Administrative, and Managerial Occupations | |
| 003 | Legislators |
| 004 | Chief executives and general administrators, public administration |
| 005 | Administrators and officials, public administration |
| 006 | Administrators, protective services |
| 007 | Financial managers |
| 008 | Personnel and labor relations managers |
| 009 | Purchasing managers |
| 013 | Managers, marketing, advertising, and public relations |
| 014 | Administrators, education and related fields |
| 015 | Managers, medicine and health |
| 016 | Postmasters and mail superintendents |
| 017 | Managers, food serving and lodging establishments |
| 018 | Managers, properties and real estate |
| 019 | Funeral directors |
| 021 | Managers, service organizations, n.e.c. |
| 022 | Managers and administrators n.e.c. |
| Management Related Occupations | |
| 023 | Accountants and auditors |
| 024 | Underwriters |
| 025 | Other financial officers |
| 026 | Management analysts |
| 027 | Personnel, training, and labor relations specialists |
| 028 | Purchasing agents and buyers, farm products |
| 029 | Buyers, wholesale and retail trade except farm products |
| 033 | Purchasing agents and buyers, n.e.c. |
| 034 | Business and promotion agents |
| 035 | Construction inspectors |
| 036 | Inspectors and compliance officers, except construction |
| 037 | Management related occupations, n.e.c. |
| PROFESSIONAL SPECIALTY OCCUPATIONS | |
| Engineers, Architects, and Surveyors | |
| 043 | Architects |
| | <i>Engineers</i> |
| 044 | Aerospace |
| 045 | Metallurgical and materials |
| 046 | Mining |
| 047 | Petroleum |
| 048 | Chemical |
| 049 | Nuclear |
| 053 | Civil |

| 1990 Census Code | Occupation |
|-------------------------|---|
| 054 | Agricultural |
| 055 | Electrical and electronic |
| 056 | Industrial |
| 057 | Mechanical |
| 058 | Marine and naval architects |
| 059 | Engineers, n.e.c. |
| 063 | Surveyors and mapping scientists |
| | <i>Mathematical and Computer Scientists</i> |
| 064 | Computer systems analysts and scientists |
| 065 | Operations and systems researchers and analysts |
| 066 | Actuaries |
| 067 | Statisticians |
| 068 | Mathematical scientists, n.e.c. |
| | <i>Natural Scientists</i> |
| 069 | Physicists and astronomers |
| 073 | Chemists, except biochemists |
| 074 | Atmospheric and space scientists |
| 075 | Geologists and geodesists |
| 076 | Physical scientists, n.e.c. |
| 077 | Agricultural and food scientists |
| 078 | Biological and life scientists |
| 079 | Forestry and conservation scientists |
| 083 | Medical scientists |
| | <i>Health Diagnosing Occupations</i> |
| 084 | Physicians |
| 085 | Dentists |
| 086 | Veterinarians |
| 087 | Optometrists |
| 088 | Podiatrists |
| 089 | Health diagnosing practitioners, n.e.c. |
| | <i>Health Assessment and Treating Occupations</i> |
| 095 | Registered nurses |
| 096 | Pharmacists |
| 097 | Dietitians |
| | <i>Therapists</i> |
| 098 | Respiratory therapists |
| 099 | Occupational therapists |
| 103 | Physical therapists |
| 104 | Speech therapists |
| 105 | Therapists, n.e.c. |
| 106 | Physicians' assistants |
| | <i>Teachers, Postsecondary</i> |
| 113 | Earth, environmental, and marine science teachers |
| 114 | Biological science teachers |
| 115 | Chemistry teachers |
| 116 | Physics teachers |

| 1990 Census Code | Occupation |
|-------------------------|--|
| 117 | Natural science teachers, n.e.c. |
| 118 | Psychology teachers |
| 119 | Economics teachers |
| 123 | History teachers |
| 124 | Political science teachers |
| 125 | Sociology teachers |
| 126 | Social science teachers, n.e.c. |
| 127 | Engineering teachers |
| 128 | Mathematical science teachers |
| 129 | Computer science teachers |
| 133 | Medical science teachers |
| 134 | Health specialties teachers |
| 135 | Business, commerce, and marketing teachers |
| 136 | Agriculture and forestry teachers |
| 137 | Art, drama, and music teachers |
| 138 | Physical education teachers |
| 139 | Education teachers |
| 143 | English teachers |
| 144 | Foreign language teachers |
| 145 | Law teachers |
| 146 | Social work teachers |
| 147 | Theology teachers |
| 148 | Trade and industrial teachers |
| 149 | Home economics teachers |
| 153 | Teachers, postsecondary, n.e.c. |
| 154 | Postsecondary teachers, subject not specified |
| | <i>Teachers, Except Postsecondary</i> |
| 155 | Teachers, prekindergarten and kindergarten |
| 156 | Teachers, elementary school |
| 157 | Teachers, secondary school |
| 158 | Teachers, special education |
| 159 | Teachers, n.e.c. |
| 163 | Counselors, educational and vocational |
| | <i>Librarians, Archivists, and Curators</i> |
| 164 | Librarians |
| 165 | Archivists and curators |
| | <i>Social Scientists and Urban Planners</i> |
| 166 | Economists |
| 167 | Psychologists |
| 168 | Sociologists |
| 169 | Social scientists, n.e.c. |
| 173 | Urban planners |
| | <i>Social, Recreation, and Religious Workers</i> |
| 174 | Social workers |
| 175 | Recreation workers |
| 176 | Clergy |
| 177 | Religious workers, n.e.c. |

| 1990 Census Code | Occupation |
|------------------|--|
| | <i>Lawyers and Judges</i> |
| 178 | Lawyers |
| 179 | Judges |
| | <i>Writers, Artists, Entertainers, and Athletes</i> |
| 183 | Authors |
| 184 | Technical writers |
| 185 | Designers |
| 186 | Musicians and composers |
| 187 | Actors and directors |
| 188 | Painters, sculptors, craft-artists, and artist printmakers |
| 189 | Photographers |
| 193 | Dancers |
| 194 | Artists, performers, and related workers, n.e.c. |
| 195 | Editors and reporters |
| 197 | Public relations specialists |
| 198 | Announcers |
| 199 | Athletes |

TECHNICAL, SALES AND ADMINISTRATIVE SUPPORT OCCUPATIONS

| | |
|-----|---|
| | Technicians and Related Support occupations |
| | <i>Health Technologists and Technicians</i> |
| 203 | Clinical laboratory technologists and technicians |
| 204 | Dental hygienists |
| 205 | Health record technologists and technicians |
| 206 | Radiologic technicians |
| 207 | Licensed practical nurses |
| 208 | Health technologists and technicians, n.e.c. |
| | <i>Technologists and Technicians, Except Health</i> |
| | Engineering and Related Technologists and Technicians |
| 213 | Electrical and electronic technicians |
| 214 | Industrial engineering technicians |
| 215 | Mechanical engineering technicians |
| 216 | Engineering technicians, n.e.c. |
| 217 | Drafting occupations |
| 218 | Surveying and mapping technicians |
| | Science Technicians |
| 223 | Biological technicians |
| 224 | Chemical technicians |
| 225 | Science technicians |
| | Technicians: Except Health, Engineering and Science |
| 226 | Airplane pilots and navigators |
| 227 | Air traffic controllers |
| 228 | Broadcast equipment operators |
| 229 | Computer programmers |
| 233 | Tool programmers, numerical control |
| 234 | Legal assistants |
| 235 | Technicians, n.e.c. |

| 1990 Census Code | Occupation |
|-------------------------|---|
| | Sales Occupations |
| 243 | Supervisors and proprietors, sales occupations |
| | <i>Sales Representatives, Finance and Business Services</i> |
| 253 | Insurance sales occupations |
| 254 | Real estate sales occupations |
| 255 | Securities and financial services sales occupations |
| 256 | Advertising and related sales occupations |
| 257 | Sales occupations, other business services |
| | <i>Sales Representatives, Commodities Except Retail</i> |
| 258 | Sales engineers |
| 259 | Sales representatives, mining, manufacturing, and wholesale |
| | <i>Sales Workers, Retail and Personal Services</i> |
| 263 | Sales workers, motor vehicles and boats |
| 264 | Sales workers, apparel |
| 265 | Sales workers, shoes |
| 266 | Sales workers, furniture and home furnishings |
| 267 | Sales workers; radio, TV, hi-fi, and appliances |
| 268 | Sales workers, hardware and building supplies |
| 269 | Sales workers, parts |
| 274 | Sales workers, other commodities |
| 275 | Sales counter clerks |
| 276 | Cashiers |
| 277 | Street and door-to-door sales workers |
| 278 | News vendors |
| | <i>Sales Related Occupations</i> |
| 283 | Demonstrators, promoters and models, sales |
| 284 | Auctioneers |
| 285 | Sales support occupations, n.e.c. |
| | Administrative Support Occupations, Including Clerical |
| | <i>Supervisors, Administrative Support Occupations</i> |
| 303 | Supervisors, general office |
| 304 | Supervisors, computer equipment operators |
| 305 | Supervisors, financial records processing |
| 306 | Chief communications operators |
| 307 | Supervisors; distribution, scheduling, and adjusting clerks |
| | <i>Computer Equipment Operators</i> |
| 308 | Computer operators |
| 309 | Peripheral equipment operators |
| | <i>Secretaries, Stenographers, and Typists</i> |
| 313 | Secretaries |
| 314 | Stenographers |
| 315 | Typists |
| | <i>Information Clerks</i> |
| 316 | Interviewers |
| 317 | Hotel clerks |

| 1990 Census Code | Occupation |
|-------------------------|---|
| 318 | Transportation ticket and reservation agents |
| 319 | Receptionists |
| 323 | Information clerks, n.e.c. |
| | <i>Records Processing Occupations, Except Financial</i> |
| 325 | Classified-ad clerks |
| 326 | Correspondence clerks |
| 327 | Order clerks |
| 328 | Personnel clerks, except payroll and timekeeping |
| 329 | Library clerks |
| 335 | File clerks |
| 336 | Records clerks |
| | <i>Financial Records Processing Occupations</i> |
| 337 | Bookkeepers, accounting, and auditing clerks |
| 338 | Payroll and timekeeping clerks |
| 339 | Billing clerks |
| 343 | Cost and rate clerks |
| 344 | Billing, posting, and calculating machine operators |
| | <i>Duplicating, Mail and Other Office Machine Operators</i> |
| 345 | Duplicating machine operators |
| 346 | Mail preparing and paper handling machine operators |
| 347 | Office machine operators, n.e.c. |
| | <i>Communications Equipment Operators</i> |
| 348 | Telephone operators |
| 353 | Communications equipment operators, n.e.c. |
| | <i>Mail and Message Distributing Occupations</i> |
| 354 | Postal clerks, exc. mail carriers |
| 355 | Mail carriers, postal service |
| 356 | Mail clerks, exc. postal service |
| 357 | Messengers |
| | <i>Material Recording, Scheduling, and Distributing Clerks</i> |
| 359 | Dispatchers |
| 363 | Production coordinators |
| 364 | Traffic, shipping, and receiving clerks |
| 365 | Stock and inventory clerks |
| 366 | Motor readers |
| 368 | Weighers, measurers, checkers and samplers |
| 373 | Expeditors |
| 374 | Material recording, scheduling, and distributing clerks, n.e.c. |
| | <i>Adjusters and Investigators</i> |
| 375 | Insurance adjusters, examiners, and investigators |
| 376 | Investigators and adjusters except insurance |
| 377 | Eligibility clerks, social welfare |
| 378 | Bill and account collectors |
| | <i>Miscellaneous Administrative Support Occupations</i> |
| 379 | General office clerks |

| 1990 Census Code | Occupation |
|------------------|--|
| 383 | Bank tellers |
| 384 | Proofreaders |
| 385 | Data-entry keyers |
| 386 | Statistical clerks |
| 387 | Teachers' aides |
| 389 | Administrative support occupations, n.e.c. |

SERVICE OCCUPATIONS

Private Household Occupations

| | |
|-----|---|
| 403 | Launderers and ironers |
| 404 | Cooks, private household |
| 405 | Housekeepers and butlers |
| 406 | Child care workers, private household |
| 407 | Private household cleaners and servants |

Protective Service Occupations

Supervisors Protective Service Occupations

| | |
|-----|---|
| 413 | Supervisors, firefighting and fire prevention occupations |
| 414 | Supervisors, police and detectives |
| 415 | Supervisors, guards |

Firefighting and Fire Prevention Occupations

| | |
|-----|---|
| 416 | Fire inspection and fire prevention occupations |
| 417 | Firefighting occupations |

Police and Detectives

| | |
|-----|--|
| 418 | Police and detectives, public service |
| 423 | Sheriffs, bailiffs, and other law enforcement officers |
| 424 | Correctional institution officers |

Guards

| | |
|-----|--|
| 425 | Crossing guards |
| 426 | Guards and police, except public service |
| 427 | Protective service occupations, n.e.c. |

Service Occupations, Except Protective and Household

Food Preparation and Service Occupations

| | |
|-----|---|
| 433 | Supervisors, food preparation and service occupations |
| 434 | Bartenders |
| 435 | Waiters and waitresses |
| 436 | Cooks |
| 438 | Food counter, fountain and related occupations |
| 439 | Kitchen workers, food preparation |
| 443 | Waiters'/waitresses' assistants |
| 444 | Miscellaneous food preparation occupations |

Health Service Occupations

| | |
|-----|--|
| 445 | Dental assistants |
| 446 | Health aides, except nursing |
| 447 | Nursing aides, orderlies, and attendants |

Cleaning and Building Service Occupations, except Household

| | |
|-----|---|
| 448 | Supervisors cleaning and building service workers |
|-----|---|

| 1990 Census Code | Occupation |
|------------------|--------------------------|
| 449 | Maids and houseman |
| 453 | Janitors and cleaners |
| 454 | Elevator operators |
| 455 | Post control occupations |

Personal Service Occupations

| | |
|-----|---|
| 456 | Supervisors, personal service occupations |
| 457 | Barbers |
| 458 | Hairdressers and cosmetologists |
| 459 | Attendants, amusement and recreation facilities |
| 461 | Guides |
| 462 | Ushers |
| 463 | Public transportation attendants |
| 464 | Baggage porters and bellhops |
| 465 | Welfare service aides |
| 466 | Family child care providers |
| 467 | Early childhood teacher's assistants |
| 468 | Child care workers, n.e.c. |
| 469 | Personal service occupations, n.e.c. |

FARMING, FORESTRY AND FISHING OCCUPATIONS

Farm Operators and Managers

| | |
|-----|---|
| 473 | Farmers, except horticultural |
| 474 | Horticultural specialty farmers |
| 475 | Managers, farms, except horticultural |
| 476 | Managers, horticultural specialty farms |

Other Agricultural and Related Occupations

Farm Occupations Except Managerial

| | |
|-----|---------------------------------|
| 477 | Supervisors, farm workers |
| 479 | Farm workers |
| 483 | Marine life cultivation workers |
| 484 | Nursery workers |

Related Agricultural Occupations

| | |
|-----|--|
| 485 | Supervisors related agricultural occupations |
| 486 | Groundskeepers and gardeners, except farm |
| 487 | Animal caretakers, except farm |
| 488 | Graders and sorters, agricultural products |
| 489 | Inspectors, agricultural products |

Forestry and Logging Occupations

| | |
|-----|--|
| 494 | Supervisors, forestry, and logging workers |
| 495 | Forestry workers, except logging |
| 496 | Timber cutting and logging occupations |

Fishers, Hunters, and Trappers

| | |
|-----|--|
| 497 | Captains and other officers, fishing vessels |
| 498 | Fishers |
| 499 | Hunters and trappers |

| 1990 Census Code | Occupation |
|--|---|
| PRECISION PRODUCTION, CRAFT, AND REPAIR OCCUPATIONS | |
| Mechanics and Repairers | |
| 503 | Supervisors, mechanics and repairers |
| <i>Mechanics and Repairers, Except Supervisors</i> | |
| | Vehicle and Mobile Equipment Mechanics and Repairers |
| 505 | Automobile mechanics |
| 506 | Automobile mechanic apprentices |
| 507 | Bus, truck, and stationary engine mechanics |
| 508 | Aircraft engine mechanics |
| 509 | Small engine repairers |
| 514 | Automobile body and related repairers |
| 515 | Aircraft mechanics, exc. engine |
| 516 | Heavy equipment mechanics |
| 517 | Farm equipment mechanics |
| 518 | Industrial machinery repairers |
| 519 | Machinery maintenance occupations |
| <i>Electrical and Electronic Equipment Repairers</i> | |
| 523 | Electronic repairers, communications and industrial equipment |
| 525 | Data processing equipment repairers |
| 526 | Household appliance and power tool repairers |
| 527 | Telephone line installers and repairers |
| 529 | Telephone installers and repairers |
| 533 | Miscellaneous electrical and electronic equipment repairers |
| 534 | Heating, air conditioning, and refrigeration mechanics |
| <i>Miscellaneous Mechanics and Repairers</i> | |
| 535 | Camera, watch, and musical instrument repairers |
| 536 | Locksmiths and safe repairers |
| 538 | Office machine repairers |
| 539 | Mechanical controls and valve repairers |
| 543 | Elevator installers and repairers |
| 544 | Millwrights |
| 547 | Specified mechanics and repairers, n.e.c. |
| 549 | Not specified mechanics and repairers |
| Construction Trades | |
| <i>Supervisors, Construction Occupations</i> | |
| 553 | Supervisors; brickmasons, stonemasons, and tile setters |
| 554 | Supervisors; carpenters and related workers |
| 555 | Supervisors; electricians and power transmission installers |
| 556 | Supervisors; painters, paperhangers, and plasterers |
| 557 | Supervisors; plumbers, pipefitters, and steamfitters |
| 558 | Supervisors; n.e.c. |
| <i>Construction Trades Except Supervisors</i> | |
| 563 | Brickmasons and stonemasons |
| 564 | Brickmason and stonemason apprentices |
| 565 | Tile setters, hard and soft |
| 566 | Carpet installers |
| 567 | Carpenters |

| 1990 Census Code | Occupation |
|--|--|
| 569 | Carpenter apprentices |
| 573 | Drywall installers |
| 575 | Electricians |
| 576 | Electrician apprentices |
| 577 | Electrical power installers and repairers |
| 579 | Painters, construction and maintenance |
| 583 | Paperhangers |
| 584 | Plasterers |
| 585 | Plumbers, pipefitters, and steamfitters |
| 587 | Plumber, pipefitter, and steamfitter apprentices |
| 588 | Concrete and terrazzo finishers |
| 589 | Glaziers |
| 593 | Insulation workers |
| 594 | Paving, surfacing, and tamping equipment operators |
| 595 | Roofers |
| 596 | Sheetmetal duct installers |
| 597 | Structural metal workers |
| 598 | Drillers, earth |
| 599 | Construction trades, n.e.c. |
| Extractive Occupations | |
| 613 | Supervisors, extractive occupations |
| 614 | Drillers, oil well |
| 615 | Explosives workers |
| 616 | Mining machine operators |
| 617 | Mining occupations, n.e.c. |
| Precision Production Occupations | |
| 628 | Supervisors, production occupations |
| <i>Precision Metal Working Occupations</i> | |
| 634 | Tool and die makers |
| 635 | Tool and die maker apprentices |
| 636 | Precision assemblers, metal |
| 637 | Machinists |
| 639 | Machinist apprentices |
| 643 | Boilermakers |
| 644 | Precision grinders, filers, and tool sharpeners |
| 645 | Patternmakers and model makers, metal |
| 646 | Lay-out workers |
| 647 | Precious stones and metals workers (Jewelers) |
| 649 | Engravers, metal |
| 653 | Sheet metal workers |
| 654 | Sheet metal worker apprentices |
| 655 | Miscellaneous precision metal workers |
| <i>Precision Woodworking Occupations</i> | |
| 656 | Patternmakers and model makers, wood |
| 657 | Cabinet makers and bench carpenters |
| 658 | Furniture and wood finishers |
| 659 | Miscellaneous precision woodworkers |

| 1990 Census Code | Occupation |
|-------------------------|--|
| | <i>Precision Textile, Apparel, and Furnishings Machine Workers</i> |
| 666 | Dressmakers |
| 667 | Tailors |
| 668 | Upholsterers |
| 669 | Shoe repairers |
| 674 | Miscellaneous precision apparel and fabric workers |
| | <i>Precision Workers, Assorted Materials</i> |
| 675 | Hand molders and shapers, except jewelers |
| 676 | Patternmakers, lay-out workers, and cutters |
| 677 | Optical goods workers |
| 678 | Dental laboratory and medical appliance technicians |
| 679 | Bookbinders |
| 683 | Electrical and electronic equipment assemblers |
| 684 | Miscellaneous precision workers, n.e.c. |
| | <i>Precision Food Production Occupations</i> |
| 686 | Butchers and meat cutters |
| 687 | Bakers |
| 688 | Food batchmakers |
| | <i>Precision Inspectors, Testers, and Related Workers</i> |
| 689 | Inspectors, testers, and graders |
| 693 | Adjusters and calibrators |
| | Plant and System Operators |
| 694 | Water and sewage treatment plant operators |
| 695 | Power plant operators |
| 696 | Stationary engineers |
| 699 | Miscellaneous plant and system operators |
| | OPERATORS, FABRICATORS, AND LABORERS |
| | Machine Operators, Assemblers, and Inspectors |
| | <i>Machine Operators and Tenders, except Precision Metal Working and Plastic Working</i> |
| | <i>Machine Operators</i> |
| 703 | Lathe and turning machine set-up operators |
| 704 | Lathe and turning machine operators |
| 705 | Milling and planing machine operators |
| 706 | Punching and stamping press machine operators |
| 707 | Rolling machine operators |
| 708 | Drilling and boring machine operators |
| 709 | Grinding, abrading, buffing, and polishing machine operators |
| 713 | Forging machine operators |
| 714 | Numerical control machine operators |
| 715 | Miscellaneous metal, plastic, stone, and glass working machine operators |
| 717 | Fabricating machine operators, n.e.c. |
| | <i>Metal and Plastic Processing Machine Operators</i> |
| 719 | Molding and canting machine operators |
| 723 | Metal plating machine operators |
| 724 | Heat treating equipment operators |
| 725 | Miscellaneous metal and plastic processing machine operators |

| 1990 Census Code | Occupation |
|-------------------------|--|
| | <i>Woodworking Machine Operators</i> |
| 726 | Wood lathe, routing, and planing machine operators |
| 727 | Sawing machine operators |
| 728 | Shaping and joining machine operators |
| 729 | Nailing and tacking machine operators |
| 733 | Miscellaneous woodworking machine operators |
| | <i>Printing Machine Operators</i> |
| 734 | Printing press operators |
| 735 | Photoengravers and lithographers |
| 736 | Typesetters and compositors |
| 737 | Miscellaneous printing machine operators |
| | <i>Textile, Apparel, and Furnishings Machine Operators</i> |
| 738 | Winding and twisting machine operators |
| 739 | Knitting, looping, taping, and weaving machine operators |
| 743 | Textile cutting machine operators |
| 744 | Textile serving machine operators |
| 745 | Shoe machine operators |
| 747 | Pressing machine operators |
| 748 | Laundering and dry cleaning machine operators |
| 749 | Miscellaneous textile machine operators |
| | <i>Machine Operators, Assorted Materials</i> |
| 753 | Cementing and gluing machine operators |
| 754 | Packaging and filling machine operators |
| 755 | Extruding and forming machine operators |
| 756 | Mixing and blending machine operators |
| 757 | Separating, filtering, and clarifying machine operators |
| 758 | Compressing and compacting machine operators |
| 759 | Painting and paint spraying machine operators |
| 763 | Roasting and baking machine operators, food |
| 764 | Washing, cleaning, and pickling machine operators |
| 765 | Folding machine operators |
| 766 | Furnace, kiln, and oven operators, exc. food |
| 768 | Crushing and grinding machine operators |
| 769 | Slicing and cutting machine operators |
| 773 | Motion picture projectionists |
| 774 | Photographic process machine operators |
| 777 | Miscellaneous machine operators, n.e.c. |
| 779 | Machine operators, not specified |
| | <i>Fabricators, Assemblers, and Hand Working Occupations</i> |
| 783 | Welders and cutters |
| 784 | Solderers and brazers |
| 785 | Assemblers |
| 786 | Hand cutting and trimming occupations |
| 787 | Hand molding, casting, and forming occupations |
| 789 | Hand painting, coating, and decorating occupations |
| 793 | Hand engraving and printing occupations |
| 795 | Miscellaneous hand working occupations |

| 1990 Census Code | Occupation |
|-------------------------|---|
| | <i>Production Inspectors, Testers, Samplers, and Weighers</i> |
| 796 | Production inspectors, checkers, and examiners |
| 797 | Production testers |
| 798 | Production samplers and weighers |
| 799 | Graders and sorters, exc. agricultural |
| | Transportation and Material Moving Occupations |
| | <i>Motor Vehicle Operators</i> |
| 803 | Supervisors, motor vehicle operators |
| 804 | Truck drivers |
| 806 | Driver-sales workers |
| 808 | Bus drivers |
| 809 | Taxicab drivers and chauffeurs |
| 813 | Parking lot attendants |
| 814 | Motor transportation occupations, n.e.c. |
| | <i>Transportation Occupations, Except Motor Vehicles</i> |
| | <i>Rail Transportation Occupations</i> |
| 823 | Railroad conductors and yardmasters |
| 824 | Locomotive operating occupations |
| 825 | Railroad brake, signal, and switch operators |
| 826 | Rail vehicle operators, n.e.c. |
| | <i>Water Transportation Occupations</i> |
| 828 | Ship captains and mates, except fishing boats |
| 829 | Sailors and deckhands |
| 833 | Marine engineers |
| 834 | Bridge, lock, and lighthouse tenders |
| | <i>Material Moving Equipment Operators</i> |
| 843 | Supervisors, material moving equipment operators |
| 844 | Operating engineers |
| 845 | Longshore equipment operators |
| 848 | Hoist and winch operators |
| 849 | Crane and tower operators |
| 853 | Excavating and loading machine operators |
| 855 | Grader, dozer, and scraper operators |
| 856 | Industrial truck and tractor equipment operators |
| 859 | Miscellaneous material moving equipment operators |
| | Handlers, Equipment Cleaners, Helpers, and Laborers |
| 864 | Supervisors, handlers, equipment cleaners, and laborers, n.e.c. |
| 865 | Helpers, mechanics and repairers |
| | <i>Helpers, Construction and Extractive Occupations</i> |
| 866 | Helpers, construction trades |
| 867 | Helpers, surveyor |
| 868 | Helpers, extractive occupations |
| 869 | Construction laborers |
| 874 | Production helpers |
| | <i>Freight, Stock, and Material Handlers</i> |
| 875 | Garbage collectors |

| 1990 Census Code | Occupation |
|-----------------------------|--|
| 876 | Stevedores |
| 877 | Stock handlers and baggers |
| 878 | Machine feeders and offbearers |
| 883 | Freight, stock, and material handlers, n.e.c. |
| 885 | Garage and service station related occupations |
| 887 | Vehicle washers and equipment cleaners |
| 888 | Hand packers and packagers |
| 889 | Laborers, except construction |
| MILITARY OCCUPATIONS | |
| 905 | Military Occupation |

Introduction to the Created Variable Appendices

The main file NLSY97 CD-ROM contains a number of created variables. These created variables will be included on each public release of the NLSY97. With very few exceptions, these variables are either commonly used items that are derived from a number of different NLSY97 survey questions or longitudinal items that require updating in each round. If they were not provided, creating either the longitudinal or the cross-sectional variables might present difficulties for some NLSY97 users. In general, the created variables present information in five topical areas: education, employment, family background and formation, geographic information, and income and assets.

The first five appendices present the programs that were used to create the NLSY97 variables in round 2. The majority of programs are in SAS; a few are in SPSS. In the interest of space, four variables were not included in these appendices. These are the following:

CV_INTERVIEW_CMONT
CV_INTERVIEW_DATE
CV_AGE_INT_DATE
CV_AGE(MONTHS)_INT_DATE

These variables are either directly picked up from the data set or relatively easy to compute. In the case of CV_INTERVIEW_CMONT and CV_AGE(MONTHS)_INT_DATE, a simple formula of $\{(\text{variable year}-1980)*12\}+\text{variable month}$ was used to translate the date into a continuous month scheme (see Appendix 7 for a further explanation of the continuous month scheme).

These programs are provided for the convenience of the user and are available electronically from NLS User Services (see the Table of Contents for contact information).

Appendices 6 and 7 provide detailed information about the creation of the event history variables, a special set of variables included on the mainfile CD-ROM. Although programs are not included, the text of Appendix 6 describes the various status arrays available and discusses the creation and editing process. Appendix 7 explains the continuous month and continuous week systems used in the creation of event history variables and provides a crosswalk of continuous month/week and actual dates so that researchers can associate event history variables with other information about the respondents.

INDEX TO CREATED VARIABLE PROGRAMS

This section lists by question name each NLSY97 created variable for which a program is provided in this document, along with the accompanying page number for the program.

| | |
|----------------------------------|------------------------------|
| CV_AMT_GOVNT_PGM_PCY.xx, 213 | CV_HH_UNDER_6, 173 |
| CV_ASSOC_CREDITS.xx, 56 | CV_HIGHEST_DEGREE_EVER, 29 |
| CV_BA_CREDITS.xx, 56 | CV_HIGHEST_DEGREE_YR, 29 |
| CV_BIO_CHILD_HH, 183 | CV_HOURS_WK_EVER, 163 |
| CV_BIO_CHILD_NR, 183 | CV_HOURS_WK_YR.xx, 160 |
| CV_CENSUS_REGION, 189 | CV_HRLY_COMPENSATION, 80 |
| CV_CHILD_BIRTH_DATE.xx_M,_Y, 183 | CV_HRLY_PAY, 80 |
| CV_CHILD_BIRTH_MONTH.xx, 183 | CV_HS_DIPLOMA, 42 |
| CV_CHILD_DEATH_DATE.xx_M,_Y, 183 | CV_INCOME_GROSS_YR, 193 |
| CV_CHILD_DEATH_MONTH.xx, 183 | CV_JOB<13_WKS, 80 |
| CV_CHILD_STATUS.xx, 183 | CV_MARRIAGES_TTL, 178 |
| CV_COHAB_TTL, 178 | CV_MARSTAT, 178 |
| CV_ENROLLSTAT, 29 | CV_MARSTAT_COLLAPSED, 178 |
| CV_FIRST_COHAB_DATE_M,_Y, 178 | CV_MSA, 189 |
| CV_FIRST_COHAB_MONTH, 178 | CV_PIAT_PERCENTILE_SCORE, 59 |
| CV_FIRST_MARRY_DATE_M,_Y, 178 | CV_PIAT_STANDARD_SCORE, 59 |
| CV_FIRST_MARRY_MONTH, 178 | CV_SCH_ATTEND_EVER, 54 |
| CV_GED, 42 | CV_SCH_ATTEND_YR, 54 |
| CV_GOVNT_PGM_EVER, 213 | CV SCHOOL_SIZE, 62 |
| CV_GOVNT_PGM_YR.xx, 213 | CV SCHOOL_TYPE, 40 |
| CV_GRADE_SKIPPED_EVER, 43 | CV_STUDENT_TEACHER_RATIO, 62 |
| CV_GRADE_SKIPPED_YR, 43 | CV_TTL_JOBS_EVER, 169 |
| CV_GRADES_REPEAT_EVER, 43 | CV_TTL_JOBS_YR.xx, 166 |
| CV_GRADES_REPEAT_YR, 43 | CV_TTL_RESIDENCES, 186 |
| CV_HGC_EVER, 29 | CV_URBAN_RURAL, 190 |
| CV_HGC_YR, 29 | CV_WKSWK_DLI, 152 |
| CV_HH_NET_WORTH, 193 | CV_WKSWK_EVER, 153 |
| CV_HH_POV_RATIO, 193 | CV_WKSWK_JOB_DLI.xx, 158 |
| CV_HH_REL_CURRENT, 174 | CV_WKSWK_JOB_YR.xx.xx, 155 |
| CV_HH_SIZE, 173 | CV_WKSWK_YR.xx, 150 |
| CV_HH_UNDER_18, 173 | UNEMPRATE-COL, 190 |

NLSY97 Appendix 1:

Education Variable Creation

ENROLLMENT STATUS, HIGHEST GRADE COMPLETED, AND HIGHEST DEGREE RECEIVED

| | | |
|---------------------------|---------------|------------------------|
| Variables Created: | CV_ENROLLSTAT | CV_HGC_YR |
| | CV_HGC_EVER | CV_HIGHEST_DEGREE_EVER |
| | | CV_HIGHEST_DEGREE_YR |

Variables Used

| Name in Program | Question Name on CD | Name in Program | Question Name on CD |
|-------------------|-------------------------|--------------------|-----------------------------|
| R1_3500 | YSCH-3500(Round1) | E9335M21-E9335M22 | YSCH-9335.02.01~M-.02~M |
| R1_5000 | YSCH-5000(Round1) | E9335Y21-E9335Y22 | YSCH-9335.02.01~Y-.02~Y |
| E1605 | YSCH-1605 | E9335M31-E9335M32 | YSCH-9335.03.01~M-.02~M |
| E16151-E16152 | YSCH-1615.01-.02 | E9335Y31-E9335Y32 | YSCH-9335.03.01~Y-.02~Y |
| E16241-E16242 | YSCH-1624.01-.02 | E9589111-E9589114 | YSCH-9589.01.01.01-.04 |
| E16271 | YSCH-1627.01 | E9589121-E9589122 | YSCH-9589.01.02.01-.02 |
| E2806 | YSCH-2806 | E9589211-E9589216 | YSCH-9589.02.01.01-.06 |
| E2857 | YSCH-2857 | E9589221 | YSCH-9589.02.02.01 |
| E3061 | YSCH-3061 | E9589311-E9589313 | YSCH-9589.03.01.01-.03 |
| E3112 | YSCH-3112 | E9589321 | YSCH-9589.03.02.01 |
| E340011-E340061 | YSCH-3400.01.01-.06.01 | E9589411 | YSCH-9589.04.01.01 |
| E3878 | YSCH-3878 | E9589511 | YSCH-9589.05.01.01 |
| E4793 | YSCH-4793 | E9946111-E9946114 | YSCH-9946.01.01.01-.04 |
| E4795 | YSCH-4795 | E9946121-E9946122 | YSCH-9946.01.02.01-.02 |
| E4951 | YSCH-4951 | E9946211-E9946216 | YSCH-9946.02.01.01-.06 |
| E67841-E67845 | YSCH-6784.01-.05 | E9946221 | YSCH-9946.02.02.01 |
| E69381-E69385 | YSCH-6938.01-.05 | E9946311-E9946313 | YSCH-9946.03.01.01-.03 |
| E69431-E69433 | YSCH-6943.01.01-.03.01 | E9946321 | YSCH-9946.03.02.01 |
| E71421-E71424 | YSCH-7142.01.01-.04.01 | E9946411 | YSCH-9946.04.01.01 |
| E71921-E71925 | YSCH-7192.01-.05 | E9946511 | YSCH-9946.05.01.01 |
| E841611-E841613 | YSCH-8416.01.01-.03 | E199M111-E199M113 | YSCH-10099.01.01.01-M-.03~M |
| E841621-E841622 | YSCH-8416.02.01-.02 | E199Y111-E199Y113 | YSCH-10099.01.01.01-Y-.03~Y |
| E841631-E841632 | YSCH-8416.03.01-.02 | E199M121, E199Y121 | YSCH-10099.01.02.01-M, ~Y |
| E841641-E841642 | YSCH-8416.04.01-.02 | E199M211, E199Y211 | YSCH-10099.02.01.01-M, ~Y |
| E841651 | YSCH-8416.05.01 | E199M212, E199Y212 | YSCH-10099.02.01.02-M, ~Y |
| E841661 | YSCH-8416.06.01 | E199M214, E199Y214 | YSCH-10099.02.01.04-M, ~Y |
| E913111-E913113 | YSCH-9131.01.01-.03 | E199M215, E199Y215 | YSCH-10099.02.01.05-M, ~Y |
| E913121-E913122 | YSCH-9131.02.01-.02 | E199M311-E199M312 | YSCH-10099.03.01.01-M-.02~M |
| E913131-E913132 | YSCH-9131.03.01-.02 | E199Y311-E199Y312 | YSCH-10099.03.01.01-Y-.02~Y |
| E913141-E913142 | YSCH-9131.04.01-.02 | E11700 | YSCH-11700 |
| E913151 | YSCH-9131.05.01 | E11900M, E11900Y | YSCH-11900~M, ~Y |
| E913161 | YSCH-9131.06.01 | E234501-E234504 | YSCH-23450.01-.04 |
| E9335M11-E9335M13 | YSCH-9335.01.01-M-.03~M | E234601M, E234601Y | YSCH-23460.01~M, ~Y |
| E9335Y11-E9335Y13 | YSCH-9335.01.01-Y-.03~Y | E273371-E273375 | YSCH-27337.01-.05 |

Codes for Created Variables

Enrollment Status (CV_ENROLLSTAT)

- 1. not enrolled, no high school degree, no GED
- 2. not enrolled, GED, but no high school degree
- 3. not enrolled, a high school degree, no GED
- 4. not enrolled, some college
- 5. not enrolled, 2 year college graduate
- 6. not enrolled, 4 year college graduate
- 7. not enrolled, graduate degree
- 8. enrolled in grades 1-12
- 9. enrolled in a 2 year college
- 10. enrolled in a 4 year program
- 11. enrolled in a graduate program

Highest Grade Completed (CV_HGC_EVER, CV_HGC_YR)

- 1. none
- 2. ungraded
- 3. first
- 4. second
- 5. third
- 12. tenth
- 13. eleventh
- 14. twelfth
- 15. first year college
- 16. second year college

- | | |
|------------|--------------------------|
| 6. fourth | 17. third year college |
| 7. fifth | 18. fourth year college |
| 8. sixth | 19. fifth year college |
| 9. seventh | 20. sixth year college |
| 10. eighth | 21. seventh year college |
| 11. ninth | 22. eighth year college |

- | | |
|--|---|
| Highest Degree Received (CV_HIGHEST_DEGREE_EVER, CV_HIGHEST_DEGREE_YR) | |
| 0. none | 4. bachelor's degree (B.A., B.S., or unspecified) |
| 1. GED | 5. master's degree |
| 2. high school diploma | 6. doctoral degree |
| 3. junior college or 2-year associate degree | 7. professional degree (M.D., L.L.D., D.D.S., J.D.) |

This program first creates an enrollment status variable for each respondent. It then identifies the highest grade completed by the respondent as of the round 2 interview date and as of June 30, 1998. Finally, it determines the highest degree or diploma received by the respondent as of the same two dates. In round 2, no respondent had completed a degree higher than a high school diploma and/or GED. Therefore, in practice, the range of actual values is limited to [0,2].

```

/* Initialize variables. */
ENROLL=-16; /* Indicator of whether respondent currently enrolled */
ENCAT=-16; /* Enrollment status variable (including educational attainment) */
NODIP=0; /* Indicator of whether respondent does not have a diploma (high school or GED) */
           /* Note: must be filled in from Round 1 */
HS=0; /* Indicator of whether respondent is currently attending grades 1 to 12 */
COLLEGE=0; /* Indicator of whether respondent is current attending 2 or 4 year college */
FLAGCOL=0; /* Indicator to flag respondents currently attending grades 1 to 12 and enrolled in college */
FLAGLOOP=0; /* Indicator to flag respondents going through old school re-enrollment loop twice */

/* Defining arrays for use later in the program */
array E1615 E16151 E16152;
array E1624 E16241 E16242;
array E3400 E340011 E340021 E340031 E340041 E340051 E340061;
array E5458 E54581 E54582 E54583 E54584 E54585;
array E6784 E67841 E67842 E67843 E67844 E67845;
array E6938 E69381 E69382 E69383 E69384 E69385;
array E6943 E69431 E69432 E69433;
array E7142 E71421 E71422 E71423 E71424;
array E7192 E71921 E71922 E71923 E71924 E71925;
array E84161 E841611 E841612 E841613;
array NE84161 NE841611 NE841612 NE841613;
array ME84161 ME841611 ME841612 ME841613;
array E84162 E841621 E841622;
array NE84162 NE841621 NE841622;
array ME84162 ME841621 ME841622;
array E84163 E841631 E841632;
array NE84163 NE841631 NE841632;
array ME84163 ME841631 ME841632;
array E84164 E841641 E841642;
array E91311 E913111 E913112 E913113;
array E91312 E913121 E913122;
array E91313 E913131 E913132;
array E91314 E913141 E913142;
array E9335M1 E9335M11 E9335M12 E9335M13;
array E9335Y1 E9335Y11 E9335Y12 E9335Y13;
array E9335M2 E9335M21 E9335M22;
array E9335Y2 E9335Y21 E9335Y22;

```

```

array E9335M3 E9335M31 E9335M32;
array E9335Y3 E9335Y31 E9335Y32;
array E958911 E9589111 E9589112 E9589113 E9589114;
array NE95811 NE958111 NE958112 NE958113;
array ME95811 ME958111 ME958112 ME958113;
array E958912 E9589121 E9589122;
array E958921 E9589211 E9589212 E9589213 E9589214 E9589215 E9589216;
array NE95821 NE958211 NE958212 NE958213 NE958214 NE958215 NE958216;
array ME95821 ME958211 ME958212 ME958213 ME958214 ME958215 ME958216;
array E958931 E9589311 E9589312 E9589313;
array NE95831 NE958311 NE958312;
array ME95831 ME958311 ME958312;
array E994611 E9946111 E9946112 E9946113 E9946114;
array E994612 E9946121 E9946122;
array E994621 E9946211 E9946212 E946213 E9946214 E9946215 E9946216;
array E994631 E9946311 E9946312 E9946313;
array E199M11 E199M111 E199M112 E199M113;
array E199Y11 E199Y111 E199Y112 E199Y113;
array E199M21 E199M211 E199M212 E199M214 E199M215 E199M216;
array E199Y21 E199Y211 E199Y212 E199Y214 E199Y215 E199Y216;
array E199M31 E199M311 E199M312;
array E199Y31 E199Y311 E199Y312;
array E23450 E234501 E234502 E234503 E234504;
array E27337 E273371 E273372 E273373 E273374 E273375;

```

******* ENROLLMENT STATUS *******

*/*Begin by defining the conditions when a respondent is enrolled. Note that summer vacations are included as part of the enrollment period. */*

/ Continuously enrolled in old school since dli */
if E4795=1 then ENROLL=1;
if E4793=1 then ENROLL=1;*

/ Not continuously enrolled and not enrolled in any other schools since */
if E4795=0 and E4951=0 then ENROLL=0;*

/ Not continuously enrolled in old school since dli, have not re-enrolled in old school since dli and do not report any new schools since dli */
if E1605=-2 then ENROLL=-2;
if E4795=0 and E1605=0 and E4951=0 then ENROLL=0;
if E4793=0 and E1605=0 and E4951=0 then ENROLL=0;
if E3878=0 then ENROLL=0;*

/ Not continuously enrolled in old school since dli, have reenrolled in old school a maximum of 2 times, have not reported any other schools attended since */
if E4793=0 and E1605=1 and E16151=1 then ENROLL=1;
if E4793=0 and E1605=1 and E16152=1 then ENROLL=1;*

/ Not continuously enrolled in old school since dli, have received degree and/or completed course work at old school (i.e. YSCH-1600=1), and has not reported new school since */
if E4793=0 and E4951=0 and E1605=-4 then ENROLL=0;*

/ RE-ENROLLMENT IN OLD SCHOOL. Note that respondents can only go through the re-enrollment loop twice and it is not possible to determine which individuals re-enrolled a third time. Individuals who have gone through the loop twice and were not continuously enrolled up to the interview during the last enrollment spell and who do not report any new schools will be coded as not enrolled. */*

```

/* Flag those who go through the re-enrollment loop twice */
if E16241=1 then FLAGLOOP=1;
if E16242=1 then FLAGLOOP=1;

if (E4795=0 and E1605=1) then do;
  if E16152=-2 then ENROLL=-2;
  if E16151=1 then ENROLL=1; /* Continuously enrolled in old school, first time through loop.*/
  if E16151=0 and E16241=0 and E16271=0 and E4951=0 then ENROLL=0; /* Not continuously enrolled in
    old school, first time through loop, no other re-enrollments at old school and no new schools reported.*/
  if E16152=0 and E16242=1 and E4951=0 then ENROLL=0; /* Not continuously enrolled in old school, second
    time through loop and no new schools reported.*/
  if E16152=1 then ENROLL=1; /* Continuously enrolled in old school, second time through loop.*/
end;

/* ENROLLMENT IN NEW SCHOOLS */
if (E4951=1 or E3878=1) then do;
  do I=1 to 5; /* Allow for up to five new schools to be recorded.*/
    if E6784(I)=1 and E7192(I)=0 then ENROLL=1; /* Continuously enrolled in new school, no other new
      schools reported.*/
    if E6784(I)=0 and E7192(I)=0 then ENROLL=0; /* Not continuously enrolled in new school, did not re-
      enroll in new school, no other new schools reported.*/
/* ENROLLMENT PERIODS IN NEW SCHOOLS */
  if (E6784(I)=0 and E6938(I)=1) then do;
    if E6943(I)=1 and E7192(I)=0 then ENROLL=1; /* Continuously re-enrolled in new school, no other new
      schools reported.*/
    if E6943(I)=0 and E7142(I)=0 and E7192(I)=0 then ENROLL=0; /* Not continuously re-enrolled in new
      school, not re-enrolled in new school for a subsequent spell, no other new schools reported.*/
  end;
end;
end;

/* ENROLLMENT STATUS CATEGORIES */
/* No respondents have GED at dli coded as don't know/refused and no respondents have between grades 9 and 12,
  are not currently attending and do not already have a GED by dli. Therefore, the GED question is skipped in
  Round 2. As a result, GED status must be filled in using variables from Round 1 and NODIP must be appropriately
  updated as well. */

/*Need to fill in NODIP from Round 1 */
if E11700<1 then NODIP=1;

/* NOT ENROLLED */
if (ENROLL=0) then do;
  if NODIP=1 then ENCATE=1; /* No high school degree, no GED */
  if E11700=1 then ENCATE=3; /* High school degree */
  if 13<=E2857<=20 then ENCATE=4; /* Highest grade attended is at least 1 year of college */
  if (E234501=1 or E234502=1 or E234503=1 or E234504=1) then ENCATE=5; /* Received junior
    college or 2 year associate degree (need to check each school, allow for up to 5 new
    schools)*/
  if (E234501=3 or E234502=3 or E234503=3 or E234504=3) then ENCATE=6; /* Received bachelor's
    degree (need to check each school, allow for up to 5 new schools)*/
  do I=1 to 4; /* Received master's, doctoral or professional degree (need to check each school, allow for
    up to 5 new schools)*/
    if 4<=E23450(I)<=6 then ENCATE=7;
  end;
end;

```

```

/* ENROLLED */
if (ENROLL=1) then do;
    if (E273372=-2 or E273373=-2)then ENCAT=-2;      /* Missing values */
    if (E273372=-2 or E273373=-2)then COLLEGE=-2;
    if (E273372=-2 or E273373=-2)then FLAGCOL=-2;
    if 1<=E2857<=12 and NODIP=1 then ENCAT=8;      /* Highest grade attended as of today less than or
        equal to 12th grade and don't have a diploma/GED */
    if (E273371=1 or E273372=1 or E273373=1 or E273374=1 or E273375=1) and (13<=E2857<95) then
        ENCAT=9;          /* Working towards an Associate/Junior College or two-year associate
        degree (allow up to five schools) and current grade is greater than high school */
    if (E273371=3 or E273372=3 or E273373=3 or E273374=3 or E273375=3) and (13<=E2857<95) then
        ENCAT=10;         /* Working towards a Bachelor's degree and current grade is greater
        than high school */
    do I=1 to 5;          /* Working towards a graduate degree and current grade is greater than high school */
        if 4<=E27337(I)<=6 and 13<=E2857<95 then ENCAT=11;
    end;
end;

```

/* The following is for respondents who are enrolled, have no evidence of a high school diploma, yet reported that highest grade attended was 1st year of college. Since there is no evidence these people have a diploma and have not attended a new school, they will be placed under enrolled in grades 1-12. */

if ENROLL=1 and E2857=13 and NODIP=1 then ENCAT=8;

/* The following accounts for the respondents who reported a new school since dli. It is assumed that for respondents who reported working towards a master's or PhD (YSCH-27337) that also report less than 2 years of college are actually still in undergraduate college. Therefore they were placed in the college and enrolled ENCAT variable. */

```

if (ENROLL=1) then do;
    do I=1 to 5;
        if 0<=E5458(I)<=3 then ENCAT=8;
        if E5458(I)=4 then ENCAT=9;
        if E5458(I)=5 then ENCAT=10;
    end;
end;

```

/* The following is for respondents who have a high school diploma, a valid date for the diploma, have not enrolled in a new school since dli, and report the highest grade completed being 12. They have been decided to be categorized as not enrolled with a high school diploma (ENCAT=3) despite the fact that these respondents claim to be continuously enrolled since dli. Since there is no evidence of college from the respondent, they have been grouped as not enrolled. */

if E11700=1 and E11900M>0 and E11900Y>0 and E4951=0 and E3112=12 then ENCAT=3;

/* The following is for respondents who report the highest grade attended as ungraded (YSCH-2857=95), show no evidence of a high school diploma, were skipped out of the homeschool question, and are enrolled. Again, since there is no evidence of a high school diploma, they are regarded as enrolled between grades 1 and 12 */

if E2857=95 and NODIP=1 and ENROLL=1 then ENCAT=8;

/* Special hand edited case */
if PUBID=5998 then ENCAT=1;

/* Flag individuals who are enrolled in grades 1 to 12 and are enrolled in 2 or 4 year colleges (could be getting credits towards GED). For created enrollment status variable above, the flagged individuals will fall into category 8. */

Appendix 1: Education Variable Creation

```
if ENROLL=1 and 1<=E2857<=12 and NODIP=1 then HS=1;
if ENROLL=1 and (E273371=1 or E273372=1 or E273373=1 or E273374=1 or E273375=1 or E273371=3 or
E273372=3 or E273373=3 or E273374=3 or E273375=3) then COLLEGE=1;
if HS=1 and COLLEGE=1 then FLAGCOL=1;

***** HIGHEST GRADE COMPLETED AS OF THE SURVEY DATE *****/
/* Start with highest grade completed at dli and update. */

if -4<R1_5000<0 then GRSURV=R1_5000; /* Missing values */

/* Highest grade at dli attended is the starting grade for GRSURV */
if R1_5000=0 then GRSURV=1;
if R1_5000=95 then GRSURV=2;
if 0<R1_5000<95 then GRSURV=R1_5000+2;

/* The following accounts for corrections in highest grade completed from Round 1. YSCH-3061 is the check that
makes sure highest grade completed from Round 1 is correct. If yes, respondent gave true highest grade completed at
Round 1 interview date at YSCH-3061. The same is repeated for R1_3500, which uses YSCH-3104 as the check for
Round1 current grade. */

if E3061>-1 then GRSURV=E3061+2;
if E3104>-1 then R1_3500=E3104;

/* Due to conflicting information, highest grade completed at survey is no longer collected at E3112. Using E3112
produced several people saying they completed 9th grade at E3112 but then saying they didn't complete 9th at
E913111 (and other questions). So, GRSURV will start at R1_5000, which is highest grade completed at Round 1.
It will be updated if and only if the respondent completes another grade and gives a valid date. Note that don't
knows or refusals of the month of completion will still be considered valid as completing that grade. Variables that
begin with "N" designate new variables created so that the original values won't be affected in the list file. */

NR1_3500=R1_3500;

/* Submit the negativity conditions */
do I=1 to 3; if -4<E91311(I)<0 then GRSURV=E91311(I); end;
do I=1 to 2;
  if -4<E91312(I)<0 then GRSURV=E91312(I);
  if -4<E91313(I)<0 then GRSURV=E91313(I);
  if -4<E91314(I)<0 then GRSURV=E91314(I);
end;
if -4<E913151<0 then GRSURV=E913151;
if -4<E913161<0 then GRSURV=E913161;

do I=1 to 3; if -4<E84161(I)<0 then GRSURV=E84161(I); end;
do I=1 to 2;
  if -4<E84162(I)<0 then GRSURV=E84162(I);
  if -4<E84163(I)<0 then GRSURV=E84163(I);
  if -4<E84164(I)<0 then GRSURV=E84164(I);
end;
if -4<E841651<0 then GRSURV=E841651;
if -4<E841661<0 then GRSURV=E841661;
do I=1 to 4; if -4<E994611(I)<0 then GRSURV=E994611(I); end;
do I=1 to 2; if -4<E994612(I)<0 then GRSURV=E994612(I); end;
do I=1 to 6; if -4<E994621(I)<0 then GRSURV=E994621(I); end;
if -4<E9946221<0 then GRSURV=E9946221;
do I=1 to 3; if -4<E994631(I)<0 then GRSURV=E994631(I); end;
if -4<E9946321<0 then GRSURV=E9946321;
```

```

if -4<E9946411<0 then GRSURV=E9946411;
if -4<E9946511<0 then GRSURV=E9946511;
do I=1 to 4; if -4<E958911(I)<0 then GRSURV=E958911(I); end;
do I=1 to 2; if -4<E958912(I)<0 then GRSURV=E958912(I); end;
do I=1 to 6; if -4<E958921(I)<0 then GRSURV=E958921(I); end;
if -4<E9589221<0 then GRSURV=E9589221;
do I=1 to 3; if -4<E958931(I)<0 then GRSURV=E958931(I); end;
if -4<E9589321<0 then GRSURV=E9589321;
if -4<E9589411<0 then GRSURV=E9589411;
if -4<E9589511<0 then GRSURV=E9589511;

/* Normally, R1_3500 or (if R1_3500 is incorrect) E2806 is used for questions about if grade completed (E913111)
and when (E9335M11). E841611 is skipped in these cases. When E841611 is not skipped, then it must take the
place of R1_3500. */

if E841611>-4 then NR1_3500=E841611;

NE841611=E841611; NE841612=E841612; NE841613=E841613;
NE841621=E841621; NE841622=E841622;
NE841631=E841631; NE841632=E841632;
NE841641=E841641;
NE841651=E841651;
NE841661=E841661;

/* Due to character limits on naming variables, created "N" variables will not be exactly the same name as the
originals */ /* The "9" is dropped to limit the name to 8 letters */
NE958111=E9589111; NE958112=E9589112; NE958113=E9589113;
NE958121=E9589121;
NE958211=E9589211; NE958212=E9589212; NE958214=E9589214; NE958215=E9589215;
NE958311=E9589311; NE958312=E9589312;

/* Assign missing values for grades not completed, invalid year information, or a non-interview case. */
if E913111<1 or E9335M11<-4 or E9335Y11<=0 then NR1_3500=.;
do I=2 to 3; if E91311(I)<1 or E9335M1(I)<-4 or E9335Y1(I)<=0 then NE84161(I)=.; end;
do I=1 to 2;
    if E91312(I)<1 or E9335M2(I)<-4 or E9335Y2(I)<=0 then NE84162(I)=.;
    if E91313(I)<1 or E9335M3(I)<-4 or E9335Y3(I)<=0 then NE84163(I)=.;
end;
if E913141<1 or E9335M41<-4 or E9335Y41<=0 then NE841641=.;
if E913151<1 or E9335M51<-4 or E9335Y51<=0 then NE841651=.;
if E913161<1 or E9335M61<-4 or E9335Y61<=0 then NE841661=;

do I=1 to 3; if E994611(I)<1 or E199M11(I)<-4 or E199Y11(I)<=0 then NE95811(I)=.; end;
if E9946121<1 or E199M121<-4 or E199Y121<=0 then NE958121=.;
do I=1 to 2; if E994621(I)<1 or E199M21(I)<-4 or E199Y21(I)<=0 then NE95821(I)=.; end;
do I=4 to 5; if E994621(I)<1 or E199M21(I)<-4 or E199Y21(I)<=0 then NE95821(I)=.; end;
do I=1 to 2; if E994631(I)<1 or E199M31(I)<-4 or E199Y31(I)<=0 then NE95831(I)=.; end;

/* Define "MAXGRAD1" as highest grade in the following grade update loop */
MAXGRAD1=max (NR1_3500, NE841612, NE841613, NE841621, NE841622, NE841631, NE841632,
NE841641, NE841651, NE841661, NE958111, NE958112, NE958113, NE958121, NE958211, NE958212,
NE958214, NE958215, NE958311, NE958312);

/* Need to limit maxgrade so that it only changes GRSURV if it is not a valid skip (-4) or an ungraded year (95) */
if -4<MAXGRAD1<95 then GRSURV=MAXGRAD1+2;

```

Appendix 1: Education Variable Creation

***** HIGHEST DEGREE RECEIVED AS OF THE SURVEY DATE *****

NOTE: Only available for those who were enrolled since the last interview. For respondents who not enrolled since the date of last interview, need highest degree as reported in Round 1 (skip to employment sect. at YSCH-3960).
/* Initialize everyone to zero */

DEGSURV=0;

```
/* if GED=1 then DEGSURV=1 GED */
if E11700=1 then DEGSURV=2; /* High school diploma */
if (E234501=1 or E234502=1 or E234503=1 or E234504=1) then DEGSURV=3; /* Junior college or two-year
associate degree */
if (E234501=3 or E234502=3 or E234503=3 or E234504=3) then DEGSURV=4; /* Bachelor's degree */
if (E234501=4 or E234502=4 or E234503=4 or E234504=4) then DEGSURV=5; /* Master's degree */
if (E234501=5 or E234502=5 or E234503=5 or E234504=5) then DEGSURV=6; /* Doctoral degree */
if (E234501=6 or E234502=6 or E234503=6 or E234504=6) then DEGSURV=7; /* Professional degree */
```

***** HIGHEST GRADE COMPLETED AS OF JUNE 30, 1998 *****

/* Start with the highest grade completed as of the date of the last interview and update. */

```
/* Initialize everyone to -16 to test missing values */
GRJUNE=-16;
```

```
if -4<R1_5000<0 then GRJUNE=R1_5000; /* Missing values */
```

```
/* Highest grade at dli attended is the starting grade for GRJUNE */
if R1_5000=0 then GRJUNE=1;
if R1_5000=95 then GRJUNE=2;
if 0<R1_5000<95 then GRJUNE=R1_5000+2;
```

```
/* If the respondent answers E3061, then GRJUNE is the value for E3061 */
if E3061>-1 then GRJUNE=E3061+2;
```

```
/* Added the condition that if the year is less than 1998 and the month of the grade completed (E9335M(I)) is -2
(don't know) does not mean GRJUNE=-2. However if the year is 1998 and the month is a -2 (don't know), then
GRJUNE should be -2. If the grade the respondent was enrolled in since the last interview was completed before
June 30, 1998, then update */
MR1_3500=R1_3500;
```

```
/* Missing values */
do I=1 to 3;
  if -4<E9335M1(I)<0 and E9335Y1(I)=1998 then GRJUNE=E9335M1(I);
  if -4<E9335Y1(I)<0 then GRJUNE=E9335Y1(I);
end;
do I=1 to 2;
  if -4<E9335M2(I)<0 and E9335Y2(I)=1998 then GRJUNE=E9335M2(I);
  if -4<E9335Y2(I)<0 then GRJUNE=E9335Y2(I);
  if -4<E9335M3(I)<0 and E9335Y3(I)=1998 then GRJUNE=E9335M3(I);
  if -4<E9335Y3(I)<0 then GRJUNE=E9335Y3(I);
end;
if -4<E9335M41<0 and E9335Y41=1998 then GRJUNE=E9335M41;
if -4<E9335Y41<0 then GRJUNE=E9335Y41;
if -4<E9335M51<0 and E9335Y51=1998 then GRJUNE=E9335M51;
if -4<E9335Y51<0 then GRJUNE=E9335Y51;
if -4<E9335M61<0 and E9335Y61=1998 then GRJUNE=E9335M61;
```

```

if -4<E9335Y61<0 then GRJUNE=E9335Y61;

do I=1 to 3; if -4<E91311(I)<0 then GRJUNE=E91311(I); end;
do I=1 to 2;
  if -4<E91312(I)<0 then GRJUNE=E91312(I);
  if -4<E91313(I)<0 then GRJUNE=E91313(I);
  if -4<E91314(I)<0 then GRJUNE=E91314(I);
end;
if -4<E913151<0 then GRJUNE=E913151;
if -4<E913161<0 then GRJUNE=E913161;

do I=1 to 3; if -4<E84161(I)<0 then GRJUNE=E84161(I); end;
do I=1 to 2;
  if -4<E84162(I)<0 then GRJUNE=E84162(I);
  if -4<E84163(I)<0 then GRJUNE=E84163(I);
  if -4<E84164(I)<0 then GRJUNE=E84164(I);
end;
if -4<E841651<0 then GRJUNE=E841651;
if -4<E841661<0 then GRJUNE=E841661;

do I=1 to 3;
  if -4<E199M11(I)<0 and E199Y11(I)=1998 then GRJUNE=E199M11(I);
  if -4<E199Y11(I)<0 then GRJUNE=E199Y11(I);
end;
if -4<E199M121<0 and E199Y121=1998 then GRJUNE=E199M121;
if -4<E199Y121<0 then GRJUNE=E199Y121;
do I=1 to 2;
  if -4<E199M21(I)<0 and E199Y21(I)=1998 then GRJUNE=E199M21(I);
  if -4<E199Y21(I)<0 then GRJUNE=E199Y21(I);
end;
do I=4 to 5;
  if -4<E199M21(I)<0 and E199Y21(I)=1998 then GRJUNE=E199M21(I);
  if -4<E199Y21(I)<0 then GRJUNE=E199Y21(I);
end;
do I=1 to 2;
  if -4<E199M31(I)<0 and E199M31(I)=1998 then GRJUNE=E199M31(I);
  if -4<E199Y31(I)<0 then GRJUNE=E199Y31(I);
end;

do I=1 to 4; if -4<E994611(I)<0 then GRJUNE=E994611(I); end;
do I=1 to 2; if -4<E994612(I)<0 then GRJUNE=E994612(I); end;
do I=1 to 6; if -4<E994621(I)<0 then GRJUNE=E994621(I); end;
if -4<E9946221<0 then GRJUNE=E9946221;
do I=1 to 3; if -4<E994631(I)<0 then GRJUNE=E994631(I); end;
if -4<E9946321<0 then GRJUNE=E9946321;
if -4<E9946411<0 then GRJUNE=E9946411;
if -4<E9946511<0 then GRJUNE=E9946511;

do I=1 to 4; if -4<E958911(I)<0 then GRJUNE=E958911(I); end;
do I=1 to 2; if -4<E958912(I)<0 then GRJUNE=E958912(I); end;
do I=1 to 6; if -4<E958921(I)<0 then GRJUNE=E958921(I); end;
if -4<E9589221<0 then GRJUNE=E9589221;
do I=1 to 3; if -4<E958931(I)<0 then GRJUNE=E958931(I); end;
if -4<E9589321<0 then GRJUNE=E9589321;
if -4<E9589411<0 then GRJUNE=E9589411;
if -4<E9589511<0 then GRJUNE=E9589511;

```

Appendix 1: Education Variable Creation

```
/* Normally, R1_3500 is used for questions about grade completed (E913111) and when (E9335M11). E841611 is
skipped in these cases. When E841611 is not skipped, then it must take the place of R1_3500. */
if E841611>-4 then MR1_3500=E841611;

ME841611=E841611; ME841612=E841612; ME841613=E841613;
ME841621=E841621; ME841622=E841622;
ME841631=E841631; ME841632=E841632;
ME841641=E841641;
ME841651=E841651;
ME841661=E841661;

/* Again, due to character limitations, created "M" variable names will not be exactly the same as the originals */
ME958111=E9589111; ME958112=E9589112; ME958113=E9589113;
ME958121=E9589121;
ME958211=E9589211; ME958212=E9589212; ME958214=E9589214; ME958215=E9589215;
ME958311=E9589311; ME958312=E9589312;

/* The following lines only count the maximum grade completed if it comes before 7/98. The first line of each loop
accounts for grades completed in 1998. During 1998, a valid month between January and June must be given to
count towards GRJUNE. However, before 1998 any month can be given, including a don't know (-2), a refusal (-1),
or an invalid skip (-3).*/
if (E913111<1 or E9335M11<=0 or E9335M11>6) and E9335Y11=1998 then MR1_3500=.;
if E913111<1 or E9335Y11>1998 or E9335Y11<=0 then MR1_3500=.;
do I=2 to 3;
  if (E91311(I)<1 or E9335M1(I)<=0 or E9335M1(I)>6) and E9335Y1(I)=1998 then ME84161(I)=.;
  if E91311(I)<1 or E9335Y1(I)>1998 or E9335Y1(I)<=0 then ME84161(I)=.;
end;
do I=1 to 2;
  if (E91312(I)<1 or E9335M2(I)<=0 or E9335M2(I)>6) and E9335Y2(I)=1998 then ME84162(I)=.;
  if E91312(I)<1 or E9335Y2(I)>1998 or E9335Y2(I)<=0 then ME84162(I)=.;
  if (E91313(I)<1 or E9335M3(I)<=0 or E9335M3(I)>6) and E9335Y3(I)=1998 then ME84163(I)=.;
  if E91313(I)<1 or E9335Y3(I)>1998 or E9335Y3(I)<=0 then ME84163(I)=.;
end;
if (E913141<1 or E9335M41<=0 or E9335M41>6) and E9335Y41=1998 then ME841641=.;
if E913141<1 or E9335Y41>1998 or E9335Y41<=0 then ME841641=.;
if (E913151<1 or E9335M51<=0 or E9335M51>6) and E9335Y51=1998 then ME841651=.;
if E913151<1 or E9335Y51>1998 or E9335Y51<=0 then ME841651=.;
if (E913161<1 or E9335M61<=0 or E9335M61>6) and E9335Y61=1998 then ME841661=.;
if E913161<1 or E9335Y61>1998 or E9335Y61<=0 then ME841661=.

do I=1 to 3;
  if (E994611(I)<1 or E199M11(I)<=0 or E199M11(I)>6) and E199Y11(I)=1998 then ME95811(I)=.;
  if E994611(I)<1 or E199Y11(I)>1998 or E199Y11(I)<=0 then ME95811(I)=.;
end;
if (E9946121<1 or E199M121<=0 or E199M121>6) and E199Y121=1998 then ME958121=.;
if E9946121<1 or E199Y121>1998 or E199Y121<=0 then ME958121=.;
do I=1 to 2;
  if (E994621(I)<1 or E199M21(I)<=0 or E199M21(I)>6) and E199Y21(I)=1998 then ME95821(I)=.;
  if E994621(I)<1 or E199Y21(I)>1998 or E199Y21(I)<=0 then ME95821(I)=.;
end;
do I=4 to 5;
  if (E994621(I)<1 or E199M21(I)<=0 or E199M21(I)>6) and E199Y21(I)=1998 then ME95821(I)=.;
  if E994621(I)<1 or E199Y21(I)>1998 or E199Y21(I)<=0 then ME95821(I)=.;
end;
do I=1 to 2;
  if (E994631(I)<1 or E199M31(I)<=0 or E199M31(I)>6) and E199Y31(I)=1998 then ME95831(I)=.;
  if E994631(I)<1 or E199Y31(I)>1998 or E199Y31(I)<=0 then ME95831(I)=.;
```

```

end;

/* Define "MAXGRAD2" as highest grade in the following grade update loop */
MAXGRAD2=max (MR1_3500, ME841612, ME841613, ME841621, ME841622, ME841631, ME841632,
ME841641, ME841651, ME841661, ME958111, ME958112, ME958113, ME958121, ME958211, ME958212,
ME958214, ME958215, ME958311, ME958312);

/* Need to limit maxgrade so that it only changes GRSURV if it is not a valid skip (-4) or an ungraded year (95) */
if -4<MAXGRAD2<95 then GRJUNE=MAXGRAD2+2;

***** HIGHEST DEGREE COMPLETED AS OF JUNE 30, 1998 *****

/* Initialize everyone to zero */
DEGJUNE=0;

/*GEDM and GEDY are month and year the GED was obtained. These variables must be added from Round 1.*/
/* if GED=1 and 0<GEDM<=7 and 0<GEDY<=1998 then DEGJUNE=1 GED */
if E11700=1 and 0<E11900M<=7 and 0<E11900Y<=1998 then DEGJUNE=2; /* High school diploma */

/* No respondent reported more than one degree since the last interview */

if E234501=1 and 0<E234601M<7 and 0<E234601Y<=1998 then DEGJUNE=3; /* Junior college or 2-year
associate degree */
if E234501=3 and 0<E234601M<7 and 0<E234601Y<=1998 then DEGJUNE=4; /* Bachelor's degree */
if E234501=4 and 0<E234601M<7 and 0<E234601Y<=1998 then DEGJUNE=5; /* Master's degree */
if E234501=5 and 0<E234601M<7 and 0<E234601Y<=1998 then DEGJUNE=6; /* Doctoral degree */
if E234501=6 and 0<E234601M<7 and 0<E234601Y<=1998 then DEGJUNE=7; /* Professional degree */

if 95>GRSURV>0 and 95>E3112>0 then do;
    SURVEY=E3112+2; DIFF=GRSURV-SURVEY;
end;
miss=0;
if DIFF ne 0 then miss=1;

/* Account for non-interview cases */
if E2857=-5 then do; encat=-5; grjune=-5; grsurv=-5; degsurv=-5; degjune=-5; end;

/* GED hand edits */
if pubid in (137,4485,7949,8303) then do; degsurv=1; degjune=1; end;

if pubid in (137,4485,8303) then do; encat=2; end;

/* Hand edits for respondents with a HS diploma in Round1 */
if pubid in (64,995,1959,4163,7478,8148) then do; degsurv= ; degjune= ; end;

if pubid in (995,7478) then do; encat=3; end;

/* Hand edit for case with R1_3500=R1_5000=-4 */
if pubid=3178 then do; GRSURV=12; GRJUNE=12; end;

endsas;

```

CURRENT OR MOST RECENT SCHOOL PRIVATE OR PAROCHIAL

Variables Created: CV_SCHOOL_TYPE

Variables Used

| Name in Program | Question Name on CD |
|--------------------|----------------------------|
| S3400R1 | YSCH-3400 |
| PUBID | PUBID |
| TYPE1-TYPE6 | NEWSCHOOL_TYPE.01-06 |
| SPERIOD1-SPERIOD6 | NEWSCHOOL_PERIODS.01-06 |
| SSTOP1M1-SSTOP1M6 | NEWSCHOOL_STOP1.01~M-.06~M |
| SSTOP1Y1-SSTOP1Y6 | NEWSCHOOL_STOP1.01~Y-.06~Y |
| SSTOP2M1-SSTOP2M4 | NEWSCHOOL_STOP2.01~M-.04~M |
| SSTOP2Y1-SSTOP2Y4 | NEWSCHOOL_STOP2.01~Y-.04~Y |
| SSTOP3M1, SSTOP3Y1 | NEWSCHOOL_STOP3.01~M, ~Y |
| SCODE1-SCODE6 | NEWSCHOOL_SCHCODE.01-06 |
| SINTERV1-SINTERV6 | NEWSCHOOL_INTERVIEW.01-06 |

Codes for Created Variable

- 1 = Public school
- 2 = Private, not parochial
- 3 = Parochial
- 4 = Other

This program creates a variable indicating whether the respondent's current or most recent school is public, private, parochial, or other. The program first determines which school is the most recent school (non-college) by examining the end dates for different enrollment periods and then picks up the school type from round 1 or round 2 for the most recent school. Users should note that school type is defined only for respondents attending grades 1-12.

```

/* Create the array variables.*/
* # of enrollment periods;
  array SPERIOD SPERIOD1-SPERIOD6;
* ending date for each period;
  array SSTOP1M SSTOP1M1-SSTOP1M6;
  array SSTOP1Y SSTOP1Y1-SSTOP1Y6;
  array SSTOP2M SSTOP2M1-SSTOP2M6;
  array SSTOP2Y SSTOP2Y1-SSTOP2Y6;
  array SSTOP3M SSTOP3M1-SSTOP3M6;
  array SSTOP3Y SSTOP3Y1-SSTOP3Y6;
* school code, i.e. elementary school, high school or college;
  array SCODE SCODE1-SCODE6;
* indicate if the school info. is from round 1 or round 2;
  array SINTERV SINTERV1-SINTERV6;
*school type for round 2;
  array TYPE TYPE1-TYPE6;
* ending time for each school;
  array SRECM SRECM1-SRECM6;
  array SRECY SRECY1-SRECY6;
*school length;
  array SLENGTH SLENGTH1-SLENGTH6;

***** Step 1).find the most recent school. *****/
do I=1 to 6;
SRECM[I]=-4;
SRECY[I]=-4;

```

```

if SPERIOD[I]=1 and SCODE[I] in (1,2,3) then do;
    SRECM[I]=SSTOP1M[I]; SRECY[I]=SSTOP1Y[I];
end;

if SPERIOD[I]=2 and SCODE[I] in (1,2,3) then do;
    SRECM[I]=SSTOP2M[I]; SRECY[I]=SSTOP2Y[I];
end;

if SPERIOD[I]=3 and SCODE[I] in (1,2,3) then do;
    SRECM[I]=SSTOP3M[I]; SRECY[I]=SSTOP3Y[I];
end;

if SRECM[I] gt 0 and SRECY[I] gt 0 then SLENGTH[I]=12*(SRECY[I]-1990)+SRECM[I];
else SLENGTH[I]=-4;
end;

MAXLENG=MAX(SLENGTH1,SLENGTH2, SLENGTH3, SLENGTH4, SLENGTH5, SLENGTH6);
if TYPE1=-5 then MAXLENG=-5;

/***************** step 2).decide the school type from round 1 or round 2 information.*******/
SLTYPE1=-4;
do I=1 to 6;
if MAXLENG>0 and SLENGTH[I]=MAXLENG then do;
    if SINTERV[I]=1 then SLTYPE1=S3400R1;
    else if SINTERV[I]=2 then SLTYPE1=TYPE[I];
    end;
end;

if SCODE1 in (1,2,3) then do;
    if SPERIOD2=-4 and SPERIOD3=-4 and SPERIOD4=-4 and SPERIOD5=-4 and SPERIOD6=-4 then do;
        if SINTERV1=1 then SLTYPE1=S3400R1;
        if SINTERV1=2 then SLTYPE1=TYPE1;
        end; /* If the respondent only goes to one school in R2,input that school type.*/
end;

if MAXLENG=-4 and SCODE1 in (4,5) then SLTYPE=S3400R1;
if SPERIOD1=-4 and SPERIOD2=-4 and SPERIOD3=-4 and SPERIOD4=-4 and SPERIOD5=-4 and SPERIOD6=-4 then SLTYPE=S3400R1;

/********************* Recode the type variable.*******/
SLTYPE=-4;
if SLTYPE1=1 then SLTYPE=1;
if SLTYPE1=5 then SLTYPE=2;
if SLTYPE1 in (3,4) then SLTYPE=3;
if SLTYPE1=2 or SLTYPE1>=6 then SLTYPE=4;
if -4<SLTYPE1<0 then SLTYPE=-3;
if TYPE1=-5 then SLTYPE=-5;

/* if one of the school code is dk or refused, input the school type as -3.*/
do I=1 to 6;
    if SPERIOD[I] NE -4 and SCODE[I] in (0,-1,-2,-3) then SLTYPE=-3;
end;

endsas;

```

DATE RECEIVED DIPLOMA OR DEGREE

Variables Created: CV_GED
CV_HS_DIPLOMA

Variables Used

| Name in Program | Question Name on CD | Name in Program | Question Name on CD |
|-----------------|---------------------|--------------------|---------------------|
| GEDR1 | CV_GED | S13500M2, S13500Y2 | YSCH-13500~M, ~Y |
| HSR1 | CV_HS_DIPLOMA | S11700R2 | YSCH-11700R2 |
| S13300R2 | YSCH-13300 | S11900M2, S11900Y2 | YSCH-11900~M, ~Y |

This program creates two continuous month variables, one identifying the month that the respondent received a GED and the other a high school diploma. In future rounds, similar variables will calculate the month an associate's, bachelor's, master's, doctoral, or professional degree was received; no respondents had received these degrees in rounds 1 or 2. For more information on the continuous month scheme, see appendix 7 in this document.

```

/* First, create the date variables using round 2 infomation only.*/
HSR2=-4; GEDR2=-4; FLAG=0;

/* The date that the respondent received high school degree.*/
if S11700R2=1 and S11900M2 gt 0 and S11900Y2 gt 0 then HSR2=(S11900Y2-1980)*12+S11900M2;
if -4<S11700R2<0 then HSR2=S11700R2;
if -4<S11900M2<0 then HSR2=S11900M2;
if -4<S11900Y2<0 then HSR2=S11900Y2;

/* For the two persons who report that they got high school diploma in 1980 and 1988, we change the data
to 1998 since they finished 12th grade in 1998 and they also reported receiving high school diploma.*/
if HSR2=6 or HSR2=101 then HSR2=(1998-1980)*12+S11900M2;

/* The date that the respondent received GED.*/
if S13300R2=1 and S13500M2>0 and S13500Y2>0 then GEDR2=(S13500Y2-1980)*12+S13500M2;
if -4<S13300R2<0 then GEDR2=S13300R2;
if -4<S13500M2<0 then GEDR2=S13500M2;
if -4<S13500Y2<0 then GEDR2=S13500Y2;

***** Second, we will check if there is some conflict between round 1 and round 2.*****
FLAGHS=0; FLAGHS1=0; FLAGGED=0;
if HSR2 not in (-4,-5) and HSR1 ne -4 and HSR2 ne HSR1 then FLAGHS=1;
if HSR2 not in (-4,-5) and HSR1 ne -4 and HSR2=HSR1 then FLAGHS1=1;
if GEDR2 not in (-4,-5) and GEDR1 ne -4 and GEDR2 ne GEDR1 then FLAGGED=1;

***** Finally, we will merge the result from round 1 and round 2 together. *****/
DTGED=-4; DTHS=-4;

if FLAGGED=0 then do;
    if GEDR1 ne -4 then DTGED=GEDR1;
    if GEDR2 ne -4 then DTGED=GEDR2; end;

if FLAGHS=0 then do;
    if HSR1 NE -4 then DTHS=HSR1;
    if HSR2 NE -4 then DTHS=HSR2; end;

/* For the one person whose answer for the date of high diploma from round 2 (209) can not match with that from
round 1 (210), we just set the date according to round 2 (most recent information)*/
if FLAGHS=1 then DTHS=HSR2;

endsas;
```

NUMBER OF GRADES REPEATED OR SKIPPED

Variables Created: CV_GRADES_REPEAT_EVER CV_GRADES_REPEAT_YR
CV_GRADE_SKIPPED_EVER CV_GRADE_SKIPPED_YR

Variables Used

| Name in Program | Question Name on CD | Name in Program | Question Name on CD |
|--------------------|-----------------------|--------------------|--------------------------|
| DATER1D, M, Y | YINF-900_D, _M, _Y | S9589121-S9589122 | YSCH-9589.01.02.01-02 |
| S3500 | YSCH-3500 | S9589211-S9589216 | YSCH-9589.02.01.01-06 |
| PUBID | PUBID | S9589221 | YSCH-9589.02.02.01 |
| DATER2D, M, Y | YINTDATE~D, ~M, ~Y | S9589311-S9589313 | YSCH-9589.03.01.01-03 |
| S3103 | YSCH-3103 | S9589321 | YSCH-9589.03.02.01 |
| S3104 | YSCH-3104 | S9589411 | YSCH-9589.04.01.01 |
| S8163G11-S8163G13 | YSCH-8163BG.01.01-03 | S9589511 | YSCH-9589.05.01.01 |
| S8163G21-S8163G22 | YSCH-8163BG.02.01-02 | S691111M, S691111Y | YSCH-9691.01.01.01-M, ~Y |
| S8163G31-S8163G32 | YSCH-8163BG.03.01-02 | S691112M, S691112Y | YSCH-9691.01.01.02-M, ~Y |
| S8163G41-S8163G42 | YSCH-8163BG.04.01-02 | S691113M, S691113Y | YSCH-9691.01.01.03-M, ~Y |
| S8163G51 | YSCH-8163BG.05.01 | S691114M, S691114Y | YSCH-9691.01.01.04-M, ~Y |
| S8163G61 | YSCH-8163BG.06.01 | S691121M, S691121Y | YSCH-9691.01.02.01-M, ~Y |
| S841611-S841613 | YSCH-8416.01.01-03 | S691122M, S691122Y | YSCH-9691.01.02.02-M, ~Y |
| S841621-S841622 | YSCH-8416.02.01-02 | S691211M, S691211Y | YSCH-9691.02.01.01-M, ~Y |
| S841631-S841632 | YSCH-8416.03.01-02 | S691212M, S691212Y | YSCH-9691.02.01.02-M, ~Y |
| S841641-S841642 | YSCH-8416.04.01-02 | S691213M, S691213Y | YSCH-9691.02.01.03-M, ~Y |
| S841651 | YSCH-8416.05.01 | S691214M, S691214Y | YSCH-9691.02.01.04-M, ~Y |
| S841661 | YSCH-8416.06.01 | S691215M, S691215Y | YSCH-9691.02.01.05-M, ~Y |
| S877411M, S877411Y | YSCH-8774.01.01-M, ~Y | S691216M, S691216Y | YSCH-9691.02.01.06-M, ~Y |
| S877412M, S877412Y | YSCH-8774.01.02-M, ~Y | S691221M, S691221Y | YSCH-9691.02.02.01-M, ~Y |
| S877413M, S877413Y | YSCH-8774.01.03-M, ~Y | S691311M, S691311Y | YSCH-9691.03.01.01-M, ~Y |
| S877421M, S877421Y | YSCH-8774.02.01-M, ~Y | S691312M, S691312Y | YSCH-9691.03.01.02-M, ~Y |
| S877422M, S877422Y | YSCH-8774.02.02-M, ~Y | S691313M, S691313Y | YSCH-9691.03.01.03-M, ~Y |
| S877431M, S877431Y | YSCH-8774.03.01-M, ~Y | S691321M, S691321Y | YSCH-9691.03.02.01-M, ~Y |
| S877432M, S877432Y | YSCH-8774.03.02-M, ~Y | S691411M, S691411Y | YSCH-9691.04.01.01-M, ~Y |
| S877441M, S877441Y | YSCH-8774.04.01-M, ~Y | S691511M, S691511Y | YSCH-9691.05.01.01-M, ~Y |
| S877442M, S877442Y | YSCH-8774.04.02-M, ~Y | S9742111 S9742114 | YSCH-9742.01.01.01-04 |
| S877451M, S877451Y | YSCH-8774.05.01-M, ~Y | S9742121 S9742122 | YSCH-9742.01.02.01-02 |
| S892611-S892613 | YSCH-8926.01.01-03 | S9742211 S9742216 | YSCH-9742.02.01.01-06 |
| S892621-S892622 | YSCH-8926.02.01-02 | S9742221 | YSCH-9742.02.02.01 |
| S892631-S892632 | YSCH-8926.03.01-02 | S9742311 S9742313 | YSCH-9742.03.01.01-03 |
| S892641-S892642 | YSCH-8926.04.01-02 | S9742321 | YSCH-9742.03.02.01 |
| S892651 | YSCH-8926.05.01 | S9742411 | YSCH-9742.04.01.01 |
| S892661 | YSCH-8926.06.01 | S9742511 | YSCH-9742.05.01.01 |
| S897711-S897712 | YSCH-8977.01.01-02 | S9793111 S9793114 | YSCH-9793.01.01.01-04 |
| S897721 | YSCH-8977.02.01 | S9793211 S9793213 | YSCH-9793.02.01.01-03 |
| S897731-S897732 | YSCH-8977.03.01-02 | S9793311 | YSCH-9793.03.01.01 |
| S897741 | YSCH-8977.04.01 | S9844111 S9844114 | YSCH-9844.01.01.01-04 |
| S902811-S902813 | YSCH-9028.01.01-03 | S9844121 S9844122 | YSCH-9844.01.02.01-02 |
| S902821-S902822 | YSCH-9028.02.01-02 | S9844211 S9844216 | YSCH-9844.02.01.01-06 |
| S902831-S902832 | YSCH-9028.03.01-02 | S9844221 | YSCH-9844.02.02.01 |
| S902841-S902842 | YSCH-9028.04.01-02 | S9844311 S9844313 | YSCH-9844.03.01.01-03 |
| S902851 | YSCH-9028.05.01 | S9844321 | YSCH-9844.03.02.01 |
| S902861 | YSCH-9028.06.01 | S9844411 | YSCH-9844.04.01.01 |
| S908011-S908013 | YSCH-9080.01.01-03 | S9844511 | YSCH-9844.05.01.01 |
| S908021-S908022 | YSCH-9080.02.01-02 | S9895111 S9895113 | YSCH-9895.01.01.01-03 |
| S908031-S908032 | YSCH-9080.03.01-02 | S9895211 S9895215 | YSCH-9895.02.01.01-05 |
| S908041 | YSCH-9080.04.01 | S9895311 | YSCH-9895.03.01.01 |
| S908051 | YSCH-9080.05.01 | S9946111 S9946114 | YSCH-9946.01.01.01-04 |
| S908061 | YSCH-9080.06.01 | S9946121 S9946122 | YSCH-9946.01.02.01-02 |
| S913111-S913113 | YSCH-9131.01.01-03 | S9946211 S9946216 | YSCH-9946.02.01.01-06 |
| S913121-S913122 | YSCH-9131.02.01-02 | S9946221 | YSCH-9946.02.02.01 |
| S913131-S913132 | YSCH-9131.03.01-02 | S9946311 S9946313 | YSCH-9946.03.01.01-03 |
| S913141-S913142 | YSCH-9131.04.01-02 | S9946321 | YSCH-9946.03.02.01 |

Appendix 1: Education Variable Creation

| | | | |
|--------------------|-----------------------|--------------------|---------------------------|
| S913151 | YSCH-9131.05.01 | S9946411 | YSCH-9946.04.01.01 |
| S913161 | YSCH-9131.06.01 | S9946511 | YSCH-9946.05.01.01 |
| S933511M, S933511Y | YSCH-9335.01.01~M, ~Y | S099111M, S099111Y | YSCH-10099.01.01.01~M, ~Y |
| S933512M, S933512Y | YSCH-9335.01.02~M, ~Y | S099112M, S099112Y | YSCH-10099.01.01.02~M, ~Y |
| S933513M, S933513Y | YSCH-9335.01.03~M, ~Y | S099113M, S099113Y | YSCH-10099.01.01.03~M, ~Y |
| S933521M, S933521Y | YSCH-9335.02.01~M, ~Y | S099121M, S099121Y | YSCH-10099.01.02.01~M, ~Y |
| S933522M, S933522Y | YSCH-9335.02.02~M, ~Y | S099211M, S099211Y | YSCH-10099.02.01.01~M, ~Y |
| S933531M, S933531Y | YSCH-9335.03.01~M, ~Y | S099212M, S099212Y | YSCH-10099.02.01.02~M, ~Y |
| S933532M, S933532Y | YSCH-9335.03.02~M, ~Y | S099214M, S099214Y | YSCH-10099.02.01.04~M, ~Y |
| S933541M, S933541Y | YSCH-9335.04.01~M, ~Y | S099215M, S099215Y | YSCH-10099.02.01.05~M, ~Y |
| S933551M, S933551Y | YSCH-9335.05.01~M, ~Y | S099311M, S099311Y | YSCH-10099.03.01.01~M, ~Y |
| S933561M, S933561Y | YSCH-9335.06.01~M, ~Y | S099312M, S099312Y | YSCH-10099.03.01.02~M, ~Y |
| S938511-S938513 | YSCH-9385.01.01~03 | START1M, START1Y | NEWSCHOOL_START1.01~M, ~Y |
| S938521-S938522 | YSCH-9385.02.01~02 | START2M, START2Y | NEWSCHOOL_START1.02~M, ~Y |
| S938531-S938532 | YSCH-9385.03.01~02 | START3M, START3Y | NEWSCHOOL_START1.03~M, ~Y |
| S938541-S938542 | YSCH-9385.04.01~02 | START4M, START4Y | NEWSCHOOL_START1.04~M, ~Y |
| S938551 | YSCH-9385.05.01 | START5M, START5Y | NEWSCHOOL_START1.05~M, ~Y |
| S938561 | YSCH-9385.06.01 | START6M, START6Y | NEWSCHOOL_START1.06~M, ~Y |
| S9589111-S9589114 | YSCH-9589.01.01.01~04 | | |

This program creates variables counting the number of grades the respondent ever repeated or skipped. The first variable in each pair counts the total number of grades ever repeated/skipped through the interview date; the second variable calculates the number through June 30, 1998.

```

***** Define arrays used later on. *****
/* For the arrays that include both school number and the period, put the same period in one array.*/
/* Start date for the first period for each school. */
array startm start1m start2m start3m start4m start5m start6m;
array starty start1y start2y start3y start4y start5y start6y;
/* School is elementary, middle or high school. */
array s8163g s8163g11 s8163g21 s8163g31 s8163g41 s8163g51 s8263g61;
/* first grade for each school and period.*/
array s84161 s841611 s841621 s841631 s841641 s841651 s841661;
array s84162 s841612 s841622 s841632 s841642 s841652 s841662;
array s84163 s841613 s841623 s841633 s841643 s841653 s841663;
/* Start date for the first grade in each school and each period.*/
array s87741m s877411m s877421m s877431m s877441m s877451m;
array s87741y s877411y s877421y s877431y s877441y s877451y;
array s87742m s877412m s877422m s877432m s877442m s877452m;
array s87742y s877412y s877422y s877432y s877442y s877452y;
array s87743m s877413m s877423m s877433m s877443m s877453m;
array s87743y s877413y s877423y s877433y s877443y s877453y;
/* Questions about skip or repeat.*/
array s89261 s892611 s892621 s892631 s892641 s892651 s892661;
array s89262 s892612 s892622 s892632 s892642 s892652 s892662;
array s89263 s892613 s892623 s892633 s892643 s892653 s892663;
array s89771 s897711 s897721 s897731 s897741 s897751 s897761;
array s89772 s897712 s897722 s897732 s897742 s897752 s897762;
array s90281 s902811 s902821 s902831 s902841 s902851 s902861;
array s90282 s902812 s902822 s902832 s902842 s902852 s902862;
array s90283 s902813 s902823 s902833 s902843 s902853 s902863;
array s90801 s908011 s908021 s908031 s908041 s908051 s908061;
array s90802 s908012 s908022 s908032 s908042 s908052 s908062;
array s90803 s908013 s908023 s908033 s908043 s908053 s908063;
/*complete the first grade or not?*/
array s91311 s913111 s913121 s913131 s913141 s913151 s913161;
array s91312 s913112 s913122 s913132 s913142 s913152 s913162;
array s91313 s913113 s913123 s913133 s913143 s913153 s913163;

```

```

/* When complete the first grade? */
array s93351m s933511m s933521m s933531m s933541m s933551m;
array s93351y s933511y s933521y s933531y s933541y s933551y;
array s93352m s933512m s933522m s933532m s933542m s933552m;
array s93352y s933512y s933522y s933532y s933542y s933552y;
array s93353m s933513m s933523m s933533m s933543m s933553m;
array s93353y s933513y s933523y s933533y s933543y s933553y;
/* Enrolled in any other grades? */
array s93851 s938511 s938521 s938531 s938541 s938551 s938561;
array s93852 s938512 s938522 s938532 s938542 s938552 s938562;
array s93853 s938513 s938523 s938533 s938543 s938553 s938563;

/* For the loops with school number, period number and the grade number, we use the period grade number for
each array.*/
/* Next grades in each school and period.*/
array s958911 s9589111 s9589211 s9589311 s9589411 s9589511;
array s958912 s9589112 s9589212 s9589312 s9589412 s9589512;
array s958913 s9589113 s9589213 s9589313 s9589413 s9589513;
array s958914 s9589114 s9589214 s9589314 s9589414 s9589514;
array s958915 s9589115 s9589215 s9589315 s9589415 s9589515;
array s958916 s9589116 s9589216 s9589316 s9589416 s9589516;
array s958921 s9589121 s9589221 s9589321 s9589421 s9589521;
array s958922 s9589122 s9589222 s9589322 s9589422 s9589522;
/* Start date for next grade.*/
array s69111m s691111m s691211m s691311m s691411m s691511m;
array s69112m s691112m s691212m s691312m s691412m s691512m;
array s69113m s691113m s691213m s691313m s691413m s691513m;
array s69114m s691114m s691214m s691314m s691414m s691514m;
array s69115m s691115m s691215m s691315m s691415m s691515m;
array s69116m s691116m s691216m s691316m s691416m s691516m;
array s69121m s691121m s691221m s691321m s691421m s691521m;
array s69122m s691122m s691222m s691322m s691422m s691522m;
array s69111y s691111y s691211y s691311y s691411y s691511y;
array s69112y s691112y s691212y s691312y s691412y s691512y;
array s69113y s691113y s691213y s691313y s691413y s691513y;
array s69114y s691114y s691214y s691314y s691414y s691514y;
array s69115y s691115y s691215y s691315y s691415y s691515y;
array s69116y s691116y s691216y s691316y s691416y s691516y;
array s69121y s691121y s691221y s691321y s691421y s691521y;
array s69122y s691122y s691222y s691322y s691422y s691522y;
/* Questions about skip or repeat grades.*/
array s974211 s9742111 s9742211 s9742311 s9742411 s9742511;
array s974212 s9742112 s9742212 s9742312 s9742412 s9742512;
array s974213 s9742113 s9742213 s9742313 s9742413 s9742513;
array s974214 s9742114 s9742214 s9742314 s9742414 s9742514;
array s974215 s9742115 s9742215 s9742315 s9742415 s9742515;
array s974216 s9742116 s9742216 s9742316 s9742416 s9742516;
array s974221 s9742121 s9742221 s9742321 s9742421 s9742521;
array s974222 s9742122 s9742222 s9742322 s9742422 s9742522;
array s979311 s9793111 s9793211 s9793311 s9793411 s9793511;
array s979312 s9793112 s9793212 s9793312 s9793412 s9793512;
array s979313 s9793113 s9793213 s9793313 s9793413 s9793513;
array s979314 s9793114 s9793214 s9793314 s9793414 s9793514;
array s984411 s9844111 s9844211 s9844311 s9844411 s9844511;
array s984412 s9844112 s9844212 s9844312 s9844412 s9844512;
array s984413 s9844113 s9844213 s9844313 s9844413 s9844513;
array s984414 s9844114 s9844214 s9844314 s9844414 s9844514;

```

```

array s984415 s9844115 s9844215 s9844315 s9844415 s9844515;
array s984416 s9844116 s9844216 s9844316 s9844416 s9844516;
array s984421 s9844121 s9844221 s9844321 s9844421 s9844521;
array s984422 s9844122 s9844222 s9844322 s9844422 s9844522;
array s989511 s9895111 s9895211 s9895311 s9895411 s9895511;
array s989512 s9895112 s9895212 s9895312 s9895412 s9895512;
array s989513 s9895113 s9895213 s9895313 s9895413 s9895513;
array s989514 s9895114 s9895214 s9895314 s9895414 s9895514;
array s989515 s9895115 s9895215 s9895315 s9895415 s9895515;
/* complete next grade or not? */
array s994611 s9946111 s9946211 s9946311 s9946411 s9946511;
array s994612 s9946112 s9946212 s9946312 s9946412 s9946512;
array s994613 s9946113 s9946213 s9946313 s9946413 s9946513;
array s994614 s9946114 s9946214 s9946314 s9946414 s9946514;
array s994615 s9946115 s9946215 s9946315 s9946415 s9946515;
array s994616 s9946116 s9946216 s9946316 s9946416 s9946516;
array s994621 s9946121 s9946221 s9946321 s9946421 s9946521;
array s994622 s9946122 s9946222 s9946322 s9946422 s9946522;
/* When complete the next grade? */
array s09911m s099111m s099211m s099311m s099411m s099511m;
array s09912m s099112m s099212m s099312m s099412m s099512m;
array s09913m s099113m s099213m s099313m s099413m s099513m;
array s09914m s099114m s099214m s099314m s099414m s099514m;
array s09915m s099115m s099215m s099315m s099415m s099515m;
array s09916m s099116m s099216m s099316m s099416m s099516m;
array s09921m s099121m s099221m s099321m s099421m s099521m;
array s09922m s099122m s099222m s099322m s099422m s099522m;
array s09911y s099111y s099211y s099311y s099411y s099511y;
array s09912y s099112y s099212y s099312y s099412y s099512y;
array s09913y s099113y s099213y s099313y s099413y s099513y;
array s09914y s099114y s099214y s099314y s099414y s099514y;
array s09915y s099115y s099215y s099315y s099415y s099515y;
array s09916y s099116y s099216y s099316y s099416y s099516y;
array s09921y s099121y s099221y s099321y s099421y s099521y;
array s09922y s099122y s099222y s099322y s099422y s099522y;
/* Array for grades completed and the grades attended in round 2, the starting time and the completing time.*/
array datestm datestm1-datestm9; array datesty datesty1-datesty9;
array datespm datespm1-datespm9; array datespy datespy1-datespy9;
array comp comp1-comp9; array attend attend1-attend9;

do i=1 to 9;
    datestm[i]=-4; datesty[i]=-4; datespm[i]=-4; datespy[i]=-4; comp[i]=-4; attend[i]=-4;
end;

***** List the grades completed and the ending date *****/
j=1;
if s913111=1 then do;
    if s841611 not in (-4,-5) then do;
        comp[j]=s841611; datespm[j]=s933511m; datespy[j]=s933511y; j=j+1; end;
    if s841611=-4 and s3104 not in (-4,-5) then do;
        comp[j]=s3104; datespm[j]=s933511m; datespy[j]=s933511y; j=j+1; end;
    if s841611=-4 and s3104=-4 and s3500 not in (-4,-5) then do;
        comp[j]=s3500; datespm[j]=s933511m; datespy[j]=s933511y; j=j+1; end;
    end;
if s9946111=1 and s9589111 not in (-4,-5) then do;
    comp[j]=s9589111; datespm[j]=s099111m; datespy[j]=s099111y; j=j+1; end;

```

```

if s9946112=1 and s9589112 not in (-4,-5) then do;
  comp[j]=s9589112; datespm[j]=s099112m; datespy[j]=s099112y; j=j+1; end;
if s9946113=1 and s9589113 not in (-4,-5) then do;
  comp[j]=s9589113; datespm[j]=s099113m; datespy[j]=s099113y; j=j+1; end;
if s9946114=1 and s9589114 not in (-4,-5) then do;
  comp[j]=s9589114; datespm[j]=s099114m; datespy[j]=s099114y; j=j+1; end;
if s913112=1 and s841612 not in (-4,-5) then do;
  comp[j]=s841612; datespm[j]=s933512m; datespy[j]=s933512y; j=j+1; end;
if s9946121=1 and s9589121 not in (-4,-5) then do;
  comp[j]=s9589121; datespm[j]=s099121m; datespy[j]=s099121y; j=j+1; end;
if s9946122=1 and s9589122 not in (-4,-5) then do;
  comp[j]=s9589122; datespm[j]=s099122m; datespy[j]=s099122y; j=j+1; end;
if s913113=1 and s841613 not in (-4,-5) then do;
  comp[j]=s841613; datespm[j]=s933513m; datespy[j]=s933513y; j=j+1; end;
if s913121=1 and s841621 not in (-4,-5) then do;
  comp[j]=s841621; datespm[j]=s933521m; datespy[j]=s933521y; j=j+1; end;
if s9946211=1 and s9589211 not in (-4,-5) then do;
  comp[j]=s9589211; datespm[j]=s099211m; datespy[j]=s099211y; j=j+1; end;
if s9946212=1 and s9589212 not in (-4,-5) then do;
  comp[j]=s9589212; datespm[j]=s099212m; datespy[j]=s099212y; j=j+1; end;
if s9946213=1 and s9589213 not in (-4,-5) then do;
  comp[j]=s9589213; datespm[j]=s099213m; datespy[j]=s099213y; j=j+1; end;
if s9946214=1 and s9589214 not in (-4,-5) then do;
  comp[j]=s9589214; datespm[j]=s099214m; datespy[j]=s099214y; j=j+1; end;
if s9946215=1 and s9589215 not in (-4,-5) then do;
  comp[j]=s9589215; datespm[j]=s099215m; datespy[j]=s099215y; j=j+1; end;
if s9946216=1 and s9589216 not in (-4,-5) then do;
  comp[j]=s9589216; datespm[j]=s099216m; datespy[j]=s099216y; j=j+1; end;
if s913122=1 and s841622 not in (-4,-5) then do;
  comp[j]=s841622; datespm[j]=s933522m; datespy[j]=s933522y; j=j+1; end;
if s9946221=1 and s9589221 not in (-4,-5) then do;
  comp[j]=s9589221; datespm[j]=s099221m; datespy[j]=s099221y; j=j+1; end;
if s913131=1 and s841631 not in (-4,-5) then do;
  comp[j]=s841631; datespm[j]=s933531m; datespy[j]=s933531y; j=j+1; end;
if s9946311=1 and s9589311 not in (-4,-5) then do;
  comp[j]=s9589311; datespm[j]=s099311m; datespy[j]=s099311y; j=j+1; end;
if s9946312=1 and s9589312 not in (-4,-5) then do;
  comp[j]=s9589312; datespm[j]=s099312m; datespy[j]=s099312y; j=j+1; end;
if s9946313=1 and s9589313 not in (-4,-5) then do;
  comp[j]=s9589313; datespm[j]=s099313m; datespy[j]=s099313y; j=j+1; end;
if s913132=1 and s841632 not in (-4,-5) then do;
  comp[j]=s841632; datespm[j]=s933532m; datespy[j]=s933532y; j=j+1; end;
if s9946321=1 and s9589321 not in (-4,-5) then do;
  comp[j]=s9589321; datespm[j]=s099321m; datespy[j]=s099321y; j=j+1; end;
if s913141=1 and s841641 not in (-4,-5) then do;
  comp[j]=s841641; datespm[j]=s933541m; datespy[j]=s933541y; j=j+1; end;
if s9946411=1 and s9589411 not in (-4,-5) then do;
  comp[j]=s9589411; datespm[j]=s099411m; datespy[j]=s099411y; j=j+1; end;
if s913142=1 and s841642 not in (-4,-5) then do;
  comp[j]=s841642; datespm[j]=s933542m; datespy[j]=s933542y; j=j+1; end;
if s913151=1 and s841651 not in (-4,-5) then do;
  comp[j]=s841651; datespm[j]=s933551m; datespy[j]=s933551y; j=j+1; end;
if s9946511=1 and s9589511 not in (-4,-5) then do;
  comp[j]=s9589511; datespm[j]=s099511m; datespy[j]=s099511y; j=j+1; end;
if s913161=1 and s841661 not in (-4,-5) then do;
  comp[j]=s841661; datespm[j]=s933561m; datespy[j]=s933561y; j=j+1; end;

```

Appendix 1: Education Variable Creation

```
***** List the grades started in round 2 and the starting dates.*****
k=1;

if s913111 not in (-4,-5) then do;
  if s841611=-4 and s3104 not in (-4,-5) then do;
    attend[k]=s3104;
    if s877411m>0 then datestm[k]=s877411m;
    if s877411y>0 then datesty[k]=s877411y;
    if s877411m=-4 and s877411y=-4 and start1m>0 then datestm[k]=start1m;
    if s877411m=-4 and s877411y=-4 and start1y>0 then datesty[k]=start1y;
    if s877411m=-4 and s877411y=-4 and start1m=-4 and start1y=-4 and dater1m>0 then datestm[k]=dater1m;
    if s877411m=-4 and s877411y=-4 and start1m=-4 and start1y=-4 and dater1y>0 then datesty[k]=dater1y;
    k=k+1;
  end;

if s841611=-4 and s3104=-4 and s3500 not in (-4,-5) then do;
  attend[k]=s3500;
  if s877411m>0 then datestm[k]=s877411m;
  if s877411y>0 then datesty[k]=s877411y;
  if s877411m=-4 and s877411y=-4 and start1m>0 then datestm[k]=start1m;
  if s877411m=-4 and s877411y=-4 and start1y>0 then datesty[k]=start1y;
  if s877411m=-4 and s877411y=-4 and start1m=-4 and start1y=-4 and dater1m>0 then datestm[k]=dater1m;
  if s877411m=-4 and s877411y=-4 and start1m=-4 and start1y=-4 and dater1y>0 then datesty[k]=dater1y;
  k=k+1;
end;

if s841611 not in (-4,-5) then do;
  attend[k]=s841611;
  if s877411m>0 then datestm[k]=s877411m;
  if s877411y>0 then datesty[k]=s877411y;
  if s877411m=-4 and s877411y=-4 and start1m>0 then datestm[k]=start1m;
  if s877411m=-4 and s877411y=-4 and start1y>0 then datesty[k]=start1y;
  if s877411m=-4 and s877411y=-4 and start1m=-4 and start1y=-4 and dater1m>0 then datestm[k]=dater1m;
  if s877411m=-4 and s877411y=-4 and start1m=-4 and start1y=-4 and dater1y>0 then datesty[k]=dater1y;
  k=k+1;
end;
end;

if s938511=1 then do; /* check if the respondent went to other grades in that period.*/
  if s9946111 ne -4 then do; attend[k]=s9589111; datestm[k]=s691111m; datesty[k]=s691111y; k=k+1; end;
  if s9946112 ne -4 then do; attend[k]=s9589112; datestm[k]=s691112m; datesty[k]=s691112y; k=k+1; end;
  if s9946113 ne -4 then do; attend[k]=s9589113; datestm[k]=s691113m; datesty[k]=s691113y; k=k+1; end;
  if s9946114 ne -4 then do; attend[k]=s9589114; datestm[k]=s691114m; datesty[k]=s691114y; k=k+1; end;
end;

if s913112 not in (-4,-5) then do; attend[k]=s841612; datestm[k]=s877412m; datesty[k]=s877412y; k=k+1; end;

if s938512=1 then do; /* check if the respondent sent to other grades.*/
  if s9946121 ne -4 then do; attend[k]=s9589121; datestm[k]=s691121m; datesty[k]=s691121y; k=k+1; end;
  if s9946122 ne -4 then do; attend[k]=s9589122; datestm[k]=s691122m; datesty[k]=s691122y; k=k+1; end;
end;

if s913113 not in (-4,-5) then do; attend[k]=s841613; datestm[k]=s877413m; datesty[k]=s877413y; k=k+1; end;

if s913121 not in (-4,-5) then do;
  attend[k]=s841621;
  if s877421m>0 and s877421y>0 then do; datestm[k]=s877421m; datesty[k]=s877421y; end;
```

```

if s877421m=-4 and s877421y=-4 and start2m>0 and start2y>0 then do;
    datestm[k]=start2m; datesty[k]=start2y; end;
    k=k+1;
end;

if s938521=1 then do; /* check if the respondent went to other grades.*/
    if s9946211 ne -4 then do; attend[k]=s9589211; datestm[k]=s691211m; datesty[k]=s691211y; k=k+1; end;
    if s9946212 ne -4 then do; attend[k]=s9589212; datestm[k]=s691212m; datesty[k]=s691212y; k=k+1; end;
    if s9946213 ne -4 then do; attend[k]=s9589213; datestm[k]=s691213m; datesty[k]=s691213y; k=k+1; end;
    if s9946214 ne -4 then do; attend[k]=s9589214; datestm[k]=s691214m; datesty[k]=s691214y; k=k+1; end;
    if s9946215 ne -4 then do; attend[k]=s9589215; datestm[k]=s691215m; datesty[k]=s691215y; k=k+1; end;
    if s9946216 ne -4 then do; attend[k]=s9589216; datestm[k]=s691216m; datesty[k]=s691216y; k=k+1; end;
end;

if s913122 not in (-4,-5) then do;
    attend[k]=s841622; datestm[k]=s877422m; datesty[k]=s877422y; k=k+1; end;

if s938522=1 then do; /* enrolled in some other grade. */
    if s9946221 ne -4 then do; attend[k]=s9589221; datestm[k]=s691221m; datesty[k]=s691221y; k=k+1; end;
end;

if s913131 not in (-4,-5) then do;
    attend[k]=s841631;
    if s877431m>0 and s877431y>0 then do; datestm[k]=s877431m; datesty[k]=s877431y; end;
    if s877431m=-4 and s877431y=-4 and start3m>0 and start3y>0 then do;
        datestm[k]=start3m; datesty[k]=start3y; end;
    k=k+1;
end;

if s938531=1 then do; /* enrolled in some other grade.*/
    if s9946311 ne -4 then do; attend[k]=s9589311; datestm[k]=s691311m; datesty[k]=s691311y; k=k+1; end;
    if s9946312 ne -4 then do; attend[k]=s9589312; datestm[k]=s691312m; datesty[k]=s691312y; k=k+1; end;
    if s9946313 ne -4 then do; attend[k]=s9589313; datestm[k]=s691313m; datesty[k]=s691313y; k=k+1; end;
end;

if s913132 not in (-4,-5) then do; attend[k]=s841632; datestm[k]=s877432m; datesty[k]=s877432y; k=k+1; end;

if s938532=1 then do; /* enrolled in other grades.*/
    if s9946321 ne -4 then do; attend[k]=s9589321; datestm[k]=s691321m; datesty[k]=s691321y; k=k+1; end;
end;

if s913141 not in (-4,-5) then do;
    attend[k]=s841641;
    if s877441m>0 and s877441y>0 then do; datestm[k]=s877441m; datesty[k]=s877441y; end;
    if s877441m=-4 and s877441y=-4 and start4m>0 and start4y>0 then do;
        datestm[k]=start4m; datesty[k]=start4y; end;
    k=k+1;
end;

if s938541=1 then do; /* enrolled in other grades */
    if s9946411 ne -4 then do; attend[k]=s9589411; datestm[k]=s691411m; datesty[k]=s691411y; k=k+1; end;
end;

if s913142 not in (-4,-5) then do; attend[k]=s841642; datestm[k]=s877442m; datesty[k]=s877442y; k=k+1; end;

if s913151 not in (-4,-5) then do;
    attend[k]=s841651;

```

Appendix 1: Education Variable Creation

```
if s877451m>0 and s877451y>0 then do; datestm[k]=s877451m; datesty[k]=s877451y; end;
if s877451m=-4 and s877451y=-4 and start5m>0 and start5y>0 then do;
    datestm[k]=start5m; datesty[k]=start5y; end;
    k=k+1;
end;

if s938551=1 then do; /* enrolled in other grades.*/
    if s9946511 ne -4 then do; attend[k]=s9589511; datestm[k]=s691511m; datesty[k]=s691511y; k=k+1; end;
end;

if s913161 not in (-4,-5) then do; attend[k]=s841661; datestm[k]=s877461m; datesty[k]=s877461y; k=k+1; end;

/** After listing all the grades started and all the grades completed that are between 9th to 12th in round 2, we next
can count the grades skipped and the grades repeated. For the grades skipped, we can look at all the grades
completed, if there is gaps between these grades, we would say that the grades are skipped. For the grades repeated,
we need to combine the start date and the ending date for each grade to calculate the time that each respondent spent
finishing up each grade. If it took the respondent more than 13 month to finish that grade, we would consider that
grade is repeated. **/

***** Count the grades skipped *****/
array skip skip1-skip6;
array skipj skipj1-skipj6;
numskip=0;
numskipj=0;
p=1;
q=1;
flag=0;

/** If the 1st grade completed is more than one greater than the 1st grade attended, count the grades skipped. ***/
if attend[1]>0 and comp[1]>0 and comp[1]>attend[1] then do; /* find the start date for comp[1]*/
    do i=1 to 9;
        if attend[i]=comp[1] then do; datestmc=datestm[i]; datestyc=datesty[i]; i=9; end;
    end;

    if comp[1]-attend[1]=1 then do;
        skip[p]=attend[1]; p=p+1;
        if datestmc+(datestyc-1990)*12<=102 then do; skipj[q]=attend[1]; q=q+1; end;
    end;

    if comp[1]-attend[1]=2 then do;
        skip[p]=attend[1]; p=p+1; skip[p]=attend[1]+1; p=p+1;
        if datestmc+(datestyc-1990)*12<=102 then do; skipj[q]=attend[1]; q=q+1; skipj[q]=attend[1]+1; q=q+1; end;
    end;

    if comp[1]-attend[1]=3 then do;
        skip[p]=attend[1]; p=p+1; skip[p]=attend[1]+1; p=p+1; skip[p]=attend[1]+2; p=p+1;
        if datestmc+(datestyc-1990)*12<=102 then do;
            skipj[q]=attend[1]; q=q+1; skipj[q]=attend[1]+1; q=q+1; skipj[q]=attend[1]+2; q=q+1; end;
        end;

    if comp[1]-attend[1]=4 then do;
        skip[p]=attend[1]; p=p+1; skip[p]=attend[1]+1; p=p+1; skip[p]=attend[1]+2; p=p+1; skip[p]=attend[1]+3;
        p=p+1;
        if datestmc+(datestyc-1990)*12<=102 then do;
            skipj[q]=attend[1]; q=q+1; skipj[q]=attend[1]+1; q=q+1; skipj[q]=attend[1]+2; q=q+1; skipj[q]=attend[1]+3;
            q=q+1; end;
        end;
    end;
```

```

if comp[1]-attend[1]=5 then do;
    skip[p]=attend[1]; p=p+1; skip[p]=attend[1]+1; p=p+1; skip[p]=attend[1]+2; p=p+1; skip[p]=attend[1]+3;
        p=p+1; skip[p]=attend[1]+4; p=p+1;
    if datestmc+(datestyc-1990)*12<=102 then do;
        skipj[q]=attend[1]; q=q+1; skipj[q]=attend[1]+1; q=q+1; skipj[q]=attend[1]+2; q=q+1; skipj[q]=attend[1]+3;
            q=q+1; skipj[q]=attend[1]+4; q=q+1; end;
    end;

    if comp[1]-attend[1]>5 then flag=1;
end;

***** The grades skipped between the grades completed. *****
do i=1 to 8;

if comp[i+1]>0 and comp[i]>0 and comp[i+1]-comp[i]>1 then do; /* find the start date for comp[i+1]*/
    do l=1 to 9;
        if attend[l]=comp[i+1] then do; datestmm=datestm[l]; datestym=datesty[l]; l=9; end;
    end;

    if comp[i+1]-comp[i]=2 then do;
        skip[p]=comp[i]+1; p=p+1;
        if datestmm+(datestym-1990)*12<=102 then do; skipj[q]=comp[i]+1; q=q+1; end;
    end;

    if comp[i+1]-comp[i]=3 then do;
        skip[p]=comp[i]+1; p=p+1; skip[p]=comp[i]+2; p=p+1;
        if datestmm+(datestym-1990)*12<=102 then do;
            skipj[q]=comp[i]+1; q=q+1; skipj[q]=comp[i]+2; q=q+1; end;
    end;

    if comp[i+1]-comp[i]=4 then do;
        skip[p]=comp[i]+1; p=p+1; skip[p]=comp[i]+2; p=p+1; skip[p]=comp[i]+3; p=p+1;
        if datestmm+(datestym-1990)*12<=102 then do;
            skipj[q]=comp[i]+1; q=q+1; skipj[q]=comp[i]+2; q=q+1; skipj[q]=comp[i]+3; q=q+1; end;
    end;

    if comp[i+1]-comp[i]=5 then do;
        skip[p]=comp[i]+1; p=p+1; skip[p]=comp[i]+2; p=p+1; skip[p]=comp[i]+3; p=p+1; skip[p]=comp[i]+4; p=p+1;
        if datestmm+(datestym-1990)*12<=102 then do;
            skipj[q]=comp[i]+1; q=q+1; skipj[q]=comp[i]+2; q=q+1; skipj[q]=comp[i]+3; q=q+1; skipj[q]=comp[i]+4;
                q=q+1; end;
    end;

    if comp[i+1]-comp[i]>5 then flag=2;
end;
end;

/*If the grade attended in the last round is more than one year greater than the grade completed in the last round.**/
/* first find the last attended and last completed grades.*/

if j>1 and k>1 and attend[k-1]>0 and comp[j-1]>0 and attend[k-1]-comp[j-1]>1 then do;
    compl=comp[j-1]; attendl=attend[k-1]; end;

if j=1 and k>1 and attend[k-1]>0 and hgcr1>0 and attend[k-1]-hgcr1>1 then do;
    compl=hgcr1; attendl=attend[k-1]; end;

```

```

if attendl>0 and compl>0 then do;
  if attendl-compl=2 then do;
    skip[p]=compl+1; p=p+1;
    if datestm[k-1]+(datesty[k-1]-1990)*12<=102 then do; skipj[q]=compl+1; q=q+1; end;
  end;

  if attendl-compl=3 then do;
    skip[p]=compl+1; p=p+1; skip[p]=compl+2; p=p+1;
    if datestm[k-1]+(datesty[k-1]-1990)*12<=102 then do;
      skipj[q]=compl+1; q=q+1; skipj[q]=compl+2; q=q+1; end;
    end;

  if attendl-compl=4 then do;
    skip[p]=compl+1; p=p+1; skip[p]=compl+2; p=p+1; skip[p]=compl+3; p=p+1;
    if datestm[k-1]+(datesty[k-1]-1990)*12<=102 then do;
      skipj[q]=compl+1; q=q+1; skipj[q]=compl+2; q=q+1; skipj[q]=compl+3; q=q+1; end;
    end;

  if attendl-compl=5 then do;
    skip[p]=compl+1; p=p+1; skip[p]=compl+2; p=p+1; skip[p]=compl+3; p=p+1; skip[p]=compl+4; p=p+1;
    if datestm[k-1]+(datesty[k-1]-1990)*12<=102 then do;
      skipj[q]=compl+1; q=q+1; skipj[q]=compl+2; q=q+1; skipj[q]=compl+3; q=q+1; skipj[q]=compl+4; q=q+1; end;
    end;

    if datestm[k-1] in (-1,-2,-3) or datesty[k-1] in (-1,-2,-3) then skipj[q]=-3;
    if attendl-compl>5 then flag=3;
  end;

do i=1 to 6;
  if skip[i]>0 and skip[i]<=12 then numskip=numskip+1;
  if skipj[i]>0 and skipj[i]<=12 then numskipj=numskipj+1;
end;

if skipj[1] in (-1,-2,-3) or skipj[2] in (-1,-2,-3) or skipj[3] in (-1,-2,-3) or skipj[4] in (-1,-2,-3) or skipj[5] in (-1,-2,-3)
or skipj[6] in (-1,-2,-3) then numskipj=-3;

do i=1 to j-1;
  if comp[i] in (-1,-2,-3) then do; numskip=-3; numskipj=-3; end;
end;

if datestmc in (-1,-2,-3) or datestycc in (-1,-2,-3) or datestmm in (-1,-2,-3) or datestym in (-1,-2,-3) then numskipj=-3;

***** Count the grades repeated *****/
array grade grade1-grade6;
array lenth lenth1-lenth6;
array stop stop1-stop6/* the stop date for each grade*/;
l=1;
numrep=0;
numrepj=0;
fflag=0;

do i=1 to 9;
  do t=1 to 9;
    if attend[t]>0 and comp[i]>0 and attend[t]=comp[i] then do; grade[l]=attend[t];
    if datespm[i]>0 and datestm[t]>0 and datespy[i]>0 and datesty[t]>0 then do;
      lenth[l]=datespm[i]-datestm[t]+12*(datespy[i]-datesty[t]); stop[l]=datespm[i]+(datespy[i]-1990)*12; end;
  end;
end;

```

```

if datespm[i] in (-1,-2,-3) or datestm[t] in (-1,-2,-3) or datespy[i] in (-1,-2,-3) or datesty[t] in (-1,-2,-3) then
    lenth[l]=-3;
if datespm[i] in (-1,-2,-3) or datespy[i] in (-1,-2,-3) then stop[l]=-3;
l=l+1;
t=9;
end;
end;
end;

do i=1 to 6;
  if grade[i]>0 and grade[i]<=12 and lenth[i]>13 then do;
    numrep=numrep+1;
    if stop[i]<=102 then numrepj=numrepj+1;
  end;
end;

***** The last grade attended that are not completed in round2 and take more than 13 month. *****
if j=1 and hgcr1>0 then hgc=hgcr1;
if j>1 and comp[j-1]>0 then hgc=comp[j-1];

if k>1 and attend[k-1]>hgc and attend[k-1]>0 and attend[k-1]<=12 then do;
  if (dater2y-datesty[k-1])*12+(dater2m-datestm[k-1])>13 then do; numrep=numrep+1; fflag=1; end;
  if (1998-datesty[k-1])*12+(6-datestm[k-1])>13 then do; numrepj=numrepj+1; fflag=2; end;
end;

if lenth1=-3 or lenth2=-3 or lenth3=-3 or lenth4=-3 or lenth5=-3 or lenth6=-3 then do;
  numrep=-3; numrepj=-3; end;

if stop1=-3 or stop2=-3 or stop3=-3 or stop4=-3 or stop5=-3 or stop6=-3 then numrepj=-3;

***** other cases *****
if s3500 in (95,14) or s3104 in (95,14) or s841611 in (95,14) or s841612 in (95,14) or s841613 in (95,14) or
  s841621 in (95,14) or s841622 in (95,14) or s841631 in (95,14) or s841632 in (95,14) or s841641 in (95,14) or
  s841642 in (95,14) or s841651 in (95,14) or s841661 in (95,14) or s9589111 in (95,14) or
  s9589112 in (95,14) or s9589113 in (95,14) or s9589114 in (95,14) or s9589121 in (95,14) or
  s9589122 in (95,14) or s9589211 in (95,14) or s9589212 in (95,14) or s9589213 in (95,14) or
  s9589214 in (95,14) or s9589215 in (95,14) or s9589216 in (95,14) or s9589221 in (95,14) or
  s9589311 in (95,14) or s9589312 in (95,14) or s9589313 in (95,14) or s9589321 in (95,14) or
  s9589411 in (95,14) or s9589511 in (95,14) then do;
  numrep=-3; numskp=-3; numrepj=-3; numskipj=-3;
end;

/* we get length negative because of the data problem.*/
if (lenth1<0 and lenth1 ne .) or (lenth2<0 and lenth2 ne .) or (lenth3<0 and lenth3 ne .) or (lenth4<0 and lenth4 ne .
  .) or (lenth5<0 and lenth5 ne .) or (lenth6<0 and lenth6 ne .) then do;
  numrep=-3; numrepj=-3; numskip=-3; numskipj=-3;
end;

if s3104=-5 then do;
  numskip=-5; numrep=-5; numskipj=-5; numrepj=-5;
end;

endsas;

```

NUMBER OF SCHOOLS ATTENDED

Variables Created: CV_SCH_ATTEND_EVER
CV_SCH_ATTEND_YR

Variables Used

| Name in Program | Question Name on CD | Name in Program | Question Name on CD |
|-------------------|-----------------------|-------------------|-----------------------------|
| CURGDR1 | YSCH-3500 | S826361 | YSCH-8263.06.01 |
| PUBID | PUBID | S841611-S841613 | YSCH-8416.01.01-.03 |
| PINF15 | PINF-015_Y | S841621-S841622 | YSCH-8416.02.01-.02 |
| NUMJR1 | CV_SCH_ATTEND_YR | S841631-S841632 | YSCH-8416.03.01-.02 |
| HGAR2 | YSCH-2857 | S841641-S841642 | YSCH-8416.04.01-.02 |
| S66821M-S66825M | YSCH-6682.01~M-.05~M | S841651 | YSCH-8416.05.01 |
| S66821Y-S66825Y | YSCH-6682.01~Y-.05~Y | S841661 | YSCH-8416.06.01 |
| S8163G11-S8163G13 | YSCH-8163BG.01.01-.03 | S9589111-S9589114 | YSCH-9589.01.01.01-.04 |
| S8163G21-S8163G22 | YSCH-8163BG.02.01-.02 | S9589121-S9589122 | YSCH-9589.01.02.01-.02 |
| S8163G31-S8163G32 | YSCH-8163BG.03.01-.02 | S9589211-S9589216 | YSCH-9589.02.01.01-.06 |
| S8163G41-S8163G42 | YSCH-8163BG.04.01-.02 | S9589221 | YSCH-9589.02.02.01 |
| S8163G51 | YSCH-8163BG.05.01 | S9589311-S9589313 | YSCH-9589.03.01.01-.03 |
| S8163G61 | YSCH-8163BG.06.01 | S9589321 | YSCH-9589.03.02.01 |
| S826311-S826313 | YSCH-8263.01.01-.03 | S9589411 | YSCH-9589.04.01.01 |
| S826321-S826322 | YSCH-8263.02.01-.02 | S9589511 | YSCH-9589.05.01.01 |
| S826331-S826332 | YSCH-8263.03.01-.02 | START1M-START6M | NEWSCHOOL_START1.01~M-.06~M |
| S826341-S826342 | YSCH-8263.04.01-.02 | START1Y-START6Y | NEWSCHOOL_START1.01~Y-.06~Y |
| S826351 | YSCH-8263.05.01 | ROUND1-ROUND6 | NEWSCHOOL_INTERVIEW.01-.06 |

This program calculates the number of schools ever attended by the respondent as of the interview date and as of June 30, 1998. The program first counts the number of new regular schools that the youth attended since DLI to the date of survey using the first enrollment date in each new school, then adds this number to the created variable in round 1 to get the total number of regular schools that the youth have ever attended. Users should note that the variable counts only schools at which the respondent attended grades 7–12.

```

array startmx S66821M S66822M S66823M S66824M S66825M;
array startyx S66821Y S66822Y S66823Y S66824Y S66825Y;
array startm start1m start2m start3m start4m start5m start6m;
array starty start1y start2y start3y start4y start5y start6y;
array start start1-start6;
array S8263 S826311 S826321 S826331 S826341 S826351 S826361;
array dumgrd dumgrd1-dumgrd6;

/*First, create the dummy variables for each school to indicate if the youth attended grade 7–12 in that school.*/
do I=1 to 6;
  dumgrd[I]=0; start[I]=-4;
end;

if (7<=S841611<=12 or 7<=S9589111<=12 or 7<=S9589112<=12 or 7<=S9589113<=12 or 7<=S9589114<=12) or
  (7<=S841612<=12 or 7<=S9589121<=12 or 7<=S9589122<=12) or 7<=S841613<=12 then dumgrd1=1;
if (7<=S841621<=12 or 7<=S9589211<=12 or 7<=S9589212<=12 or 7<=S9589213<=12 or 7<=S9589214<=12 or
  7<=S9589215<=12 or 7<=S9589216<=12) or (7<=S841622<=12 or 7<=S9589221<=12) then dumgrd2=1;
if (7<=S841631<=12 or 7<=S9589311<=12 or 7<=S9589312<=12 or 7<=S9589313<=12) or (7<=S841632<=12 or
  7<=S9589321<=12) then dumgrd3=1;
if (7<=S841641<=12 OR 7<=S9589411<=12) or 7<=S841642<=12 then dumgrd4=1;
if (7<=S841651<=12 or 7<=S9589511<=12) then dumgrd5=1;
if 7<=S841661<=12 then dumgrd6=1;

if (-3<=S841611<=-1 or -3<=S9589111<=-1 or -3<=S95891-1<=-1 or -3<=S9589113<=-1 or -3<=S9589114<=-1) or
  (-3<=S8416-1<=-1 or -3<=S9589-11<=-1 or -3<=S9589-12<=-1) or -3<=S841613<=-1 then dumgrd1=-3;

```

Appendix 1: Education Variable Creation

```
if (-3<=S841621<=-1 or -3<=S9589211<=-1 or -3<=S95892-1<=-1 or -3<=S9589213<=-1 or -3<=S9589214<=-1 or  
-3<=S9589215<=-1 or -3<=S9589216<=-1) or (-3<=S841622<=-1 or -3<=S9589221<=-1) then dumgrd2=-3;  
if (-3<=S841631<=-1 or -3<=S9589311<=-1 or -3<=S95893-1<=-1 or -3<=S9589313<=-1) or (-3<=S841632<=-1 or  
-3<=S9589321<=-1) then dumgrd3=-3;  
if (-3<=S841641<=-1 or -3<=S9589411<=-1) or -3<=S841642<=-1 then dumgrd4=-3;  
if (-3<=S841651<=-1 or -3<=S9589511<=-1) then dumgrd5=-3;  
if -3<=S841661<=-1 then dumgrd6=-3;  
  
/*Next, calculate numjr2 and numr2.*/  
/* Calculate the accumulated month of the start date. */  
do I=1 to 6;  
    if starty[I]>0 and startm[I]>0 then start[I]=12*(starty[I]-1990)+startm[I];  
end;  
  
numjr2=0;  
numr2=0;  
do I=1 to 6;  
    if S8263[I]=0 then do;  
        if dumgrd[I]=1 then numr2=numr2+1;  
        if dumgrd[I]=1 and (0<start[I]<=102 or 0<starty[I]<1998) then numjr2=numjr2+1;  
    end;  
end;  
  
if dumgrd1=-3 or dumgrd2=-3 or dumgrd3=-3 or dumgrd4=-3 or dumgrd5=-3 or dumgrd6=-3 then do;  
    numr2=-3; numjr2=-3; end;  
  
if (S826311=0 and dumgrd1=1 and ((start1y=1998 and -4<start1m<0) or -4<start1y<0)) or (S826321=0 and  
dumgrd2=1 and ((start2y=1998 and -4<start2m<0) or -4<start2y<0)) or (S826331=0 and dumgrd3=1 and  
((start3y=1998 and -4<start3m<0) or -4<start3y<0)) or (S826341=0 and dumgrd4=1 and ((start4y=1998 and -4<start4m<0) or -4<start4y<0)) or (S826351=0 and dumgrd5=1 and ((start5y=1998 and -4<start5m<0) or -4<start5y<0)) or (S826361=0 and dumgrd6=1 and ((start6y=1998 and -4<start6m<0) or -4<start6y<0))  
    then numjr2=-3;  
  
/*We then add the above two numbers to the corresponding variables from round 1.*/  
if numjr1>=0 then do;  
    if numjr2>=0 then numj=numjr2+numjr1;  
    if numr2>=0 then nums=numr2+numjr1;  
    if numjr2=-3 then numj=-3;  
    if numr2=-3 then nums=-3;  
end;  
  
/*For the youth who didn't answer the number of schools in round 1, we distinguish them into three cases 1)If the  
grade currently attended in round 1 is <=7 and the highest grade attended in round 2 >= 7, we use the number from  
round 2 as the total number. 2)If the grade currently attended in round 1 >7, we use -3 for the total number. 3)If  
both the grade currently attended in round 1 and the highest grade attended in round 2 are <7, we use -4 for the  
total number.*/  
  
if PINF15=-4 then do;  
    if curgdr1<=7 then do; numj=numjr2; nums=numr2; end;  
    if curgdr1>7 then do; numj=-3; nums=-3; end;  
    if curgdr1<7 and hgar2<7 then do; numj=-4; nums=-4; end;  
end;  
  
if -4<numjr1<0 then do; numj=numjr1; nums=numjr1; end;  
  
if S826311=-5 then do; numjr2=-5; numr2=-5; numj=-5; nums=-5; end;  
endsas;
```

TOTAL FRACTION OF CREDITS EARNED TOWARDS BACHELORS/ASSOCIATE DEGREE

Variables Created: CV_BA_CREDITS.01-.05
CV_ASSOC_CREDITS.01-.05

Variables Used

| Name in Program | Question Name on CD | Name in Program | Question Name on CD |
|-----------------|---------------------|-------------------|----------------------|
| R1DTYPE | YSCH-1800 | APCRED1-APCRED4 | YSCH-26878.01-.04 |
| R1TCRED | YSCH-16000.01 | HSCRED1-HSCRED4 | YSCH-26929.01-.04 |
| R1CRED1 | YSCH-22800.01 | E273371-E273375 | YSCH-27337.01-.05 |
| PUBID | PUBID | E273881-E273885 | YSCH-27388.01-.05 |
| R1ASCRED | CV_ASSOC_CREDITS | E2070011-E2070015 | YSCH-20700.01.01-.05 |
| R1BACRED | CV_BA_CREDITS | E2070021-E2070025 | YSCH-20700.02.01-.05 |
| E16200 | YSCH-16200.01 | E2070031-E2070035 | YSCH-20700.03.01-.05 |
| E25807 | YSCH-25807.01 | E2070041 | YSCH-20700.04.01 |
| E25909 | YSCH-25909.01 | E2070051 | YSCH-20700.05.01 |
| E265211-E265215 | YSCH-26521.01-.05 | E230001-E230003 | YSCH-23000.01-.03 |
| XCRED1-XCRED4 | YSCH-26827.01-.04 | | |

Codes for Created Variable

These variables are created on a continuous scale. They have 2 implied decimal places.

This program calculates the fraction of credits the respondent has completed toward an associate's degree or a bachelor's degree.

/* Note that for all variables below the 1st loop is a school reported in Round1. Loops 2 through 5 are schools reported in Round2. */

```

array E26521 (i) E265211-E265215;
array XCRED (i) XCRED1-XCRED4;
array APCRED (i) APCRED1-APCRED4;
array HSCRED (i) HSCRED1-HSCRED4;
array E27337 (i) E273371-E273375;
array E27388 (i) E273881-E273885;
array INCRED (i) INCRED1-INCRED5;
array E207001 (i) E2070011-E2070015;
array E207002 (i) E2070021-E2070025;
array E207003 (i) E2070031-E2070035;
array E207004 (i) E2070041-E2070045;
array E207005 (i) E2070051-E2070055;
array cred (i) cred1-cred5;
array tcred (i) tcred1-tcred5;
array tascred (i) tascred1-tascred5;
array tbacred (i) tbacred1-tbacred5;
array asfrac (i) asfrac1-asfrac5;           /* Created variable for associate credits */
array bafrac (i) bafrac1-bafrac5;           /* Created variable for bachelor credits */

```

/* Round1 created variables are in fraction form times 100. R1CRED1 is the total earned credits variable. R1DTYPE is type of degree respondent is working toward. */

```

if E273371=1 and R1CRED1=>0 then do; R1ASCRED=R1CRED1; end;
if R1DTYPE=4 and E273371=-4 and R1CRED1=>0 then do; R1ASCRED=R1CRED1; end;
if E273371=3 and R1CRED1=>0 then do; R1BACRED=R1CRED1; end;
if R1DTYPE=5 and E273371=-4 and R1CRED1=>0 then do; R1BACRED=R1CRED1; end;

```

```
/* Hand edit case */ if pubid=4163 then R1ASCRED=-4;
```

Appendix 1: Education Variable Creation

```
/* If a respondent reported missing information for incoming credits and then give a valid answer in E16200, then
we will include that value to the Round1 credit count. If E16200 was not answered, then incoming credits for school
1 is simply the Round1 created variable. */

if R1BACRED=>0 and E16200=>0 then do; INCRED1=R1BACRED+E16200; end;
if R1ASCRED=>0 and E16200=>0 then do; INCRED1=R1ASCRED+E16200; end;
if R1ASCRED=>0 then do; INCRED1=R1ASCRED; end;
if R1BACRED=>0 then do; INCRED1=R1BACRED; end;

do i=1 to 5;
  if E26521=>0 then do; INCRED=E26521; end;
end;

/* 1st school */
if E2070011=>0 then do; cred1=E2070011; end;
if E2070011=>0 and E2070012=>0 then do; cred1=E2070011+E2070012; end;
if E2070011=>0 and E2070012=>0 and E2070013=>0 then do; cred1=E2070011+E2070012+E2070013; end;
if E2070011=>0 and E2070012=>0 and E2070013=>0 and E2070014=>0 then do;
  cred1=E2070011+E2070012+E2070013+E2070014; end;
if E2070011=>0 and E2070012=>0 and E2070013=>0 and E2070014=>0 and E2070015=>0 then do;
  cred1=E2070011+E2070012+E2070013+E2070014+E2070015; end;

/* 2nd school */
if E2070021=>0 then do; cred2=E2070021; end;
if E2070021=>0 and E2070022=>0 then do; cred2=E2070021+E2070022; end;
if E2070021=>0 and E2070022=>0 and E2070023=>0 then do; cred2=E2070021+E2070022+E2070023; end;
if E2070022=>0 and E2070022=>0 and E2070023=>0 and E2070024=>0 then do;
  cred2=E2070021+E2070022+E2070023+E2070024; end;
if E2070021=>0 and E2070022=>0 and E2070023=>0 and E2070024=>0 and E2070025=>0 then do;
  cred2=E2070021+E2070022+E2070023+E2070024+E2070025; end;

/* 3rd school */
if E2070031=>0 then do; cred3=E2070031; end;
if E2070031=>0 and E2070032=>0 then do; cred3=E2070031+E2070032; end;
if E2070031=>0 and E2070032=>0 and E2070033=>0 then do; cred3=E2070031+E2070032+E2070033; end;
if E2070031=>0 and E2070032=>0 and E2070033=>0 and E2070034=>0 then do;
  cred3=E2070031+E2070032+E2070033+E2070034; end;
if E2070031=>0 and E2070032=>0 and E2070033=>0 and E2070034=>0 and E2070035=>0 then do;
  cred3=E2070031+E2070032+E2070033+E2070034+E2070035; end;

/* 4th school */
if E2070041=>0 then do; cred4=E2070041; end;

/* 5th school */
if E2070051=>0 then do; cred5=E2070051; end;

/* Sum incoming credits with earned credits */
do over INCRED;
  if INCRED=>0 then do; TCRED=INCRED; end;
  if CRED=>0 then do; TCRED=CRED; end;
  if INCRED=>0 and CRED=>0 then do; TCRED=INCRED+CRED; end;
end;

/* E23000(1-3) accounts for incorrect tabulation of Round2 credits earned. */
array E23000 (i) E230001-E230003;
array OTCRED (i) OTCRED1-OTCRED3;
```

```
do i=1 to 3;
  OTCRED=.;
  if E23000>-4 then do; OTCRED=TCRED; TCRED=E23000; end;
end;

/* Here we add the Round1 created variable for fraction of credits with the Round2 count. They are separated by the
responent's answer to E27337(1-5), which is the type of degree question.*/
/* Initialixe both created variables.*/
do i=1 to 5;
  asfrac=0;
  bafrac=0;
end;

/* Associates Degree */
do i=1 to 5;
  if E27337=1 and TCRED=>0 then do; TASCRED=TCRED; end;
  if -4<E27388<0 and E27337=1 then do; asfrac=E27388; end;
  if TASCRED=>0 and E27388>0 then do; asfrac=(TASCRED/E27388)*100; end;
  asfrac=round(asfrac,1);
end;

/* Bachelors Degree */
do i=1 to 5;
  if E27337=3 and TCRED=>0 then do; TBACRED=TCRED; end;
  if -4<E27388<0 and E27337=3 then do; bafrac=E27388; end;
  if TBACRED=>0 and E27388>0 then do; bafrac=(TBACRED/E27388)*100; end;
  bafrac=round(bafrac,1);
end;

/* Non-interview cases */
if E16200=-5 then do;
  do i=1 to 5;
    bafrac=-5; asfrac=-5; end;
end;

endsas;
```

YOUTH'S MATH PIAT SCORE

Variables Created: CV_PIAT_STANDARD_SCORE
CV_PIAT_PERCENTILE_SCORE

Variables Used

| Name in Program | Question Name on CD |
|------------------|-------------------------------|
| CURD, CURM, CURY | SYMBOL!CURDATE~D, ~M, ~Y |
| DOB01D, M, Y | KEY!BDATE_D, _M, _Y (round 1) |
| PIATRAW | PIAT_RAW_SCORE |

This program calculates the respondent's age utilizing the interview date information and the respondent's date of birth information. It then uses the age data and the raw PIAT score to create a standard PIAT score and percentile rank for each respondent.

```
/**calculate the interview date (continuous months format) <initialize everyone to valid skip>**/

intday=-4;      intmonth=-4;      intyear=-4;

/**if there is a valid day, month, and year interview date in round 2 use it.*/
if (curm>=0) and (curd>=0) then do;
  intday=curd;
  intmonth=curm;
  intyear=cury;
end;

contmint=((intyear-1980)*12)+intmonth;

*****respondent age section*****<if there is a valid date of birth for respondent, use it.>**

dobday=dob01d;
dobmonth=dob01m;
dobyear=dob01y;

/**<ensure all respondents have an in range value.>**/
if (dobday<0) then dobday=15;
if (dobmonth<0) then dobmonth=6;
if (dobyear<1970) then dobyear=1983;

/**<calculate the youth's age in continuous months>/
yagemnth=((intyear-dobyear)*12)+(intmonth-dobmonth);
if (intday-dobday<0) then yagemnth=yagemnth-1;

*****piat standard score section*****<initialize everyone to valid skip>**/

piatstnd=-4;
```

/* At this point the program loops through each three-month age range and assigns appropriate standard scores based on the respondent's raw score. This code is quite lengthy and is not reproduced here; instead, we include a table that provides the standard score and percentile rank associated with each raw score in each age range. Users who need the precise programming code should contact NLS User Services. */

Appendix 1: Education Variable Creation

| Standard Score | Percentile Rank | Raw Score by Age Range (Years-Months) | | | | | | | | | | | | |
|----------------|-----------------|---------------------------------------|--------------|--------------|---------------|--------------|--------------|--------------|---------------|--------------|--------------|--------------|---------------|--|
| | | 13-0 to 13-2 | 13-3 to 13-5 | 13-6 to 13-8 | 13-9 to 13-11 | 14-0 to 14-2 | 14-3 to 14-5 | 14-6 to 14-8 | 14-9 to 14-11 | 15-0 to 15-2 | 15-3 to 15-5 | 15-6 to 15-8 | 15-9 to 15-11 | |
| 55 | 0 | ≤30 | ≤31 | ≤32 | ≤33 | ≤34 | ≤35 | ≤36 | ≤37 | ≤38 | ≤39 | ≤40 | ≤41 | |
| 56 | 0 | 31 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 | 41 | 42 | |
| 57 | 0 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 | 41 | 42 | | |
| 58 | 0 | 33 | 34 | 35 | 36 | 37 | 38 | | 40 | 41 | 42 | 43 | 43 | |
| 59 | 0 | 34 | 35 | 36 | 37 | | | 39 | | | 43 | | 44 | |
| 60 | 0 | 35 | 36 | | | 38 | 39 | 40 | 41 | 42 | | 44 | 45 | |
| 61 | 0 | 36 | 37 | 37 | 38 | 39 | 40 | 41 | 42 | 43 | 44 | 45 | 46 | |
| 62 | 1 | 37 | | 38 | 39 | 40 | 41 | 42 | 43 | 44 | 45 | 46 | 47 | |
| 63 | 1 | | 38 | 39 | 40 | 41 | 42 | 43 | 44 | 45 | 46 | 47 | 48 | |
| 64 | 1 | 38 | 39 | 40 | 41 | 42 | 43 | 44 | 45 | 46 | 47 | 48 | 49 | |
| 65 | 1 | 39 | 40 | 41 | 42 | 43 | 44 | 45 | 46 | 47 | 48 | 49 | 50 | |
| 66 | 1 | 40 | 41 | 42 | 43 | 44 | 45 | 46 | 47 | 48 | 49 | 50 | 51 | |
| 67 | 1 | 41 | 42 | 43 | 44 | 45 | 46 | 47 | 48 | 49 | 50 | 51 | 52 | |
| 68 | 2 | 42 | 43 | 44 | 45 | 46 | 47 | 48 | 49 | 50 | 51 | 52 | 53 | |
| 69 | 2 | 43 | 44 | 45 | 46 | 47 | 48 | 49 | 50 | 51 | 52 | 53 | 54 | |
| 70 | 2 | 44 | 45 | 46 | 47 | 48 | 49 | 50 | 51 | 52 | 53 | 54 | 55 | |
| 71 | 3 | 45 | 46 | 47 | 48 | 49 | 50 | 51 | 52 | 53 | 54 | 55 | 56 | |
| 72 | 3 | 46 | 47 | 48 | 49 | 50 | 51 | 52 | 53 | 54 | 55 | 56 | 57 | |
| 73 | 4 | 47 | 48 | 49 | 50 | 51 | 52 | 53 | 54 | 55 | 56 | 57 | | |
| 74 | 4 | 48 | 49 | 50 | 51 | 52 | 53 | 54 | 55 | 56 | 57 | 58 | 58 | |
| 75 | 5 | 49 | 50 | 51 | 52 | 53 | 54 | 55 | 56 | 57 | 58 | | 59 | |
| 76 | 5 | 50 | 51 | 52 | 53 | 54 | 55 | 56 | 57 | 58 | 59 | 59 | 60 | |
| 77 | 6 | | 52 | | 54 | | 56 | | | | | 60 | 61 | |
| 78 | 7 | 51 | 53 | 53 | 55 | 55 | 57 | 57 | 58 | 59 | 60 | 61 | 62 | |
| 79 | 8 | 52 | | 54 | | 56 | | 58 | 59 | 60 | 61 | 62 | 63 | |
| 80 | 9 | 53 | 54 | 55 | 56 | 57 | 58 | 59 | 60 | 61 | 62 | 63 | 64 | |
| 81 | 10 | 54 | 55 | 56 | 57 | 58 | 59 | 60 | 61 | 62 | 63 | 64 | 65 | |
| 82 | 12 | 55 | 56 | 57 | 58 | 59 | 60 | 61 | 62 | 63 | 64 | 65 | 66 | |
| 83 | 13 | 56 | 57 | 58 | 59 | 60 | 61 | 62 | 63 | 64 | 65 | 66 | | |
| 84 | 14 | 57 | 58 | 59 | 60 | 61 | 62 | 63 | 64 | 65 | 66 | 67 | 67 | |
| 85 | 16 | 58 | 59 | 60 | 61 | 62 | 63 | 64 | 65 | 66 | 67 | | 68 | |
| 86 | 18 | 59 | 60 | 61 | 62 | 63 | 64 | 65 | 66 | 67 | | 68 | 69 | |
| 87 | 19 | | | | | | 65 | | | | | 68 | 69 | |
| 88 | 21 | 60 | 61 | 62 | 63 | 64 | | 66 | 67 | 68 | 69 | 70 | 71 | |
| 89 | 23 | 61 | 62 | 63 | 64 | 65 | 66 | 67 | 68 | 69 | 70 | 71 | 72 | |
| 90 | 25 | 62 | 63 | 64 | 65 | 66 | 67 | 68 | 69 | 70 | 71 | 72 | | |
| 91 | 27 | | | 65 | 66 | 67 | 68 | 69 | 70 | | | | 73 | |
| 92 | 30 | 63 | 64 | | | | 69 | | | 71 | 72 | 73 | 74 | |
| 93 | 32 | 64 | 65 | 66 | 67 | 68 | | 70 | 71 | 72 | 73 | 74 | 75 | |
| 94 | 34 | 65 | 66 | 67 | 68 | 69 | 70 | 71 | 72 | 73 | 74 | 75 | | |
| 95 | 37 | | | 68 | 69 | 70 | 71 | 72 | 73 | 74 | 75 | | 76 | |
| 96 | 39 | 66 | 67 | | | | | | | | | 76 | 77 | |
| 97 | 42 | 67 | 68 | 69 | 70 | 71 | 72 | 73 | 74 | 75 | 76 | 77 | | |
| 98 | 45 | | 69 | 70 | 71 | 72 | 73 | 74 | 75 | 76 | 77 | | 78 | |
| 99 | 47 | 68 | | | | | | | | | | 78 | 79 | |
| 100 | 50 | 69 | 70 | 71 | 72 | 73 | 74 | 75 | 76 | 77 | 78 | 79 | 80 | |

Appendix 1: Education Variable Creation

| Standard Score | Percentile Rank | Raw Score by Age Range (Years-Months) | | | | | | | | | | | |
|----------------|-----------------|---------------------------------------|--------------|--------------|---------------|--------------|--------------|--------------|---------------|--------------|--------------|--------------|---------------|
| | | 13-0 to 13-2 | 13-3 to 13-5 | 13-6 to 13-8 | 13-9 to 13-11 | 14-0 to 14-2 | 14-3 to 14-5 | 14-6 to 14-8 | 14-9 to 14-11 | 15-0 to 15-2 | 15-3 to 15-5 | 15-6 to 15-8 | 15-9 to 15-11 |
| 101 | 53 | | 71 | 72 | 73 | 74 | 75 | 76 | 77 | 78 | 79 | 80 | |
| 102 | 55 | 70 | 72 | 73 | 74 | 75 | 76 | 77 | 78 | 79 | 80 | | 81 |
| 103 | 58 | 71 | | | 75 | | | 78 | | | | 81 | 82 |
| 104 | 61 | 72 | 73 | 74 | | 76 | 77 | | 79 | 80 | 81 | 82 | 83 |
| 105 | 63 | 73 | 74 | 75 | 76 | 77 | 78 | 79 | 80 | 81 | 82 | 83 | 84 |
| 106 | 66 | 74 | 75 | 76 | 77 | 78 | 79 | 80 | 81 | 82 | 83 | 84 | |
| 107 | 68 | | 76 | 77 | 78 | 79 | 80 | 81 | 82 | 83 | 84 | | 85 |
| 108 | 70 | 75 | 77 | 78 | 79 | 80 | 81 | 82 | 83 | 84 | 85 | 85 | 86 |
| 109 | 73 | 76 | | | | 81 | 82 | 83 | | | | 86 | 87 |
| 110 | 75 | 77 | 78 | 79 | 80 | 82 | 83 | 84 | 84 | 85 | 86 | 87 | 88 |
| 111 | 77 | 78 | 79 | 80 | 81 | | | | 85 | 86 | 87 | 88 | 89 |
| 112 | 79 | 79 | 80 | 81 | 82 | 83 | 84 | 85 | 86 | 87 | 88 | 89 | 90 |
| 113 | 81 | 80 | 81 | 82 | 83 | 84 | 85 | 86 | 87 | 88 | 89 | 90 | 91 |
| 114 | 82 | 81 | 82 | 83 | 84 | 85 | 86 | 87 | 88 | 89 | 90 | 91 | 92 |
| 115 | 84 | 82 | 83 | 84 | 85 | 86 | 87 | 88 | 89 | 90 | 91 | 92 | 93 |
| 116 | 86 | 83 | 84 | 85 | 86 | 87 | 88 | 89 | 90 | 91 | 92 | 93 | 94 |
| 117 | 87 | 84 | 85 | 86 | 87 | 88 | 89 | 90 | 91 | 92 | 93 | 94 | |
| 118 | 88 | | 86 | 87 | 88 | 89 | 90 | 91 | 92 | 93 | 94 | 95 | 95 |
| 119 | 90 | 85 | 87 | 88 | 89 | 90 | 91 | 92 | 93 | 94 | 95 | 96 | 96 |
| 120 | 91 | 86 | 88 | 89 | 90 | 91 | 92 | 93 | 94 | 95 | 96 | | 97 |
| 121 | 92 | 87 | | 90 | 91 | 92 | 93 | 94 | 95 | 96 | | 97 | 98 |
| 122 | 93 | 88 | 89 | 91 | 92 | 93 | 94 | 95 | 96 | 97 | 97 | 98 | 99 |
| 123 | 94 | 89 | 90 | | | | | | | | | 98 | 99 |
| 124 | 95 | 90 | 91 | 92 | 93 | 94 | 95 | 96 | 97 | 98 | 99 | 100 | |
| 125 | 95 | | 92 | 93 | 94 | 95 | 96 | 97 | 98 | 99 | 100 | | |
| 126 | 96 | 91 | 93 | 94 | 95 | 96 | 97 | 98 | 99 | 100 | | | |
| 127 | 96 | 92 | | 95 | 96 | 97 | 98 | 99 | 100 | | | | |
| 128 | 97 | 93 | 94 | 96 | 97 | 98 | 99 | 100 | | | | | |
| 129 | 97 | 94 | 95 | | | | | | | | | | |
| 130 | 98 | 95 | 96 | 97 | 98 | 99 | 100 | | | | | | |
| 131 | 98 | | 97 | 98 | 99 | 100 | | | | | | | |
| 132 | 98 | 96 | 98 | 99 | 100 | | | | | | | | |
| 133 | 99 | 97 | | 100 | | | | | | | | | |
| 134 | 99 | 98 | 99 | | | | | | | | | | |
| 135 | 99 | 99 | 100 | | | | | | | | | | |
| 136 | 99 | 100 | | | | | | | | | | | |

Note: This table of *PIAT* scores is based on the information provided by the testing company. Users interested in more information about the *PIAT-Math Assessment* may wish to consult the following reference:

Markwardt, Frederick C. Jr. *Peabody Individual Achievement Test-Revised*. Circle Pines, Minn.: American Guidance Service, Inc., 1989.

QED DATA: SCHOOL SIZE AND STUDENT-TEACHER RATIO

Variables Created: CV_SCHOOL_SIZE
CV_STUDENT_TEACHER_RATIO

Variables Used

| Name in Program | Question Name on CD |
|-----------------|---------------------|
| GENDER | KEY!SEX |
| RACE | KEY!RACE |
| E2857 | YSCH-2857 |
| E21625 | YSCH-21625 |
| AGE | SYMBOL!KEY!AGE |
| REGION | CV_CENSUS_REGION |
| CV_HGC | CV_HGC_EVER |

Codes for Created Variable

| School Size | Student-teacher ratio |
|-------------|-----------------------|
| 1 = <100 | 1 = <14 |
| 2 = 100-299 | 2 = 14-17 |
| 3 = 300-499 | 3 = 18-21 |
| 4 = 500-749 | 4 = 22+ |
| 5 = 750-999 | |
| 6 = 1000+ | |

This program merges school identification information from the NLSY97 data with data from the “National Education Database,” provided under copyright by Quality Education Data (QED), Inc. It then creates two variables that provide information about the respondent’s school.

/* Before creating the QED variables, survey staff used dates of enrollment data to identify the respondent’s current or most recent school and then merged that information with the QED data. This information is excluded here due to length and confidentiality restrictions. Researchers needing more information about this process should contact NLS User Services. */

```

/* Initialize variables */
SCHSIZE=-4;
STUDTEAC=-4;

/* create SCHSIZE and STUDTEAC while checking
that the grade reported in the QED coincides with that
being reported in the NLSY97 */

if GRADSPAN='2' then GRADSPAN='3';
if GRADSPAN='A' then GRADSPAN='2';
if GRADSPAN='B' or GRADSPAN='C' or
   GRADSPAN='D' or GRADSPAN='E' or
   GRADSPAN='*' then GRADSPAN='0';

/* Redefine each GRADSPAN to GRADSPN to
eliminate character value problems */
if GRADSPAN='0' then GRADSPN=0;
if GRADSPAN='1' then GRADSPN=1;
if GRADSPAN='2' then GRADSPN=2;
if GRADSPAN='3' then GRADSPN=3;
if GRADSPAN='4' then GRADSPN=4;
if GRADSPAN='5' then GRADSPN=5;
if GRADSPAN='6' then GRADSPN=6;

if GRADSPN='7' then GRADSPN=7;
if GRADSPN='8' then GRADSPN=8;
if GRADSPN='9' then GRADSPN=9;

/* Initialize the dummies to zero. */
dummy1=0;      dummy4=0;
dummy5=0;      dummy6=0;
dummy7=0;      dummy8=0;
dummy9=0;      dummy10=0;

if E2857 ge 1 and E2857 le 3 then do;
   dummy1=1;      dummy4=1;
   dummy5=1;      dummy6=0;
   dummy7=0;      dummy8=0;
   dummy9=0;      dummy10=0;
end;

if E2857 ge 4 and E2857 le 6 then do;
   dummy1=1;      dummy4=1;
   dummy5=1;      dummy6=1;
   dummy7=0;      dummy8=0;
   dummy9=0;      dummy10=0;
end;

if E2857 ge 7 and E2857 le 8 then do;

```

```

dummy1=1;    dummy4=0;
dummy5=1;    dummy6=1;
dummy7=1;    dummy8=1;
dummy9=0;    dummy10=0;
end;
if E2857 eq 9 then do;
  dummy1=1;    dummy4=0;
  dummy5=0;    dummy6=0;
  dummy7=1;    dummy8=1;
  dummy9=1;    dummy10=0;
end;
if E2857 ge 10 and E2857 le 12 then do;
  dummy1=1;    dummy4=0;
  dummy5=0;    dummy6=0;
  dummy7=0;    dummy8=1;
  dummy9=1;    dummy10=1;
end;

correct=0;
if GRADSPN=1 and dummy1=1 then correct=1;
if GRADSPN=4 and dummy4=1 then correct=1;
if GRADSPN=5 and dummy5=1 then correct=1;
if GRADSPN=6 and dummy6=1 then correct=1;
if GRADSPN=7 and dummy7=1 then correct=1;
if GRADSPN=8 and dummy8=1 then correct=1;
if GRADSPN=9 and dummy9=1 then correct=1;
if GRADSPN=2 and dummy10=1 then correct=1;

/* If the grades reported in the QED and the NLSY97
do not coincide, we make both created variables equal
to -3; otherwise, we proceed to create them according
to the definition provided at the top of the program */

/* Need to check parameters for STUDRANG */
if STUDRANG=0 or STUDRANG=1 then
  SCHSIZE=1;
if STUDRANG=2 then SCHSIZE=2;
if STUDRANG=3 then SCHSIZE=3;
if STUDRANG=4 then SCHSIZE=4;
if STUDRANG=5 then SCHSIZE=5;
if STUDRANG=6 or STUDRANG=7 or
  STUDRANG=8 or STUDRANG=9 then
  SCHSIZE=6;
if STUDRANG=. then SCHSIZE=-3;

```

```

if STUDENT1 gt 0 and TEACHERS gt 0 then
  STUDTEA1=STUDENT1/TEACHERS;
if STUDTEA1 lt 14 then STUDTEAC=1;
if STUDTEA1 ge 14 and STUDTEA1 lt 18 then
  STUDTEAC=2;
if STUDTEA1 ge 18 and STUDTEA1 lt 22 then
  STUDTEAC=3;
if STUDTEA1 ge 22 then STUDTEAC=4;
if STUDTEA1 eq . then STUDTEAC=-3;

if correct=0 then SCHSIZE=-3;
if correct=0 then STUDTEAC=-3;

/* We also make sure that there are at least six schools
in each cell determined by the region, type of school,
control, school size and student-teacher ratio to avoid
any possibilities of identifying the schools. We first run
the tabulations, then assign -3 to the created variables
for schools that fall in cells of five or less schools. */

INDEX=REGION||TYPESCH||E21625||SCHSIZE||
  STUDTEAC;
proc freq;
  tables REGION*TYPESCH*E21625*SCHSIZE*
    STUDTEAC / OUT=A;
  proc freq data=A noprint;
    tables COUNT;
  data A;
    set A;
    set A;

INDEX=REGION||TYPESCH||E21625||SCHSIZE||
  STUDTEAC;
proc freq;
  tables INDEX / OUT=INDEXFRQ;
  proc freq data=INDEXFRQ noprint;
    tables COUNT;

if COUNT le 5 then SCHSIZE=-3;
if COUNT le 5 then STUDTEAC=-3;
if pin=-5 then do;
  studteac=-5; schsize=-5;
end;
endsas;

```

NLSY97 Appendix 2:

Employment Variable Creation

INTRODUCTION

A number of the created employment variables use the same program(s) as input. The two programs in this section are referred to throughout the employment variables. Thus, for example, to create the “Weeks Worked at Employee Job #x during 19xx” variables, survey staff first run the program below titled “emp_begin.sas” and then run the program included in the weeks worked section of this appendix.

EMP_BEGIN.SAS

This program calculates total weeks worked at each job for each respondent. It converts start and stop dates for jobs and within-job gaps to continuous week numbers, subtracts within-job gaps, and finally counts the total weeks worked.

| Name in Program | Question Name on CD | Name in Program | Question Name on CD |
|---------------------|------------------------------|---------------------------|------------------------------|
| dli_d, dli_m, dli_y | YINF-900_D, _M, _Y | BGDY3_2-BGDY3_4 | YEMP-102400.03.02~D-.03.04~D |
| int_d, int_m, int_y | YINTDATE~D, ~M, ~Y | BGMO3_2-BGMO3_4 | YEMP-102400.03.02~M-.03.04~M |
| PUBID | PUBID | BGYR3_2-BGYR3_4 | YEMP-102400.03.02~Y-.03.04~Y |
| stardy1-stardy9 | YEMP_STARTDATE.01~D-.09~D | BGDY4_2, BGMO4_2, BGYR4_2 | YEMP-102400.04.02~D, ~M, ~Y |
| starmo1-starmo9 | YEMP_STARTDATE.01~M-.09~M | BGDY5_2, BGMO5_2, BGYR5_2 | YEMP-102400.05.02~D, ~M, ~Y |
| staryr1-staryr9 | YEMP_STARTDATE.01~Y-.09~Y | EGDY1_1-EGDY1_9 | YEMP-102700.01.01~D-.01.09~D |
| stopdy1-stopdy9 | YEMP_STOPDATE.01~D-.09~D | EGMO1_1-EGMO1_9 | YEMP-102700.01.01~M-.01.09~M |
| stopmo1-stopmo9 | YEMP_STOPDATE.01~M-.09~M | EGYR1_1-EGYR1_9 | YEMP-102700.01.01~Y-.01.09~Y |
| stopyr1-stopyr9 | YEMP_STOPDATE.01~Y-.09~Y | EGDY2_1-EGDY2_5 | YEMP-102700.02.01~D-.02.05~D |
| UID1-UID9 | YEMP_UID.01~09 | EGMO2_1-EGMO2_5 | YEMP-102700.02.01~M-.02.05~M |
| BGDY1_1-BGDY6_1 | YEMP-102200.01.01~D-.06.01~D | EGYR2_1-EGYR2_5 | YEMP-102700.02.01~Y-.02.05~Y |
| BGMO1_1-BGMO6_1 | YEMP-102200.01.01~M-.06.01~M | EGDY3_1-EGDY3_4 | YEMP-102700.03.01~D-.03.04~D |
| BGYR1_1-BGYR6_1 | YEMP-102200.01.01~Y-.06.01~Y | EGMO3_1-EGMO3_4 | YEMP-102700.03.01~M-.03.04~M |
| BGDY1_2-BGDY1_9 | YEMP-102400.01.02~D-.01.09~D | EGYR3_1-EGYR3_4 | YEMP-102700.03.01~Y-.03.04~Y |
| BGMO1_2-BGMO1_9 | YEMP-102400.01.02~M-.01.09~M | EGDY4_1, EGMO4_1, EGYR4_1 | YEMP-102700.04.01~D, ~M, ~Y |
| BGYR1_2-BGYR1_9 | YEMP-102400.01.02~Y-.01.09~Y | EGDY4_2, EGMO4_2, EGYR4_2 | YEMP-102700.04.02~D, ~M, ~Y |
| BGDY2_2-BGDY2_5 | YEMP-102400.02.02~D-.02.05~D | EGDY5_1, EGMO5_1, EGYR5_1 | YEMP-102700.05.01~D, ~M, ~Y |
| BGMO2_2-BGMO2_5 | YEMP-102400.02.02~M-.02.05~M | EGDY5_2, EGMO5_2, EGYR5_2 | YEMP-102700.05.02~D, ~M, ~Y |
| BGYR2_2-BGYR2_5 | YEMP-102400.02.02~Y-.02.05~Y | EGDY6_1, EGMO6_1, EGYR6_1 | YEMP-102700.06.01~D, ~M, ~Y |

***** SECTION 1: The program first converts interview dates into continuous week numbers.*****

```

/* The following are hand edit cases for round 2 int date */
if PUBID=1818 then do; INT_D=2; INT_M=1; INT_Y=1999; end;
if PUBID=5294 then do; INT_D=12; INT_M=12; INT_Y=1998; end;
/* The following are hand edit cases for round 1 int date */
if (pubid=2 or pubid=5615 or pubid=5902) then do; dli_y=1997; end;

/* Convert Interview Date to week number */
/*Convert interview month and day to total days (intdays)*/
if INT_M>0 and INT_D>0 then do;
    if INT_M=1 then intdays=INT_D;
    if INT_M=3 then intdays=INT_D+59;
    if INT_M=5 then intdays=INT_D+120;
    if INT_M=7 then intdays=INT_D+181;
    if INT_M=9 then intdays=INT_D+243;
    if INT_M=11 then intdays=INT_D+304;
end;

/* Convert days into week numbers*/
/* Basic Formula: weekno=endweek{specific year}+ceil[(totdays+{# of days remaining in DEC})/7] */

```

```

/* Default interview week = 9999 */
intwk=9999;
if INT_Y>0 and intdays>0 then do;
  if INT_Y=1998 then do; INTWK=939+ceil((intdays+4)/7); end;
  end;
if INT_Y>0 and intdays>0 then do;
  if INT_Y=1999 then do; INTWK=991+ceil((intdays+5)/7); end;
  end;

/* Convert Date of Last Interview to week number */
/* Convert interview month and day to total days (dlidays) */
if DLI_M>0 and DLI_D>0 then do;
  if DLI_M=1 then dlidays=DLI_D;           if DLI_M=2 then dlidays=DLI_D+31;
  if DLI_M=3 then dlidays=DLI_D+59;         if DLI_M=4 then dlidays=DLI_D+90;
  if DLI_M=5 then dlidays=DLI_D+120;        if DLI_M=6 then dlidays=DLI_D+151;
  if DLI_M=7 then dlidays=DLI_D+181;        if DLI_M=8 then dlidays=DLI_D+212;
  if DLI_M=9 then dlidays=DLI_D+243;        if DLI_M=10 then dlidays=DLI_D+273;
  if DLI_M=11 then dlidays=DLI_D+304;       if DLI_M=12 then dlidays=DLI_D+334;
  end;

/*Convert days into week numbers*/
/* Basic Formula: weekno=endweek{specific year}+ceil[(totdays+[# of days remaining in DEC])/7] */

/* Default interview week = 9999 */
DLIWK=9999;
if DLI_Y>0 and dlidays>0 then do;
  if DLI_Y=1997 then do; DLIWK=887+ceil((dlidays+3)/7); end;
  end;
if DLI_Y>0 and dlidays>0 then do;
  if DLI_Y=1998 then do; DLIWK=939+ceil((dlidays+4)/7); end;
  end;

/* Hand edited cases */
if PUBID=471 or PUBID=476 then INTWK=990;
if PUBID=7315 then DLIWK=970;

/* Done to create a seamless flow between rounds */
intwk=intwk-1;

/* SECTION 2: This part converts employment start and stop dates into continuous week numbers.*/

/*Invalid start and stop data lead to imputed values that are marked by dummy variables for invalid start
dates and invalid end dates. The dummy variables uflag1-uflag9 denotes updating of start dates of jobs
worked in both rounds to the Round1 interview date. This is done to create a separate created variable for
Round1 and Round2 which will be added in the end. */

/* Hand edit for respondent with UID8=0. By inspection this is assumed to be mistaken for 9801. */
if pubid=1665 then UID8=9801;

/* ostarm represents the "old" start dates, used when start dates are updated to interview dates. */
array ostarm (i) ostarmo1-ostarmo9;      array ostardt (i) ostardy1-ostardy9;
array ostarty (i) ostaryr1-ostaryr9;      array ostopm (i) ostopmo1-ostopmo9;

```

```

array ostoppd (i) ostoppdy1-ostoppdy9;
array startm (i) starmo1-starmo9;
array starty (i) staryr1-staryr9;
array stopd (i) stopdy1-stopdy9;
array UID (i) UID1-UID9;

array ostopy (i) ostopyr1-ostopyr9;
array startd (i) stardy1-stardy9;
array stopm (i) stopmo1-stopmo9;
array stopy (i) stopyr1-stopyr9;

array sttdays (i) sttday1-sttday9; /* total days in that year from startdate (to Jan 1) */
array stpdays (i) stpday1-stpday9; /* total days in that year from stopdate (to Jan 1) */
array startwk (i) starw1-starw9;
array stopwk (i) stopw1-stopw9;
array srflag (i) srflg1-srflg10;
array spflag (i) spflg1-spflg10;
array uflag (i) uflag1-uflag9; /* uflag=1 when job startdate is updated */
array smofl (i) smofl1-smofl9; /* dummy equals 1 when a start month is imputed */
array emofl (i) emofl1-emofl9; /* dummy equals 1 when a stop month is imputed */

/* Initialize smofl and emofl */
do i=1 to 9; smofl=0; emofl=0; end;

/* Two hand edit cases for respondents who reported jobs in Round1 that lasted up to Round1 interview date. In Round2, respondents indicated that these jobs ended before Round1 interview date, and therefore should not be on the Round2 job roster. Therefore, these jobs are erased from the Round2 roster. */
if pubid=2461 then do;
    stardy2=-4; starmo2=-4; staryr2=-4; uid2=-4; stopdy2=-4; stopmo2=-4; stopyr2=-4; end;
if pubid=5276 then do;
    stardy1=-4; starmo1=-4; staryr1=-4; uid1=-4; stopdy1=-4; stopmo1=-4; stopyr1=-4; end;

/* Define old start and stop dates */
do i=1 to 9;
    ostartm=startm;          ostardt=startd;
    ostarty=starty;           ostopm=stopm;
    ostoppd=stopd;           ostopy=stopy;
end;

/* Fill-in start/stop day for those missing*/
/* flag1 = impute start day (valid month)          flag2 = impute start month (valid day)
   flag3 = impute start day and month            flag4 = impute stop day (valid month)
   flag5 = impute stop month (valid day)           flag6 = impute stop day and month      */
flag1=0;         flag2=0;         flag3=0;
do over starty;
    if starty>0 then do;
        if startm>0 and startd<=0 then do; startd=15; flag1=1; srflag=1; end;
        if startm<=0 and startd>0 then do; startm=6; flag2=1; srflag=1; smofl=1; end;
        if startm<=0 and startd<=0 then do; startm=6; startd=15; flag3=1; srflag=1; smofl=1; end;
        /* keep imputed values for start months and days from preceding the Round1 interview date. */
        if starty=dli_y and startm=dli_m and startd<dli_d and srflag=1 then do; startd=dli_d; end;
        if starty=dli_y and startm<dli_m and srflag=1 then do; startm=dli_m; startd=dli_d; end;
    end;
end;

flag4=0;         flag5=0;         flag6=0;

```

```

do over stopy;
if stopy>0 then do;
  if stopm>0 and stopd<=0 then do; stopd=15; flag4=1; spflag=1; end;
  if stopm<=0 and stopd>0 then do; stopm=6; flag5=1; spflag=1; emofl=1; end;
  if stopm<=0 and stopd<=0 then do; stopm=6; stopd=15; flag6=1; spflag=1; emofl=1; end;
/* keep imputed values for stop months and days from exceeding the Round2 interview date. */
  if stopy=int_y and stopm=int_m and stopd>int_d and spflag=1 then do; stopd=int_d; end;
  if stopy=int_y and stopm>int_m then do; stopm=int_m; stopd=int_d; end;
end;
end;

```

/ The following lines of code account for the cases where the respondent has a start date for a job reported in Round 1 that comes before dli in the Round 2 roster. These cases will have the original startdate reported on the Round 2 roster that is used to count the number of weeks employed. The idea is to only count the weeks employed from Round1 interview date to today. Then, the activity from Round 1 will be added to the Round 2 activity to get the full event history. To achieve this, all jobs reported in Round 1 (UID's begin with 97) will have their startdates updated to the dli. The start week and stop week for these jobs will be counted the same. */*

```

do over uflag; uflag=0; end; /* Initialize uflag */
do i=1 to 9;
  if 9700<UID<9800 then do;
    if starty=dli_y and startm=dli_m and startd<dli_d then do; startd=dli_d; uflag=1; end;
    if starty=dli_y and startm<dli_m then do; startm=dli_m; startd=dli_d; uflag=1; end;
    if starty<dli_y then do; starty=dli_y; startm=dli_m; startd=dli_d; uflag=1; end;
  end;
end;

```

*/*Convert START month and day to total days*/*

```

do over startm;
  if startm>0 and startd>0 then do;
    if startm=1 then sttdays=startd;
    if startm=3 then sttdays=startd+59;
    if startm=5 then sttdays=startd+120;
    if startm=7 then sttdays=startd+181;
    if startm=9 then sttdays=startd+243;
    if startm=11 then sttdays=startd+304;
  end;
end;

```

```

  if startm=2 then sttdays=startd+31;
  if startm=4 then sttdays=startd+90;
  if startm=6 then sttdays=startd+151;
  if startm=8 then sttdays=startd+212;
  if startm=10 then sttdays=startd+273;
  if startm=12 then sttdays=startd+334;

```

*/*Account for leap years*/*

```

do over starty;
  if starty=1980 or starty=1984 or starty=1988 or starty=1992 or starty=1996 then do;
    if startm>0 and startd>0 then do;
      if startm=1 then sttdays=startd;
      if startm=3 then sttdays=startd+60;
      if startm=5 then sttdays=startd+121;
      if startm=7 then sttdays=startd+182;
      if startm=9 then sttdays=startd+244;
      if startm=11 then sttdays=startd+305;
    end;
  end;

```

```

      if startm=2 then sttdays=startd+31;
      if startm=4 then sttdays=startd+91;
      if startm=6 then sttdays=startd+152;
      if startm=8 then sttdays=startd+213;
      if startm=10 then sttdays=startd+274;
      if startm=12 then sttdays=startd+335;

```

```

end;

/*Convert STOP month and day to total days*/
do over stopm;
  if stopm>0 and stopd>0 then do;
    if stopm=1 then stpdays=stopd;
    if stopm=3 then stpdays=stopd+59;
    if stopm=5 then stpdays=stopd+120;
    if stopm=7 then stpdays=stopd+181;
    if stopm=9 then stpdays=stopd+243;
    if stopm=11 then stpdays=stopd+304;
    end;
  end;

/*Account for leap years*/
do over stopy;
  if stopy=1980 or stopy=1984 or stopy=1988 or stopy=1992 or stopy=1996 then do;
    if stopm>0 and stopd>0 then do;
      if stopm=1 then stpdays=stopd;
      if stopm=3 then stpdays=stopd+60;
      if stopm=5 then stpdays=stopd+121;
      if stopm=7 then stpdays=stopd+182;
      if stopm=9 then stpdays=stopd+244;
      if stopm=11 then stpdays=stopd+305;
      end;
    end;
  end;

/*Convert days into week numbers*/
/** Basic Formula: weekno=endweek{specific year}+ceil[(totdays+{# of days remaining in DEC})/7] ***/
do over starty;
  if starty>0 and sttdays>0 then do;
    if starty=1980 then do; startwk=ceil((sttdays+2)/7); end;
    if starty=1981 then do; startwk=52+ceil((sttdays+4)/7); end;
    if starty=1982 then do; startwk=104+ceil((sttdays+5)/7); end;
    if starty=1983 then do; startwk=156+ceil((sttdays+6)/7); end;
    if starty=1984 then do; startwk=209+ceil((sttdays)/7); end;
    if starty=1985 then do; startwk=261+ceil((sttdays+2)/7); end;
    if starty=1986 then do; startwk=313+ceil((sttdays+3)/7); end;
    if starty=1987 then do; startwk=365+ceil((sttdays+4)/7); end;
    if starty=1988 then do; startwk=417+ceil((sttdays+5)/7); end;
    if starty=1989 then do; startwk=470+ceil((sttdays)/7); end;
    if starty=1990 then do; startwk=522+ceil((sttdays+1)/7); end;
    if starty=1991 then do; startwk=574+ceil((sttdays+2)/7); end;
    if starty=1992 then do; startwk=626+ceil((sttdays+3)/7); end;
    if starty=1993 then do; startwk=678+ceil((sttdays+5)/7); end;
    if starty=1994 then do; startwk=730+ceil((sttdays+6)/7); end;
    if starty=1995 then do; startwk=783+ceil((sttdays)/7); end;
    if starty=1996 then do; startwk=835+ceil((sttdays+1)/7); end;
    if starty=1997 then do; startwk=887+ceil((sttdays+3)/7); end;
    if starty=1998 then do; startwk=939+ceil((sttdays+4)/7); end;
    if starty=1999 then do; startwk=991+ceil((sttdays+5)/7); end;

```

```

end;
if starty<0 and starty>-4 then do; startwk=-3; end;
end;

do over stopy;
  if stopy>0 and stpdays>0 then do;
    if stopy=1980 then do; stopwk=ceil((stpdays+2)/7); end;
    if stopy=1981 then do; stopwk=52+ceil((stpdays+4)/7); end;
    if stopy=1982 then do; stopwk=104+ceil((stpdays+5)/7); end;
    if stopy=1983 then do; stopwk=156+ceil((stpdays+6)/7); end;
    if stopy=1984 then do; stopwk=209+ceil((stpdays)/7); end;
    if stopy=1985 then do; stopwk=261+ceil((stpdays+2)/7); end;
    if stopy=1986 then do; stopwk=313+ceil((stpdays+3)/7); end;
    if stopy=1987 then do; stopwk=365+ceil((stpdays+4)/7); end;
    if stopy=1988 then do; stopwk=417+ceil((stpdays+5)/7); end;
    if stopy=1989 then do; stopwk=470+ceil((stpdays)/7); end;
    if stopy=1990 then do; stopwk=522+ceil((stpdays+1)/7); end;
    if stopy=1991 then do; stopwk=574+ceil((stpdays+2)/7); end;
    if stopy=1992 then do; stopwk=626+ceil((stpdays+3)/7); end;
    if stopy=1993 then do; stopwk=678+ceil((stpdays+5)/7); end;
    if stopy=1994 then do; stopwk=730+ceil((stpdays+6)/7); end;
    if stopy=1995 then do; stopwk=783+ceil((stpdays)/7); end;
    if stopy=1996 then do; stopwk=835+ceil((stpdays+1)/7); end;
    if stopy=1997 then do; stopwk=887+ceil((stpdays+3)/7); end;
    if stopy=1998 then do; stopwk=939+ceil((stpdays+4)/7); end;
    if stopy=1999 then do; stopwk=991+ceil((stpdays+5)/7); end;
  end;
  if stopy<0 and stopy>-4 then do; stopwk=-3; end;
end;

```

/* The following code decreases the Round 2 interview date by one so that, in Round 3, the Round 2 int. week will not be counted twice for respondents with a job during that time. The actual Round 2 int. week will counted as part of the job tenure in Round 3. The same procedure was used in Round 1. */

```

do over stopwk;
  if stopwk>0 and UID>0 then do; if stopwk>intwk then do; stopwk=intwk; end; end;
end;

```

/* The following lines considers jobs that begin the same week as the Round2 interview date. Since we are updating the Round2 interview week by -1, we need to account for jobs that start in the same week or tenures of -1 will result. */

```

do over startwk;
  if startwk>0 and UID>0 then do; if startwk>intwk then do; startwk=intwk; end; end;
end;

```

/* Two hand edit cases result from stop dates being greater than Round2 interview dates. Stop dates updated back to interview dates for each respondent. These have been subtracted by one. */

```

if pubid=2019 then stopw1=988;
if pubid=8995 then stopw1=984;

```

/* To account for respondents interviewed in Round1 but not Round2 */

```

if starmo1=-5 then do;

```

```

do over startwk; startwk=-5; stopwk=-5; end; end;

/* Correcting for imputed values that resulted in the start date being later than the stop date. In these
cases, the imputed date will be updated to the good date. */
do i=1 to 9;
  if startwk>stopwk and (flag1=1 or flag2=1 or flag3=1) then do; startwk=stopwk; end;
  if startwk>stopwk and (flag4=1 or flag5=1 or flag6=1) then do; stopwk=startwk; end;
end;

/* SECTION 3: This part converts within-job gap start & stop dates into continuous week numbers.*/

/* Reads in raw data on within-job gaps and converts the gap dates into NLSY97 week numbers. The flag
variables equal one for invalid gap data and zero otherwise. */

/* JOB 1 GAPS */
/* These variables are read as follows:    BGDY1_1 = Begin day of within-job gap 1 on job 1
                                         EGMO1_6 = End month of within-job gap 6 on job 1
                                         BGAP1_3 = Begin week of within-job gap 3 on job 1 [CREATED] */
array bgdy BGDY1_1-BGDY1_9;           array bgmo BGMO1_1-BGMO1_9;
array bgyr BGYR1_1-BGYR1_9;           array egdy EGDY1_1-EGDY1_9;
array egmo EGMO1_1-EGMO1_9;          array egyr EGYR1_1-EGYR1_9;
array bdays bday1_1-bday1_9;          /* begin day of job1_gap# (internal calculation)*/
array edays eday1_1-eday1_9;          /* end day of job1_gap# (internal calculation)*/
array bweek bgap1_1-bgap1_9;          /* begin week of job1_gap# (created) */
array eweek egap1_1-egap1_9;          /* end week of job1_gap# (created) */
array bflag bflg1_1-bflg1_9;
array eflag eflg1_1-eflg1_9;
array bgfl bgfl1_1-bgfl1_9;
array egfl egfl1_1-egfl1_9;

/* only impute start/stop dates if day is missing */
/* Fill-in start day for those missing */
do over bgyr;
  if bgyr>0 then do;
    if bgmo>0 and bgdy<=0 then do; bgdy=15; bflag=1; end;
  /* Account for beginning gap dates before job */
    if bgyr=staryr1 and bgmo=starmo1 and bgdy<stardy1 and bflag=1 then do; bgdy=stardy1; end;
  end;
end;

/* Fill-in stop day for those missing */
do over egyr;
  if egyr>0 then do;
    if egmo>0 and egdy<=0 then do; egdy=15; eflag=1; end;
  /* Account for end gap dates after job end */
    if egyr=stopyr1 and egmo=stopmo1 and egdy>stopdy1 and eflag=1 then do; egdy=stopdy1; end;
  end;
end;

/*Set flag for gap exists but invalid data*/
do over bgyr;
  bgfl=-4;

```

```

if (-4 < bgmo < 0) or (-4 < bgyr < 0) then do; bgfl=1; end;
end;

do over egyr;
  egfl=-4;
  if (-4 < egyr < 0) or (-4 < egmo < 0) then do; egfl=1; end;
end;

/**Identify within-job gaps on JOB 1; Convert gap dates to week numbers ***/
/*Convert START month and day to total days (BDAYS)*/
do over bgmo;
  if bgmo>0 and bgdy>0 then do;
    if bgmo=1 then bdays=bgdy;
    if bgmo=3 then bdays=bgdy+59;
    if bgmo=5 then bdays=bgdy+120;
    if bgmo=7 then bdays=bgdy+181;
    if bgmo=9 then bdays=bgdy+243;
    if bgmo=11 then bdays=bgdy+304;
    end;
  end;

/**Account for leap years***/
do over bgyr;
  if bgyr=1980 or bgyr=1984 or bgyr=1988 or bgyr=1992 or bgyr=1996 then do;
    if bgmo>0 and bgdy>0 then do;
      if bgmo=1 then bdays=bgdy;
      if bgmo=3 then bdays=bgdy+60;
      if bgmo=5 then bdays=bgdy+121;
      if bgmo=7 then bdays=bgdy+182;
      if bgmo=9 then bdays=bgdy+244;
      if bgmo=11 then bdays=bgdy+305;
    end;
  end;
end;

/**Convert STOP month and day to total days (EDAYS)***/
do over egmo;
  if egmo>0 and egdy>0 then do;
    if egmo=1 then edays=egdy;
    if egmo=3 then edays=egdy+59;
    if egmo=5 then edays=egdy+120;
    if egmo=7 then edays=egdy+181;
    if egmo=9 then edays=egdy+243;
    if egmo=11 then edays=egdy+304;
  end;
end;

/**Account for leap years***/
do over egyr;
  if egyr=1980 or egyr=1984 or egyr=1988 or egyr=1992 or egyr=1996 then do;
    if egmo>0 and egdy>0 then do;
      if egmo=1 then edays=egdy;
      if egmo=2 then edays=egdy+31;
    end;
  end;
end;

```

```

if egmo=3 then edays=egdy+60;
if egmo=5 then edays=egdy+121;
if egmo=7 then edays=egdy+182;
if egmo=9 then edays=egdy+244;
if egmo=11 then edays=egdy+305;
end;
end;
end;

/**Convert days into week numbers*/
/** Basic Formula: weekno=endweek{specific year}+ceil[(totdays+{# of days remaining in DEC})/7] */
/** Note: Use this program takes the week following the actual start of the gap as the measure of when the
non-working period begins. */
do over bgyr;
if bgyr>0 and bdays>0 then do;
  if bgyr=1980 then bweek=ceil((bdays+2)/7);      if bgyr=1981 then bweek=52+ceil((bdays+4)/7);
  if bgyr=1982 then bweek=104+ceil((bdays+5)/7);    if bgyr=1983 then bweek=156+ceil((bdays+6)/7);
  if bgyr=1984 then bweek=209+ceil((bdays)/7);       if bgyr=1985 then bweek=261+ceil((bdays+2)/7);
  if bgyr=1986 then bweek=313+ceil((bdays+3)/7);     if bgyr=1987 then bweek=365+ceil((bdays+4)/7);
  if bgyr=1988 then bweek=417+ceil((bdays+5)/7);     if bgyr=1989 then bweek=470+ceil((bdays)/7);
  if bgyr=1990 then bweek=522+ceil((bdays+1)/7);     if bgyr=1991 then bweek=574+ceil((bdays+2)/7);
  if bgyr=1992 then bweek=626+ceil((bdays+3)/7);     if bgyr=1993 then bweek=678+ceil((bdays+5)/7);
  if bgyr=1994 then bweek=730+ceil((bdays+6)/7);     if bgyr=1995 then bweek=783+ceil((bdays)/7);
  if bgyr=1996 then bweek=835+ceil((bdays+1)/7);     if bgyr=1997 then bweek=887+ceil((bdays+3)/7);
  if bgyr=1998 then bweek=939+ceil((bdays+4)/7);     if bgyr=1999 then bweek=991+ceil((bdays+5)/7);
  if bweek>0 then do; bweek=bweek+1; end;
end;
end;

do over egyr;
if egyr>0 and edays>0 then do;
  if egyr=1980 then eweek=ceil((edays+2)/7);        if egyr=1981 then eweek=52+ceil((edays+4)/7);
  if egyr=1982 then eweek=104+ceil((edays+5)/7);    if egyr=1983 then eweek=156+ceil((edays+6)/7);
  if egyr=1984 then eweek=209+ceil((edays)/7);       if egyr=1985 then eweek=261+ceil((edays+2)/7);
  if egyr=1986 then eweek=313+ceil((edays+3)/7);     if egyr=1987 then eweek=365+ceil((edays+4)/7);
  if egyr=1988 then eweek=417+ceil((edays+5)/7);     if egyr=1989 then eweek=470+ceil((edays)/7);
  if egyr=1990 then eweek=522+ceil((edays+1)/7);     if egyr=1991 then eweek=574+ceil((edays+2)/7);
  if egyr=1992 then eweek=626+ceil((edays+3)/7);     if egyr=1993 then eweek=678+ceil((edays+5)/7);
  if egyr=1994 then eweek=730+ceil((edays+6)/7);     if egyr=1995 then eweek=783+ceil((edays)/7);
  if egyr=1996 then eweek=835+ceil((edays+1)/7);     if egyr=1997 then eweek=887+ceil((edays+3)/7);
  if egyr=1998 then eweek=939+ceil((edays+4)/7);     if egyr=1999 then eweek=991+ceil((edays+5)/7);
  if eweek>0 then do; eweek=eweek-1; end;
end;
end;

/* The following lines omit gap start and stop dates for gaps less than one work week (5 days) */
do over bdays;
  if edays-bdays<5 and bweek>eweek and bdays ne . and edays ne . then do; bweek=.; eweek=.; end;
end;

/* The following omits cases where bweek>eweek, which are caused when missing values are substituted
in. For example, when the day of a beginning gap is unknown, the program uses the 15th. This can cause

```

bweek>eweek, which will cause problems when writing programs for the created variables for the government. This situation will be fixed by making eweek and bweek the same. */

```
do over bweek; if bweek>eweek then bweek=eweek; end;
```

```
/* To correct for bad gap information */
```

```
do over bdays;
  if eweek>stopw1 and eweek ne . then eweek=stopw1;
  if bweek<starw1 and bweek ne . then eweek=starw1;
end;
```

***** At this point in the program, this code is repeated for gaps within each of the respondent's jobs (job 2, job 3, etc.). Due to space constraints the complete program is not included here; researchers should contact NLS User Services if additional information is required.*****

******* SECTION 4: The program finally counts weeks worked between 1980 and 1999. *******

/* By creating a dummy variables for each week in from 1980 to 1999, this program counts the weeks worked by the respondent and removes the within job gaps. It will place a "1" into weeks where the respondent was employed and a "0" into weeks were the respondent was not employed. There are 9 different event histories, one for each possible job on the roster. Weeks range from the first week of 1980 to the last week of 1999, for a total of 1044 weeks.

```
array job1wks (i) wk1_1-wk1_1044;      array job2wks (i) wk2_1-wk2_1044;
array job3wks (i) wk3_1-wk3_1044;      array job4wks (i) wk4_1-wk4_1044;
array job5wks (i) wk5_1-wk5_1044;      array job6wks (i) wk6_1-wk6_1044;
array job7wks (i) wk7_1-wk7_1044;      array job8wks (i) wk8_1-wk8_1044;
array job9wks (i) wk9_1-wk9_1044;      array stopw (i) stopw1-stopw9;
```

```
/* Default Settings*/
```

```
do i=1 to 1044;
  job1wks=0;    job2wks=0;    job3wks=0;
  job4wks=0;    job5wks=0;    job6wks=0;
  job7wks=0;    job8wks=0;    job9wks=0;
end;
```

/* Define rd2wk as the maximum of dliwk and age14wk. This is used for bad start/stop weeks. */
if dliwk>age14wk then do; rd2wk=dliwk; end;
if age14wk=>dliwk then do; rd2wk=age14wk; end;

```
/** total weeks worked on job 1 **/
```

```
/* Set up starfl dummy for invalid dates used later */
starfl_1=0;      stopfl_1=0;
```

```
if starw1=-3 and uid1 ne -5 then do; starw1=rd2wk; starfl_1=1; end;
if stopw1=-3 and uid1 ne -5 then do; stopw1=intwk; stopfl_1=1; end;
if smofl1=1 then do; starfl_1=1; end;
if emofl1=1 then do; stopfl_1=1; end;
```

```
if starw1>0 and stopw1>0 then do; /* [1] */
  do i=(starw1) to (stopw1); job1wks=1; end;
```

```
/* Remove gap 1 on job 1 */
if bgap1_1>0 & egap1_1>0 then do; do i=(bgap1_1) to (egap1_1); job1wks=0; end; end;
/* Remove gap 2 on job 1 */
if bgap1_2>0 & egap1_2>0 then do; do i=(bgap1_2) to (egap1_2); job1wks=0; end; end;
/* and so on through gap 9: variables bgap1_9 and egap1_9 */

/** end of normal gaps, begin missing gap information **/
if bgap1_1=. then bgap1_1=10000;      if egap1_1=. then egap1_1=0;
/* and so on for gaps 2, 3, etc., through */
if bgap1_9=. then bgap1_9=10000;      if egap1_9=. then egap1_9=0;

/* Remove gap 1 on job 1 - beginning gap date bad */
if bgf1_1=1 & egap1_1>0 then do; do i=(starw1) to (egap1_1); job1wks=-3; gpfl1_1=1; end; end;
/* Remove gap 1 on job 1 - end gap date bad */
if bgap1_1>0 & egf1_1=1 then do; do i=(bgap1_1) to (stopw1); job1wks=-3; gpfl1_1=1; end; end;
/* Remove gap 1 on job 1 - both gap dates bad */
if bgf1_1=1 & egf1_1=1 then do; do i=(starw1) to (stopw1); job1wks=-3; gpfl1_1=1; end; end;

/* At this point the program loops through the above three lines of code for each gap 2-9 (for example, the
gap 2 variables are bgap1_2, egap1_2, bgf1_2, egf1_2, and gpfl1_2). This code is deleted here due to
space considerations; users who need the complete program should contact NLS User Services. */

if (starfl_1=1) then do;
  do i=(starw1) to min(stopw1, bgap1_1-1, bgap1_2-1, bgap1_3-1, bgap1_4-1, bgap1_5-1, bgap1_6-1,
    bgap1_7, bgap1_8, bgap1_9);
    job1wks=-3;
  end;
end;

if (stopfl_1=1) then do;
  do i=max(starw1, egap1_1+1, egap1_2+1, egap1_3+1, egap1_4+1, egap1_5+1, egap1_6+1, egap1_7+1,
    egap1_8+1, egap1_9+1) to (stopw1);
    job1wks=-3;
  end;
end;

***** At this point in the program, this code is repeated for to calculate weeks worked, excluding
gaps, for each of the respondent's jobs. Due to space constraints the complete program is not included
here; researchers should contact NLS User Services if additional information is required.*****
```

endsas;

BDATE1.SAS

This program changes the respondent's birthday and 14th birthday to a continuous week number. The variables used are the following:

| Name in Program | Question Name on CD |
|-----------------|---------------------|
| birthdy | KEY!BDATE_D |
| birthmo | KEY!BDATE_M |
| birthyr | KEY!BDATE_Y |
| norcid | YNORCID |

```

***** Calulate Age 14 year *****/
AGE14YR=birthyr+14;

/** Convert Age 14 Birthdate to week number **/

/*change age14 month/day to total days(bbdays)*/
if birthmo>0 and birthdy>0 then do;
  if birthmo=1 then bbdays=birthdy;
  if birthmo=2 then bbdays=birthdy+31;
  if birthmo=3 then bbdays=birthdy+59;
  if birthmo=4 then bbdays=birthdy+90;
  if birthmo=5 then bbdays=birthdy+120;
  if birthmo=6 then bbdays=birthdy+151;
  if birthmo=7 then bbdays=birthdy+181;
  if birthmo=8 then bbdays=birthdy+212;
  if birthmo=9 then bbdays=birthdy+243;
  if birthmo=10 then bbdays=birthdy+273;
  if birthmo=11 then bbdays=birthdy+304;
  if birthmo=12 then bbdays=birthdy+334;
end;

/*Account for leap years*/
if age14yr=1980 or age14yr=1984 or
   age14yr=1988 or age14yr=1992 or
   age14yr=1996 then do;
  if birthmo>0 and birthdy>0 then do;
    if birthmo=1 then bbdays=birthdy;
    if birthmo=2 then bbdays=birthdy+31;
    if birthmo=3 then bbdays=birthdy+60;
    if birthmo=4 then bbdays=birthdy+91;
    if birthmo=5 then bbdays=birthdy+121;
    if birthmo=6 then bbdays=birthdy+152;
    if birthmo=7 then bbdays=birthdy+182;
    if birthmo=8 then bbdays=birthdy+213;
    if birthmo=9 then bbdays=birthdy+244;
    if birthmo=10 then bbdays=birthdy+274;
    if birthmo=11 then bbdays=birthdy+305;
    if birthmo=12 then bbdays=birthdy+335;
  end;
end;

/* Convert days into week numbers */

```

```

/** Basic Formula: weekno=endweek{specific
year}+ceil[(totdays+{# of days remaining in
DEC})/7] **/

age14wk=9999; /* Default age 14 week = 9999 */
if age14yr>0 and bbdays>0 then do;
  if age14yr=1980 then do;
    age14wk=ceil((bbdays+2)/7); end;
  if age14yr=1981 then do;
    age14wk=52+ceil((bbdays+4)/7); end;
  if age14yr=1982 then do;
    age14wk=104+ceil((bbdays+5)/7); end;
  if age14yr=1983 then do;
    age14wk=156+ceil((bbdays+6)/7); end;
  if age14yr=1984 then do;
    age14wk=209+ceil((bbdays)/7); end;
  if age14yr=1985 then do;
    age14wk=261+ceil((bbdays+2)/7); end;
  if age14yr=1986 then do;
    age14wk=313+ceil((bbdays+3)/7); end;
  if age14yr=1987 then do;
    age14wk=365+ceil((bbdays+4)/7); end;
  if age14yr=1988 then do;
    age14wk=417+ceil((bbdays+5)/7); end;
  if age14yr=1989 then do;
    age14wk=470+ceil((bbdays)/7); end;
  if age14yr=1990 then do;
    age14wk=522+ceil((bbdays+1)/7); end;
  if age14yr=1991 then do;
    age14wk=574+ceil((bbdays+2)/7); end;
  if age14yr=1992 then do;
    age14wk=626+ceil((bbdays+3)/7); end;
  if age14yr=1993 then do;
    age14wk=678+ceil((bbdays+5)/7); end;
  if age14yr=1994 then do;
    age14wk=730+ceil((bbdays+6)/7); end;
  if age14yr=1995 then do;
    age14wk=783+ceil((bbdays)/7); end;
  if age14yr=1996 then do;
    age14wk=835+ceil((bbdays+1)/7); end;
  if age14yr=1997 then do;
    age14wk=887+ceil((bbdays+3)/7); end;

```

```

if age14yr=1998 then do;
  age14wk=939+ceil((bbdays+4)/7); end;
if age14yr=1999 then do;
  age14wk=991+ceil((bbdays+5)/7); end;
end;

/** Convert Birthdate to week number */
/*Default birthdate week=0 if bdate <12/30/79*/
birthwk=0;

if birthyr>0 and bbdays>0 then do;
  if birthyr=1980 then do;
    birthwk=ceil((bbdays+2)/7); end;
  if birthyr=1981 then do;
    birthwk=52+ceil((bbdays+4)/7); end;
  if birthyr=1982 then do;
    birthwk=104+ceil((bbdays+5)/7); end;
  if birthyr=1983 then do;
    birthwk=156+ceil((bbdays+6)/7); end;
  if birthyr=1984 then do;
    birthwk=209+ceil((bbdays)/7); end;
  if birthyr=1985 then do;
    birthwk=261+ceil((bbdays+2)/7); end;
  if birthyr=1986 then do;
    birthwk=313+ceil((bbdays+3)/7); end;
  if birthyr=1987 then do;
    birthwk=365+ceil((bbdays+4)/7); end;
  if birthyr=1988 then do;
    birthwk=417+ceil((bbdays+5)/7); end;
  if birthyr=1989 then do;
    birthwk=470+ceil((bbdays)/7); end;
  if birthyr=1990 then do;
    birthwk=522+ceil((bbdays+1)/7); end;
  if birthyr=1991 then do;
    birthwk=574+ceil((bbdays+2)/7); end;
  if birthyr=1992 then do;
    birthwk=626+ceil((bbdays+3)/7); end;
  if birthyr=1993 then do;
    birthwk=678+ceil((bbdays+5)/7); end;
  if birthyr=1994 then do;
    birthwk=730+ceil((bbdays+6)/7); end;
  if birthyr=1995 then do;
    birthwk=783+ceil((bbdays)/7); end;
  if birthyr=1996 then do;
    birthwk=835+ceil((bbdays+1)/7); end;
  if birthyr=1997 then do;
    birthwk=887+ceil((bbdays+3)/7); end;
  if birthyr=1998 then do;
    birthwk=939+ceil((bbdays+4)/7); end;
  if birthyr=1999 then do;
    birthwk=991+ceil((bbdays+5)/7); end;
end;
endsas;

```

Appendix 2: Employment Variable Creation

HOURLY RATE OF PAY, HOURLY MONETARY COMPENSATION AND JOB LENGTH < 13 WEEKS

Variables Created:

- CV_HRLY_PAY
- CV_HRLY_COMPENSATION
- CV_JOB<13_WKS

Variables Used

| Name in Program | Question Name on CD | Name in Program | Question Name on CD |
|--------------------|----------------------|-----------------------------|---------------------------------|
| PUBID | PUBID | E2120011-E2120016 | YEMP-21200.01~000001 - ~000006 |
| E200 | YEMP-200 | E2120021-E2120026 | YEMP-21200.02~000001 - ~000006 |
| E6001-E60010 | YEMP-600.01-.10 | E2120031-E2120036 | YEMP-21200.03~000001 - ~000006 |
| E192001-E192009 | YEMP-19200.01-.09 | E2120041-E2120046 | YEMP-21200.04~000001 - ~000006 |
| E207001-E207009 | YEMP-20700.01-.09 | E2120051-E2120056 | YEMP-21200.05~000001 - ~000006 |
| E229001-E229006 | YEMP-22900.01-.06 | E2120061-E2120066 | YEMP-21200.06~000001 - ~000006 |
| E230001-E230009 | YEMP-23000.01-.09 | E2260411-E2260419, E2260410 | YEMP-22604.01~000001 - ~000010 |
| E232001-E232004 | YEMP-23200.01-.04 | E2260421-E2260429, E2260420 | YEMP-22604.02~000001 - ~000010 |
| E239001-E239004 | YEMP-23900.01-.04 | E2260431-E2260439, E2260430 | YEMP-22604.03~000001 - ~000010 |
| E239011-E239019 | YEMP-23901.01-.09 | E2260441-E2260449, E2260440 | YEMP-22604.04~000001 - ~000010 |
| E245011-E245016 | YEMP-24501.01-.06 | E1022511-E1022519, E1022510 | YEMP-100225.01~000001 - ~000010 |
| E245021-E245026 | YEMP-24502.01-.06 | E1022521-E1022529, E1022520 | YEMP-100225.02~000001 - ~000010 |
| E245141-E245146 | YEMP-24514.01-.06 | E1022531-E1022539, E1022530 | YEMP-100225.03~000001 - ~000010 |
| E24514B3 | YEMP-24514B.03 | E1022541-E1022549, E1022540 | YEMP-100225.04~000001 - ~000010 |
| E334001-E334007 | YEMP-33400.01-.07 | E1022551-E1022559, E1022550 | YEMP-100225.05~000001 - ~000010 |
| E335001-E335005 | YEMP-33500.01-.05 | E3840741, E3840744 | YEMP-38407.04.01, .04 |
| E336001-E336004 | YEMP-33600.01-.04 | E3840753 | YEMP-38407.05.03 |
| E344021-E344027 | YEMP-34402.01-.07 | E3841611-E3841615 | YEMP-38416.01.01-.05 |
| E34402B1-E34402B4 | YEMP-34402AB.01-.04 | E3841621-E3841625 | YEMP-38416.02.01-.05 |
| E344281-E344285 | YEMP-34428.01-.05 | E3841631-E3841634 | YEMP-38416.03.01-.04 |
| E344301 | YEMP-34430.01 | E3841641, E3841644 | YEMP-38416.04.01, .04 |
| E361001-E361005 | YEMP-36100.01-.05 | E3841653 | YEMP-38416.05.03 |
| E362001-E362007 | YEMP-36200.01-.07 | E831001-E831007 | YEMP-83100.01-.07 |
| E2160011-E2160015 | YEMP-21600.01.02-.06 | E868001-E868005 | YEMP-86800.01-.05 |
| E2160021-E2160025 | YEMP-21600.02.02-.06 | E869001-E869006 | YEMP-86900.01-.06 |
| E2160031-E2160034 | YEMP-21600.03.02-.05 | E871001-E871005 | YEMP-87100.01-.05 |
| E2160041-E2160043 | YEMP-21600.04.02-.04 | E885011-E885015 | YEMP-88501.01-.05 |
| E2160051, E2160052 | YEMP-21600.05.02,.03 | E885021-E885025 | YEMP-88502.01-.05 |
| E2160061, E2160062 | YEMP-21600.06.02,.03 | E885121-E885125 | YEMP-88512.01-.05 |
| E2250011-E2250015 | YEMP-22500.01.02-.06 | E973001-E973007 | YEMP-97300.01-.07 |
| E2250021-E2250025 | YEMP-22500.02.02-.06 | E974001-E974005 | YEMP-97400.01-.05 |
| E2250031-E2250034 | YEMP-22500.03.02-.05 | E975001-E975004 | YEMP-97500.01-.04 |
| E2250041-E2250043 | YEMP-22500.04.02-.04 | E983001-E983004 | YEMP-98300.01-.04 |
| E2250051, E2250052 | YEMP-22500.05.02,.03 | E984021-E984027 | YEMP-98402.01-.07 |
| E2250061, E2250062 | YEMP-22500.06.02,.03 | E98402D1-E98402D4 | YEMP-98402D.01-.04 |
| E37901B1-E37901B9 | YEMP-37901B.01-.09 | E984031-E984035 | YEMP-98403.01-.05 |
| E3800B1-E3800B9 | YEMP-38000B.01-.09 | E984041-E984045 | YEMP-98404.01-.05 |
| E3800F1-E3800F6 | YEMP-38000F.01-.06 | E984141-E984145 | YEMP-98414.01-.05 |
| E380131-E380139 | YEMP-38013.01-.09 | E98414B1 | YEMP-98414B.01 |
| E380141-E380146 | YEMP-38014.01-.06 | E984291-E984295 | YEMP-98429.01-.05 |
| E380231-E380236 | YEMP-38023.01-.06 | E995001-E995004 | YEMP-99500.01-.04 |
| E380271, E380272 | YEMP-38027.01,.02 | E1001001-E1001007 | YEMP-100100.01-.07 |
| E381011-E381016 | YEMP-38101.01-.06 | E1020511-E1020514 | YEMP-100205.01.02-.05 |
| E381021, E381022 | YEMP-38102.01,.02 | E1020521-E1020525 | YEMP-100205.02.02-.06 |
| E381031, E381032 | YEMP-38103.01,.02 | E1020531-E1020534 | YEMP-100205.03.02-.05 |
| E381041, E381042 | YEMP-38104.01,.02 | E1020541-E1020543 | YEMP-100205.04.02-.04 |
| E381051, E381052 | YEMP-38105.01,.02 | E1020551-E1020555 | YEMP-100205.05.02-.06 |
| E381061-E381066 | YEMP-38106.01-.06 | E1021411-E1021414 | YEMP-100214.01.02-.05 |
| E381071-E381076 | YEMP-38107.01-.06 | E1021421-E1021425 | YEMP-100214.02.02-.06 |
| E381161-E381166 | YEMP-38116.01-.06 | E1021431-E1021434 | YEMP-100214.03.02-.05 |
| E38116B1 | YEMP-38116B.01 | E1021441, E1021442 | YEMP-100214.04.02, .03 |
| E382011-E382016 | YEMP-38201.01-.06 | E1021432 | YEMP-100214.04.04 |
| E382021-E382024 | YEMP-38202.01-.04 | E1021451-E1021455 | YEMP-100214.05.02-.06 |

Appendix 2: Employment Variable Creation

| | | | |
|-------------------|----------------------|--------------------|---------------------------|
| E382111-E382114 | YEMP-38211.01-04 | E226091-E226094 | YEMP-22609.01.01-.04.01 |
| E599001-E599007 | YEMP-59900.01-.07 | E226101-E226103 | YEMP-22610.01.02-.03.02 |
| E344031-E344035 | YEMP-34403.01-.05 | E226111 | YEMP-22611.01.03 |
| E344041-E344045 | YEMP-34404.01-.05 | E226121, E226122 | YEMP-22612.01.04, .02.04 |
| E34413C1-E34413C5 | YEMP-34413C.01-.05 | E226131 | YEMP-22613.01.05 |
| E34413E1 | YEMP-34413E.01 | E226151, E226152 | YEMP-22615.01.07, .02.07 |
| E380011-E380019 | YEMP-38001.01-.09 | E226172 | YEMP-22617.02.09 |
| E380021-E380025 | YEMP-38002.01-.05 | E226261-E226264 | YEMP-22626.01.01-.04.01 |
| E380031-E380035 | YEMP-38003.01-.05 | E226271, E226272 | YEMP-22627.01.04, .02.04 |
| E380121-E380125 | YEMP-38012.01-.05 | E226281 | YEMP-22628.01.05 |
| E38012B1 | YEMP-38012B.01 | E226301, E226302 | YEMP-22630.01.07, .02.07 |
| E381011-E381016 | YEMP-38101.01-.06 | E226322 | YEMP-22632.02.09 |
| E381021, E381022 | YEMP-38102.01, .02 | E377011-E377019 | YEMP-37701.01-.09 |
| E381051, E381052 | YEMP-38105.01, .02 | E1002301-E1002304 | YEMP-100230.01.01-.04.01 |
| E382011-E382016 | YEMP-38201.01-.06 | E1002311, E1002313 | YEMP-100231.01.02, .03.02 |
| E382021-E382024 | YEMP-38202.01-.04 | E1002335 | YEMP-100233.05.04 |
| E382111-E382114 | YEMP-38211.01-.04 | E1002341 | YEMP-100234.01.05 |
| E383131-E383139 | YEMP-38313.01-.09 | E1002361 | YEMP-100236.01.07 |
| E383291-E383295 | YEMP-38329.01-.05 | E1002392 | YEMP-100239.02.09 |
| E38329B1-E38329B3 | YEMP-38329B.01-.03 | E1002481-E1002484 | YEMP-100248.01.01-.04.01 |
| E38329D1-E38329D3 | YEMP-38329D.01-.03 | E1002495 | YEMP-100249.05.04 |
| E383301-E383304 | YEMP-38330.01-.04 | E1002501 | YEMP-100250.01.05 |
| E3840711-E3840715 | YEMP-38407.01.01-.05 | E1002521 | YEMP-100252.01.07 |
| E3840721-E3840725 | YEMP-38407.02.01-.05 | E1002542 | YEMP-100254.02.09 |
| E3840731-E3840734 | YEMP-38407.03.01-.04 | | |

Codes for Created Variable

Note that hourly rate of pay is reported with two implied decimal places.

This program creates the hourly rate of pay for NLSY97 respondents. The hourly rate of pay is constructed from stop date information for respondents who have a job lasting more than 13 weeks. For all other respondents the start wage is used.

In addition, this program creates an hourly monetary compensation variable for NLSY97 respondents. This variable that includes information about all compensation received by the respondent, such as tips, bonuses, commissions, overtime, etc., in the calculation. Hourly monetary compensation differs from hourly rate of pay variable, which calculates only the base pay rate. This variable is constructed from stop date information for respondents with jobs longer than 13 weeks and start date information for other.

Finally, a variable indicating whether the jobs lasted more than 13 weeks is also created. There are up to 9 jobs reported, so each variable is created for 9 jobs.

/* DECLARING THE ARRAYS TO BE LATER ON USED IN THE PROGRAM */

```

array E19200 E192001-E192009;
array E59900 E599001-E599009;
array E23000 E230001-E230009;
array E23900 E239001-E239009;
array E34402 E344021-E344029;
array E33600 E336001-E336009;
array E34428 E344281-E344289;
array E36100 E361001-E361009;
array E20700 E207001-E207009;
array E24514 E245141-E245149;
array E38013 E380131-E380139;
array E38023 E380231-E380239;
array E38107 E381071-E381079;
array E3800B E3800B1-E3800B9;

array E37901B E37901B1-E37901B9;
array E22900 E229001-E229009;
array E23200 E232001-E232009;
array E33400 E334001-E334009;
array E34402B E34402B1-E34402B9;
array E33500 E335001-E335009;
array E34430 E344301-E344309;
array E36200 E362001-E362009;
array E24501 E245011-E245019;
array E24514B E24514B1-E24514B9;
array E38014 E380141-E380149;
array E38106 E381061-E381069;
array E38116 E381161-E381169;
array E38027 E380271-E380279;

```

Appendix 2: Employment Variable Creation

array E23901 E239011-E239019;
array E38101 E381011-E381019;
array E38102 E381021-E381029;
array E38201 E382011-E382019;
array E24502 E245021-E245029;
array E38211 E382111-E382119;
array E38211B E38211B1-E38211B9;
array E38313 E383131-E383139;
array E86800 E868001-E868009;
array E87100 E871001-E871009;
array E97300 E973001-E973009;
array E97500 E975001-E975009;
array E98402 E984021-E984029;
array E98429 E984291-E984299;
array E100000 E1000001-E1000009;
array E99500 E995001-E995009;
array E88502 E885021-E885029;
array E88512B E88512B1-E88512B9;
array E34404 E344041-E344049;
array E34413E E34413E1-E34413E9;
array E98404 E984041-E984049;
array E98414B E98414B1-E98414B9;
array E38002 E380021-E380029;
array E38012 E380121-E380129;
array E38329B E38329B1-E38329B9;
array E38329 E383291-E383299;
array E35600 E356001-E356009;
array E22610 E226101-E226109;
array E22612 E226121-E226129;
array E22614 E226141-E226149;
array E22616 E226161-E226269;
array E22626 E226261-E226269;
array E22628 E226281-E226289;
array E22630 E226301-E226309;
array E22632 E226321-E226329;
array E100231 E1002311-E1002319;
array E100233 E1002331-E1002339;
array E100235 E1002351-E1002359;
array E100237 E1002371-E1002379;
array E100248 E1002481-E1002489;
array E100250 E1002501-E1002509;
array E100252 E1002521-E1002529;
array E100254 E1002541-E1002549;
array E37701 E377011-E377019;
array E58201 E582011-E582019;

array E3800F E3800F1-E3800F9;
array E3811B E3811B1-E3811B9;
array E38103 E381031-E381039;
array E38105 E381051-E381059;
array E38202 E382021-E382029;
array E38103F E38103F1-E38103F9;
array E34400 E344001-E344009;
array E83100 E831001-E831009;
array E86900 E869001-E869009;
array E87800 E878001-E878009;
array E97400 E974001-E974009;
array E98300 E983001-E983009;
array E98402D E98402D1-E98402D9;
array E98429E E98429E1-E98429E9;
array E100100 E1001001-E1001009;
array E88501 E885011-E885019;
array E88512 E885121-E885129;
array E34403 E344031-E344039;
array E34413C E34413C1-E34413C9;
array E98403 E984031-E984039;
array E98414 E984141-E984149;
array E38001 E380011-E380019;
array E38003 E380031-E380039;
array E38012B E38012B1-E38012B9;
array E38329D E38329D1-E38329D9;
array E38330 E383301-E383309;
array E22609 E226091-E226099;
array E22611 E226111-E226119;
array E22613 E226131-E226139;
array E22615 E226151-E226159;
array E22617 E226171-E226179;
array E22627 E226271-E226279;
array E22629 E226291-E226299;
array E22631 E226311-E226319;
array E100230 E1002301-E1002309;
array E100232 E1002321-E1002329;
array E100234 E1002341-E1002349;
array E100236 E1002361-E1002369;
array E100239 E1002391-E1002399;
array E100249 E1002491-E1002499;
array E100251 E1002511-E1002519;
array E100253 E1002531-E1002539;
array E36802 E368021-E368029;
array E58401 E584011-E584019;

array E225001 E2250011 E2250021 E2250031 E2250041 E2250051 E2250061 E2250071 E2250081 E2250091;
array E225002 E2250012 E2250022 E2250032 E2250042 E2250052 E2250062 E2250072 E2250082 E2250092;
array E225003 E2250013 E2250023 E2250033 E2250043 E2250053 E2250063 E2250073 E2260083 E2250093;
array E225004 E2250014 E2250024 E2250034 E2250044 E2250054 E2250064 E2250074 E2250084 E2250094;
array E225005 E2250015 E2250025 E2250035 E2250045 E2250055 E2250065 E2250075 E2250085 E2250095;
array E216001 E2160011 E2160021 E2160031 E2160041 E2160051 E2160061 E2160071 E2160081 E2160091;
array E216002 E2160012 E2160022 E2160032 E2160042 E2160052 E2160062 E2160072 E2160082 E2160092;
array E216003 E2160013 E2160023 E2160033 E2160043 E2160053 E2160063 E2160073 E2160083 E2160093;
array E216004 E2160014 E2160024 E2160034 E2160044 E2160054 E2160064 E2160074 E2160084 E2160094;
array E216005 E2160015 E2160025 E2160035 E2160045 E2160055 E2160065 E2160075 E2160085 E2160095;
array E212001 E2120011 E2120021 E2120031 E2120041 E2120051 E2120061 E2120071 E2120081 E2120091;

```

array E212002 E2120012 E2120022 E2120032 E2120042 E2120052 E2120062 E2120072 E2120082 E2120092;
array E212003 E2120013 E2120023 E2120033 E2120043 E2120053 E2120063 E2120073 E2120083 E2120093;
array E212004 E2120014 E2120024 E2120034 E2120044 E2120054 E2120064 E2120074 E2120084 E2120094;
array E212005 E2120015 E2120025 E2120035 E2120045 E2120055 E2120065 E2120075 E2120085 E2120095;
array E212006 E2120016 E2120026 E2120036 E2120046 E2120056 E2120066 E2120076 E2120086 E2120096;
array E384161 E3841611 E3841621 E3841631 E3841641 E3841651 E3841661 E3841671 E3841681 E3841691;
array E384162 E3841612 E3841622 E3841632 E3841642 E3841652 E3841662 E3841672 E3841682 E3841692;
array E384163 E3841613 E3841623 E3841633 E3841643 E3841653 E3841663 E3841673 E3841683 E3841693;
array E384164 E3841614 E3841624 E3841634 E3841644 E3841654 E3841664 E3841674 E3841684 E3841694;
array E384165 E3841615 E3841625 E3841635 E3841645 E3841655 E3841665 E3841675 E3841685 E3841695;
array E384071 E3840711 E3840721 E3840731 E3840741 E3840751 E3840761 E3840771 E3840781 E3840791;
array E384072 E3840712 E3840722 E3840732 E3840742 E3840752 E3840762 E3840772 E3840782 E3840792;
array E384073 E3840713 E3840723 E3840733 E3840743 E3840753 E3840763 E3840773 E3840783 E3840793;
array E384074 E3840714 E3840724 E3840734 E3840744 E3840754 E3840764 E3840774 E3840784 E3840794;
array E384075 E3840715 E3840725 E3840735 E3840745 E3840755 E3840765 E3840775 E3840785 E3840795;
array E102051 E1020511 E1020521 E1020531 E1020541 E1020551 E1020561 E1020571 E1020581 E1020591;
array E102052 E1020512 E1020522 E1020532 E1020542 E1020552 E1020562 E1020572 E1020582 E1020592;
array E102053 E1020513 E1020523 E1020533 E1020543 E1020553 E1020563 E1020573 E1020583 E1020593;
array E102054 E1020514 E1020524 E1020534 E1020544 E1020554 E1020564 E1020574 E1020584 E1020594;
array E102055 E1020515 E1020525 E1020535 E1020545 E1020555 E1020565 E1020575 E1020585 E1020595;
array E102141 E1021411 E1021421 E1021431 E1021441 E1021451 E1021461 E1021471 E1021481 E1021491;
array E102142 E1021412 E1021422 E1021432 E1021442 E1021452 E1021462 E1021472 E1021482 E1021492;
array E102143 E1021413 E1021423 E1021433 E1021443 E1021453 E1021463 E1021473 E1021483 E1021493;
array E102144 E1021414 E1021424 E1021434 E1021444 E1021454 E1021464 E1021474 E1021484 E1021494;
array E102145 E1021415 E1021425 E1021435 E1021445 E1021455 E1021465 E1021475 E1021485 E1021495;
array E226041 E2260411 E2260421 E2260431 E2260441 E2260451 E2260461 E2260471 E2260481 E2260491;
array E226042 E2260412 E2260422 E2260432 E2260442 E2260452 E2260462 E2260472 E2260482 E2260492;
array E226043 E2260413 E2260423 E2260433 E2260443 E2260453 E2260463 E2260473 E2260083 E2260493;
array E226044 E2260414 E2260424 E2260434 E2260444 E2260454 E2260464 E2260474 E2260484 E2260494;
array E226045 E2260415 E2260425 E2260435 E2260445 E2260455 E2260465 E2260475 E2260485 E2260495;
array E226046 E2260416 E2260426 E2260436 E2260446 E2260456 E2260466 E2260476 E2260486 E2260496;
array E226047 E2260417 E2260427 E2260437 E2260447 E2260457 E2260467 E2260477 E2260487 E2260497;
array E226048 E2260418 E2260428 E2260438 E2260448 E2260458 E2260468 E2260478 E2260088 E2260498;
array E226049 E2260419 E2260429 E2260439 E2260449 E2260459 E2260469 E2260479 E2260489 E2260499;
array E226040 E2260410 E2260420 E2260430 E2260440 E2260450 E2260460 E2260470 E2260480 E2260490;
array E102251 E1022511 E1022521 E1022531 E1022541 E1022551 E1022561 E1022571 E1022581 E1022591;
array E102252 E1022512 E1022522 E1022532 E1022542 E1022552 E1022562 E1022572 E1022582 E1022592;
array E102253 E1022513 E1022523 E1022533 E1022543 E1022553 E1022563 E1022573 E2260083 E1022593;
array E102254 E1022514 E1022524 E1022534 E1022544 E1022554 E1022564 E1022574 E1022584 E1022594;
array E102255 E1022515 E1022525 E1022535 E1022545 E1022555 E1022565 E1022575 E1022585 E1022595;
array E102256 E1022516 E1022526 E1022536 E1022546 E1022556 E1022566 E1022576 E1022586 E1022596;
array E102257 E1022517 E1022527 E1022537 E1022547 E1022557 E1022567 E1022577 E1022587 E1022597;
array E102258 E1022518 E1022528 E1022538 E1022548 E1022558 E1022568 E1022578 E2260088 E1022598;
array E102259 E1022519 E1022529 E1022539 E1022549 E1022559 E1022569 E1022579 E1022589 E1022599;
array E102250 E1022510 E1022520 E1022530 E1022540 E1022550 E1022560 E1022570 E1022580 E1022590;

```

***** Section 1: START WAGES FOR THE YOUTH *****

**** Part 1: Start hourly rate of pay ****/

/* For each time unit, the information could either be given in the earlier part (from E19200) or in the later part (from E83100); it also depends on whether the respondent has other compensation. */

```

/*Respondents reporting an HOURLY wage*/
HRWAGE01=-4;          HRWAGE02=-4;          HRWAGE03=-4;
HRWAGE04=-4;          HRWAGE05=-4;          HRWAGE06=-4;
HRWAGE07=-4;          HRWAGE08=-4;          HRWAGE09=-4;

```

Appendix 2: Employment Variable Creation

```

array HRWAGE HRWAGE01 HRWAGE02 HRWAGE03 HRWAGE04 HRWAGE05 HRWAGE06
      HRWAGE07 HRWAGE08 HRWAGE09;

do I=1 to 9;
  if E19200[I]=1 then do;
    if (E22900[I]>=0) then HRWAGE[I]=E22900[I];
    if (E23000[I]>=0) then HRWAGE[I]=E23000[I];
    if (E22900[I]=-2 or E23000[I]=-2) then HRWAGE[I]=E23200[I];
    if (E22900[I]=-3 or E23000[I]=-3) then HRWAGE[I]=E23200[I];
    if (E22900[I]=-1 or E23000[I]=-1) then HRWAGE[I]=-1;
    if E23900[I]>=0 then HRWAGE[I]=E23900[I];
  end;

  if E83100[I]=1 then do;
    if (E86800[I]>=0) then HRWAGE[I]=E86800[I];
    if (E86900[I]>=0) then HRWAGE[I]=E86900[I];
    if (E86800[I]=-2 or E86900[I]=-2) then HRWAGE[I]=E87100[I];
    if (E86800[I]=-3 or E86900[I]=-3) then HRWAGE[I]=E87100[I];
    if (E86800[I]=-1 or E86900[I]=-1) then HRWAGE[I]=-1;
    if E87800[I]>=0 then HRWAGE[I]=E87800[I];
  end;
end;

/*Respondents reporting a DAILY wage*/
DAILY01=-4;           DAILY02=-4;           DAILY03=-4;
DAILY04=-4;           DAILY05=-4;           DAILY06=-4;
DAILY07=-4;           DAILY08=-4;           DAILY09=-4;

array DAILY DAILY01 DAILY02 DAILY03 DAILY04 DAILY05 DAILY06 DAILY07 DAILY08 DAILY09;

do I=1 to 9; /* daily start wage divided by the number of hours worked per week*/
if E19200[I]=2 then do;

/*no compensation*/
  if (E33400[I]>=0 and E34402[I]>0 and E34402B[I]>0) then DAILY[I]=(E33400[I]*E34402B[I]/E34402[I]);
  if (E33400[I]=-2 and E33600[I]>=0 and E34402[I]>0 and E34402B[I]>0) then
    DAILY[I]=(E33600[I]*E34402B[I]/E34402[I]);
  if E34400[I]>=0 and E34402[I]>0 and E34402B[I]>0 then DAILY[I]=E34400[I]*E34402B[I]/E34402[I];
  /* missing value*/ if -4<E34402[I]<0 then DAILY[I]=E34402[I];

/*received compensation*/
/*without overtime*/
  if (E33500[I]>=0 and E34402[I]>0 and E34402B[I]>0) then DAILY[I]=(E33500[I]*E34402B[I]/E34402[I]);
  if (E33500[I]=-2 and E33600[I]>=0 and E34402[I]>0 and E34402B[I]>0) then
    DAILY[I]=(E33600[I]*E34402B[I]/E34402[I]);
  if E34400[I]>=0 and E34402[I]>0 and E34402B[I]>0 then DAILY[I]=E34400[I]*E34402B[I]/E34402[I];
  /*missing values*/
  if -4<E34402[I]<0 then DAILY[I]=E34402[I];
  if (E33400[I]>=0 or E33500[I]>=0 or E33600[I]>=0 or E34400[I]>=0) and E34402[I]=0 then DAILY[I]=-3;
  if -4<E34402B[I]<0 then DAILY[I]=E34402B[I];
  if E34402B[I]=0 then DAILY[I]=-3;

/*with overtime*/
  if (E33500[I]>=0 and E34428[I]>0 and E34430[I]>0) then DAILY[I]=(E33500[I]*E34430[I]/E34428[I]);
  if (E33500[I]=-2 and E33600[I]>=0 and E34428[I]>0 and E34430[I]>0) then
    DAILY[I]=(E33600[I]*E34430[I]/E34428[I]);
  if E34400[I]>=0 and E34428[I]>0 and E34430[I]>0 then DAILY[I]=E34400[I]*E34430[I]/E34428[I];

```

```

/*missing values*/
if -4<E34428[I]<0 then DAILY[I]=E34428[I];
if (E33500[I]>=0 or E33600[I]>=0 or E34400[I]>=0) and E34428[I]=0 then DAILY[I]=-3;
if -4<E34430[I]<0 then DAILY[I]=E34430[I];
if E34430[I]=0 then DAILY[I]=-3;

/*if still paid hourly...*/
if E36200[I]>=0 then DAILY[I]=E36200[I];
if E36100[I]>=0 then DAILY[I]=E36100[I];
end;

if E83100[I]=2 then do;

/*no compensation*/
if (E97300[I]>=0 and E98402[I]>0 and E98402D[I]>0) then DAILY[I]=(E97300[I]*E98402D[I]/E98402[I]);
if (E97300[I]=-2 and E97500[I]>=0 and E98402[I]>0 and E98402D[I]>0) then
    DAILY[I]=(E97500[I]*E98402D[I]/E98402[I]);
if E98300[I]>=0 and E98402D[I]>0 and E98402D[I]>0 then DAILY[I]=E98300[I]*E98402D[I]/E98402D[I];
/* missing value*/ if -4<E98402D[I]<0 then DAILY[I]=E98402D[I];

/*received compensation*/
/*with overtime */
if (E97400[I]>=0 and E98429[I]>0 and E98429E[I]>0) then DAILY[I]=(E97400[I]*E98429E[I]/E98429[I]);
if (E97400[I]=-2 and E97500[I]>=0 and E98429[I]>0 and E98429E[I]>0) then
    DAILY[I]=(E97500[I]*E98429E[I]/E98429[I]);
if E98300[I]>=0 and E98429D[I]>0 and E98429E[I]>0 then DAILY[I]=E98300[I]*E98429E[I]/E98429D[I];
/*missing values*/
if -4<E98429D[I]<0 then DAILY[I]=E98429D[I];
if (E97400[I]>=0 or E97500[I]>=0 or E98300[I]>=0) and E98429D[I]=0 then DAILY[I]=-3;
if -4<E98429E[I]<0 then DAILY[I]=E98429E[I];
if E98429E[I]=0 then DAILY[I]=-3;

/* without overtime */
if (E97400[I]>=0 and E98402D[I]>0 and E98402D[I]>0) then DAILY[I]=(E97400[I]*E98402D[I]/E98402D[I]);
if (E97400[I]=-2 and E97500[I]>=0 and E98402D[I]>0 and E98402D[I]>0) then
    DAILY[I]=(E97500[I]*E98402D[I]/E98402D[I]);
if E98300[I]>=0 and E98402D[I]>0 and E98402D[I]>0 then DAILY[I]=E98300[I]*E98402D[I]/E98402D[I];
/*missing values*/
if -4<E98402D[I]<0 then DAILY[I]=E98402D[I];
if (E97300[I]>=0 or E97400[I]>=0 or E97500[I]>=0 or E98300[I]>=0) and E98402D[I]=0 then DAILY[I]=-3;
if -4<E98402D[I]<0 then DAILY[I]=E98402D[I];
if E98402D[I]=0 then DAILY[I]=-3;

/*if still paid hourly...*/
if E100100[I]>=0 then DAILY[I]=E100100[I];
if E100000[I]>=0 then DAILY[I]=E100000[I];
end;
end;

/*Respondents reporting a WEEKLY wage*/
WEEKLY01=-4;          WEEKLY02=-4;          WEEKLY03=-4;
WEEKLY04=-4;          WEEKLY05=-4;          WEEKLY06=-4;
WEEKLY07=-4;          WEEKLY08=-4;          WEEKLY09=-4;

array WEEKLY WEEKLY01 WEEKLY02 WEEKLY03 WEEKLY04 WEEKLY05 WEEKLY06
    WEEKLY07 WEEKLY08 WEEKLY09;

```

Appendix 2: Employment Variable Creation

```
do I=1 to 9; /* weekly start wage divided by the number of hours worked per week*/
if E19200[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) then do;

/*no compensation*/
if (E33400[I]>=0 and E34402[I]>0) then WEEKLY[I]=(E33400[I]/E34402[I]);
if (E33400[I]=-2 and E33600[I]>=0 and E34402[I]>0) then WEEKLY[I]=(E33600[I]/E34402[I]);
if E34400[I]>=0 and E34402[I]>0 then WEEKLY[I]=E34400[I]/E34402[I];
/*missing value*/ if -4<E34402[I]<0 then WEEKLY[I]=E34402[I];

/*received compensation*/
/*with overtime*/
if (E33500[I]>=0 and E34428[I]>0) then WEEKLY[I]=(E33500[I]/E34428[I]);
if (E33500[I]=-2 and E33600[I]>=0 and E34428[I]>0) then WEEKLY[I]=(E33600[I]/E34428[I]);
if E34400[I]>=0 and E34428[I]>0 then WEEKLY[I]=E34400[I]/E34428[I];
/*missing values*/
if -4<E34428[I]<0 then WEEKLY[I]=E34428[I];
if (E33500[I]>=0 or E33600[I]>=0 or E34400[I]>=0) and E34428[I]=0 then WEEKLY[I]=-3;

/*without overtime*/
if (E33500[I]>=0 and E34402[I]>0) then WEEKLY[I]=(E33500[I]/E34402[I]);
if (E33500[I]=-2 and E33600[I]>=0 and E34402[I]>0) then WEEKLY[I]=(E33600[I]/E34402[I]);
if E34400[I]>=0 and E34402[I]>0 then WEEKLY[I]=E34400[I]/E34402[I];
/*missing values*/
if -4<E34402[I]<0 then WEEKLY[I]=E34402[I];
if (E33400[I]>=0 or E33500[I]>=0 or E33600[I]>=0 or E34400[I]>=0) and E34402[I]=0 then WEEKLY[I]=-3;

/*if still paid hourly...*/
if E36200[I]>=0 then WEEKLY[I]=E36200[I];
if E36100[I]>=0 then WEEKLY[I]=E36100[I];
end;

if E83100[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) then do;

/*no compensation*/
if (E97300[I]>=0 and E98402[I]>0) then WEEKLY[I]=(E97300[I]/E98402[I]);
if (E97300[I]=-2 and E97500[I]>=0 and E98402[I]>0) then WEEKLY[I]=(E97500[I]/E98402[I]);
if E98300[I]>=0 and E98402[I]>0 then WEEKLY[I]=E98300[I]/E98402[I];
/*missing value*/ if -4<E98402[I]<0 then WEEKLY[I]=E98402[I];

/*received compensation*/
/*with overtime*/
if (E97400[I]>=0 and E98429[I]>0) then WEEKLY[I]=(E97400[I]/E98429[I]);
if (E97400[I]=-2 and E97500[I]>=0 and E98429[I]>0) then WEEKLY[I]=(E97500[I]/E98429[I]);
if E98300[I]>=0 and E98429[I]>0 then WEEKLY[I]=E98300[I]/E98429[I];
/*missing values*/
if -4<E98429[I]<0 then WEEKLY[I]=E98429[I];
if (E97400[I]>=0 or E97500[I]>=0 or E98300[I]>=0) and E98429[I]=0 then WEEKLY[I]=-3;

/*without overtime*/
if (E97400[I]>=0 and E98402[I]>0) then WEEKLY[I]=(E97400[I]/E98402[I]);
if (E97400[I]=-2 and E97500[I]>=0 and E98402[I]>0) then WEEKLY[I]=(E97500[I]/E98402[I]);
if E98300[I]>=0 AND E98402[I]>0 then WEEKLY[I]=E98300[I]/E98402[I];
/*missing values*/
if -4<E98402[I]<0 then WEEKLY[I]=E98402[I];
if (E97300[I]>=0 or E97400[I]>=0 or E97500[I]>=0 or E98300[I]>=0) and E98402[I]=0 then WEEKLY[I]=-3;

/*if still paid hourly...*/
```

```

if E100100[I]>=0 then WEEKLY[I]=E100100[I];
  if E100000[I]>=0 then WEEKLY[I]=E100000[I];
end;
end;

/*Respondents reporting a BIWEEKLY wage*/
BIWKLY01=-4;           BIWKLY02=-4;           BIWKLY03=-4;
BIWKLY04=-4;           BIWKLY05=-4;           BIWKLY06=-4;
BIWKLY07=-4;           BIWKLY08=-4;           BIWKLY09=-4;

array BIWKLY BIWKLY01 BIWKLY02 BIWKLY03 BIWKLY04 BIWKLY05 BIWKLY06 BIWKLY07
      BIWKLY08 BIWKLY09;

do I=1 to 9; /* biwkly start wage divided by the number of hours worked per week*/
if E19200[I]=4 then do;

/*no compensation*/
  if (E33400[I]>=0 and E34402[I]>0) then BIWKLY[I]=E33400[I]/(2*E34402[I]);
  if (E33400[I]=-2 and E33600[I]>=0 and E34402[I]>0) then BIWKLY[I]=E33600[I]/(2*E34402[I]);
  if E34400[I]>=0 and E34402[I]>0 then BIWKLY[I]=E34400[I]/(2*E34402[I]);
  /*missing value*/ if -4<E34402[I]<0 then BIWKLY[I]=E34402[I];

/*received compensation*/
/*with overtime*/
  if (E33500[I]>=0 and E34428[I]>0) then BIWKLY[I]=E33500[I]/(2*E34428[I]);
  if (E33500[I]=-2 and E33600[I]>=0 and E34428[I]>0) then BIWKLY[I]=E33600[I]/(2*E34428[I]);
  if E34400[I]>=0 and E34428[I]>0 then BIWKLY[I]=E34400[I]/(2*E34428[I]);
  /*missing values*/
  if -4<E34428[I]<0 then BIWKLY[I]=E34428[I];
  if (E33500[I]>=0 or E33600[I]>=0 or E34400[I]>=0) and E34428[I]=0 then BIWKLY[I]=-3;

/*without overtime*/
  if (E33500[I]>=0 and E34402[I]>0) then BIWKLY[I]=E33500[I]/(2*E34402[I]);
  if (E33500[I]=-2 and E33600[I]>=0 and E34402[I]>0) then BIWKLY[I]=E33600[I]/(2*E34402[I]);
  if E34400[I]>=0 and E34402[I]>0 then BIWKLY[I]=E34400[I]/(2*E34402[I]);
  /*missing values*/
  if -4<E34402[I]<0 then BIWKLY[I]=E34402[I];
  if (E33400[I]>=0 or E33500[I]>=0 or E33600[I]>=0 or E34400[I]>=0) and E34402[I]=0 then BIWKLY[I]=-3;

/*if still paid hourly...*/
  if E36200[I]>=0 then BIWKLY[I]=E36200[I];
  if E36100[I]>=0 then BIWKLY[I]=E36100[I];
end;

if E83100[I]=4 then do;

/*no compensation*/
  if (E97300[I]>=0 and E98402[I]>0) then BIWKLY[I]=E97300[I]/(2*E98402[I]);
  if (E97300[I]=-2 and E97500[I]>=0 and E98402[I]>0) then BIWKLY[I]=E97500[I]/(2*E98402[I]);
  if E98300[I]>=0 and E98402[I]>0 then BIWKLY[I]=E98300[I]/(2*E98402[I]);
  /*missing value*/ if -4<E98402[I]<0 then BIWKLY[I]=E98402[I];

/*received compensation*/
/*with overtime*/
  if (E97400[I]>=0 and E98429[I]>0) then BIWKLY[I]=E97400[I]/(2*E98429[I]);
  if (E97400[I]=-2 and E97500[I]>=0 and E98429[I]>0) then BIWKLY[I]=E97500[I]/(2*E98429[I]);
  if E98300[I]>=0 and E98429[I]>0 then BIWKLY[I]=E98300[I]/(2*E98429[I]);

```

Appendix 2: Employment Variable Creation

```
/*missing values*/
if -4<E98429[I]<0 then BIWKLY[I]=E98429[I];
if (E97400[I]>=0 or E97500[I]>=0 or E98300[I]>=0) and E98429[I]=0 then BIWKLY[I]=-3;

/*without overtime*/
if (E97400[I]>=0 and E98402[I]>0) then BIWKLY[I]=E97400[I]/(2*E98402[I]);
if (E97400[I]=-2 and E97500[I]>=0 and E98402[I]>0) then BIWKLY[I]=E97500[I]/(2*E98402[I]);
if E98300[I]>=0 and E98402[I]>0 then BIWKLY[I]=E98300[I]/(2*E98402[I]);
/*missing values*/
if -4<E98402[I]<0 then BIWKLY[I]=E98402[I];
if (E97300[I]>=0 or E97400[I]>=0 or E97500[I]>=0 or E98300[I]>=0) and E98402[I]=0 then BIWKLY[I]=-3;

/*if still paid hourly...*/
if E100100[I]>=0 then BIWKLY[I]=E100100[I];
if E100000[I]>=0 then BIWKLY[I]=E100000[I];
end;
end;

/*Respondents reporting a MONTHLY wage*/
MONTH01=-4;           MONTH02=-4;           MONTH03=-4;
MONTH04=-4;           MONTH05=-4;           MONTH06=-4;
MONTH07=-4;           MONTH08=-4;           MONTH09=-4;

array MONTH MONTH01 MONTH02 MONTH03 MONTH04 MONTH05 MONTH06 MONTH07
      MONTH08 MONTH09;

do I=1 to 9; /* month start wage divided by the number of hours worked per week*/
if E19200[I]=5 then do;

/*no compensation*/
if (E33400[I]>=0 and E34402[I]>0) then MONTH[I]=E33400[I]/(4.3*E34402[I]);
if (E33400[I]=-2 and E33600[I]>=0 and E34402[I]>0) then MONTH[I]=E33600[I]/(4.3*E34402[I]);
if E34400[I]>=0 and E34402[I]>0 then MONTH[I]=E34400[I]/(4.3*E34402[I]);
/*missing value*/ if -4<E34402[I]<0 then MONTH[I]=E34402[I];

/*received compensation*/
/*with overtime*/
if (E33500[I]>=0 and E34428[I]>0) then MONTH[I]=E33500[I]/(4.3*E34428[I]);
if (E33500[I]=-2 and E33600[I]>=0 and E34428[I]>0) then MONTH[I]=E33600[I]/(4.3*E34428[I]);
if E34400[I]>=0 and E34428[I]>0 then MONTH[I]=E34400[I]/(4.3*E34428[I]);
/*missing values*/
if -4<E34428[I]<0 then MONTH[I]=E34428[I];
if (E33500[I]>=0 or E33600[I]>=0 or E34400[I]>=0) and E34428[I]=0 then MONTH[I]=-3;

/*without overtime*/
if (E33500[I]>=0 and E34402[I]>0) then MONTH[I]=E33500[I]/(4.3*E34402[I]);
if (E33500[I]=-2 and E33600[I]>=0 and E34402[I]>0) then MONTH[I]=E33600[I]/(4.3*E34402[I]);
if E34400[I]>=0 and E34402[I]>0 then MONTH[I]=E34400[I]/(4.3*E34402[I]);
/*missing values*/
if -4<E34402[I]<0 then MONTH[I]=E34402[I];
if (E33400[I]>=0 or E33500[I]>=0 or E33600[I]>=0 or E34400[I]>=0) and E34402[I]=0 then MONTH[I]=-3;

/*if still paid hourly...*/
if E36200[I]>=0 then MONTH[I]=E36200[I];
if E36100[I]>=0 then MONTH[I]=E36100[I];
end;
```

```

if E83100[I]=5 then do;

/*no compensation*/
  if (E97300[I]>=0 and E98402[I]>0) then MONTH[I]=E97300[I]/(4.3*E98402[I]);
  if (E97300[I]=-2 and E97500[I]>=0 and E98402[I]>0) then MONTH[I]=E97500[I]/(4.3*E98402[I]);
  if E98300[I]>=0 and E98402[I]>0 then MONTH[I]=E98300[I]/(4.3*E98402[I]);
  /*missing value*/ if -4<E98402[I]<0 then MONTH[I]=E98402[I];

/*received compensation*/
/*with overtime*/
  if (E97400[I]>=0 and E98429[I]>0) then MONTH[I]=E97400[I]/(4.3*E98429[I]);
  if (E97400[I]=-2 and E97500[I]>=0 and E98429[I]>0) then MONTH[I]=E97500[I]/(4.3*E98429[I]);
  if E98300[I]>=0 and E98429[I]>0 then MONTH[I]=E98300[I]/(4.3*E98429[I]);
  /*missing values*/
  if -4<E98429[I]<0 then MONTH[I]=E98429[I];
  if (E97400[I]>=0 or E97500[I]>=0 or E98300[I]>=0) and E98429[I]=0 then MONTH[I]=-3;

/*without overtime*/
  if (E97400[I]>=0 and E98402[I]>0) then MONTH[I]=E97400[I]/(4.3*E98402[I]);
  if (E97400[I]=-2 and E97500[I]>=0 and E98402[I]>0) then MONTH[I]=E97500[I]/(4.3*E98402[I]);
  if E98300[I]>=0 and E98402[I]>0 then MONTH[I]=E98300[I]/(4.3*E98402[I]);
  /*missing values*/
  if -4<E98402[I]<0 then MONTH[I]=E98402[I];
  if (E97300[I]>=0 or E97400[I]>=0 or E97500[I]>=0 or E98300[I]>=0) and E98402[I]=0 then MONTH[I]=-3;

/*if still paid hourly...*/
  if E100100[I]>=0 then MONTH[I]=E100100[I];
  if E100000[I]>=0 then MONTH[I]=E100000[I];
end;
end;

/*Respondents reporting an ANNUAL wage*/
ANNUAL01=-4;           ANNUAL02=-4;           ANNUAL03=-4;
ANNUAL04=-4;           ANNUAL05=-4;           ANNUAL06=-4;
ANNUAL07=-4;           ANNUAL08=-4;           ANNUAL09=-4;

array ANNUAL ANNUAL01 ANNUAL02 ANNUAL03 ANNUAL04 ANNUAL05 ANNUAL06
      ANNUAL07 ANNUAL08 ANNUAL09;

do I=1 to 9; /* annual start wage divided by the number of hours worked per week*/
if E19200[I]=6 then do;

/*no compensation*/
  if (E33400[I]>=0 and E34402[I]>0 and E35600[I]>0) then ANNUAL[I]=E33400[I]/(E35600[I]*E34402[I]);
  if (E33400[I]=-2 and E33600[I]>=0 and E34402[I]>0 and E35600[I]>0) then
    ANNUAL[I]=E33600[I]/(E35600[I]*E34402[I]);
  if E34400[I]>=0 and E34402[I]>0 and E35600[I]>0 then ANNUAL[I]=E34400[I]/(E35600[I]*E34402[I]);
  /*missing value*/ if -4<E34402[I]<0 then ANNUAL[I]=E34402[I];

/*received compensation*/
/*with overtime*/
  if (E33500[I]>=0 and E34428[I]>0 and E35600[I]>0) then ANNUAL[I]=E33500[I]/(E35600[I]*E34428[I]);
  if (E33500[I]=-2 and E33600[I]>=0 and E34428[I]>0 and E35600[I]>0) then
    ANNUAL[I]=E33600[I]/(E35600[I]*E34428[I]);
  if E34400[I]>=0 and E34428[I]>0 and E35600[I]>0 then ANNUAL[I]=E34400[I]/(E35600[I]*E34428[I]);
  /*missing values*/
  if -4<E34428[I]<0 then ANNUAL[I]=E34428[I];

```

Appendix 2: Employment Variable Creation

```
if (E33500[I]>=0 or E33600[I]>=0 or E34400[I]>=0) and E34428[I]=0 then ANNUAL[I]=-3;  
  
/*without overtime*/  
if (E33500[I]>=0 and E34402[I]>0 and E35600[I]>0) then ANNUAL[I]=E33500[I]/(E35600[I]*E34402[I]);  
if (E33500[I]=-2 and E33600[I]>=0 and E34402[I]>0 and E35600[I]>0) then  
    ANNUAL[I]=E33600[I]/(E35600[I]*E34402[I]);  
if E34400[I]>=0 and E34402[I]>0 and E35600[I]>0 then ANNUAL[I]=E34400[I]/(E35600[I]*E34402[I]);  
/*missing values*/  
if -4<E34402[I]<0 then ANNUAL[I]=E34402[I];  
if (E33400[I]>=0 or E33500[I]>=0 or E33600[I]>=0 or E34400[I]>=0) and E34402[I]=0 then ANNUAL[I]=-3;  
if -4<E35600[I]<0 then ANNUAL[I]=E35600[I];  
if E35600[I]=0 then ANNUAL[I]=-3;  
  
/*if still paid hourly...*/  
if E36200[I]>=0 then ANNUAL[I]=E36200[I];  
if E36100[I]>=0 then ANNUAL[I]=E36100[I];  
end;  
  
if E83100[I]=6 then do;  
  
/*no compensation*/  
if (E97300[I]>=0 and E98402[I]>0 and E99500[I]>0) then ANNUAL[I]=E97300[I]/(E99500[I]*E98402[I]);  
if (E97300[I]=-2 and E97500[I]>=0 and E98402[I]>0 and E99500[I]>0) then  
    ANNUAL[I]=E97500[I]/(E99500[I]*E98402[I]);  
if E98300[I]>=0 and E98402[I]>0 and E99500[I]>0 then ANNUAL[I]=E98300[I]/(E99500[I]*E98402[I]);  
/*missing value*/ if -4<E98402[I]<0 then ANNUAL[I]=E98402[I];  
  
/*received compensation*/  
/*with overtime*/  
if (E97400[I]>=0 and E98429[I]>0 and E99500[I]>0) then ANNUAL[I]=E97400[I]/(E99500[I]*E98429[I]);  
if (E97400[I]=-2 and E97500[I]>=0 and E98429[I]>0 and E99500[I]>0) then  
    ANNUAL[I]=E97500[I]/(E99500[I]*E98429[I]);  
if E98300[I]>=0 and E98429[I]>0 and E99500[I]>0 then ANNUAL[I]=E98300[I]/(E99500[I]*E98429[I]);  
/*missing values*/  
if -4<E99500[I]<0 then ANNUAL[I]=E99500[I];  
if -4<E98429[I]<0 then ANNUAL[I]=E98429[I];  
if (E97400[I]>=0 or E97500[I]>=0 or E98300[I]>=0) and E98429[I]=0 then ANNUAL[I]=-3;  
  
/* without overtime*/  
if (E97400[I]>=0 and E98402[I]>0 and E99500[I]>0) then ANNUAL[I]=E97400[I]/(E99500[I]*E98402[I]);  
if (E97400[I]=-2 and E97500[I]>=0 and E98402[I]>0 and E99500[I]>0) then  
    ANNUAL[I]=E97500[I]/(E99500[I]*E98402[I]);  
if E98300[I]>=0 and E98402[I]>0 and E99500[I]>0 then ANNUAL[I]=E98300[I]/(E99500[I]*E98402[I]);  
/*missing values*/  
if -4<E99500[I]<0 then ANNUAL[I]=E99500[I];  
if -4<E98402[I]<0 then ANNUAL[I]=E98402[I];  
if (E97300[I]>=0 or E97400[I]>=0 or E97500[I]>=0 or E98300[I]>=0) and E98402[I]=0 then ANNUAL[I]=-3;  
if -4<E99500[I]<0 then ANNUAL[I]=E99500[I];  
if E99500[I]=0 then ANNUAL[I]=-3;  
  
/*if still paid hourly...*/  
if E100100[I]>=0 then ANNUAL[I]=E100100[I];  
if E100000[I]>=0 then ANNUAL[I]=E100000[I];  
end;  
end;  
  
/*Respondents reporting a SEMIMONTHLY wage*/
```

Appendix 2: Employment Variable Creation

```

SEMIM01=-4;           SEMIM02=-4;           SEMIM03=-4;
SEMIM04=-4;           SEMIM05=-4;           SEMIM06=-4;
SEMIM07=-4;           SEMIM08=-4;           SEMIM09=-4;

array SEMIM SEMIM01 SEMIM02 SEMIM03 SEMIM04 SEMIM05 SEMIM06 SEMIM07 SEMIM08
      SEMIM09;

do I=1 to 9; /* semim start wage divided by the number of hours worked per week*/
if E19200[I]=8 then do;

/*no compensation*/
if (E33400[I]>=0 and E34402[I]>0) then SEMIM[I]=E33400[I]/(2.15*E34402[I]);
if (E33400[I]=-2 and E33600[I]>=0 and E34402[I]>0) then SEMIM[I]=E33600[I]/(2.15*E34402[I]);
if E34400[I]>=0 and E34402[I]>0 then SEMIM[I]=E34400[I]/(2.15*E34402[I]);
/*missing value*/ if -4<E34402[I]<0 then SEMIM[I]=E34402[I];

/*received compensation*/
/*with overtime*/
if (E33500[I]>=0 and E34428[I]>0) then SEMIM[I]=E33500[I]/(2.15*E34428[I]);
if (E33500[I]=-2 and E33600[I]>=0 and E34428[I]>0) then SEMIM[I]=E33600[I]/(2.15*E34428[I]);
if E34400[I]>=0 and E34428[I]>0 then SEMIM[I]=E34400[I]/(2.15*E34428[I]);
/*missing values*/
if -4<E34428[I]<0 then SEMIM[I]=E34428[I];
if (E33500[I]>=0 or E33600[I]>=0 or E34400[I]>=0) and E34428[I]=0 then SEMIM[I]=-3;

/*without overtime*/
if (E33500[I]>=0 and E34402[I]>0) then SEMIM[I]=E33500[I]/(2.15*E34402[I]);
if (E33500[I]=-2 and E33600[I]>=0 and E34402[I]>0) then SEMIM[I]=E33600[I]/(2.15*E34402[I]);
if E34400[I]>=0 and E34402[I]>0 then SEMIM[I]=E34400[I]/(2.15*E34402[I]);
/*missing values*/
if -4<E34402[I]<0 then SEMIM[I]=E34402[I];
if (E33400[I]>=0 or E33500[I]>=0 or E33600[I]>=0 or E34400[I]>=0) and E34402[I]=0 then SEMIM[I]=-3;

/*if still paid hourly...*/
if E36200[I]>=0 then SEMIM[I]=E36200[I];
if E36100[I]>=0 then SEMIM[I]=E36100[I];
end;

if E83100[I]=8 then do;

/*no compensation*/
if (E97300[I]>=0 and E98402[I]>0) then SEMIM[I]=E97300[I]/(2.15*E98402[I]);
if (E97300[I]=-2 and E97500[I]>=0 and E98402[I]>0) then SEMIM[I]=E97500[I]/(2.15*E98402[I]);
if E98300[I]>=0 and E98402[I]>0 then SEMIM[I]=E98300[I]/(2.15*E98402[I]);
/*missing value*/ if -4<E98402[I]<0 then SEMIM[I]=E98402[I];

/*received compensation*/
/*with overtime*/
if (E97400[I]>=0 and E98429[I]>0) then SEMIM[I]=E97400[I]/(2.15*E98429[I]);
if (E97400[I]=-2 and E97500[I]>=0 and E98429[I]>0) then SEMIM[I]=E97500[I]/(2.15*E98429[I]);
if E98300[I]>=0 and E98429[I]>0 then SEMIM[I]=E98300[I]/(2.15*E98429[I]);
/*missing values*/
if -4<E98429[I]<0 then SEMIM[I]=E98429[I];
if (E97400[I]>=0 or E97500[I]>=0 or E98300[I]>=0) and E98429[I]=0 then SEMIM[I]=-3;

/*without overtime*/
if (E97400[I]>=0 and E98402[I]>0) then SEMIM[I]=E97400[I]/(2.15*E98402[I]);

```

Appendix 2: Employment Variable Creation

```
if (E97400[I]=-2 and E97500[I]>=0 and E98402[I]>0) then SEMIM[I]=E97500[I]/(2.15*E98402[I]);  
if E98300[I]>=0 and E98402[I]>0 then SEMIM[I]=E98300[I]/(2.15*E98402[I]);  
/*missing values*/  
if -4<E98402[I]<0 then SEMIM[I]=E98402[I];  
if (E97300[I]>=0 or E97400[I]>=0 or E97500[I]>=0 or E98300[I]>=0) and E98402[I]=0 then SEMIM[I]=-3;  
  
/*if still paid hourly...*/  
if E100100[I]>=0 then SEMIM[I]=E100100[I];  
if E100000[I]>=0 then SEMIM[I]=E100000[I];  
end;  
end;  
  
/*Respondents reporting an OTHER wage*/  
OTHERF1=0; OTHERF2=0; OTHERF3=0;  
OTHERF4=0; OTHERF5=0; OTHERF6=0;  
OTHERF7=0; OTHERF8=0; OTHERF9=0;  
  
array OTHERF OTHERF1-OTHERF9;  
do I=1 to 9;  
  
if E19200[I] in (0,7,12,13,15,16,17,14,99,21,22,23,24,25,26,28) or E83100[I] in  
                 (0,7,12,13,15,16,17,14,99,21,22,23,24,25,26,28) then OTHERF[I]=OTHERF[I]+1;  
end;  
  
***** Part 2: Create the Hourly Rate of Pay based on the start wage *****  
  
HRWG01=-4; HRWG02=-4; HRWG03=-4;  
HRWG04=-4; HRWG05=-4; HRWG06=-4;  
HRWG07=-4; HRWG08=-4; HRWG09=-4;  
  
array HRWG HRWG01 HRWG02 HRWG03 HRWG04 HRWG05 HRWG06 HRWG07 HRWG08 HRWG09;  
  
do I=1 to 9; /* Report hourly wage to be 0 for the family business*/  
if E19200[I] in (9,14) or E83100[I] in (9,14) then HRWG[I]=0;  
end;  
  
do I=1 to 9; /* report hourly rate -1 or -2 if amount is -1 or -2*/  
if E19200[I] IN (2,3,4,5,6,7,8,0,12,13,15,16,17,99,21,22,23,24,25,25,28,-1,-2) then do;  
  if E33400[I]=-1 or E33500[I]=-1 then HRWG[I]=-1;  
  if (E33400[I]=-2 and E33600[I]=-2) or (E33500[I]=-2 and E33600[I]=-2) then HRWG[I]=-2;  
  if E33400[I]=-3 or E33500[I]=-3 then HRWG[I]=-3;  
end;  
  
if E83100[I] in (2,3,4,5,6,7,8,0,12,13,15,16,17,99,21,22,23,24,25,25,28,-1,-2) then do;  
  if E97300[I]=-1 or E97400[I]=-1 then HRWG[I]=-1;  
  if (E97300[I]=-2 and E97500[I]=-2) or (E97400[I]=-2 and E97500[I]=-2) then HRWG[I]=-2;  
  if E97300[I]=-3 or E97400[I]=-3 then HRWG[I]=-3;  
end;  
end;  
  
do I=1 to 9; /* report hourly rate -3 if no hours reported*/  
if E212001[I]=1 and E34428[I]=-4 then HRWG[I]=-3;  
end;  
  
do I=1 to 9;  
  
if ANNUAL[I] ge 0 then HRWG[I]=ANNUAL[I];   f MONTH[I] ge 0 then HRWG[I]=MONTH[I];
```

Appendix 2: Employment Variable Creation

```

if BIWKLY[I] ge 0 then HRWG[I]=BIWKLY[I];      if WEEKLY[I] ge 0 then HRWG[I]=WEEKLY[I];
if DAILY[I] ge 0 then HRWG[I]=DAILY[I];        if HRWAGE[I] ge 0 then HRWG[I]=HRWAGE[I];
if SEMIM[I] ge 0 then HRWG[I]=SEMIM[I];
if HRWAGE[I] eq -1 or DAILY[I]=-1 or WEEKLY[I] eq -1 or BIWKLY[I] eq -1 or MONTH[I] eq -1 or
    ANNUAL[I] eq -1 or SEMIM[I]=-1 then HRWG[I]=-1;
if HRWAGE[I] eq -2 or DAILY[I]=-2 or WEEKLY[I] eq -2 or BIWKLY[I] eq -2 or MONTH[I] eq -2 or
    ANNUAL[I] eq -2 or SEMIM[I]=-2 then HRWG[I]=-2;
if HRWAGE[I] eq -3 or DAILY[I]=-3 or WEEKLY[I] eq -3 or BIWKLY[I] eq -3 or MONTH[I] eq -3 or
    ANNUAL[I] eq -3 or SEMIM[I]=-3 then HRWG[I]=-3;
end;

```

*******Part 3: Report the corrected wage if the correction is made *******

```

do I=1 to 9;
if E226041[I]=1 and E226042[I]=0 then do; /* rate incorrect but hours correct*/
    /* no overtime, use E34402 for hours */
    if E22609[I]=1 and E22626[I]>=0 then HRWG[I]=E22626[I];
    if E22609[I]=2 and E34402[I]>0 and E34402B[I]>0 and E22626[I]>=0 then
        HRWG[I]=E22626[I]*E34402B[I]/E34402[I];
    if E22609[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and E34402[I]>0 and E22626[I]>=0 then
        HRWG[I]=E22626[I]/E34402[I];
    if E22609[I]=4 and E34402[I]>0 and E22626[I]>=0 then HRWG[I]=E22626[I]/(2*E34402[I]);
    if E22609[I]=5 and E34402[I]>0 and E22626[I]>=0 then HRWG[I]=E22626[I]/(4.3*E34402[I]);
    if E22609[I]=6 and E34402[I]>0 and E35600[I]>0 and E22626[I]>=0 then
        HRWG[I]=E22626[I]/(E35600[I]*E34402[I]);
    if E22609[I]=8 and E34402[I]>0 and E22626[I]>=0 then HRWG[I]=E22626[I]/(2.15*E34402[I]);
    if E22609[I] in (9,14) then HRWG[I]=0;
    if E22609[I]=2 and E34402B[I] le 0 then HRWG[I]=-3;
    if E22609[I]=2 and -4<E34402B[I]<0 then HRWG[I]=E34402B[I];
    if E22609[I]=6 and E35600[I] le 0 then HRWG[I]=-3;
    if E22609[I]=6 and -4<E35600[I]<0 then HRWG[I]=E35600[I];
    if E22609[I] in (1,2,3,4,5,6,7,8,0,12,13,15,16,17,99,21,22,223,24,25,25,28,-1,-2) and -4<E22626[I]<0 then
        HRWG[I]=E22626[I];
    if E22609[I] in (2,3,4,5,6,7,8,0,12,13,15,16,17,99,21,22,223,24,25,25,28,-2) and E34402[I]=0 then
        HRWG[I]=-3;
    if E22609[I] in (2,3,4,5,6,7,8,0,12,13,15,16,17,99,21,22,223,24,25,25,28,-2) and -4<E34402[I]<0 then
        HRWG[I]=E34402[I];

/* overtime, use E34428 for hours*/
if E22609[I]=1 and E22626[I]>=0 then HRWG[I]=E22626[I];
if E22609[I]=2 and E34428[I]>0 and E34430[I]>0 and E22626[I]>=0 then
    HRWG[I]=E22626[I]*E34430[I]/E34428[I];
if E22609[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and E34428[I]>0 and E22626[I]>=0 then
    HRWG[I]=E22626[I]/E34428[I];
if E22609[I]=4 and E34428[I]>0 and E22626[I]>=0 then HRWG[I]=E22626[I]/(2*E34428[I]);
if E22609[I]=5 and E34428[I]>0 and E22626[I]>=0 then HRWG[I]=E22626[I]/(4.3*E34428[I]);
if E22609[I]=6 and E34428[I]>0 and E35600[I]>0 and E22626[I]>=0 then
    HRWG[I]=E22626[I]/(E35600[I]*E34428[I]);
if E22609[I]=8 and E34428[I]>0 and E22626[I]>=0 then HRWG[I]=E22626[I]/(2.15*E34428[I]);
if E22609[I] in (9,14) then HRWG[I]=0;
if E22609[I]=2 and E34430[I] le 0 then HRWG[I]=-3;
if E22609[I]=2 and -4<E34430[I]<0 then HRWG[I]=E34430[I];
if E22609[I] in (1,2,3,4,5,6,7,8,0,12,13,15,16,17,99,21,22,223,24,25,25,28,-1,-2) and -4<E22626[I]<0 then
    HRWG[I]=E22626[I];
if E22609[I] in (2,3,4,5,6,7,8,0,12,13,15,16,17,99,21,22,223,24,25,25,28,-2) and E34428[I]=0 then
    HRWG[I]=-3;

```

Appendix 2: Employment Variable Creation

```
if E22609[I] in (2,3,4,5,6,7,8,0,12,13,15,16,17,99,21,22,223,24,25,25,28,-2) and -4<E34428[I]<0 then
    HRWG[I]=E34428[I];
end;

if E226041[I]=0 and E226042[I]=1 then do; /*rate correct but hours incorrect*/
    if E19200[I] ne 1 and E22610[I]>0 and E34402[I]>0 and HRWG[I]>=0 then
        HRWG[I]=HRWG[I]*E34402[I]/E22610[I];
    if E19200[I] ne 1 and E22610[I]>0 and E34428[I]>0 and HRWG[I]>=0 then
        HRWG[I]=HRWG[I]*E34428[I]/E22610[I];
    if E19200[I] ne 1 and E22610[I]=0 then HRWG[I]=-3;
    if E19200[I] ne 1 and -4<E22610[I]<0 then HRWG[I]=E22610[I];
end;

if E226041[I]=1 and E226042[I]=1 then do; /* neither rate nor hours is correct*/
    if E22609[I]=1 and E22626[I]>=0 then HRWG[I]=E22626[I];
    if E22609[I]=2 and E22610[I]>0 and E34402B[I]>0 and E22626[I]>=0 then
        HRWG[I]=E22626[I]*E34402B[I]/E22610[I];
    if E22609[I]=2 and E22610[I]>0 and E34430[I]>0 and E22626[I]>=0 then
        HRWG[I]=E22626[I]*E34430[I]/E22610[I];
    if E22609[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and E22610[I]>0 and E22626[I]>=0 then
        HRWG[I]=E22626[I]/E22610[I];
    if E22609[I]=4 and E22610[I]>0 and E22626[I]>=0 then HRWG[I]=E22626[I]/(2*E22610[I]);
    if E22609[I]=5 and E22610[I]>0 and E22626[I]>=0 then HRWG[I]=E22626[I]/(4.3*E22610[I]);
    if E22609[I]=6 and E22610[I]>0 and E35600[I]>0 and E22626[I]>=0 then
        HRWG[I]=E22626[I]/(E35600[I]*E22610[I]);
    if E22609[I]=8 and E22610[I]>0 and E22626[I]>=0 then HRWG[I]=E22626[I]/(2.15*E22610[I]);
    if E22609[I] in (9,14) then HRWG[I]=0;
    if E22609[I]=2 and E34430[I] le 0 then HRWG[I]=-3;
    if E22609[I]=2 and E34402B[I] le 0 then HRWG[I]=-3;
    if E22609[I]=2 and -4<E34430[I]<0 then HRWG[I]=E34430[I];
    if E22609[I]=2 and -4<E34402B[I]<0 then HRWG[I]=E34402B[I];
    if E22609[I]=6 and E35600[I] le 0 then HRWG[I]=-3;
    if E22609[I]=6 and -4<E35600[I]<0 then HRWG[I]=E35600[I];
    if E22609[I] in (1,2,3,4,5,6,7,8,0,12,13,15,16,17,99,21,22,23,24,25,26,28,-1,-2) and -4<E22626[I]<0 then
        HRWG[I]=E22626[I];
    if E22609[I] in (2,3,4,5,6,7,8,0,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and E22610[I]=0 then HRWG[I]=-3;
    if E22609[I] in (2,3,4,5,6,7,8,0,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and -4<E22610[I]<0 then
        HRWG[I]=E22610[I];
end;
end;

do I=1 to 9;
if E102251[I]=1 and E102252[I]=0 then do; /* rate incorrect but hours correct*/
    /* no overtime, use E98402 for hours */
    if E100230[I]=1 and E100248[I]>=0 then HRWG[I]=E100248[I];
    if E100230[I]=2 and E98402D[I]>0 and E98402D[I]>0 and E100248[I]>=0 then
        HRWG[I]=E100248[I]*E98402D[I]/E98402D[I];
    if E100230[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and E98402D[I]>0 and E100248[I]>=0 then
        HRWG[I]=E100248[I]/E98402D[I];
    if E100230[I]=4 and E98402D[I]>0 and E100248[I]>=0 then HRWG[I]=E100248[I]/(2*E98402D[I]);
    if E100230[I]=5 and E98402D[I]>0 and E100248[I]>=0 then HRWG[I]=E100248[I]/(4.3*E98402D[I]);
    if E100230[I]=6 and E98402D[I]>0 and E99500[I]>0 and E100248[I]>=0 then
        HRWG[I]=E100248[I]/(E99500[I]*E98402D[I]);
    if E100230[I]=8 and E98402D[I]>0 and E100248[I]>=0 then HRWG[I]=E100248[I]/(2.15*E98402D[I]);
    if E100230[I] in (9,14) then HRWG[I]=0;
    if E100230[I]=2 and E98402D[I] le 0 then HRWG[I]=-3;
```

```

if E100230[I]=2 and -4<E98402D[I]<0 then HRWG[I]=E98402D[I];
if E100230[I]=6 and E99500[I] le 0 then HRWG[I]=-3;
if E100230[I]=6 and -4<E99500[I]<0 then HRWG[I]=E99500[I];
if E100230[I] in (1,2,3,4,5,6,7,8,0,12,13,15,16,17,99,21,22,23,24,25,26,28,-1,-2) and -4<E100248[I]<0 then
    HRWG[I]=E100248[I];
if E100230[I] in (2,3,4,5,6,7,8,0,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and E98402[I]=0 then
    HRWG[I]=-3;
if E100230[I] in (2,3,4,5,6,7,8,0,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and -4<E98402[I]<0 then
    HRWG[I]=E98402[I];

/* overtime, use E98429 for hours*/
if E100230[I]=1 and E100248[I]>=0 then HRWG[I]=E100248[I];
if E100230[I]=2 and E98429[I]>0 and E98429E[I]>0 and E100248[I]>=0 then
    HRWG[I]=E100248[I]*E98429E[I]/E98429[I];
if E100230[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and E98429[I]>0 and E100248[I]>=0 then
    HRWG[I]=E100248[I]/E98429[I];
if E100230[I]=4 and E98429[I]>0 and E100248[I]>=0 then HRWG[I]=E100248[I]/(2*E98429[I]);
if E100230[I]=5 and E98429[I]>0 and E100248[I]>=0 then HRWG[I]=E100248[I]/(4.3*E98429[I]);
if E100230[I]=6 and E98429[I]>0 and E99500[I]>0 and E100248[I]>=0 then
    HRWG[I]=E100248[I]/(E99500[I]*E98429[I]);
if E100230[I]=8 and E98429[I]>0 and E100248[I]>=0 then HRWG[I]=E100248[I]/(2.15*E98429[I]);
if E100230[I] in (9,14) then HRWG[I]=0;
if E100230[I]=2 and E98429E[I] le 0 then HRWG[I]=-3;
if E100230[I]=2 and -4<E98429E[I]<0 then HRWG[I]=E98429E[I];
if E100230[I] in (1,2,3,4,5,6,7,8,0,12,13,15,16,17,99,21,22,23,24,25,26,28,-1,-2) and -4<E100248[I]<0 then
    HRWG[I]=E100248[I];
if E100230[I] in (2,3,4,5,6,7,8,0,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and E98429[I]=0 then
    HRWG[I]=-3;
if E100230[I] in (2,3,4,5,6,7,8,0,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and -4<E98429[I]<0 then
    HRWG[I]=E98429[I];
end;

if E102251[I]=0 and E102252[I]=1 then do; /*rate correct but hours incorrect*/
    if E83100[I] ne 1 and E100231[I]>0 and E98402[I]>0 and HRWG[I]>=0 then
        HRWG[I]=HRWG[I]*E98402[I]/E100231[I];
    if E83100[I] ne 1 and E100231[I]>0 and E98429[I]>0 and HRWG[I]>=0 then
        HRWG[I]=HRWG[I]*E98429[I]/E100231[I];
    if E83100[I] ne 1 and E100231[I]=0 then HRWG[I]=-3;
    if E83100[I] ne 1 and -4<E100231[I]<0 then HRWG[I]=E100231[I];
end;

if E102251[I]=1 and E102252[I]=1 then do; /* neither rate nor hours is correct*/
    if E100230[I]=1 and E100248[I]>=0 then HRWG[I]=E100248[I];
    if E100230[I]=2 and E100231[I]>0 and E98402D[I]>0 and E100248[I]>=0 then
        HRWG[I]=E100248[I]*E98402D[I]/E100231[I];
    if E100230[I]=2 and E100231[I]>0 and E98429E[I]>0 and E100248[I]>=0 then
        HRWG[I]=E100248[I]*E98429E[I]/E100231[I];
    if E100230[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and E100231[I]>0 and E100248[I]>=0 then
        HRWG[I]=E100248[I]/E100231[I];
    if E100230[I]=4 and E100231[I]>0 and E100248[I]>=0 then HRWG[I]=E100248[I]/(2*E100231[I]);
    if E100230[I]=5 and E100231[I]>0 and E100248[I]>=0 then HRWG[I]=E100248[I]/(4.3*E100231[I]);
    if E100230[I]=6 and E100231[I]>0 and E99500[I]>0 and E100248[I]>=0 then
        HRWG[I]=E100248[I]/(E99500[I]*E100231[I]);
    if E100230[I]=8 and E100231[I]>0 and E100248[I]>=0 then HRWG[I]=E100248[I]/(2.15*E100231[I]);
    if E100230[I] in (9,14) then HRWG[I]=0;
    if E100230[I]=2 and E98429E[I] le 0 then HRWG[I]=-3;
    if E100230[I]=2 and E98402D[I] le 0 then HRWG[I]=-3;

```

Appendix 2: Employment Variable Creation

```

if E100230[I]=2 and -4<E98429E[I]<0 then HRWG[I]=E98429E[I];
if E100230[I]=2 and -4<E98402D[I]<0 then HRWG[I]=E98402D[I];
if E100230[I]=6 and E99500[I] le 0 then HRWG[I]=-3;
if E100230[I]=6 and -4<E99500[I]<0 then HRWG[I]=E99500[I];
if E100230[I] in (1,2,3,4,5,6,7,8,0,12,13,15,16,17,99,21,22,23,24,25,26,28,-1,-2) and -4<E100248[I]<0 then
    HRWG[I]=E100248[I];
if E100230[I] in (2,3,4,5,6,7,8,0,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and E100231[I]=0 then
    HRWG[I]=-3;
if E100230[I] in (2,3,4,5,6,7,8,0,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and -4<E100231[I]<0 then
    HRWG[I]=E100231[I];
end;
end;

```

******Part 4: Add the start compensation ****/**

```

OT1=-4;          OT2=-4;          OT3=-4;          OT4=-4;          OT5=-4;
OT6=-4;          OT7=-4;          OT8=-4;          OT9=-4;

```

```

OTF1=0;          OTF2=0;          OTF3=0;          OTF4=0;          OTF5=0;
OTF6=0;          OTF7=0;          OTF8=0;          OTF9=0;

```

array OT OT1-OT9;

array OTF OTF1-OTF9;

do I=1 to 9; /* if report HOURLY rate of pay */

```

if E24501[I]>0 and E24502[I]=1 and E24514[I]>=0 then OT[I]=E24514[I];
if E24501[I]>0 and E24502[I]=2 and E24514B[I]>0 and E24514[I]>=0 then
    OT[I]=E24514[I]*E24514B[I]/E24501[I];
if E24501[I]>0 and E24502[I] in (3,7,12,13,15,16,17,21,22,23,24,25,26,28,-2) and E24514[I]>=0 then
    OT[I]=E24514[I]/E24501[I];
if E24501[I]>0 and E24502[I]=4 and E24514[I]>=0 then OT[I]=E24514[I]/(2*E24501[I]);
if E24501[I]>0 and E24502[I]=5 and E24514[I]>=0 then OT[I]=E24514[I]/(4.3*E24501[I]);
if E24501[I]>0 and E24502[I]=6 and E24514[I]>=0 then OT[I]=-3; /* Since the weeks per year is unknown.*/
if E24501[I]>0 and E24502[I]=8 and E24514[I]>=0 then OT[I]=E24514[I]/(2.15*E24501[I]);
if E24501[I]>0 and E24502[I] in (0,7,12,13,14,15,16,17,21,22,23,24,25,26,28,-2) then OTF[I]=OTF[I]+1;
if E24501[I]>0 and E24502[I]=10 and HRWG[I]>=0 then OT[I]=HRWG[I]*1.5;
if E24501[I]>0 and E24502[I]=11 and HRWG[I]>=0 then OT[I]=HRWG[I]*2;
if E24502[I]=14 then OT[I]=0;
/*MISSING VALUE*/;
if -4<E24501[I]<0 then OT[I]=E24501[I];
if -4<E24514[I]<0 then OT[I]=E24514[I];
if E24514[I]>=0 and E24501[I]=0 then OT[I]=-3;
if E24502[I]=2 and E24514B[I]=0 then OT[I]=-3;
if E24502[I]=2 and -4<E24514B[I]<0 then OT[I]=E24514B[I];

```

end;

do I=1 to 9; /*if report payment in OTHER units*/

```

if E34403[I]>0 and E34404[I]=1 and E34413C[I]>=0 then OT[I]=E34413C[I];
if E34403[I]>0 and E34404[I]=2 and E34413E[I]>0 and E34413C[I]>=0 then
    OT[I]=E34413C[I]*E34413E[I]/E34403[I];
if E34403[I]>0 and E34404[I] in (3,7,12,13,15,16,17,21,22,23,24,25,26,28,-2) and E34413C[I]>=0 then
    OT[I]=E34413C[I]/E34403[I];
if E34403[I]>0 and E34404[I]=4 and E34413C[I]>=0 then OT[I]=E34413C[I]/(2*E34403[I]);
if E34403[I]>0 and E34404[I]=5 and E34413C[I]>=0 then OT[I]=E34413C[I]/(4.3*E34403[I]);

```

Appendix 2: Employment Variable Creation

```
if E34403[I]>0 and E34404[I]=6 and E34413C[I]>=0 then OT[I]=-3; /* Since there is no weeks per year. */  
if E34403[I]>0 and E34404[I]=8 and E34413C[I]>=0 then OT[I]=E34413C[I]/(2.15*E34403[I]);  
if E34403[I]>0 and E34404[I] in (0,7,12,13,14,15,16,17,21,22,23,24,25,26,28,-2) then OTF[I]=OTF[I]+1;  
if E34403[I]>0 and E34404[I]=10 and HRWG[I]>=0 then OT[I]=HRWG[I]*1.5;  
if E34403[I]>0 and E34404[I]=11 and HRWG[I]>=0 then OT[I]=HRWG[I]*2;  
if E34404[I]=14 then OT[I]=0;  
/*missing value*/;  
if -4<E34403[I]<0 then OT[I]=E34403[I];  
if -4<E34413C[I]<0 then OT[I]=E34413C[I];  
if E34413C[I]>=0 and E34403[I]=0 then OT[I]=-3;  
if E34404[I]=2 and E34413E[I]=0 then OT[I]=-3;  
if E34404[I]=2 and -4<E34413E[I]<0 then OT[I]=E34413E[I];  
  
end;  
  
do I=1 to 9; /*if report overtime pay later*/  
  
if E88501[I]>0 and E88502[I]=1 and E88512[I]>=0 then OT[I]=E88512[I];  
if E88501[I]>0 and E88502[I]=2 and E88512B[I]>0 and E88512[I]>=0 then  
    OT[I]=E88512[I]*E88512B[I]/E88501[I];  
if E88501[I]>0 and E88502[I] in (3,7,12,13,15,16,17,21,22,23,24,25,26,28,-2) and E88512[I]>=0 then  
    OT[I]=E88512[I]/E88501[I];  
if E88501[I]>0 and E88502[I]=4 and E88512[I]>=0 then OT[I]=E88512[I]/(2*E88501[I]);  
if E88501[I]>0 and E88502[I]=5 and E88512[I]>=0 then OT[I]=E88512[I]/(4.3*E88501[I]);  
if E88501[I]>0 and E88502[I]=6 and E88512[I]>=0 then OT[I]=-3; /* Since there is no weeks per year. */  
if E88501[I]>0 and E88502[I]=8 and E88512[I]>=0 then OT[I]=E88512[I]/(2.15*E88501[I]);  
if E88501[I]>0 and E88502[I] in (0,7,12,13,14,15,16,17,21,22,23,24,25,26,28,-2) then OTF[I]=OTF[I]+1;  
if E88501[I]>0 and E88502[I]=10 and HRWG[I]>=0 then OT[I]=HRWG[I]*1.5;  
if E88501[I]>0 and E88502[I]=11 and HRWG[I]>=0 then OT[I]=HRWG[I]*2;  
if E88502[I]=14 then OT[I]=0;  
/*missing value*/;  
if -4<E88501[I]<0 then OT[I]=E88501[I];  
if -4<E88512[I]<0 then OT[I]=E88512[I];  
if E88512[I]>=0 and E88501[I]=0 then OT[I]=-3;  
  
end;  
  
do I=1 to 9;  
  
if E98403[I]>0 and E98404[I]=1 and E98414[I]>=0 then OT[I]=E98414[I];  
if E98403[I]>0 and E98404[I]=2 and E98414B[I]>0 and E98414[I]>=0 then  
    OT[I]=E98414[I]*E98414B[I]/E98403[I];  
if E98403[I]>0 and E98404[I] in (3,7,12,13,15,16,17,21,22,23,24,25,26,28,-2) and E98414[I]>=0 then  
    OT[I]=E98414[I]/E98403[I];  
if E98403[I]>0 and E98404[I]=4 and E98414[I]>=0 then OT[I]=E98414[I]/(2*E98403[I]);  
if E98403[I]>0 and E98404[I]=5 and E98414[I]>=0 then OT[I]=E98414[I]/(4.3*E98403[I]);  
if E98403[I]>0 and E98404[I]=6 and E98414[I]>=0 then OT[I]=-3; /* Since there is no weeks per year. */  
if E98403[I]>0 and E98404[I]=8 and E98414[I]>=0 then OT[I]=E98414[I]/(2.15*E98403[I]);  
if E98403[I]>0 and E98404[I] in (0,7,12,13,14,15,16,17,21,22,23,24,25,26,28,-2) then OTF[I]=OTF[I]+1;  
if E98403[I]>0 and E98404[I]=10 and HRWG[I]>=0 then OT[I]=HRWG[I]*1.5;  
if E98403[I]>0 and E98404[I]=11 and HRWG[I]>=0 then OT[I]=HRWG[I]*2;  
if E98404[I]=14 then OT[I]=0;  
/*MISSING VALUE*/;  
if -4<E98403[I]<0 then OT[I]=E98403[I];  
if -4<E98414[I]<0 then OT[I]=E98414[I];  
if E98414[I]>=0 and E98403[I]=0 then OT[I]=-3;
```

Appendix 2: Employment Variable Creation

end;

do I=1 to 9; /* report the corrected overtime payment if the correction is made */

if E226044[I]=1 and E226043[I]=0 then do; /*rate incorrect but hours correct*/

 if E19200[I]=1 then do; /* report hourly payrate*/

 if E22612[I]=1 and E22627[I]>=0 then OT[I]=E22627[I];

 if E22612[I]=2 and E22627[I]>=0 and E24514B[I]>0 and E24501[I]>0 then

 OT[I]=E22627[I]*E24514B[I]/E24501[I];

 if E22612[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and E22627[I]>=0 and E24501[I]>0 then

 OT[I]=E22627[I]/E24501[I];

 if E22612[I]=4 and E22627[I]>=0 and E24501[I]>0 then OT[I]=E22627[I]/(2*E24501[I]);

 if E22612[I]=5 and E22627[I]>=0 and E24501[I]>0 then OT[I]=E22627[I]/(4.3*E24501[I]);

 if E22612[I]=6 and E22627[I]>=0 and E24501[I]>0 then OT[I]=-3; /* Since there is no weeks per year. */

 if E22612[I]=8 and E22627[I]>=0 and E24501[I]>0 then OT[I]=E22627[I]/(2.15*E24501[I]);

 if E22612[I] in (9,14) then OT[I]=0;

 if E24501[I]=0 then OT[I]=-3;

 if -4<E24501[I]<0 then OT[I]=E24501[I];

 if -4<E22627[I]<0 then OT[I]=E22627[I];

 if E22612[I]=2 and E24514B[I]<0 then OT[I]=-3;

 if E22612[I]=2 and -4<E24514B[I]<0 then OT[I]=E24514B[I];

 end;

 if E19200[I] ne 1 then do; /*report non-hourly payrate*/

 if E22612[I]=1 and E22627[I]>=0 then OT[I]=E22627[I];

 if E22612[I]=2 and E22627[I]>=0 and E34413E[I]>0 and E34403[I]>0 then

 OT[I]=E22627[I]*E34413E[I]/E34403[I];

 if E22612[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and E22627[I]>=0 and E34403[I]>0 then

 OT[I]=E22627[I]/E34403[I];

 if E22612[I]=4 and E22627[I]>=0 and E34403[I]>0 then OT[I]=E22627[I]/(2*E34403[I]);

 if E22612[I]=5 and E22627[I]>=0 and E34403[I]>0 then OT[I]=E22627[I]/(4.3*E34403[I]);

 if E22612[I]=6 and E22627[I]>=0 and E34403[I]>0 then OT[I]=-3; /* Since there is no weeks per year.*/

 if E22612[I]=8 and E22627[I]>=0 and E34403[I]>0 then OT[I]=E22627[I]/(2.15*E34403[I]);

 if E22612[I] in (9,14) then OT[I]=0;

 if E34403[I]=0 then OT[I]=-3;

 if -4<E34403[I]<0 then OT[I]=E34403[I];

 if -4<E22627[I]<0 then OT[I]=E22627[I];

 if E22612[I]=2 and E34413E[I]<0 then OT[I]=-3;

 if E22612[I]=2 and -4<E34413E[I]<0 then OT[I]=E34413E[I];

 end;

end;

if E226044[I]=0 and E226043[I]=1 then do; /*rate correct but hours incorrect*/

 if E24502[I] ne 1 and OT[I]>=0 and E24501[I]>0 and E22611[I]>0 then OT[I]=OT[I]*E24501[I]/E22611[I];

 if E34404[I] ne 1 and OT[I]>=0 and E34403[I]>0 and E22611[I]>0 then OT[I]=OT[I]*E34403[I]/E22611[I];

 if E24502[I] ne 1 and E34404[I] ne 1 and E22611[I]=0 then OT[I]=-3;

 if E24502[I] ne 1 and E34404[I] ne 1 and -4<E22611[I]<0 then OT[I]=E22611[I];

end;

if E226044[I]=1 and E226043[I]=1 then do; /* neither rate nor the hours is correct*/

 if E22612[I]=1 and E22627[I]>=0 then OT[I]=E22627[I];

 if E22612[I]=2 and E22627[I]>=0 and E24514B[I]>0 and E22611[I]>0 then

 OT[I]=E22627[I]*E24514B[I]/E22611[I];

 if E22612[I]=2 and E22627[I]>=0 and E34413E[I]>0 and E22611[I]>0 then

 OT[I]=E22627[I]*E34413E[I]/E22611[I];

 if E22612[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and E22627[I]>=0 and E22611[I]>0 then

 OT[I]=E22627[I]/E22611[I];

 if E22612[I]=4 and E22627[I]>=0 and E22611[I]>0 then OT[I]=E22627[I]/(2*E22611[I]);

```

if E22612[I]=5 and E22627[I]>=0 and E22611[I]>0 then OT[I]=E22627[I]/(4.3*E22611[I]);
if E22612[I]=6 and E22627[I]>=0 and E22611[I]>0 then OT[I]=-3; /* Since there is no weeks per year. */
if E22612[I]=8 and E22627[I]>=0 and E22611[I]>0 then OT[I]=E22627[I]/(2.15*E22611[I]);
if E22612[I] in (9,14) then OT[I]=0;
if E22612[I]=2 and E24514B[I] le 0 then OT[I]=-3;
if E22612[I]=2 and E34413E[I] le 0 then OT[I]=-3;
if E22612[I]=2 and -4<E24514B[I]<0 then OT[I]=E24514B[I];
if E22612[I]=2 and -4<E34413E[I]<0 then OT[I]=E34413E[I];
if E22612[I] in (2,3,4,5,6,7,8,0,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and E22611[I]=0 then OT[I]=-3;
if E22612[I] in (2,3,4,5,6,7,8,0,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and -4<E22611[I]<0 then
    OT[I]=E22611[I];
if E22612[I] in (1,2,3,4,5,6,7,8,0,12,13,15,16,17,99,21,22,23,24,25,26,28,-1,-2) and -4<E22627[I]<0 then
    OT[I]=E22627[I];
end;
end;

```

******* Part 5: Amount paid in tips, commissions, bonuses, incentive pay or other *******

```

array OTHPAY1 OTHPAY11 OTHPAY21 OTHPAY31 OTHPAY41 OTHPAY51 OTHPAY61 OTHPAY71
    OTHPAY81 OTHPAY91;
array OTHPAY2 OTHPAY12 OTHPAY22 OTHPAY32 OTHPAY42 OTHPAY52 OTHPAY62 OTHPAY72
    OTHPAY82 OTHPAY92;
array OTHPAY3 OTHPAY13 OTHPAY23 OTHPAY33 OTHPAY43 OTHPAY53 OTHPAY63 OTHPAY73
    OTHPAY83 OTHPAY93;
array OTHPAY4 OTHPAY14 OTHPAY24 OTHPAY34 OTHPAY44 OTHPAY54 OTHPAY64 OTHPAY74
    OTHPAY84 OTHPAY94;
array OTHPAY5 OTHPAY15 OTHPAY25 OTHPAY35 OTHPAY45 OTHPAY55 OTHPAY65 OTHPAY75
    OTHPAY85 OTHPAY95;
array OTHPF1 OTHPF11 OTHPF21 OTHPF31 OTHPF41 OTHPF51 OTHPF61 OTHPF71 OTHPF81
    OTHPF91;
array OTHPF2 OTHPF12 OTHPF22 OTHPF32 OTHPF42 OTHPF52 OTHPF62 OTHPF72 OTHPF82
    OTHPF92;
array OTHPF3 OTHPF13 OTHPF23 OTHPF33 OTHPF43 OTHPF53 OTHPF63 OTHPF73 OTHPF83
    OTHPF93;
array OTHPF4 OTHPF14 OTHPF24 OTHPF34 OTHPF44 OTHPF54 OTHPF64 OTHPF74 OTHPF84
    OTHPF94;
array OTHPF5 OTHPF15 OTHPF25 OTHPF35 OTHPF45 OTHPF55 OTHPF65 OTHPF75 OTHPF85
    OTHPF95;

do I=1 to 9;
    OTHPAY1[I]=-4;          OTHPAY2[I]=-4;          OTHPAY3[I]=-4;
    OTHPAY4[I]=-4;          OTHPAY5[I]=-4;
    OTHPF1[I]=0;            OTHPF2[I]=0;            OTHPF3[I]=0;
    OTHPF4[I]=0;            OTHPF5[I]=0;
end;

/* do it for non-overtime payment*/
***** FOR TIPS *****/
do I=1 to 9; /*with overtime*/
    if E34428[I]>0 and E216001[I]=1 and E225001[I]>=0 then OTHPAY1[I]=E225001[I];
    if E34428[I]>0 and E216001[I]=2 and E34430[I]>0 and E225001[I]>=0 then
        OTHPAY1[I]=(E225001[I]*E34430[I])/E34428[I];
    if E34428[I]>0 and E216001[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and E225001[I]>=0 then
        OTHPAY1[I]=E225001[I]/E34428[I];
    if E34428[I]>0 and E216001[I]=4 and E225001[I]>=0 then OTHPAY1[I]=E225001[I]/(2*E34428[I]);
    if E34428[I]>0 and E216001[I]=5 and E225001[I]>=0 then OTHPAY1[I]=E225001[I]/(4.3*E34428[I]);

```

Appendix 2: Employment Variable Creation

```
if E34428[I]>0 and E216001[I]=6 and E225001[I]>=0 and E35600[I]>0 then
    OTHPAY1[I]=E225001[I]/(E35600[I] *E34428[I]);
if E34428[I]>0 and E216001[I]=8 and E225001[I]>=0 then OTHPAY1[I]=E225001[I]/(2.15*E34428[I]);
/*missing value*/;
if E225001[I]>=0 and -4<E34428[I]<0 then OTHPAY1[I]=E34428[I];
if -4<E225001[I]<0 then OTHPAY1[I]=E225001[I];
if E225001[I]>=0 and E34428[I]=0 then OTHPAY1[I]=-3;
end;

do I=1 to 9; /*without overtime*/
if E34402[I]>0 and E216001[I]=1 and E225001[I]>=0 then OTHPAY1[I]=E225001[I];
if E34402[I]>0 and E216001[I]=2 and E34402B[I]>0 and E225001[I]>=0 then
    OTHPAY1[I]=(E225001[I]*E34402B[I])/E34402[I];
if E34402[I]>0 and E216001[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and E225001[I]>=0 then
    OTHPAY1[I]=E225001[I]/E34402[I];
if E34402[I]>0 and E216001[I]=4 and E225001[I]>=0 then OTHPAY1[I]=E225001[I]/(2*E34402[I]);
if E34402[I]>0 and E216001[I]=5 and E225001[I]>=0 then OTHPAY1[I]=E225001[I]/(4.3*E34402[I]);
if E34402[I]>0 and E216001[I]=6 and E225001[I]>=0 and E35600[I]>0 then
    OTHPAY1[I]=E225001[I]/(E35600[I]*E34402[I]);
if E34402[I]>0 and E216001[I]=8 and E225001[I]>=0 then OTHPAY1[I]=E225001[I]/(2.15*E34402[I]);
if E216001[I] in (9,14) then OTHPAY1[I]=0;
if E216001[I] in (0,7,12,13,15,16,17,14,99,21,22,23,24,25,26,28) then OTHPF1[I]=OTHPF1[I]+1;
/*missing value*/;
if E225001[I]>=0 and -4<E34402[I]<0 then OTHPAY1[I]=E34402[I];
if E225001[I]>=0 and E34402[I]=0 then OTHPAY1[I]=-3;
if E216001[I]=2 and E34402B[I] le 0 and E34430[I] le 0 then OTHPAY1[I]=-3;
if E216001[I]=2 and -4<E34402B[I]<0 then OTHPAY1[I]=E34402B[I];
if E216001[I]=2 and -4<E34430[I]<0 then OTHPAY1[I]=E34430[I];
if E216001[I]=6 and E35600[I] le 0 then OTHPAY1[I]=-3;
if E216001[I]=6 and -4<E35600[I]<0 then OTHPAY1[I]=E35600[I];
end;

/* if tips is corrected in the later part */
do I=1 to 9;
if E226045[I]=1 then do;
if E22613[I]=1 and E22628[I]>=0 then OTHPAY1[I]=E22628[I]; /* with overtime*/
if E22613[I]=2 and E22628[I]>=0 and E34430[I]>0 and E34428[I]>0 then
    OTHPAY1[I]=E22628[I]*E34430[I]/E34428[I];
if E22613[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and E22628[I]>=0 and E34428[I]>0 then
    OTHPAY1[I]=E22628[I]/E34428[I];
if E22613[I]=4 and E22628[I]>=0 and E34428[I]>0 then OTHPAY1[I]=E22628[I]/(2*E34428[I]);
if E22613[I]=5 and E22628[I]>=0 and E34428[I]>0 then OTHPAY1[I]=E22628[I]/(4.3*E34428[I]);
if E22613[I]=6 and E22628[I]>=0 and E34428[I]>0 and E35600[I]>0 then
    OTHPAY1[I]=E22628[I]/(E35600[I]*E34428[I]);
if E22613[I]=8 and E22628[I]>=0 and E34428[I]>0 then OTHPAY1[I]=E22628[I]/(2.15*E34428[I]);
if E22613[I] in (9,14) then OTHPAY1[I]=0;
/*missing value*/
if -4<E34428[I]<0 then OTHPAY1[I]=E34428[I];
if -4<E22628[I]<0 then OTHPAY1[I]=E22628[I];
if E34428[I]=0 then OTHPAY1[I]=-3;

if E22613[I]=1 and E22628[I]>=0 then OTHPAY1[I]=E22628[I]; /* without overtime*/
if E22613[I]=2 and E22628[I]>=0 and E34402B[I]>0 and E34402[I]>0 then
    OTHPAY1[I]=E22628[I]*E34402B[I]/E34402[I];
if E22613[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and E22628[I]>=0 and E34402B[I]>0 then
    OTHPAY1[I]=E22628[I]/E34402B[I];
if E22613[I]=4 and E22628[I]>=0 and E34402B[I]>0 then OTHPAY1[I]=E22628[I]/(2*E34402B[I]);
```

```

if E22613[I]=5 and E22628[I]>=0 and E34402[I]>0 then OTHPAY1[I]=E22628[I]/(4.3*E34402[I]);
if E22613[I]=6 and E22628[I]>=0 and E34402[I]>0 and E35600[I]>0 then
    OTHPAY1[I]=E22628[I]/(E35600[I]*E34402[I]);
if E22613[I]=8 and E22628[I]>=0 and E34402[I]>0 then OTHPAY1[I]=E22628[I]/(2.15*E34402[I]);
if E22613[I] in (9,14) then OTHPAY1[I]=0;
/*missing value*/
if -4<E34402[I]<0 then OTHPAY1[I]=E34402[I];
if E34402[I]=0 then OTHPAY1[I]=-3;
if E22613[I]=2 and E34402B[I] le 0 and E34430[I] le 0 then OTHPAY1[I]=-3;
if E22613[I]=2 and -4<E34402B[I]<0 then OTHPAY1[I]=E34402B[I];
if E22613[I]=2 and -4<E34430[I]<0 then OTHPAY1[I]=E34430[I];
if E22613[I]=6 and E35600[I] le 0 then OTHPAY1[I]=-3;
if E22613[I]=6 and -4<E35600[I]<0 then OTHPAY1[I]=E35600[I];
end;
end;

***** FOR COMMISSIONS *****/
do I=1 to 9; /*with overtime*/
if E34428[I]>0 and E216002[I]=1 and E225002[I]>=0 then OTHPAY2[I]=E225002[I];
if E34428[I]>0 and E216002[I]=2 and E34430[I]>0 and E225002[I]>=0 then
    OTHPAY2[I]=(E225002[I]*E34430[I])/E34428[I];
if E34428[I]>0 and E216002[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and E225002[I]>=0 then
    OTHPAY2[I]=E225002[I]/E34428[I];
if E34428[I]>0 and E216002[I]=4 and E225002[I]>=0 then OTHPAY2[I]=E225002[I]/(2*E34428[I]);
if E34428[I]>0 and E216002[I]=5 and E225002[I]>=0 then OTHPAY2[I]=E225002[I]/(4.3*E34428[I]);
if E34428[I]>0 and E216002[I]=6 and E225002[I]>=0 and E35600[I]>0 then
    OTHPAY2[I]=E225002[I]/(E35600[I]*E34428[I]);
if E34428[I]>0 and E216002[I]=8 and E225002[I]>=0 then OTHPAY2[I]=E225002[I]/(2.15*E34428[I]);
/*missing value*/;
if E225002[I]>=0 and -4<E34428[I]<0 then OTHPAY2[I]=E34428[I];
if -4<E225002[I]<0 then OTHPAY2[I]=E225002[I];
if E225002[I]>=0 and E34428[I]=0 then OTHPAY2[I]=-3;
end;

do I=1 to 9; /*without overtime*/
if E34402[I]>0 and E216002[I]=1 and E225002[I]>=0 then OTHPAY2[I]=E225002[I];
if E34402[I]>0 and E216002[I]=2 and E34402B[I]>0 and E225002[I]>=0 then
    OTHPAY2[I]=(E225002[I]*E34402B[I])/E34402[I];
if E34402[I]>0 and E216002[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and E225002[I]>=0 then
    OTHPAY2[I]=E225002[I]/E34402[I];
if E34402[I]>0 and E216002[I]=4 and E225002[I]>=0 then OTHPAY2[I]=E225002[I]/(2*E34402[I]);
if E34402[I]>0 and E216002[I]=5 and E225002[I]>=0 then OTHPAY2[I]=E225002[I]/(4.3*E34402[I]);
if E34402[I]>0 and E216002[I]=6 and E225002[I]>=0 and E35600[I]>0 then
    OTHPAY2[I]=E225002[I]/(E35600[I]*E34402[I]);
if E34402[I]>0 and E216002[I]=8 and E225002[I]>=0 then OTHPAY2[I]=E225002[I]/(2.15*E34402[I]);
if E216002[I] in (9,14) then OTHPAY2[I]=0;
if E216002[I] in (0,7,12,13,15,16,17,14,99,21,22,23,24,25,26,28) then OTHPF2[I]=OTHPF2[I]+1;
/*missing value*/
if E225002[I]>=0 and -4<E34402[I]<0 then OTHPAY2[I]=E34402[I];
if E225002[I]>=0 and E34402[I]=0 then OTHPAY2[I]=-3;
if E216002[I]=2 and E34402B[I] le 0 and E34430[I] le 0 then OTHPAY2[I]=-3;
if E216002[I]=2 and -4<E34402B[I]<0 then OTHPAY2[I]=E34402B[I];
if E216002[I]=2 and -4<E34430[I]<0 then OTHPAY2[I]=E34430[I];
if E216002[I]=6 and E35600[I] le 0 then OTHPAY2[I]=-3;
if E216002[I]=6 and -4<E35600[I]<0 then OTHPAY2[I]=E35600[I];
end;

```

```

/* if commissions is corrected in the later part */
do I=1 to 9;
if E226046[I]=1 then do;

  if E22614[I]=1 and E22629[I]>=0 then OTHPAY2[I]=E22629[I]; /* with overtime*/
  if E22614[I]=2 and E22629[I]>=0 and E34430[I]>0 and E34428[I]>0 then
    OTHPAY2[I]=E22629[I]*E34430[I]/E34428[I];
  if E22614[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and E22629[I]>=0 and E34428[I]>0 then
    OTHPAY2[I]=E22629[I]/E34428[I];
  if E22614[I]=4 and E22629[I]>=0 and E34428[I]>0 then OTHPAY2[I]=E22629[I]/(2*E34428[I]);
  if E22614[I]=5 and E22629[I]>=0 and E34428[I]>0 then OTHPAY2[I]=E22629[I]/(4.3*E34428[I]);
  if E22614[I]=6 and E22629[I]>=0 and E34428[I]>0 and E35600[I]>0 then
    OTHPAY2[I]=E22629[I]/(E35600[I]*E34428[I]);
  if E22614[I]=8 and E22629[I]>=0 and E34428[I]>0 then OTHPAY2[I]=E22629[I]/(2.15*E34428[I]);
  if E22614[I] in (9,14) then OTHPAY2[I]=0;

/*missing value*/
  if -4<E34428[I]<0 then OTHPAY2[I]=E34428[I];
  if -4<E22629[I]<0 then OTHPAY2[I]=E22629[I];
  if E34428[I]=0 then OTHPAY2[I]=-3;

  if E22614[I]=1 and E22629[I]>=0 then OTHPAY2[I]=E22629[I]; /* without overtime*/
  if E22614[I]=2 and E22629[I]>=0 and E34402B[I]>0 and E34402[I]>0 then
    OTHPAY2[I]=E22629[I]*E34402B[I]/E34402[I];
  if E22614[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and E22629[I]>=0 and E34402[I]>0 then
    OTHPAY2[I]=E22629[I]/E34402[I];
  if E22614[I]=4 and E22629[I]>=0 and E34402B[I]>0 then OTHPAY2[I]=E22629[I]/(2*E34402B[I]);
  if E22614[I]=5 and E22629[I]>=0 and E34402B[I]>0 then OTHPAY2[I]=E22629[I]/(4.3*E34402B[I]);
  if E22614[I]=6 and E22629[I]>=0 and E34402B[I]>0 and E35600[I]>0 then
    OTHPAY2[I]=E22629[I]/(E35600[I]*E34402B[I]);
  if E22614[I]=8 and E22629[I]>=0 and E34402B[I]>0 then OTHPAY2[I]=E22629[I]/(2.15*E34402B[I]);
  if E22614[I] in (9,14) then OTHPAY2[I]=0;

/*missing value*/
  if -4<E34402B[I]<0 then OTHPAY2[I]=E34402B[I];
  if E34402B[I]=0 then OTHPAY2[I]=-3;
  if E22614[I]=2 and E34402B[I] le 0 and E34430[I] le 0 then OTHPAY2[I]=-3;
  if E22614[I]=2 and -4<E34402B[I]<0 then OTHPAY2[I]=E34402B[I];
  if E22614[I]=2 and -4<E34430[I]<0 then OTHPAY2[I]=E34430[I];
  if E22614[I]=6 and E35600[I] le 0 then OTHPAY2[I]=-3;
  if E22614[I]=6 and -4<E35600[I]<0 then OTHPAY2[I]=E35600[I];

end;
end;

***** FOR BONUSES *****/
do I=1 to 9; /*with overtime*/
if E34428[I]>0 and E216003[I]=1 in (1,21) and E225003[I]>=0 then OTHPAY3[I]=E225003[I];
if E34428[I]>0 and E216003[I]=2 in (2,22) and E34430[I]>0 and E225003[I]>=0 then
  OTHPAY3[I]=(E225003[I]*E34430[I])/E34428[I];
if E34428[I]>0 and E216003[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and E225003[I]>=0 then
  OTHPAY3[I]=E225003[I]/E34428[I];
if E34428[I]>0 and E216003[I]=4 and E225003[I]>=0 then OTHPAY3[I]=E225003[I]/(2*E34428[I]);
if E34428[I]>0 and E216003[I]=5 and E225003[I]>=0 then OTHPAY3[I]=E225003[I]/(4.3*E34428[I]);
if E34428[I]>0 and E216003[I]=6 and E225003[I]>=0 and E35600[I]>0 then
  OTHPAY3[I]=E225003[I]/(E35600[I]*E34428[I]);
if E34428[I]>0 and E216003[I]=8 and E225003[I]>=0 then OTHPAY3[I]=E225003[I]/(2.15*E34428[I]);
*MISSING VALUE*/;
if E225003[I]>=0 and -4<E34428[I]<0 then OTHPAY3[I]=E34428[I];

```

```

if -4<E225003[I]<0 then OTHPAY3[I]=E225003[I];
if E225003[I]>=0 and E34428[I]=0 then OTHPAY3[I]=-3;
end;

do I=1 to 9; /*without overtime*/
if E34402[I]>0 and E216003[I]=1 and E225003[I]>=0 then OTHPAY3[I]=E225003[I];
if E34402[I]>0 and E216003[I]=2 and E34402B[I]>0 and E225003[I]>=0 then
    OTHPAY3[I]=(E225003[I]*E34402B[I])/E34402[I];
if E34402[I]>0 and E216003[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and E225003[I]>=0 then
    OTHPAY3[I]=E225003[I]/E34402[I];
if E34402[I]>0 and E216003[I]=4 and E225003[I]>=0 then OTHPAY3[I]=E225003[I]/(2*E34402[I]);
if E34402[I]>0 and E216003[I]=5 and E225003[I]>=0 then OTHPAY3[I]=E225003[I]/(4.3*E34402[I]);
if E34402[I]>0 and E216003[I]=6 and E225003[I]>=0 and E35600[I]>0 then
    OTHPAY3[I]=E225003[I]/(E35600[I]*E34402[I]);
if E34402[I]>0 and E216003[I]=8 and E225003[I]>=0 then OTHPAY3[I]=E225003[I]/(2.15*E34402[I]);
if E216003[I] in (9,14) then OTHPAY3[I]=0;
if E216003[I] in (0,7,12,13,15,16,17,14,99,21,22,23,24,25,26,28) then OTHPF3[I]=OTHPF3[I]+1;
/*MISSING VALUE*/;
if E225003[I]>=0 and -4<E34402[I]<0 then OTHPAY3[I]=E34402[I];
if E225003[I]>=0 and E34402[I]=0 then OTHPAY3[I]=-3;
if E216003[I]=2 and E34402B[I] le 0 and E34430[I] le 0 then OTHPAY3[I]=-3;
if E216003[I]=2 and -4<E34402B[I]<0 then OTHPAY3[I]=E34402B[I];
if E216003[I]=2 and -4<E34430[I]<0 then OTHPAY3[I]=E34430[I];
if E216003[I]=6 and E35600[I] le 0 then OTHPAY3[I]=-3;
if E216003[I]=6 and -4<E35600[I]<0 then OTHPAY3[I]=E35600[I];
end;

/* if bonus is corrected in the later part */
do I=1 to 9;
if E226047[I]=1 then do;
if E22615[I]=1 and E22630[I]>=0 then OTHPAY3[I]=E22630[I]; /* with overtime*/
if E22615[I]=2 and E22630[I]>=0 and E34430[I]>0 and E34428[I]>0 then
    OTHPAY3[I]=E22630[I]*E34430[I]/E34428[I];
if E22615[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and E22630[I]>=0 and E34428[I]>0 then
    OTHPAY3[I]=E22630[I]/E34428[I];
if E22615[I]=4 and E22630[I]>=0 and E34428[I]>0 then OTHPAY3[I]=E22630[I]/(2*E34428[I]);
if E22615[I]=5 and E22630[I]>=0 and E34428[I]>0 then OTHPAY3[I]=E22630[I]/(4.3*E34428[I]);
if E22615[I]=6 and E22630[I]>=0 and E34428[I]>0 and E35600[I]>0 then
    OTHPAY3[I]=E22630[I]/(E35600[I]*E34428[I]);
if E22615[I]=8 and E22630[I]>=0 and E34428[I]>0 then OTHPAY3[I]=E22630[I]/(2.15*E34428[I]);
if E22615[I] in (9,14) then OTHPAY3[I]=0;
/*missing value*/
if -4<E34428[I]<0 then OTHPAY3[I]=E34428[I];
if -4<E22630[I]<0 then OTHPAY3[I]=E22630[I];
if E34428[I]=0 then OTHPAY3[I]=-3;

if E22615[I]=1 and E22630[I]>=0 then OTHPAY3[I]=E22630[I]; /* without overtime*/
if E22615[I]=2 and E22630[I]>=0 and E34402B[I]>0 and E34402[I]>0 then
    OTHPAY3[I]=E22630[I]*E34402B[I]/E34402[I];
if E22615[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and E22630[I]>=0 and E34402[I]>0 then
    OTHPAY3[I]=E22630[I]/E34402[I];
if E22615[I]=4 and E22630[I]>=0 and E34402[I]>0 then OTHPAY3[I]=E22630[I]/(2*E34402[I]);
if E22615[I]=5 and E22630[I]>=0 and E34402[I]>0 then OTHPAY3[I]=E22630[I]/(4.3*E34402[I]);
if E22615[I]=6 and E22630[I]>=0 and E34402[I]>0 and E35600[I]>0 then
    OTHPAY3[I]=E22630[I]/(E35600[I]*E34402[I]);
if E22615[I]=8 and E22630[I]>=0 and E34402[I]>0 then OTHPAY3[I]=E22630[I]/(2.15*E34402[I]);
if E22615[I] in (9,14) then OTHPAY3[I]=0;

```

```

/*missing value*/
if -4<E34402[I]<0 then OTHPAY3[I]=E34402[I];
if E34402[I]=0 then OTHPAY3[I]=-3;
if E22615[I]=2 and E34402B[I] le 0 and E34430[I] le 0 then OTHPAY3[I]=-3;
if E22615[I]=2 and -4<E34402B[I]<0 then OTHPAY3[I]=E34402B[I];
if E22615[I]=2 and -4<E34430[I]<0 then OTHPAY3[I]=E34430[I];
if E22615[I]=6 and E35600[I] le 0 then OTHPAY3[I]=-3;
if E22615[I]=6 and -4<E35600[I]<0 then OTHPAY3[I]=E35600[I];
end;
end;

***** FOR INCENTIVE PAY *****/
do I=1 to 9; /*with overtime*/
if E34428[I]>0 and E216004[I]=1 and E225004[I]>=0 then OTHPAY4[I]=E225004[I];
if E34428[I]>0 and E216004[I]=2 and E34430[I]>0 and E225004[I]>=0 then
    OTHPAY4[I]=(E225004[I]*E34430[I])/E34428[I];
if E34428[I]>0 and E216004[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and E225004[I]>=0 then
    OTHPAY4[I]=E225004[I]/E34428[I];
if E34428[I]>0 and E216004[I]=4 and E225004[I]>=0 then OTHPAY4[I]=E225004[I]/(2*E34428[I]);
if E34428[I]>0 and E216004[I]=5 and E225004[I]>=0 then OTHPAY4[I]=E225004[I]/(4.3*E34428[I]);
if E34428[I]>0 and E216004[I]=6 and E225004[I]>=0 and E35600[I]>0 then
    OTHPAY4[I]=E225004[I]/(E35600[I]*E34428[I]);
if E34428[I]>0 and E216004[I]=8 and E225004[I]>=0 then OTHPAY4[I]=E225004[I]/(2.15*E34428[I]);
/*missing value*/;
if E225004[I]>=0 and -4<E34428[I]<0 then OTHPAY4[I]=E34428[I];
if -4<E225004[I]<0 then OTHPAY4[I]=E225004[I];
if E225004[I]>=0 and E34428[I]=0 then OTHPAY4[I]=-3;
end;

do I=1 to 9; /*without overtime*/
if E34402[I]>0 and E216004[I]=1 and E225004[I]>=0 then OTHPAY4[I]=E225004[I];
if E34402[I]>0 and E216004[I]=2 and E34402B[I]>0 and E225004[I]>=0 then
    OTHPAY4[I]=(E225004[I]*E34402B[I])/E34402[I];
if E34402[I]>0 and E216004[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and E225004[I]>=0 then
    OTHPAY4[I]=E225004[I]/E34402[I];
if E34402[I]>0 and E216004[I]=4 and E225004[I]>=0 then OTHPAY4[I]=E225004[I]/(2*E34402[I]);
if E34402[I]>0 and E216004[I]=5 and E225004[I]>=0 then OTHPAY4[I]=E225004[I]/(4.3*E34402[I]);
if E34402[I]>0 and E216004[I]=6 and E225004[I]>=0 and E35600[I]>0 then
    OTHPAY4[I]=E225004[I]/(E35600[I]*E34402[I]);
if E34402[I]>0 and E216004[I]=8 and E225004[I]>=0 then OTHPAY4[I]=E225004[I]/(2.15*E34402[I]);
if E216004[I] in (9,14) then OTHPAY4[I]=0;
if E216004[I] in (0,7,12,13,15,16,17,14,99,21,22,23,24,25,26,28) then OTHPF4[I]=OTHPF4[I]+1;
/*missing value*/;
if E225004[I]>=0 and -4<E34402[I]<0 then OTHPAY4[I]=E34402[I];
if E225004[I]>=0 and E34402[I]=0 then OTHPAY4[I]=-3;
if E216004[I]=2 and E34402B[I] le 0 and E34430[I] le 0 then OTHPAY4[I]=-3;
if E216004[I]=2 and -4<E34402B[I]<0 then OTHPAY4[I]=E34402B[I];
if E216004[I]=2 and -4<E34430[I]<0 then OTHPAY4[I]=E34430[I];
if E216004[I]=6 and E35600[I] le 0 then OTHPAY4[I]=-3;
if E216004[I]=6 and -4<E35600[I]<0 then OTHPAY4[I]=E35600[I];
end;

/* if incentive pay is corrected in the later part */
do I=1 to 9;
if E226048[I]=1 then do;
if E22616[I]=1 and E22631[I]>=0 then OTHPAY4[I]=E22631[I]; /* with overtime*/

```

```

if E22616[I]=2 and E22631[I]>=0 and E34430[I]>0 and E34428[I]>0 then
    OTHPAY4[I]=E22631[I]*E34430[I]/E34428[I];
if E22616[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and E22631[I]>=0 and E34428[I]>0 then
    OTHPAY4[I]=E22631[I]/E34428[I];
if E22616[I]=4 and E22631[I]>=0 and E34428[I]>0 then OTHPAY4[I]=E22631[I]/(2*E34428[I]);
if E22616[I]=5 and E22631[I]>=0 and E34428[I]>0 then OTHPAY4[I]=E22631[I]/(4.3*E34428[I]);
if E22616[I]=6 and E22631[I]>=0 and E34428[I]>0 and E35600[I]>0 then
    OTHPAY4[I]=E22631[I]/(E35600[I]*E34428[I]);
if E22616[I]=8 and E22631[I]>=0 and E34428[I]>0 then OTHPAY4[I]=E22631[I]/(2.15*E34428[I]);
if E22616[I] in (9,14) then OTHPAY4[I]=0;
/*missing value*/
if -4<E34428[I]<0 then OTHPAY4[I]=E34428[I];
if -4<E22631[I]<0 then OTHPAY4[I]=E22631[I];
if E34428[I]=0 then OTHPAY4[I]=-3;

if E22616[I]=1 and E22631[I]>=0 then OTHPAY4[I]=E22631[I]; /* without overtime*/
if E22616[I]=2 and E22631[I]>=0 and E34402B[I]>0 and E34402[I]>0 then
    OTHPAY4[I]=E22631[I]*E34402B[I]/E34402[I];
if E22616[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and E22631[I]>=0 and E34402[I]>0 then
    OTHPAY4[I]=E22631[I]/E34402[I];
if E22616[I]=4 and E22631[I]>=0 and E34402[I]>0 then OTHPAY4[I]=E22631[I]/(2*E34402[I]);
if E22616[I]=5 and E22631[I]>=0 and E34402[I]>0 then OTHPAY4[I]=E22631[I]/(4.3*E34402[I]);
if E22616[I]=6 and E22631[I]>=0 and E34402[I]>0 and E35600[I]>0 then
    OTHPAY4[I]=E22631[I]/(E35600[I]*E34402[I]);
if E22616[I]=8 and E22631[I]>=0 and E34402[I]>0 then OTHPAY4[I]=E22631[I]/(2.15*E34402[I]);
if E22616[I] in (9,14) then OTHPAY4[I]=0;
/*missing value*/
if -4<E34402[I]<0 then OTHPAY4[I]=E34402[I];
if E34402[I]=0 then OTHPAY4[I]=-3;
if E22616[I]=2 and E34402B[I]<0 and E34430[I]<0 then OTHPAY4[I]=-3;
if E22616[I]=2 and -4<E34402B[I]<0 then OTHPAY4[I]=E34402B[I];
if E22616[I]=2 and -4<E34430[I]<0 then OTHPAY4[I]=E34430[I];
if E22616[I]=6 and E35600[I]<0 then OTHPAY4[I]=-3;
if E22616[I]=6 and -4<E35600[I]<0 then OTHPAY4[I]=E35600[I];
end;
end;

***** FOR OTHERS *****/
do I=1 to 9; /*with overtime*/
    if E34428[I]>0 and E216005[I]=1 and E225005[I]>=0 then OTHPAY5[I]=E225005[I];
    if E34428[I]>0 and E216005[I]=2 and E34430[I]>0 and E225005[I]>=0 then
        OTHPAY5[I]=(E225005[I]*E34430[I])/E34428[I];
    if E34428[I]>0 and E216005[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and E225005[I]>=0 then
        OTHPAY5[I]=E225005[I]/E34428[I];
    if E34428[I]>0 and E216005[I]=4 and E225005[I]>=0 then OTHPAY5[I]=E225005[I]/(2*E34428[I]);
    if E34428[I]>0 and E216005[I]=5 and E225005[I]>=0 then OTHPAY5[I]=E225005[I]/(4.3*E34428[I]);
    if E34428[I]>0 and E216005[I]=6 and E225005[I]>=0 and E35600[I]>0 then
        OTHPAY5[I]=E225005[I]/(E35600[I]*E34428[I]);
    if E34428[I]>0 and E216005[I]=8 and E225005[I]>=0 then OTHPAY5[I]=E225005[I]/(2.15*E34428[I]);
    /*missing value*/
    if E225005[I]>0 and -4<E34428[I]<0 then OTHPAY5[I]=E34428[I];
    if -4<E225005[I]<0 then OTHPAY5[I]=E225005[I];
    if E225005[I]>0 and E34428[I]=0 then OTHPAY5[I]=-3;
end;

do I=1 to 9; /*without overtime*/

```

Appendix 2: Employment Variable Creation

```

if E34402[I]>0 and E216005[I]=1 and E225005[I]>=0 then OTHPAY5[I]=E225005[I];
if E34402[I]>0 and E216005[I]=2 and E34402B[I]>0 and E225005[I]>=0 then
    OTHPAY5[I]=(E225005[I]*E34402B[I])/E34402[I];
if E34402[I]>0 and E216005[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and E225005[I]>=0 then
    OTHPAY5[I]=E225005[I]/E34402[I];
if E34402[I]>0 and E216005[I]=4 and E225005[I]>=0 then OTHPAY5[I]=E225005[I]/(2*E34402[I]);
if E34402[I]>0 and E216005[I]=5 and E225005[I]>=0 then OTHPAY5[I]=E225005[I]/(4.3*E34402[I]);
if E34402[I]>0 and E216005[I]=6 and E225005[I]>=0 and E35600[I]>0 then
    OTHPAY5[I]=E225005[I]/(E35600[I]*E34402[I]);
if E34402[I]>0 and E216005[I]=8 and E225005[I]>=0 then OTHPAY5[I]=E225005[I]/(2.15*E34402[I]);
if E216005[I] in (9,14) then OTHPAY5[I]=0;
if E216005[I] in (0,7,12,13,15,16,17,14,99,21,22,23,24,25,26,28) then OTHPF5[I]=OTHPF5[I]+1;
/*missing value*/
if E225005[I]>=0 and -4<E34402[I]<0 then OTHPAY5[I]=E34402[I];
if E225005[I]>=0 and E34402[I]=0 then OTHPAY5[I]=-3;
if E216005[I]=2 and E34402B[I]<=0 and E34430[I]<=0 then OTHPAY5[I]=-3;
if E216005[I]=2 and -4<E34402B[I]<0 then OTHPAY5[I]=E34402B[I];
if E216005[I]=2 and -4<E34430[I]<0 then OTHPAY5[I]=E34430[I];
if E216005[I]=6 and E35600[I]<=0 then OTHPAY5[I]=-3;
if E216005[I]=6 and -4<E35600[I]<0 then OTHPAY5[I]=E35600[I];
end;

/* if other compensation is corrected in the later part */
do I=1 to 9;
if E226049[I]=1 then do;
if E22617[I]=1 and E22632[I]>=0 then OTHPAY5[I]=E22632[I]; /* with overtime*/
if E22617[I]=2 and E22632[I]>=0 and E34430[I]>0 and E34428[I]>0 then
    OTHPAY5[I]=E22632[I]*E34430[I]/E34428[I];
if E22617[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and E22632[I]>=0 and E34428[I]>0 then
    OTHPAY5[I]=E22632[I]/E34428[I];
if E22617[I]=4 and E22632[I]>=0 and E34428[I]>0 then OTHPAY5[I]=E22632[I]/(2*E34428[I]);
if E22617[I]=5 and E22632[I]>=0 and E34428[I]>0 then OTHPAY5[I]=E22632[I]/(4.3*E34428[I]);
if E22617[I]=6 and E22632[I]>=0 and E34428[I]>0 and E35600[I]>0 then
    OTHPAY5[I]=E22632[I]/(E35600[I]*E34428[I]);
if E22617[I]=8 and E22632[I]>=0 and E34428[I]>0 then OTHPAY5[I]=E22632[I]/(2.15*E34428[I]);
if E22617[I] in (9,14) then OTHPAY5[I]=0;
/*missing value*/
if -4<E34428[I]<0 then OTHPAY5[I]=E34428[I];
if -4<E22632[I]<0 then OTHPAY5[I]=E22632[I];
if E34428[I]=0 then OTHPAY5[I]=-3;

if E22617[I]=1 and E22632[I]>=0 then OTHPAY5[I]=E22632[I]; /* without overtime*/
if E22617[I]=2 and E22632[I]>=0 and E34402B[I]>0 and E34402[I]>0 then
    OTHPAY5[I]=E22632[I]*E34402B[I]/E34402[I];
if E22617[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and E22632[I]>=0 and E34402[I]>0 then
    OTHPAY5[I]=E22632[I]/E34402[I];
if E22617[I]=4 and E22632[I]>=0 and E34402[I]>0 then OTHPAY5[I]=E22632[I]/(2*E34402[I]);
if E22617[I]=5 and E22632[I]>=0 and E34402[I]>0 then OTHPAY5[I]=E22632[I]/(4.3*E34402[I]);
if E22617[I]=6 and E22632[I]>=0 and E34402[I]>0 and E35600[I]>0 then
    OTHPAY5[I]=E22632[I]/(E35600[I]*E34402[I]);
if E22617[I]=8 and E22632[I]>=0 and E34402[I]>0 then OTHPAY5[I]=E22632[I]/(2.15*E34402[I]);
if E22617[I] in (9,14) then OTHPAY5[I]=0;
/*missing value*/
if -4<E34402[I]<0 then OTHPAY5[I]=E34402[I];
if E34402[I]=0 then OTHPAY5[I]=-3;
if E22617[I]=2 and E34402B[I]<=0 and E34430[I]<=0 then OTHPAY5[I]=-3;
if E22617[I]=2 and -4<E34402B[I]<0 then OTHPAY5[I]=E34402B[I];

```

Appendix 2: Employment Variable Creation

```
if E22617[I]=2 and -4<E34430[I]<0 then OTHPAY5[I]=E34430[I];
if E22617[I]=6 and E35600[I] le 0 then OTHPAY5[I]=-3;
if E22617[I]=6 and -4<E35600[I]<0 then OTHPAY5[I]=E35600[I];
end;
end;

***** do it for non-overtime payment if answered in the later part *****

***** FOR TIPS *****
do I=1 to 9; /*with overtime*/
if E98429[I]>0 and E102051[I]=1 and E102141[I]>=0 then OTHPAY1[I]=E102141[I];
if E98429[I]>0 and E102051[I]=2 and E98429E[I]>0 and E102141[I]>=0 then
    OTHPAY1[I]=(E102141[I]*E98429E[I])/E98429[I];
if E98429[I]>0 and E102051[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and E102141[I]>=0 then
    OTHPAY1[I]=E102141[I]/E98429[I];
if E98429[I]>0 and E102051[I]=4 and E102141[I]>=0 then OTHPAY1[I]=E102141[I]/(2*E98429[I]);
if E98429[I]>0 and E102051[I]=5 and E102141[I]>=0 then OTHPAY1[I]=E102141[I]/(4.3*E98429[I]);
if E98429[I]>0 and E102051[I]=6 and E102141[I]>=0 and E99500[I]>0 then
    OTHPAY1[I]=E102141[I]/(E99500[I]*E98429[I]);
if E98429[I]>0 and E102051[I]=8 and E102141[I]>=0 then OTHPAY1[I]=E102141[I]/(2.15*E98429[I]);
/*missing value*/
if E102141[I]>=0 and -4<E98429[I]<0 then OTHPAY1[I]=E98429[I];
if -4<E102141[I]<0 then OTHPAY1[I]=E102141[I];
if E102141[I]>=0 and E98429[I]=0 then OTHPAY1[I]=-3;
end;

do I=1 to 9; /*without overtime*/
if E98402[I]>0 and E102051[I]=1 and E102141[I]>=0 then OTHPAY1[I]=E102141[I];
if E98402[I]>0 and E102051[I]=2 and E98402D[I]>0 and E102141[I]>=0 then
    OTHPAY1[I]=(E102141[I]*E98402D[I])/E98402[I];
if E98402[I]>0 and E102051[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and E102141[I]>=0 then
    OTHPAY1[I]=E102141[I]/E98402[I];
if E98402[I]>0 and E102051[I]=4 and E102141[I]>=0 then OTHPAY1[I]=E102141[I]/(2*E98402[I]);
if E98402[I]>0 and E102051[I]=5 and E102141[I]>=0 then OTHPAY1[I]=E102141[I]/(4.3*E98402[I]);
if E98402[I]>0 and E102051[I]=6 and E102141[I]>=0 and E99500[I]>0 then
    OTHPAY1[I]=E102141[I]/(52*E98402[I]);
if E98402[I]>0 and E102051[I]=8 and E102141[I]>=0 then OTHPAY1[I]=E102141[I]/(2.15*E98402[I]);
if E102051[I] in (0,7,12,13,15,16,17,14,99,21,22,23,24,25,26,28) then OTHPF1[I]=OTHPF1[I]+1;
if E102051[I] in (9,14) then OTHPAY1[I]=0;
/*missing value*/
if E102141[I]>=0 and -4<E98402[I]<0 then OTHPAY1[I]=E98402[I];
if E102141[I]>=0 and E98402[I]=0 then OTHPAY1[I]=-3;
if E102051[I]=2 and E98402D[I] le 0 and E98429E[I] le 0 then OTHPAY1[I]=-3;
if E102051[I]=2 and -4<E98402D[I]<0 then OTHPAY1[I]=E98402D[I];
if E102051[I]=2 and -4<E98429E[I]<0 then OTHPAY1[I]=E98429E[I];
if E102051[I]=6 and E99500[I] le 0 then OTHPAY1[I]=-3;
if E102051[I]=6 and -4<E99500[I]<0 then OTHPAY1[I]=E99500[I];
end;

/* if tips is corrected in the later part */
do I=1 to 9;
if E102255[I]=1 then do;
if E100234[I]=1 and E100250[I]>=0 then OTHPAY1[I]=E100250[I]; /* with overtime*/
if E100234[I]=2 and E100250[I]>=0 and E98429E[I]>0 and E98429[I]>0 then
    OTHPAY1[I]=E100250[I]*E98429E[I]/E98429[I];
if E100234[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and E100250[I]>=0 and E98429[I]>0 then
    OTHPAY1[I]=E100250[I]/E98429[I];
```

Appendix 2: Employment Variable Creation

```

if E100234[I]=4 and E100250[I]>=0 and E98429[I]>0 then OTHPAY1[I]=E100250[I]/(2*E98429[I]);
if E100234[I]=5 and E100250[I]>=0 and E98429[I]>0 then OTHPAY1[I]=E100250[I]/(4.3*E98429[I]);
if E100234[I]=6 and E100250[I]>=0 and E98429[I]>0 and E99500[I]>0 then
    OTHPAY1[I]=E100250[I]/(E99500[I]*E98429[I]);
if E100234[I]=8 and E100250[I]>=0 and E98429[I]>0 then OTHPAY1[I]=E100250[I]/(2.15*E98429[I]);
if E100234[I] in (9,14) then OTHPAY1[I]=0;
/*missing value*/
if -4<E98429[I]<0 then OTHPAY1[I]=E98429[I];
if -4<E100250[I]<0 then OTHPAY1[I]=E100250[I];
if E98429[I]=0 then OTHPAY1[I]=-3;

if E100234[I]=1 and E100250[I]>=0 then OTHPAY1[I]=E100250[I]; /* without overtime*/
if E100234[I]=2 and E100250[I]>=0 and E98402D[I]>0 and E98402[I]>0 then
    OTHPAY1[I]=E100250[I]*E98402D[I]/E98402[I];
if E100234[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and E100250[I]>=0 and E98402[I]>0 then
    OTHPAY1[I]=E100250[I]/E98402[I];
if E100234[I]=4 and E100250[I]>=0 and E98402[I]>0 then OTHPAY1[I]=E100250[I]/(2*E98402[I]);
if E100234[I]=5 and E100250[I]>=0 and E98402[I]>0 then OTHPAY1[I]=E100250[I]/(4.3*E98402[I]);
if E100234[I]=6 and E100250[I]>=0 and E98402[I]>0 and E99500[I]>0 then
    OTHPAY1[I]=E100250[I]/(E99500[I]*E98402[I]);
if E100234[I]=8 and E100250[I]>=0 and E98402[I]>0 then OTHPAY1[I]=E100250[I]/(2.15*E98402[I]);
if E100234[I] in (9,14) then OTHPAY1[I]=0;
/*missing value*/
if -4<E98402[I]<0 then OTHPAY1[I]=E98402[I];
if E98402[I]=0 then OTHPAY1[I]=-3;
if E100234[I]=2 and E98402D[I] le 0 and E98429E[I] le 0 then OTHPAY1[I]=-3;
if E100234[I]=2 and -4<E98402D[I]<0 then OTHPAY1[I]=E98402D[I];
if E100234[I]=2 and -4<E98429E[I]<0 then OTHPAY1[I]=E98429E[I];
if E100234[I]=6 and E99500[I] le 0 then OTHPAY1[I]=-3;
if E100234[I]=6 and -4<E99500[I]<0 then OTHPAY1[I]=E99500[I];
end;
end;

/********** FOR COMMISSIONS *****/
do I=1 to 9; /*with overtime*/
if E98429[I]>0 and E102052[I]=1 and E102142[I]>=0 then OTHPAY2[I]=E102142[I];
if E98429[I]>0 and E102052[I]=2 and E98429E[I]>0 and E102142[I]>=0 then
    OTHPAY2[I]=(E102142[I]*E98429E[I])/E98429[I];
if E98429[I]>0 and E102052[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and E102142[I]>=0 then
    OTHPAY2[I]=E102142[I]/E98429[I];
if E98429[I]>0 and E102052[I]=4 and E102142[I]>=0 then OTHPAY2[I]=E102142[I]/(2*E98429[I]);
if E98429[I]>0 and E102052[I]=5 and E102142[I]>=0 then OTHPAY2[I]=E102142[I]/(4.3*E98429[I]);
if E98429[I]>0 and E102052[I]=6 and E102142[I]>=0 and E99500[I]>0 then
    OTHPAY2[I]=E102142[I]/(E99500[I]*E98429[I]);
if E98429[I]>0 and E102052[I]=8 and E102142[I]>=0 then OTHPAY2[I]=E102142[I]/(2.15*E98429[I]);
/*missing value*/
if E102142[I]>=0 and -4<E98429[I]<0 then OTHPAY2[I]=E98429[I];
if -4<E102142[I]<0 then OTHPAY2[I]=E102142[I];
if E102142[I]>=0 and E98429[I]=0 then OTHPAY2[I]=-3;
end;

do I=1 to 9; /*without overtime*/
if E98402[I]>0 and E102052[I]=1 and E102142[I]>=0 then OTHPAY2[I]=E102142[I];
if E98402[I]>0 and E102052[I]=2 and E98402D[I]>0 and E102142[I]>=0 then
    OTHPAY2[I]=(E102142[I]*E98402D[I])/E98402[I];
if E98402[I]>0 and E102052[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and E102142[I]>=0 then
    OTHPAY2[I]=E102142[I]/E98402[I];

```

```

if E98402[I]>0 and E102052[I]=4 and E102142[I]>=0 then OTHPAY2[I]=E102142[I]/(2*E98402[I]);
if E98402[I]>0 and E102052[I]=5 and E102142[I]>=0 then OTHPAY2[I]=E102142[I]/(4.3*E98402[I]);
if E98402[I]>0 and E102052[I]=6 and E102142[I]>=0 and E99500[I]>0 then
    OTHPAY2[I]=E102142[I]/(E99500[I]*E98402[I]);
if E98402[I]>0 and E102052[I]=8 and E102142[I]>=0 then OTHPAY2[I]=E102142[I]/(2.15*E98402[I]);
if E102052[I] in (0,7,12,13,15,16,17,14,99,21,22,23,24,25,26,28) then OTHPF2[I]=OTHPF2[I]+1;
if E102052[I] in (9,14) then OTHPAY2[I]=0;
/*missing value*/
if E102142[I]>=0 and -4<E98402[I]<0 then OTHPAY2[I]=E98402[I];
if E102142[I]>=0 and E98402[I]=0 then OTHPAY2[I]=-3;
if E102052[I]=2 and E98402D[I] le 0 and E98429E[I] le 0 then OTHPAY2[I]=-3;
if E102052[I]=2 and -4<E98402D[I]<0 then OTHPAY2[I]=E98402D[I];
if E102052[I]=2 and -4<E98429E[I]<0 then OTHPAY2[I]=E98429E[I];
if E102052[I]=6 and E99500[I] le 0 then OTHPAY2[I]=-3;
if E102052[I]=6 and -4<E99500[I]<0 then OTHPAY2[I]=E99500[I];
end;

/* if commissions is corrected in the later part */
do I=1 to 9;
if E102256[I]=1 then do;
if E100235[I]=1 and E100251[I]>=0 then OTHPAY2[I]=E100251[I]; /* with overtime*/
if E100235[I]=2 and E100251[I]>=0 and E98429E[I]>0 and E98429[I]>0 then
    OTHPAY2[I]=E100251[I]*E98429E[I]/E98429[I];
if E100235[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and E100251[I]>=0 and E98429[I]>0 then
    OTHPAY2[I]=E100251[I]/E98429[I];
if E100235[I]=4 and E100251[I]>=0 and E98429[I]>0 then OTHPAY2[I]=E100251[I]/(2*E98429[I]);
if E100235[I]=5 and E100251[I]>=0 and E98429[I]>0 then OTHPAY2[I]=E100251[I]/(4.3*E98429[I]);
if E100235[I]=6 and E100251[I]>=0 and E98429[I]>0 and E99500[I]>0 then
    OTHPAY2[I]=E100251[I]/(E99500[I]*E98429[I]);
if E100235[I]=8 and E100251[I]>=0 and E98429[I]>0 then OTHPAY2[I]=E100251[I]/(2.15*E98429[I]);
if E100235[I] in (9,14) then OTHPAY2[I]=0;
/*missing value*/
if -4<E98429[I]<0 then OTHPAY2[I]=E98429[I];
if -4<E100251[I]<0 then OTHPAY2[I]=E100251[I];
if E98429[I]=0 then OTHPAY2[I]=-3;

if E100235[I]=1 and E100251[I]>=0 then OTHPAY2[I]=E100251[I]; /* without overtime*/
if E100235[I]=2 and E100251[I]>=0 and E98402D[I]>0 and E98402[I]>0 then
    OTHPAY2[I]=E100251[I]*E98402D[I]/E98402[I];
if E100235[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and E100251[I]>=0 and E98402[I]>0 then
    OTHPAY2[I]=E100251[I]/E98402[I];
if E100235[I]=4 and E100251[I]>=0 and E98402[I]>0 then OTHPAY2[I]=E100251[I]/(2*E98402[I]);
if E100235[I]=5 and E100251[I]>=0 and E98402[I]>0 then OTHPAY2[I]=E100251[I]/(4.3*E98402[I]);
if E100235[I]=6 and E100251[I]>=0 and E98402[I]>0 and E99500[I]>0 then
    OTHPAY2[I]=E100251[I]/(E99500[I]*E98402[I]);
if E100235[I]=8 and E100251[I]>=0 and E98402[I]>0 then OTHPAY2[I]=E100251[I]/(2.15*E98402[I]);
if E100235[I] in (9,14) then OTHPAY2[I]=0;
/*missing value*/
if -4<E98402[I]<0 then OTHPAY2[I]=E98402[I];
if E98402[I]=0 then OTHPAY2[I]=-3;
if E100235[I]=2 and E98402D[I] le 0 and E98429E[I] le 0 then OTHPAY2[I]=-3;
if E100235[I]=2 and -4<E98402D[I]<0 then OTHPAY2[I]=E98402D[I];
if E100235[I]=2 and -4<E98429E[I]<0 then OTHPAY2[I]=E98429E[I];
if E100235[I]=6 and E99500[I] le 0 then OTHPAY2[I]=-3;
if E100235[I]=6 and -4<E99500[I]<0 then OTHPAY2[I]=E99500[I];
end;
end;

```

```

***** FOR BONUSES *****/
do I=1 to 9; /* with overtime*/
if E98429[I]>0 and E102053[I]=1 and E102143[I]>=0 then OTHPAY3[I]=E102143[I];
if E98429[I]>0 and E102053[I]=2 and E98429E[I]>0 and E102143[I]>=0 then
    OTHPAY3[I]=(E102143[I]*E98429E[I])/E98429[I];
if E98429[I]>0 and E102053[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and E102143[I]>=0 then
    OTHPAY3[I]=E102143[I]/E98429[I];
if E98429[I]>0 and E102053[I]=4 and E102143[I]>=0 then OTHPAY3[I]=E102143[I]/(2*E98429[I]);
if E98429[I]>0 and E102053[I]=5 and E102143[I]>=0 then OTHPAY3[I]=E102143[I]/(4.3*E98429[I]);
if E98429[I]>0 and E102053[I]=6 and E102143[I]>=0 and E99500[I]>0 then
    OTHPAY3[I]=E102143[I]/(E99500[I]*E98429[I]);
if E98429[I]>0 and E102053[I]=8 and E102143[I]>=0 then OTHPAY3[I]=E102143[I]/(2.15*E98429[I]);
/*missing value*/
if E102143[I]>=0 and -4<E98429[I]<0 then OTHPAY3[I]=E98429[I];
if -4<E102143[I]<0 then OTHPAY3[I]=E102143[I];
if E102143[I]>=0 and E98429[I]=0 then OTHPAY3[I]=-3;
end;

do I=1 to 9; /*without overtime*/
if E98402[I]>0 and E102053[I]=1 and E102143[I]>=0 then OTHPAY3[I]=E102143[I];
if E98402[I]>0 and E102053[I]=2 and E98402D[I]>0 and E102143[I]>=0 then
    OTHPAY3[I]=(E102143[I]*E98402D[I])/E98402[I];
if E98402[I]>0 and E102053[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and E102143[I]>=0 then
    OTHPAY3[I]=E102143[I]/E98402[I];
if E98402[I]>0 and E102053[I]=4 and E102143[I]>=0 then OTHPAY3[I]=E102143[I]/(2*E98402[I]);
if E98402[I]>0 and E102053[I]=5 and E102143[I]>=0 then OTHPAY3[I]=E102143[I]/(4.3*E98402[I]);
if E98402[I]>0 and E102053[I]=6 and E102143[I]>=0 and E99500[I]>0 then
    OTHPAY3[I]=E102143[I]/(E99500[I]*E98402[I]);
if E98402[I]>0 and E102053[I]=8 and E102143[I]>=0 then OTHPAY3[I]=E102143[I]/(2.15*E98402[I]);
if E102053[I] in (0,7,12,13,15,16,17,14,99,21,22,23,24,25,26,28) then OTHPF3[I]=OTHPF3[I]+1;
if E102053[I] in (9,14) then OTHPAY3[I]=0;
/*missing value*/
if E102143[I]>=0 and -4<E98402[I]<0 then OTHPAY3[I]=E98402[I];
if E102143[I]>=0 and E98402[I]=0 then OTHPAY3[I]=-3;
if E102053[I]=2 and E98402D[I]<=0 and E98429E[I]<=0 then OTHPAY3[I]=-3;
if E102053[I]=2 and -4<E98402D[I]<0 then OTHPAY3[I]=E98402D[I];
if E102053[I]=2 and -4<E98429E[I]<0 then OTHPAY3[I]=E98429E[I];
if E102053[I]=6 and E99500[I]<=0 then OTHPAY3[I]=-3;
if E102053[I]=6 and -4<E99500[I]<0 then OTHPAY3[I]=E99500[I];
end;

/* if bonus is corrected in the later part */
do I=1 to 9;
if E102257[I]=1 then do;
if E100236[I]=1 and E100252[I]>=0 then OTHPAY3[I]=E100252[I]; /* with overtime*/
if E100236[I]=2 and E100252[I]>=0 and E98429E[I]>0 and E98429[I]>0 then
    OTHPAY3[I]=E100252[I]*E98429E[I]/E98429[I];
if E100236[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and E100252[I]>=0 and E98429[I]>0 then
    OTHPAY3[I]=E100252[I]/E98429[I];
if E100236[I]=4 and E100252[I]>=0 and E98429[I]>0 then OTHPAY3[I]=E100252[I]/(2*E98429[I]);
if E100236[I]=5 and E100252[I]>=0 and E98429[I]>0 then OTHPAY3[I]=E100252[I]/(4.3*E98429[I]);
if E100236[I]=6 and E100252[I]>=0 and E98429[I]>0 and E99500[I]>0 then
    OTHPAY3[I]=E100252[I]/(E99500[I]*E98429[I]);
if E100236[I]=8 and E100252[I]>=0 and E98429[I]>0 then OTHPAY3[I]=E100252[I]/(2.15*E98429[I]);
if E100236[I] in (9,14) then OTHPAY3[I]=0;
/*missing value*/

```

```

if -4<E98429[I]<0 then OTHPAY3[I]=E98429[I];
if -4<E100252[I]<0 then OTHPAY3[I]=E100252[I];
if E98429[I]=0 then OTHPAY3[I]=-3;

if E100236[I]=1 and E100252[I]>=0 then OTHPAY3[I]=E100252[I]; /* without overtime*/
if E100236[I]=2 and E100252[I]>=0 and E98402D[I]>0 and E98402[I]>0 then
    OTHPAY3[I]=E100252[I]*E98402D[I]/E98402[I];
if E100236[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and E100252[I]>=0 and E98402[I]>0 then
    OTHPAY3[I]=E100252[I]/E98402[I];
if E100236[I]=4 and E100252[I]>=0 and E98402D[I]>0 then OTHPAY3[I]=E100252[I]/(2*E98402D[I]);
if E100236[I]=5 and E100252[I]>=0 and E98402D[I]>0 then OTHPAY3[I]=E100252[I]/(4.3*E98402D[I]);
if E100236[I]=6 and E100252[I]>=0 and E98402D[I]>0 and E99500[I]>0 then
    OTHPAY3[I]=E100252[I]/(E99500[I]*E98402D[I]);
if E100236[I]=8 and E100252[I]>=0 and E98402D[I]>0 then OTHPAY3[I]=E100252[I]/(2.15*E98402D[I]);
if E100236[I] in (9,14) then OTHPAY3[I]=0;
/*missing value*/
if -4<E98402D[I]<0 then OTHPAY3[I]=E98402D[I];
if E98402D[I]=0 then OTHPAY3[I]=-3;
if E100236[I]=2 and E98402D[I]<0 and E98429E[I]<0 then OTHPAY3[I]=-3;
if E100236[I]=2 and -4<E98402D[I]<0 then OTHPAY3[I]=E98402D[I];
if E100236[I]=2 and -4<E98429E[I]<0 then OTHPAY3[I]=E98429E[I];
if E100236[I]=6 and E99500[I]<0 then OTHPAY3[I]=-3;
if E100236[I]=6 and -4<E99500[I]<0 then OTHPAY3[I]=E99500[I];
end;
end;

/***** FOR INCENTIVE PAY *****/
do I=1 to 9; /* with overtime*/
if E98429[I]>0 and E102054[I]=1 and E102144[I]>=0 then OTHPAY4[I]=E102144[I];
if E98429[I]>0 and E102054[I]=2 and E98429E[I]>0 and E102144[I]>=0 then
    OTHPAY4[I]=(E102144[I]*E98429E[I])/E98429[I];
if E98429[I]>0 and E102054[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and E102144[I]>=0 then
    OTHPAY4[I]=E102144[I]/E98429[I];
if E98429[I]>0 and E102054[I]=4 and E102144[I]>=0 then OTHPAY4[I]=E102144[I]/(2*E98429[I]);
if E98429[I]>0 and E102054[I]=5 and E102144[I]>=0 then OTHPAY4[I]=E102144[I]/(4.3*E98429[I]);
if E98429[I]>0 and E102054[I]=6 and E102144[I]>=0 and E99500[I]>0 then
    OTHPAY4[I]=E102144[I]/(E99500[I]*E98429[I]);
if E98429[I]>0 and E102054[I]=8 and E102144[I]>=0 then OTHPAY4[I]=E102144[I]/(2.15*E98429[I]);
/*missing value*/
if E102144[I]>=0 and -4<E98429[I]<0 then OTHPAY4[I]=E98429[I];
if -4<E102144[I]<0 then OTHPAY4[I]=E102144[I];
if E102144[I]>=0 and E98429[I]=0 then OTHPAY4[I]=-3;
end;

do I=1 to 9; /*without overtime*/
if E98402D[I]>0 and E102054[I]=1 and E102144[I]>=0 then OTHPAY4[I]=E102144[I];
if E98402D[I]>0 and E102054[I]=2 and E98402D[I]>0 and E102144[I]>=0 then
    OTHPAY4[I]=(E102144[I]*E98402D[I])/E98402D[I];
if E98402D[I]>0 and E102054[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and E102144[I]>=0 then
    OTHPAY4[I]=E102144[I]/E98402D[I];
if E98402D[I]>0 and E102054[I]=4 and E102144[I]>=0 then OTHPAY4[I]=E102144[I]/(2*E98402D[I]);
if E98402D[I]>0 and E102054[I]=5 and E102144[I]>=0 then OTHPAY4[I]=E102144[I]/(4.3*E98402D[I]);
if E98402D[I]>0 and E102054[I]=6 and E102144[I]>=0 and E99500[I]>0 then
    OTHPAY4[I]=E102144[I]/(E99500[I]*E98402D[I]);
if E98402D[I]>0 and E102054[I]=8 and E102144[I]>=0 then OTHPAY4[I]=E102144[I]/(2.15*E98402D[I]);
if E102054[I] in (0,7,12,13,15,16,17,14,99,21,22,23,24,25,26,28) then OTHPF4[I]=OTHPF4[I]+1;
if E102054[I] in (9,14) then OTHPAY4[I]=0;

```

```

*missing value*;
if E102144[I]>=0 and -4<E98402[I]<0 then OTHPAY4[I]=E98402[I];
if E102144[I]>=0 and E98402[I]=0 then OTHPAY4[I]=-3;
if E102054[I]=2 and E98402D[I] le 0 and E98429E[I] le 0 then OTHPAY4[I]=-3;
if E102054[I]=2 and -4<E98402D[I]<0 then OTHPAY4[I]=E98402D[I];
if E102054[I]=2 and -4<E98429E[I]<0 then OTHPAY4[I]=E98429E[I];
if E102054[I]=6 and E99500[I] le 0 then OTHPAY4[I]=-3;
if E102054[I]=6 and -4<E99500[I]<0 then OTHPAY4[I]=E99500[I];
end;

/* if incentive pay is corrected in the later part */
do I=1 to 9;
if E102258[I]=1 then do;
if E100237[I]=1 and E100253[I]>=0 then OTHPAY4[I]=E100253[I]; /* with overtime*/
if E100237[I]=2 and E100253[I]>=0 and E98429E[I]>0 and E98429[I]>0 then
    OTHPAY4[I]=E100253[I]*E98429E[I]/E98429[I];
if E100237[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and E100253[I]>=0 and E98429[I]>0 then
    OTHPAY4[I]=E100253[I]/E98429[I];
if E100237[I]=4 and E100253[I]>=0 and E98429[I]>0 then OTHPAY4[I]=E100253[I]/(2*E98429[I]);
if E100237[I]=5 and E100253[I]>=0 and E98429[I]>0 then OTHPAY4[I]=E100253[I]/(4.3*E98429[I]);
if E100237[I]=6 and E100253[I]>=0 and E98429[I]>0 and E99500[I]>0 then
    OTHPAY4[I]=E100253[I]/(E99500[I]*E98429[I]);
if E100237[I]=8 and E100253[I]>=0 and E98429[I]>0 then OTHPAY4[I]=E100253[I]/(2.15*E98429[I]);
if E100237[I] in (9,14) then OTHPAY4[I]=0;
/*missing value*/
if -4<E98429[I]<0 then OTHPAY4[I]=E98429[I];
if -4<E100253[I]<0 then OTHPAY4[I]=E100253[I];
if E98429[I]=0 then OTHPAY4[I]=-3;

if E100237[I]=1 and E100253[I]>=0 then OTHPAY4[I]=E100253[I]; /* without overtime*/
if E100237[I]=2 and E100253[I]>=0 and E98402D[I]>0 and E98402[I]>0 then
    OTHPAY4[I]=E100253[I]*E98402D[I]/E98402[I];
if E100237[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and E100253[I]>=0 and E98402[I]>0 then
    OTHPAY4[I]=E100253[I]/E98402[I];
if E100237[I]=4 and E100253[I]>=0 and E98402[I]>0 then OTHPAY4[I]=E100253[I]/(2*E98402[I]);
if E100237[I]=5 and E100253[I]>=0 and E98402[I]>0 then OTHPAY4[I]=E100253[I]/(4.3*E98402[I]);
if E100237[I]=6 and E100253[I]>=0 and E98402[I]>0 and E99500[I]>0 then
    OTHPAY4[I]=E100253[I]/(E99500[I]*E98402[I]);
if E100237[I]=8 and E100253[I]>=0 and E98402[I]>0 then OTHPAY4[I]=E100253[I]/(2.15*E98402[I]);
if E100237[I] in (9,14) then OTHPAY4[I]=0;
/*missing value*/
if -4<E98402[I]<0 then OTHPAY4[I]=E98402[I];
if E98402[I]=0 then OTHPAY4[I]=-3;
if E100237[I]=2 and E98402D[I] le 0 and E98429E[I] le 0 then OTHPAY4[I]=-3;
if E100237[I]=2 and -4<E98402D[I]<0 then OTHPAY4[I]=E98402D[I];
if E100237[I]=2 and -4<E98429E[I]<0 then OTHPAY4[I]=E98429E[I];
if E100237[I]=6 and E99500[I] le 0 then OTHPAY4[I]=-3;
if E100237[I]=6 and -4<E99500[I]<0 then OTHPAY4[I]=E99500[I];
end;
end;

***** FOR OTHERS *****/
do I=1 to 9; /*with overtime*/
if E98429[I]>0 and E102055[I]=1 and E102145[I]>=0 then OTHPAY5[I]=E102145[I];
if E98429[I]>0 and E102055[I]=2 and E98429E[I]>0 and E102145[I]>=0 then
    OTHPAY5[I]=(E102145[I]*E98429E[I])/E98429[I];

```

```

if E98429[I]>0 and E102055[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and E102145[I]>=0 then
    OTHPAY5[I]=E102145[I]/E98429[I];
if E98429[I]>0 and E102055[I]=4 and E102145[I]>=0 then OTHPAY5[I]=E102145[I]/(2*E98429[I]);
if E98429[I]>0 and E102055[I]=5 and E102145[I]>=0 then OTHPAY5[I]=E102145[I]/(4.3*E98429[I]);
if E98429[I]>0 and E102055[I]=6 and E102145[I]>=0 and E99500[I]>0 then
    OTHPAY5[I]=E102145[I]/(E99500[I]*E98429[I]);
if E98429[I]>0 and E102055[I]=8 and E102145[I]>=0 then OTHPAY5[I]=E102145[I]/(2.15*E98429[I]);
/*missing value*/
if E102145[I]>=0 and -4<E98429[I]<0 then OTHPAY5[I]=E98429[I];
if -4<E102145[I]<0 then OTHPAY1[I]=E102145[I];
if E102145[I]>=0 and E98429[I]=0 then OTHPAY5[I]=-3;
end;

do I=1 to 9; /*without overtime*/
if E98402[I]>0 and E102055[I]=1 and E102145[I]>=0 then OTHPAY5[I]=E102145[I];
if E98402[I]>0 and E102055[I]=2 and E98402D[I]>0 and E102145[I]>=0 then
    OTHPAY5[I]=(E102145[I]*E98402D[I])/E98402[I];
if E98402[I]>0 and E102055[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and E102145[I]>=0 then
    OTHPAY5[I]=E102145[I]/E98402[I];
if E98402[I]>0 and E102055[I]=4 and E102145[I]>=0 then OTHPAY5[I]=E102145[I]/(2*E98402[I]);
if E98402[I]>0 and E102055[I]=5 and E102145[I]>=0 then OTHPAY5[I]=E102145[I]/(4.3*E98402[I]);
if E98402[I]>0 and E102055[I]=6 and E102145[I]>=0 and E99500[I]>0 then
    OTHPAY5[I]=E102145[I]/(E99500[I]*E98402[I]);
if E98402[I]>0 and E102055[I]=8 and E102145[I]>=0 then OTHPAY5[I]=E102145[I]/(2.15*E98402[I]);
if E102055[I] in (0,7,12,13,15,16,17,14,99,21,22,23,24,25,26,28) then OTHPF5[I]=OTHPF5[I]+1;
if E102055[I] in (9,14) then OTHPAY5[I]=0;
/*missing value*/;
if E102145[I]>=0 and -4<E98402[I]<0 then OTHPAY5[I]=E98402[I];
if E102145[I]>=0 and E98402[I]=0 then OTHPAY5[I]=-3;
if E102055[I]=2 and E98402D[I] le 0 and E98429E[I] le 0 then OTHPAY5[I]=-3;
if E102055[I]=2 and -4<E98402D[I]<0 then OTHPAY5[I]=E98402D[I];
if E102055[I]=2 and -4<E98429E[I]<0 then OTHPAY5[I]=E98429E[I];
if E102055[I]=6 and E99500[I] le 0 then OTHPAY5[I]=-3;
if E102055[I]=6 and -4<E99500[I]<0 then OTHPAY5[I]=E99500[I];
end;

/* if other compensation is corrected in the later part */
do I=1 to 9;
if E102259[I]=1 then do;
if E100239[I]=1 and E100254[I]>=0 then OTHPAY5[I]=E100254[I]; /* with overtime*/
if E100239[I]=2 and E100254[I]>=0 and E98429E[I]>0 and E98429[I]>0 then
    OTHPAY5[I]=E100254[I]*E98429E[I]/E98429[I];
if E100239[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and E100254[I]>=0 and E98429[I]>0 then
    OTHPAY5[I]=E100254[I]/E98429[I];
if E100239[I]=4 and E100254[I]>=0 and E98429[I]>0 then OTHPAY5[I]=E100254[I]/(2*E98429[I]);
if E100239[I]=5 and E100254[I]>=0 and E98429[I]>0 then OTHPAY5[I]=E100254[I]/(4.3*E98429[I]);
if E100239[I]=6 and E100254[I]>=0 and E98429[I]>0 and E99500[I]>0 then
    OTHPAY5[I]=E100254[I]/(E99500[I]*E98429[I]);
if E100239[I]=8 and E100254[I]>=0 and E98429[I]>0 then OTHPAY5[I]=E100254[I]/(2.15*E98429[I]);
if E100239[I] in (9,14) then OTHPAY5[I]=0;
/*missing value*/
if -4<E98429[I]<0 then OTHPAY5[I]=E98429[I];
if -4<E100254[I]<0 then OTHPAY5[I]=E100254[I];
if E98429[I]=0 then OTHPAY5[I]=-3;

if E100239[I]=1 and E100254[I]>=0 then OTHPAY5[I]=E100254[I]; /* without overtime*/

```

Appendix 2: Employment Variable Creation

```
if E100239[I]=2 and E100254[I]>=0 and E98402D[I]>0 and E98402[I]>0 then
    OTHPAY5[I]=E100254[I]*E98402D[I]/E98402[I];
if E100239[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28,-2) and E100254[I]>=0 and E98402[I]>0 then
    OTHPAY5[I]=E100254[I]/E98402[I];
if E100239[I]=4 and E100254[I]>=0 and E98402[I]>0 then OTHPAY5[I]=E100254[I]/(2*E98402[I]);
if E100239[I]=5 and E100254[I]>=0 and E98402[I]>0 then OTHPAY5[I]=E100254[I]/(4.3*E98402[I]);
if E100239[I]=6 and E100254[I]>=0 and E98402[I]>0 and E99500[I]>0 then
    OTHPAY5[I]=E100254[I]/(E99500[I]*E98402[I]);
if E100239[I]=8 and E100254[I]>=0 and E98402[I]>0 then OTHPAY5[I]=E100254[I]/(2.15*E98402[I]);
if E100239[I] in (9,14) then OTHPAY5[I]=0;
/*missing value*/
if -4<E98402[I]<0 then OTHPAY5[I]=E98402[I];
if E98402[I]=0 then OTHPAY5[I]=-3;
if E100239[I]=2 and E98402D[I] le 0 and E98429E[I] le 0 then OTHPAY5[I]=-3;
if E100239[I]=2 and -4<E98402D[I]<0 then OTHPAY5[I]=E98402D[I];
if E100239[I]=2 and -4<E98429E[I]<0 then OTHPAY5[I]=E98429E[I];
if E100239[I]=6 and E99500[I] le 0 then OTHPAY5[I]=-3;
if E100239[I]=6 and -4<E99500[I]<0 then OTHPAY5[I]=E99500[I];
end;
end;
```

***** Part 6: OVERALL START HOURLY COMPENSATION *****

```
array HRCOMP HRCOMP01 HRCOMP02 HRCOMP03 HRCOMP04 HRCOMP05 HRCOMP06
HRCOMP07 HRCOMP08 HRCOMP09;
do I=1 to 9;
    HRCOMP[I]=0;
    if HRWG[I] ge 0 then HRCOMP[I]=HRCOMP[I]+HRWG[I];
    if OT[I] ge 0 then HRCOMP[I]=HRCOMP[I]+OT[I];
    if OTHPAY1[I] ge 0 then HRCOMP[I]=HRCOMP[I]+OTHPAY1[I];
    if OTHPAY2[I] ge 0 then HRCOMP[I]=HRCOMP[I]+OTHPAY2[I];
    if OTHPAY3[I] ge 0 then HRCOMP[I]=HRCOMP[I]+OTHPAY3[I];
    if OTHPAY4[I] ge 0 then HRCOMP[I]=HRCOMP[I]+OTHPAY4[I];
    if OTHPAY5[I] ge 0 then HRCOMP[I]=HRCOMP[I]+OTHPAY5[I];
    if -4<HRWG[I]<0 or -4<OT[I]<0 or -4<OTHPAY1[I]<0 or -4<OTHPAY2[I]<0 or -4<OTHPAY3[I]<0 or -
        4<OTHPAY4[I]<0 or -4<OTHPAY5[I]<0 then HRCOMP[I]=-3;
    if HRWG[I]=-4 then HRCOMP[I]=-4;
end;
```

***** Section 2: END WAGES FOR YOUTH *****

***** Part 1: End hourly rate of pay *****

```
/*Respondents reporting an HOURLY wage*/
do I=1 to 9;
if (E37901B[I]=1 or E59900[I]=1) then do;
    if (E38013[I]=1 and E38014[I]=1) then do; /* without overtime at the beginning */
        if E38023[I]>=0 then HRWAGE[I]=E38023[I];
        /*missing value*/
        if -4<E38023[I]<0 then HRWAGE[I]=E38023[I];
    end;
    if (E38106[I]=1 and E38107[I]=1) then do; /* with overtime at the beginning */
        if E38116[I]>=0 then HRWAGE[I]=E38116[I];
        /*missing value*/ if -4<E38106[I]<0 then HRWAGE[I]=E38106[I];
    end;
end;
```

```

end;

/*Respondents reporting a DAILY wage*/
do I=1 to 9;
if (E37901B[I]=1 or E59900[I]=1) then do;
  if (E38013[I]=1 and E38014[I]=2) then do; /*without overtime at the beginning*/
    if E38023[I]>=0 and E3800B[I]=1 and E38027[I]>0 and E34402[I]>0 then
      DAILY[I]=E38023[I]*E38027[I]/E34402[I];
    if E38023[I]>=0 and E3800F[I]>0 and E38027[I]>0 then DAILY[I]=E38023[I]*E38027[I]/E3800F[I];
    /*missing value*/
    if E38023[I]>=0 and -4<E38027[I]<0 then DAILY[I]=E38027[I];
    if E38023[I]>=0 and E3800B[I]=1 and -4<E34402[I]<0 then DAILY[I]=E34402[I];
    if E38023[I]>=0 and -4<E3800F[I]<0 then DAILY[I]=E3800F[I];
    if E38023[I]>=0 and (E34402[I]=0 or E3800F[I]=0) then DAILY[I]=-3;
  end;
  if (E38106[I]=1 and E38107[I]=2) then do; /*with overtime at the beginning*/
    if E38116[I]>=0 and (E38102[I] ne 1 and E38102[I] ne 3) and E38116B[I]>0 and E34428[I]>0 then
      DAILY[I]=E38116B[I]*E38116[I]/E34428[I];
    if E38116[I]>=0 and (E38102[I]=1 or E38102[I]=3) and E38103[I]>0 and E38116B[I]>0 then
      DAILY[I]=E38116B[I]*E38116[I]/E38103[I];
    /*missing value*/
    if E38116[I]>=0 and -4<E38116B[I]<0 then DAILY[I]=E38116B[I];
    if E38116[I]>=0 and (E38102[I] ne 1 and E38102[I] ne 3) and -4<E34428[I]<0 then DAILY[I]=E34428[I];
    if E38116[I]>=0 and (E38102[I]=1 or E38102[I]=3) and -4<E38103[I]<0 then DAILY[I]=E38103[I];
    if E38116[I]>=0 and (E34428[I]=0 or E38103[I]=0) then DAILY[I]=-3;
  end;
end;
end;

/*Respondents reporting a WEEKLY wage*/
do I=1 to 9;
if (E37901B[I]=1 or E59900[I]=1) then do;
/*without overtime at the beginning */
  if (E38013[I]=1 and E38014[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28)) then do;
    if E38023[I]>=0 and E3800B[I]=1 and E34402[I]>0 then WEEKLY[I]=E38023[I]/E34402[I];
    if E38023[I]>=0 and E3800F[I]>0 then WEEKLY[I]=E38023[I]/E3800F[I];
    /*missing value*/
    if E38023[I]>=0 and E3800B[I]=1 and -4<E34402[I]<0 then WEEKLY[I]=E34402[I];
    if E38023[I]>=0 and -4<E3800F[I]<0 then WEEKLY[I]=E3800F[I];
    if E38023[I]>=0 and (E34402[I]=0 or E3800F[I]=0) then WEEKLY[I]=-3;
  end;
/* with overtime at the beginning */
  if (E38106[I]=1 and E38107[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28)) then do;
    if E38116[I]>=0 and (E38102[I] ne 1 and E38102[I] ne 3) and E34428[I]>0 then
      WEEKLY[I]=E38116[I]/E34428[I];
    if E38116[I]>=0 and (E38102[I]=1 or E38102[I]=3) and E38103[I]>0 then WEEKLY[I]=E38116[I]/E38103[I];
    /*missing value*/
    if E38116[I]>=0 and (E38102[I] ne 1 and E38102[I] ne 3) and -4<E34428[I]<0 then WEEKLY[I]=E34428[I];
    if E38116[I]>=0 and (E38102[I]=1 or E38102[I]=3) and -4<E38103[I]<0 then WEEKLY[I]=E38103[I];
    if E38116[I]>=0 and (E34428[I]=0 or E38103[I]=0) then WEEKLY[I]=-3;
  end;
end;
end;

/*Respondents reporting a BIWEEKLY wage*/
do I=1 to 9;
if (E37901B[I]=1 or E59900[I]=1) then do;

```

Appendix 2: Employment Variable Creation

```
if (E38013[I]=1 and E38014[I]=4) then do; /* without overtime at the beginning */
  if E38023[I]>=0 and E3800B[I]=1 and E34402[I]>0 then BIWKLY[I]=E38023[I]/(2*E34402[I]);
  if E38023[I]>=0 and E3800F[I]>0 then BIWKLY[I]=E38023[I]/(2*E3800F[I]);
  /*missing value*/
  if E38023[I]>=0 and E3800B[I]=1 and -4<E34402[I]<0 then BIWKLY[I]=E34402[I];
  if E38023[I]>=0 and -4<E3800F[I]<0 then BIWKLY[I]=E3800F[I];
  if E38023[I]>=0 and (E34402[I]=0 or E3800F[I]=0) then BIWKLY[I]=-3;
end;
if (E38106[I]=1 and E38107[I]=4) then do; /* with overtime at the beginning*/
  if E38116[I]>=0 and (E38102[I] ne 1 and E38102[I] ne 3) and E34428[I]>0 then
    BIWKLY[I]=E38116[I]/(2*E34428[I]);
  if E38116[I]>=0 and (E38102[I]=1 or E38102[I]=3) and E38103[I]>0 then
    BIWKLY[I]=E38116[I]/(2*E38103[I]);
  /*missing value*/
  if E38116[I]>=0 and (E38102[I] ne 1 and E38102[I] ne 3) and -4<E34428[I]<0 then BIWKLY[I]=E34428[I];
  if E38116[I]>=0 and (E38102[I]=1 or E38102[I]=3) and -4<E38103[I]<0 then BIWKLY[I]=E38103[I];
  if E38116[I]>=0 and (E34428[I]=0 or E38103[I]=0) then BIWKLY[I]=-3;
end;
end;
/*Respondents reporting a MONTHLY wage*/
do I=1 to 9;
if (E37901B[I]=1 or E59900[I]=1) then do;
  if (E38013[I]=1 and E38014[I]=5) then do; /* without overtime at the beginning */
    if E38023[I]>=0 and E3800B[I]=1 and E34402[I]>0 then MONTH[I]=E38023[I]/(4.3*E34402[I]);
    if E38023[I]>=0 and E3800F[I]>0 then MONTH[I]=E38023[I]/(4.3*E3800F[I]);
    /*missing value*/
    if E38023[I]>=0 and E3800B[I]=1 and -4<E34402[I]<0 then MONTH[I]=E34402[I];
    if E38023[I]>=0 and -4<E3800F[I]<0 then MONTH[I]=E3800F[I];
    if E38023[I]>=0 and (E34402[I]=0 or E3800F[I]=0) then MONTH[I]=-3;
  end;
  if (E38106[I]=1 and E38107[I]=5) then do; /* with overtime at the beginning */
    if E38116[I]>=0 and (E38102[I] ne 1 and E38102[I] ne 3) and E34428[I]>0 then
      MONTH[I]=E38116[I]/(4.3*E34428[I]);
    if E38116[I]>=0 and (E38102[I]=1 or E38102[I]=3) and E38103[I]>0 then
      MONTH[I]=E38116[I]/(4.3*E38103[I]);
    /*missing value*/
    if E38116[I]>=0 and (E38102[I] ne 1 and E38102[I] ne 3) and -4<E34428[I]<0 then MONTH[I]=E34428[I];
    if E38116[I]>=0 and (E38102[I]=1 or E38102[I]=3) and -4<E38103[I]<0 then MONTH[I]=E38103[I];
    if E38116[I]>=0 and (E34428[I]=0 or E38103[I]=0) then MONTH[I]=-3;
  end;
end;
/*Respondents reporting an ANNUAL wage*/
do I=1 to 9;
if (E37901B[I]=1 or E59900[I]=1) then do;
  if (E38013[I]=1 and E38014[I]=6) then do; /* without overtime at the beginning */
    if E38023[I]>=0 and E3800B[I]=1 and E34402[I]>0 and E35600[I]>0 then
      ANNUAL[I]=E38023[I]/(E35600[I]*E34402[I]);
    if E38023[I]>=0 and E3800F[I]>0 and E35600[I]>0 then ANNUAL[I]=E38023[I]/(E35600[I]*E3800F[I]);
    /*missing value*/
    if E38023[I]>=0 and E3800B[I]=1 and -4<E34402[I]<0 then ANNUAL[I]=E34402[I];
    if E38023[I]>=0 and -4<E3800F[I]<0 then ANNUAL[I]=E3800F[I];
    if E38023[I]>=0 and (E34402[I]=0 or E3800F[I]=0) then ANNUAL[I]=-3;
    if E35600[I] le 0 then ANNUAL[I]=-3;
```

```

if -4<E35600[I]<0 then ANNUAL[I]=E35600[I];
end;
if (E38106[I]=1 and E38107[I]=6) then do; /* with overtime at the beginning */
if E38116[I]>=0 and (E38102[I] ne 1 and E38102[I] ne 3) and E34428[I]>0 and E35600[I]>0 then
    ANNUAL[I]=E38116[I]/(E35600[I]*E34428[I]);
if E38116[I]>=0 and (E38102[I]=1 or E38102[I]=3) and E38103[I]>0 and E35600[I]>0 then
    ANNUAL[I]=E38116[I]/(E35600[I]*E38103[I]);
/*missing value*/
if E38116[I]>=0 and (E38102[I] ne 1 and E38102[I] ne 3) and -4<E34428[I]<0 then ANNUAL[I]=E34428[I];
if E38116[I]>=0 and (E38102[I]=1 or E38102[I]=3) and -4<E38103[I]<0 then ANNUAL[I]=E38103[I];
if E38116[I]>=0 and (E34428[I]=0 or E38103[I]=0) then ANNUAL[I]=-3;
if E35600[I] le 0 then ANNUAL[I]=-3;
if -4<E35600[I]<0 then ANNUAL[I]=E35600[I];
end;
end;
end;

/*Respondents reporting a SEMIMONTHLY wage*/
do I=1 to 9;
if (E37901B[I]=1 or E59900[I]=1) then do;
if (E38013[I]=1 and E38014[I]=8) then do; /* without overtime at the beginning */
if E38023[I]>=0 and E3800B[I]=1 and E34402[I]>0 then SEMIM[I]=E38023[I]/(2.15*E34402[I]);
if E38023[I]>=0 and E3800F[I]>0 then SEMIM[I]=E38023[I]/(2.15*E3800F[I]);
/*missing value*/
if E38023[I]>=0 and E3800B[I]=1 and -4<E34402[I]<0 then SEMIM[I]=E34402[I];
if E38023[I]>=0 and -4<E3800F[I]<0 then SEMIM[I]=E3800F[I];
if E38023[I]>=0 and (E34402[I]=0 or E3800F[I]=0) then SEMIM[I]=-3;
end;
if (E38106[I]=1 and E38107[I]=8) then do; /* with overtime at the beginning */
if E38116[I]>=0 and (E38102[I] ne 1 and E38102[I] ne 3) and E34428[I]>0 then
    SEMIM[I]=E38116[I]/(2.15*E34428[I]);
if E38116[I]>=0 and (E38102[I]=1 or E38102[I]=3) and E38103[I]>0 then
    SEMIM[I]=E38116[I]/(2.15*E38103[I]);
/*missing value*/
if E38116[I]>=0 and (E38102[I] ne 1 and E38102[I] ne 3) and -4<E34428[I]<0 then SEMIM[I]=E34428[I];
if E38116[I]>=0 and (E38102[I]=1 or E38102[I]=3) and -4<E38103[I]<0 then SEMIM[I]=E38103[I];
if E38116[I]>=0 and (E34428[I]=0 or E38103[I]=0) then SEMIM[I]=-3;
end;
end;
end;
end;

```

***** Part 2: Create Hourly Rate of Pay based on the from end wage *****

```

do I=1 to 9;
if E37901B[I]=1 or E59900[I]=1 then do;
if ANNUAL[I] ge 0 then HRWG[I]=ANNUAL[I];
if MONTH[I] ge 0 then HRWG[I]=MONTH[I];
if BIWKLY[I] ge 0 then HRWG[I]=BIWKLY[I];
if WEEKLY[I] ge 0 then HRWG[I]=WEEKLY[I];
if DAILY[I] ge 0 then HRWG[I]=DAILY[I];
if HRWAGE[I] ge 0 then HRWG[I]=HRWAGE[I];
if SEMIM[I] ge 0 then HRWG[I]=SEMIM[I];
if HRWAGE[I] eq -1 or DAILY[I]==-1 or WEEKLY[I] eq -1 or BIWKLY[I] eq -1 or MONTH[I] eq -1 or
    ANNUAL[I] eq -1 or SEMIM[I]==-1 then HRWG[I]=-1;
if HRWAGE[I] eq -2 or DAILY[I]==-2 or WEEKLY[I] eq -2 or BIWKLY[I] eq -2 or MONTH[I] eq -2 or
    ANNUAL[I] eq -2 or SEMIM[I]==-2 then HRWG[I]=-2;

```

Appendix 2: Employment Variable Creation

```
if HRWAGE[I] eq -3 or DAILY[I]=-3 or WEEKLY[I] eq -3 or BIWKLY[I] eq -3 or MONTH[I] eq -3 or
    ANNUAL[I] eq -3 or SEMIM[I]=-3 then HRWG[I]=-3;
end;
end;

/*set up the hourly wage for youths who report their wage in other manners */
do I=1 to 9.;
if (E37901B[I]=1 or E59900[I]=1) then do;

/* if job was not offering any comp at the end */
if (E38013[I]=1 and E38014[I] in (7,0,12,13,14,15,16,17,99,21,22,23,24,25,26,28)) then
    OTHERF[I]=OTHERF[I]+1;
if (E38013[I]=1 and E38014[I] in (9,14)) then HRWG[I]=0;
if E38013[I]=1 and -4<E38023[I]<0 then HRWG[I]=E38023[I];
/* if job was offering a comp at the end */
if (E38106[I]=1 and E38107[I] in (7,0,12,13,14,15,16,17,99,21,22,23,24,25,26,28)) then
    OTHERF[I]=OTHERF[I]+1;
if (E38106[I]=1 and E38107[I] in (9,14)) then HRWG[I]=0;
if E38106[I]=1 and -4<E38116[I]<0 then HRWG[I]=E38116[I];
end;
end;

/* REPORT -1,-2 or -3 if THE AMOUNT REPORTED IS -1, -2 or -3*/
do I=1 to 9;
if E37901B[I]=1 or E59900[I]=1 then do;
    if -4<E38023[I]<0 then HRWG[I]=E38023[I];
    if -4<E38116[I]<0 then HRWG[I]=E38116[I];
end;
end;

/* The end wage if job lasts for >=13 weeks and report the same amount but diff. hours. NOTE: For this case, we
only change hourly rate HRWG[I] without changing HRWAGE[I], DAILY[I], WEEKLW[I] BIWKLY[I],
MONTH[I], ANNUAL[I] or SEMIM[I]*/
do I=1 to 9;
if ((E37901B[I]=1 or E59900[I]=1) and E38013[I]=0 and E3800B[I]=0) and HRWG[I]>=0 and E34402[I]>0 and
    E3800F[I]>0 then HRWG[I]=HRWG[I]*E34402[I]/E3800F[I];
if ((E37901B[I]=1 or E59900[I]=1) and E38013[I]=0 and E3800B[I]=0) and E3800F[I]=0 then HRWG[I]=-3;
if ((E37901B[I]=1 or E59900[I]=1) and E38013[I]=0 and E3800B[I]=0) and -4<E3800F[I]<0 then
    HRWG[I]=E3800F[I];
if ((E37901B[I]=1 or E59900[I]=1) and E38106[I]=0 and (E38102[I]=1 or E38102[I]=3)) and HRWG[I]>=0 and
    E34428[I]>0 and E38103[I]>0 then HRWG[I]=HRWG[I]*E34428[I]/E38103[I];
if ((E37901B[I]=1 or E59900[I]=1) and E38106[I]=0 and (E38102[I]=1 or E38102[I]=3)) and HRWG[I]>=0 and
    E34402[I]>0 and E38103[I]>0 then HRWG[I]=HRWG[I]*E34402[I]/E38103[I];
if ((E37901B[I]=1 or E59900[I]=1) and E38106[I]=0 and (E38102[I]=1 or E38102[I]=3)) and E38103[I]=0 then
    HRWG[I]=-3;
if ((E37901B[I]=1 or E59900[I]=1) and E38106[I]=0 and (E38102[I]=1 or E38102[I]=3)) and -4<E38103[I]<0
    then HRWG[I]=E38103[I];
end;

***** Part 3: Add the end compensation *****
***** With paid overtime *****
do I=1 to 9;
if (E37901B[I]=1 or E59900[I]=1) then do;

if E38001[I]=0 then OT[I]=0; ***** without compensation at the beginning *****
```

```

else do;
if E38002[I]>0 and E38003[I]=1 and E38012[I]>=0 then OT[I]=E38012[I];
if E38002[I]>0 and E38003[I]=2 and E38012[I]>=0 and E38012B[I]>0 then
    OT[I]=E38012[I]*E38012B[I]/E38002[I];
if E38002[I]>0 and E38003[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E38012[I]>=0 then
    OT[I]=E38012[I]/E38002[I];
if E38002[I]>0 and E38003[I]=4 and E38012[I]>=0 then OT[I]=E38012[I]/(2*E38002[I]);
if E38002[I]>0 and E38003[I]=5 and E38012[I]>=0 then OT[I]=E38012[I]/(4.3*E38002[I]);
if E38002[I]>0 and E38003[I]=6 and E38012[I]>=0 then OT[I]=-3; /* Since no weeks per year available. */
if E38002[I]>0 and E38003[I]=8 and E38012[I]>=0 then OT[I]=E38012[I]/(2.15*E38002[I]);
if E38003[I] in (9,14) then OT[I]=0;
if E38002[I]>0 and E38003[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then OTF[I]=OTF[I]+1;
/*missing value*/
if -4<E38002[I]<0 then OT[I]=E38002[I];
if E38012[I]>=0 and E38002[I]=0 then OT[I]=-3;
if -4<E38012[I]<0 then OT[I]=E38012[I];
end;

if E38201[I]=1 then do; /***** with compensation at the beginning *****/
if E38101[I]=1 then do; /*same no. of hours as at the beginning */
if E19200[I]=1 then do; /* report hourly rate of pay at the beginning */
    if E24501[I]>0 and E38202[I]=1 and E38211[I]>=0 then OT[I]=E38211[I];
    if E24501[I]>0 and E38202[I]=2 and E38211B[I]>0 and E38211[I]>=0 then
        OT[I]=E38211[I]*E38211B[I]/E24501[I];
    if E24501[I]>0 and E38202[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E38211[I]>=0 then
        OT[I]=E38211[I]/E24501[I];
    if E24501[I]>0 and E38202[I]=4 and E38211[I]>=0 then OT[I]=E38211[I]/(2*E24501[I]);
    if E24501[I]>0 and E38202[I]=5 and E38211[I]>=0 then OT[I]=E38211[I]/(4.3*E24501[I]);
    if E24501[I]>0 and E38202[I]=6 and E38211[I]>=0 then OT[I]=-3; /* Since no weeks per year available. */
    if E24501[I]>0 and E38202[I]=8 and E38211[I]>=0 then OT[I]=E38211[I]/(2.15*E24501[I]);
    if E38202[I] in (9,14) then OT[I]=0;
    if E24501[I]>0 and E38202[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then OTF[I]=OTF[I]+1;
    /*missing value*/;
    if -4<E24501[I]<0 then OT[I]=E24501[I];
    if E38211[I]>=0 and E24501[I]=0 then OT[I]=-3;
    if -4<E38211[I]<0 then OT[I]=E38211[I];
end;

if E19200[I] ne 1 then do; /*report payment in other units at the begining*/
if E34403[I]>0 and E38202[I]=1 and E38211[I]>=0 then OT[I]=E38211[I];
if E34403[I]>0 and E38202[I]=2 and E38211B[I]>0 and E38211[I]>=0 then
    OT[I]=E38211[I]*E38211B[I]/E34403[I];
if E34403[I]>0 and E38202[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E38211[I]>=0 then
    OT[I]=E38211[I]/E34403[I];
if E34403[I]>0 and E38202[I]=4 and E38211[I]>=0 then OT[I]=E38211[I]/(2*E34403[I]);
if E34403[I]>0 and E38202[I]=5 and E38211[I]>=0 then OT[I]=E38211[I]/(4.3*E34403[I]);
if E34403[I]>0 and E38202[I]=6 and E38211[I]>=0 then OT[I]=-3; /* Since no weeks per year available. */
if E34403[I]>0 and E38202[I]=8 and E38211[I]>=0 then OT[I]=E38211[I]/(2.15*E34403[I]);
if E38202[I] in (9,14) then OT[I]=0;
if E34403[I]>0 and E38202[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then OTF[I]=OTF[I]+1;
/*missing value*/;
if -4<E34403[I]<0 then OT[I]=E34403[I];
if E38211[I]>=0 and E34403[I]=0 then OT[I]=-3;
if -4<E38211[I]<0 then OT[I]=E38211[I];
end;
end;

```

```

else do; /* different no. of hours from the beginning*/
if E38105[I]>0 and E38202[I]=1 and E38211[I]>=0 then OT[I]=E38211[I];
if E38105[I]>0 and E38202[I]=2 and E38211B[I]>0 and E38211[I]>=0 then
    OT[I]=E38211[I]*E38211B[I]/E38105[I];
if E38105[I]>0 and E38202[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E38211[I]>=0 then
    OT[I]=E38211[I]/E38105[I];
if E38105[I]>0 and E38202[I]=4 and E38211[I]>=0 then OT[I]=E38211[I]/(2*E38105[I]);
if E38105[I]>0 and E38202[I]=5 and E38211[I]>=0 then OT[I]=E38211[I]/(4.3*E38105[I]);
if E38105[I]>0 and E38202[I]=6 and E38211[I]>=0 then OT[I]=-3; /* Since no weeks per year available. */
if E38105[I]>0 and E38202[I]=8 and E38211[I]>=0 then OT[I]=E38211[I]/(2.15*E38105[I]);
if E38202[I] in (9,14) then OT[I]=0;
if E38105[I]>0 and E38202[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then OTF[I]=OTF[I]+1;
/*missing value*/;
if -4<E38105[I]<0 then OT[I]=E38105[I];
if E38211[I]>=0 and E38105[I]=0 then OT[I]=-3;
if -4<E38211[I]<0 then OT[I]=E38211[I];
end;
end;
end;
end;

***** non-overtime payment *****
do I=1 to 9;
if (E37901B[I]=1 or E59900[I]=1) then do;

*** case i. without overtime at the beginning, same no. of hours. ***
if E38313[I]=1 and E3800B[I]=1 and E212001[I] ne 1 then do;

/* report hourly wage at the beginning*/
/* for tips*/
if E23901[I]>0 and E384071[I]=1 and E384161[I]>=0 then OTHPAY1[I]=E384161[I];
if E23901[I]>0 and E384071[I]=2 and E34402B[I]>0 and E384161[I]>=0 then
    OTHPAY1[I]=(E384161[I]*E34402B[I])/E23901[I];
if E23901[I]>0 and E384071[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E384161[I]>=0 then
    OTHPAY1[I]=E384161[I]/E23901[I];
if E23901[I]>0 and E384071[I]=4 and E384161[I]>=0 then OTHPAY1[I]=E384161[I]/(2*E23901[I]);
if E23901[I]>0 and E384071[I]=5 and E384161[I]>=0 then OTHPAY1[I]=E384161[I]/(4.3*E23901[I]);
if E23901[I]>0 and E384071[I]=6 and E384161[I]>=0 and E35600[I]>0 then
    OTHPAY1[I]=(E384161[I]/(E35600[I])*E23901[I]);
if E23901[I]>0 and E384071[I]=8 and E384161[I]>=0 then OTHPAY1[I]=E384161[I]/(2.15*E23901[I]);
if E23901[I]>0 and E384071[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF1[I]=OTHPF1[I]+1;
if E384071[I] in (9,14) then OTHPAY1[I]=0;
/*missing value*/;
if E384161[I]>=0 and -4<E23901[I]<0 then OTHPAY1[I]=E23901[I];
if E384161[I]>=0 and E23901[I]=0 then OTHPAY1[I]=-3;
if -4<E384161[I]<0 then OTHPAY1[I]=E384161[I];
if E384071[I]=2 and E34402B[I] le 0 then OTHPAY1[I]=-3;
if E384071[I]=2 and -4<E34402B[I]<0 then OTHPAY1[I]=E34402B[I];
if E384071[I]=6 and E35600[I] le 0 then OTHPAY1[I]=-3;
if E384071[I]=6 and -4<E35600[I]<0 then OTHPAY1[I]=E35600[I];

/*for commissions*/
if E23901[I]>0 and E384072[I]=1 and E384162[I]>=0 then OTHPAY2[I]=E384162[I];
if E23901[I]>0 and E384072[I]=2 and E34402B[I]>0 and E384162[I]>=0 then
    OTHPAY2[I]=(E384162[I]*E34402B[I])/E23901[I];

```

```

if E23901[I]>0 and E384072[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E384162[I]>=0 then
    OTHPAY2[I]=E384162[I]/E23901[I];
if E23901[I]>0 and E384072[I]=4 and E384162[I]>=0 then OTHPAY2[I]=E384162[I]/(2*E23901[I]);
if E23901[I]>0 and E384072[I]=5 and E384162[I]>=0 then OTHPAY2[I]=E384162[I]/(4.3*E23901[I]);
if E23901[I]>0 and E384072[I]=6 and E384162[I]>=0 and E35600[I]>0 then
    OTHPAY2[I]=E384162[I]/(E35600[I]*E23901[I]);
if E23901[I]>0 and E384072[I]=8 and E384162[I]>=0 then OTHPAY2[I]=E384162[I]/(2.15*E23901[I]);
if E23901[I]>0 and E384072[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF2[I]=OTHPF2[I]+1;
if E384072[I] in (9,14) then OTHPAY2[I]=0;
/*missing value*/;
if E384162[I]>=0 and -4<E23901[I]<0 then OTHPAY2[I]=E23901[I];
if E384162[I]>=0 and E23901[I]=0 then OTHPAY2[I]=-3;
if -4<E384162[I]<0 then OTHPAY2[I]=E384162[I];
if E384072[I]=2 and E34402B[I] le 0 then OTHPAY2[I]=-3;
if E384072[I]=2 and -4<E34402B[I]<0 then OTHPAY2[I]=E34402B[I];
if E384072[I]=6 and E35600[I] le 0 then OTHPAY2[I]=-3;
if E384072[I]=6 and -4<E35600[I]<0 then OTHPAY2[I]=E35600[I];

/*for bonuses*/
if E23901[I]>0 and E384073[I]=1 and E384163[I]>=0 then OTHPAY3[I]=E384163[I];
if E23901[I]>0 and E384073[I]=2 and E34402B[I]>0 and E384163[I]>=0 then
    OTHPAY3[I]=(E384163[I]*E34402B[I])/E23901[I];
if E23901[I]>0 and E384073[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E384163[I]>=0 then
    OTHPAY3[I]=E384163[I]/E23901[I];
if E23901[I]>0 and E384073[I]=4 and E384163[I]>=0 then OTHPAY3[I]=E384163[I]/(2*E23901[I]);
if E23901[I]>0 and E384073[I]=5 and E384163[I]>=0 then OTHPAY3[I]=E384163[I]/(4.3*E23901[I]);
if E23901[I]>0 and E384073[I]=6 and E384163[I]>=0 and E35600[I]>0 then
    OTHPAY3[I]=E384163[I]/(E35600[I]*E23901[I]);
if E23901[I]>0 and E384073[I]=8 and E384163[I]>=0 then OTHPAY3[I]=E384163[I]/(2.15*E23901[I]);
if E23901[I]>0 and E384073[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF3[I]=OTHPF3[I]+1;
if E384073[I] in (9,14) then OTHPAY3[I]=0;
/*missing value*/;
if E384163[I]>=0 and -4<E23901[I]<0 then OTHPAY3[I]=E23901[I];
if E384163[I]>=0 and E23901[I]=0 then OTHPAY3[I]=-3;
if -4<E384163[I]<0 then OTHPAY3[I]=E384163[I];
if E384073[I]=2 and E34402B[I] le 0 then OTHPAY3[I]=-3;
if E384073[I]=2 and -4<E34402B[I]<0 then OTHPAY3[I]=E34402B[I];
if E384073[I]=6 and E35600[I] le 0 then OTHPAY3[I]=-3;
if E384073[I]=6 and -4<E35600[I]<0 then OTHPAY3[I]=E35600[I];

/*for incentive pay*/
if E23901[I]>0 and E384074[I]=1 and E384164[I]>=0 then OTHPAY4[I]=E384164[I];
if E23901[I]>0 and E384074[I]=2 and E34402B[I]>0 and E384164[I]>=0 then
    OTHPAY4[I]=(E384164[I]*E34402B[I])/E23901[I];
if E23901[I]>0 and E384074[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E384164[I]>=0 then
    OTHPAY4[I]=E384164[I]/E23901[I];
if E23901[I]>0 and E384074[I]=4 and E384164[I]>=0 then OTHPAY4[I]=E384164[I]/(2*E23901[I]);
if E23901[I]>0 and E384074[I]=5 and E384164[I]>=0 then OTHPAY4[I]=E384164[I]/(4.3*E23901[I]);
if E23901[I]>0 and E384074[I]=6 and E384164[I]>=0 and E35600[I]>0 then
    OTHPAY4[I]=E384164[I]/(E35600[I]*E23901[I]);
if E23901[I]>0 and E384074[I]=8 and E384164[I]>=0 then OTHPAY4[I]=E384164[I]/(2.15*E23901[I]);
if E23901[I]>0 and E384074[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF4[I]=OTHPF4[I]+1;
if E384074[I] in (9,14) then OTHPAY4[I]=0;
/*missing value*/;
```

```

if E384164[I]>=0 and -4<E23901[I]<0 then OTHPAY4[I]=E23901[I];
if E384164[I]>=0 and E23901[I]=0 then OTHPAY4[I]=-3;
if -4<E384164[I]<0 then OTHPAY4[I]=E384164[I];
if E384074[I]=2 and E34402B[I] le 0 then OTHPAY4[I]=-3;
if E384074[I]=2 and -4<E34402B[I]<0 then OTHPAY4[I]=E34402B[I];
if E384074[I]=6 and E35600[I] le 0 then OTHPAY4[I]=-3;
if E384074[I]=6 and -4<E35600[I]<0 then OTHPAY4[I]=E35600[I];

/*for others*/
if E23901[I]>0 and E384075[I]=1 and E384165[I]>=0 then OTHPAY5[I]=E384165[I];
if E23901[I]>0 and E384075[I]=2 and E34402B[I]>0 and E384165[I]>=0 then
    OTHPAY5[I]=(E384165[I]*E34402B[I])/E23901[I];
if E23901[I]>0 and E384075[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E384165[I]>=0 then
    OTHPAY5[I]=E384165[I]/E23901[I];
if E23901[I]>0 and E384075[I]=4 and E384165[I]>=0 then OTHPAY5[I]=E384165[I]/(2*E23901[I]);
if E23901[I]>0 and E384075[I]=5 and E384165[I]>=0 then OTHPAY5[I]=E384165[I]/(4.3*E23901[I]);
if E23901[I]>0 and E384075[I]=6 and E384165[I]>=0 and E35600[I]>0 then
    OTHPAY5[I]=E384165[I]/(E35600[I]*E23901[I]);
if E23901[I]>0 and E384075[I]=8 and E384165[I]>=0 then OTHPAY5[I]=E384165[I]/(2.15*E23901[I]);
if E23901[I]>0 and E384075[I] in (0,7,12,13,14,15,16,17,00,21,22,23,24,25,26,28) then
    OTHPF5[I]=OTHPF5[I]+1;
if E384075[I] in (9,14) then OTHPAY5[I]=0;
/*missing value*/;

if E384165[I]>=0 and -4<E23901[I]<0 then OTHPAY5[I]=E23901[I];
if E384165[I]>=0 and E23901[I]=0 then OTHPAY5[I]=-3;
if -4<E384165[I]<0 then OTHPAY5[I]=E384165[I];
if E384075[I]=2 and E34402B[I] le 0 then OTHPAY5[I]=-3;
if E384075[I]=2 and -4<E34402B[I]<0 then OTHPAY5[I]=E34402B[I];
if E384075[I]=6 and E35600[I] le 0 then OTHPAY5[I]=-3;
if E384075[I]=6 and -4<E35600[I]<0 then OTHPAY5[I]=E35600[I];

/* report nonhourly wage at the beginning*/
/* for tips*/
if E34402[I]>0 and E384071[I]=1 and E384161[I]>=0 then OTHPAY1[I]=E384161[I];
if E34402[I]>0 and E384071[I]=2 and E34402B[I]>0 and E384161[I]>=0 then
    OTHPAY1[I]=(E384161[I]*E34402B[I])/E34402[I];
if E34402[I]>0 and E384071[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E384161[I]>=0 then
    OTHPAY1[I]=E384161[I]/E34402[I];
if E34402[I]>0 and E384071[I]=4 and E384161[I]>=0 then OTHPAY1[I]=E384161[I]/(2*E34402[I]);
if E34402[I]>0 and E384071[I]=5 and E384161[I]>=0 then OTHPAY1[I]=E384161[I]/(4.3*E34402[I]);
if E34402[I]>0 and E384071[I]=6 and E384161[I]>=0 and E35600[I]>0 then
    OTHPAY1[I]=E384161[I]/(E35600[I]*E34402[I]);
if E34402[I]>0 and E384071[I]=8 and E384161[I]>=0 then OTHPAY1[I]=E384161[I]/(2.15*E34402[I]);
if E34402[I]>0 and E384071[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF1[I]=OTHPF1[I]+1;
if E384071[I] in (9,14) then OTHPAY1[I]=0;
/*missing value*/;

if E384161[I]>=0 and -4<E34402[I]<0 then OTHPAY1[I]=E34402[I];
if E384161[I]>=0 and E34402[I]=0 then OTHPAY1[I]=-3;
if -4<E384161[I]<0 then OTHPAY1[I]=E384161[I];

/*for commissions*/
if E34402[I]>0 and E384072[I]=1 and E384162[I]>=0 then OTHPAY2[I]=E384162[I];
if E34402[I]>0 and E384072[I]=2 and E34402B[I]>0 and E384162[I]>=0 then
    OTHPAY2[I]=(E384162[I]*E34402B[I])/E34402[I];
if E34402[I]>0 and E384072[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E384162[I]>=0 then
    OTHPAY2[I]=E384162[I]/E34402[I];

```

Appendix 2: Employment Variable Creation

```

if E34402[I]>0 and E384072[I]=4 and E384162[I]>=0 then OTHPAY2[I]=E384162[I]/(2*E34402[I]);
if E34402[I]>0 and E384072[I]=5 and E384162[I]>=0 then OTHPAY2[I]=E384162[I]/(4.3*E34402[I]);
if E34402[I]>0 and E384072[I]=6 and E384162[I]>=0 and E35600[I]>0 then
    OTHPAY2[I]=E384162[I]/(E35600[I]*E34402[I]);
if E34402[I]>0 and E384072[I]=8 and E384162[I]>=0 then OTHPAY2[I]=E384162[I]/(2.15*E34402[I]);
if E34402[I]>0 and E384072[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF2[I]=OTHPF2[I]+1;
if E384072[I] in (9,14) then OTHPAY2[I]=0;
*missing value*/;
if E384162[I]>=0 and -4<E34402[I]<0 then OTHPAY2[I]=E34402[I];
if E384162[I]>=0 and E34402[I]=0 then OTHPAY2[I]=-3;
if -4<E384162[I]<0 then OTHPAY2[I]=E384162[I];

/*for bonuses*/
if E34402[I]>0 and E384073[I]=1 and E384163[I]>=0 then OTHPAY3[I]=E384163[I];
if E34402[I]>0 and E384073[I]=2 and E34402B[I]>0 and E384163[I]>=0 then
    OTHPAY3[I]=(E384163[I]*E34402B[I])/E34402[I];
if E34402[I]>0 and E384073[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E384163[I]>=0 then
    OTHPAY3[I]=E384163[I]/E34402[I];
if E34402[I]>0 and E384073[I]=4 and E384163[I]>=0 then OTHPAY3[I]=E384163[I]/(2*E34402[I]);
if E34402[I]>0 and E384073[I]=5 and E384163[I]>=0 then OTHPAY3[I]=E384163[I]/(4.3*E34402[I]);
if E34402[I]>0 and E384073[I]=6 and E384163[I]>=0 and E35600[I]>0 then
    OTHPAY3[I]=E384163[I]/(E35600[I]*E34402[I]);
if E34402[I]>0 and E384073[I]=8 and E384163[I]>=0 then OTHPAY3[I]=E384163[I]/(2.15*E34402[I]);
if E34402[I]>0 and E384073[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF3[I]=OTHPF3[I]+1;
if E384073[I] in (9,14) then OTHPAY3[I]=0;
*missing value*/;
if E384163[I]>=0 and -4<E34402[I]<0 then OTHPAY3[I]=E34402[I];
if E384163[I]>=0 and E34402[I]=0 then OTHPAY3[I]=-3;
if -4<E384163[I]<0 then OTHPAY3[I]=E384163[I];

/*for incentive pay*/
if E34402[I]>0 and E384074[I]=1 and E384164[I]>=0 then OTHPAY4[I]=E384164[I];
if E34402[I]>0 and E384074[I]=2 and E34402B[I]>0 and E384164[I]>=0 then
    OTHPAY4[I]=(E384164[I]*E34402B[I])/E34402[I];
if E34402[I]>0 and E384074[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E384164[I]>=0 then
    OTHPAY4[I]=E384164[I]/E34402[I];
if E34402[I]>0 and E384074[I]=4 and E384164[I]>=0 then OTHPAY4[I]=E384164[I]/(2*E34402[I]);
if E34402[I]>0 and E384074[I]=5 and E384164[I]>=0 then OTHPAY4[I]=E384164[I]/(4.3*E34402[I]);
if E34402[I]>0 and E384074[I]=6 and E384164[I]>=0 and E35600[I]>0 then
    OTHPAY4[I]=E384164[I]/(E35600[I]*E34402[I]);
if E34402[I]>0 and E384074[I]=8 and E384164[I]>=0 then OTHPAY4[I]=E384164[I]/(2.15*E34402[I]);
if E34402[I]>0 and E384074[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF4[I]=OTHPF4[I]+1;
if E384074[I] in (9,14) then OTHPAY4[I]=0;
*missing value*/;
if E384164[I]>=0 and -4<E34402[I]<0 then OTHPAY4[I]=E34402[I];
if E384164[I]>=0 and E34402[I]=0 then OTHPAY4[I]=-3;
if -4<E384164[I]<0 then OTHPAY4[I]=E384164[I];

/*for others*/
if E34402[I]>0 and E384075[I]=1 and E384165[I]>=0 then OTHPAY5[I]=E384165[I];
if E34402[I]>0 and E384075[I]=2 and E34402B[I]>0 and E384165[I]>=0 then
    OTHPAY5[I]=(E384165[I]*E34402B[I])/E34402[I];
if E34402[I]>0 and E384075[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E384165[I]>=0 then
    OTHPAY5[I]=E384165[I]/E34402[I];

```

Appendix 2: Employment Variable Creation

```

if E34402[I]>0 and E384075[I]=4 and E384165[I]>=0 then OTHPAY5[I]=E384165[I]/(2*E34402[I]);
if E34402[I]>0 and E384075[I]=5 and E384165[I]>=0 then OTHPAY5[I]=E384165[I]/(4.3*E34402[I]);
if E34402[I]>0 and E384075[I]=6 and E384165[I]>=0 and E35600[I]>0 then
    OTHPAY5[I]=E384165[I]/(E35600[I]*E34402[I]);
if E34402[I]>0 and E384075[I]=8 and E384165[I]>=0 then OTHPAY5[I]=E384165[I]/(2.15*E34402[I]);
if E34402[I]>0 and E384075[I] in (0,7,12,13,14,15,16,17,21,22,23,24,25,26,28) then
    OTHPF5[I]=OTHPF5[I]+1;
if E384075[I] in (9,14) then OTHPAY5[I]=0;
/*missing value*/;
if E384165[I]>=0 and -4<E34402[I]<0 then OTHPAY5[I]=E34402[I];
if E384165[I]>=0 and E34402[I]=0 then OTHPAY5[I]=-3;
if -4<E384165[I]<0 then OTHPAY5[I]=E384165[I];
end;

/** case ii. without overtime at the beginning, diff no. of hours.***/
if E38313[I]=1 and E212001[I] ne 1 then do;

/* for tips*/
if E3800F[I]>0 and E384071[I]=1 and E384161[I]>=0 then OTHPAY1[I]=E384161[I];
if E3800F[I]>0 and E384071[I]=2 and E38027[I]>0 and E384161[I]>=0 then
    OTHPAY1[I]=(E384161[I]*E38027[I])/E3800F[I];
if E3800F[I]>0 and E384071[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E384161[I]>=0 then
    OTHPAY1[I]=E384161[I]/E3800F[I];
if E3800F[I]>0 and E384071[I]=4 and E384161[I]>=0 then OTHPAY1[I]=E384161[I]/(2*E3800F[I]);
if E3800F[I]>0 and E384071[I]=5 and E384161[I]>=0 then OTHPAY1[I]=E384161[I]/(4.3*E3800F[I]);
if E3800F[I]>0 and E384071[I]=6 and E384161[I]>=0 and E35600[I]>0 then
    OTHPAY1[I]=E384161[I]/(E35600[I]*E3800F[I]);
if E3800F[I]>0 and E384071[I]=8 and E384161[I]>=0 then OTHPAY1[I]=E384161[I]/(2.15*E3800F[I]);
if E3800F[I]>0 and E384071[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF1[I]=OTHPF1[I]+1;
if E384071[I] in (9,14) then OTHPAY1[I]=0;
/*missing value*/;
if E384161[I]>=0 and -4<E3800F[I]<0 then OTHPAY1[I]=E3800F[I];
if E384161[I]>=0 and E3800F[I]=0 then OTHPAY1[I]=-3;
if -4<E384161[I]<0 then OTHPAY1[I]=E384161[I];
if E384071[I]=2 and E38027[I]=0 then OTHPAY1[I]=-3;
if E384071[I]=2 and -4<E38027[I]<0 then OTHPAY1[I]=E38027[I];
if E384071[I]=6 and E35600[I] le 0 then OTHPAY1[I]=-3;
if E384071[I]=6 and -4<E35600[I]<0 then OTHPAY1[I]=E35600[I];

/*for commissions*/
if E3800F[I]>0 and E384072[I]=1 and E384162[I]>=0 then OTHPAY2[I]=E384162[I];
if E3800F[I]>0 and E384072[I]=2 and E38027[I]>0 and E384162[I]>=0 then
    OTHPAY2[I]=(E384162[I]*E38027[I])/E3800F[I];
if E3800F[I]>0 and E384072[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E384162[I]>=0 then
    OTHPAY2[I]=E384162[I]/E3800F[I];
if E3800F[I]>0 and E384072[I]=4 and E384162[I]>=0 then OTHPAY2[I]=E384162[I]/(2*E3800F[I]);
if E3800F[I]>0 and E384072[I]=5 and E384162[I]>=0 then OTHPAY2[I]=E384162[I]/(4.3*E3800F[I]);
if E3800F[I]>0 and E384072[I]=6 and E384162[I]>=0 and E35600[I]>0 then
    OTHPAY2[I]=E384162[I]/(E35600[I]*E3800F[I]);
if E3800F[I]>0 and E384072[I]=8 and E384162[I]>=0 then OTHPAY2[I]=E384162[I]/(2.15*E3800F[I]);
if E3800F[I]>0 and E384072[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF2[I]=OTHPF2[I]+1;
if E384072[I] in (9,14) then OTHPAY2[I]=0;
/*missing value*/;
if E384162[I]>=0 and -4<E3800F[I]<0 then OTHPAY2[I]=E3800F[I];
if E384162[I]>=0 and E3800F[I]=0 then OTHPAY2[I]=-3;

```

```

if -4<E384162[I]<0 then OTHPAY2[I]=E384162[I];
if E384072[I]=2 and E38027[I]=0 then OTHPAY2[I]=-3;
if E384072[I]=2 and -4<E38027[I]<0 then OTHPAY2[I]=E38027[I];
if E384072[I]=6 and E35600[I] le 0 then OTHPAY2[I]=-3;
if E384072[I]=6 and -4<E35600[I]<0 then OTHPAY2[I]=E35600[I];

/*for bonuses*/
if E3800F[I]>0 and E384073[I]=1 and E384163[I]>=0 then OTHPAY3[I]=E384163[I];
if E3800F[I]>0 and E384073[I]=2 and E38027[I]>0 and E384163[I]>=0 then
    OTHPAY3[I]=(E384163[I]*E38027[I])/E3800F[I];
if E3800F[I]>0 and E384073[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E384163[I]>=0 then
    OTHPAY3[I]=E384163[I]/E3800F[I];
if E3800F[I]>0 and E384073[I]=4 and E384163[I]>=0 then OTHPAY3[I]=E384163[I]/(2*E3800F[I]);
if E3800F[I]>0 and E384073[I]=5 and E384163[I]>=0 then OTHPAY3[I]=E384163[I]/(4.3*E3800F[I]);
if E3800F[I]>0 and E384073[I]=6 and E384163[I]>=0 and E35600[I]>0 then
    OTHPAY3[I]=E384163[I]/(E35600[I]*E3800F[I]);
if E3800F[I]>0 and E384073[I]=8 and E384163[I]>=0 then OTHPAY3[I]=E384163[I]/(2.15*E3800F[I]);
if E3800F[I]>0 and E384073[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF3[I]=OTHPF3[I]+1;
if E384073[I] in (9,14) then OTHPAY3[I]=0;
*missing value*;

if E384163[I]>=0 and -4<E3800F[I]<0 then OTHPAY3[I]=E3800F[I];
if E384163[I]>=0 and E3800F[I]=0 then OTHPAY3[I]=-3;
if -4<E384163[I]<0 then OTHPAY1[I]=E384163[I];
if E384073[I]=2 and E38027[I]=0 then OTHPAY3[I]=-3;
if E384073[I]=2 and -4<E38027[I]<0 then OTHPAY3[I]=E38027[I];
if E384073[I]=6 and E35600[I] le 0 then OTHPAY3[I]=-3;
if E384073[I]=6 and -4<E35600[I]<0 then OTHPAY3[I]=E35600[I];

/*for incentive pay*/
if E3800F[I]>0 and E384074[I]=1 and E384164[I]>=0 then OTHPAY4[I]=E384164[I];
if E3800F[I]>0 and E384074[I]=2 and E38027[I]>0 and E384164[I]>=0 then
    OTHPAY4[I]=(E384164[I]*E38027[I])/E3800F[I];
if E3800F[I]>0 and E384074[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E384164[I]>=0 then
    OTHPAY4[I]=E384164[I]/E3800F[I];
if E3800F[I]>0 and E384074[I]=4 and E384164[I]>=0 then OTHPAY4[I]=E384164[I]/(2*E3800F[I]);
if E3800F[I]>0 and E384074[I]=5 and E384164[I]>=0 then OTHPAY4[I]=E384164[I]/(4.3*E3800F[I]);
if E3800F[I]>0 and E384074[I]=6 and E384164[I]>=0 and E35600[I]>0 then
    OTHPAY4[I]=E384164[I]/(E35600[I]*E3800F[I]);
if E3800F[I]>0 and E384074[I]=8 and E384164[I]>=0 then OTHPAY4[I]=E384164[I]/(2.15*E3800F[I]);
if E3800F[I]>0 and E384074[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF4[I]=OTHPF4[I]+1;
if E384074[I] in (9,14) then OTHPAY4[I]=0;
*missing value*;

if E384164[I]>=0 and -4<E3800F[I]<0 then OTHPAY4[I]=E3800F[I];
if E384164[I]>=0 and E3800F[I]=0 then OTHPAY4[I]=-3;
if -4<E384164[I]<0 then OTHPAY4[I]=E384164[I];
if E384074[I]=2 and E38027[I]=0 then OTHPAY4[I]=-3;
if E384074[I]=2 and -4<E38027[I]<0 then OTHPAY4[I]=E38027[I];
if E384074[I]=6 and E35600[I] le 0 then OTHPAY4[I]=-3;
if E384074[I]=6 and -4<E35600[I]<0 then OTHPAY4[I]=E35600[I];

/*for others*/
if E3800F[I]>0 and E384075[I]=1 and E384165[I]>=0 then OTHPAY5[I]=E384165[I];
if E3800F[I]>0 and E384075[I]=2 and E38027[I]>0 and E384165[I]>=0 then
    OTHPAY5[I]=(E384165[I]*E38027[I])/E3800F[I];

```

Appendix 2: Employment Variable Creation

```

if E3800F[I]>0 and E384075[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E384165[I]>=0 then
    OTHPAY5[I]=E384165[I]/E3800F[I];
if E3800F[I]>0 and E384075[I]=4 and E384165[I]>=0 then OTHPAY5[I]=E384165[I]/(2*E3800F[I]);
if E3800F[I]>0 and E384075[I]=5 and E384165[I]>=0 then OTHPAY5[I]=E384165[I]/(4.3*E3800F[I]);
if E3800F[I]>0 and E384075[I]=6 and E384165[I]>=0 and E35600[I]>0 then
    OTHPAY5[I]=E384165[I]/(E35600[I]*E3800F[I]);
if E3800F[I]>0 and E384075[I]=8 and E384165[I]>=0 then OTHPAY5[I]=E384165[I]/(2.15*E3800F[I]);
if E3800F[I]>0 and E384075[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF5[I]=OTHPF5[I]+1;
if E384075[I] in (9,14) then OTHPAY5[I]=0;
/*missing value*/;
if E384165[I]>=0 and -4<E3800F[I]<0 then OTHPAY5[I]=E3800F[I];
if E384165[I]>=0 and E3800F[I]=0 then OTHPAY5[I]=-3;
if -4<E384165[I]<0 then OTHPAY5[I]=E384165[I];
if E384075[I]=2 and E38027[I]=0 then OTHPAY5[I]=-3;
if E384075[I]=2 and -4<E38027[I]<0 then OTHPAY5[I]=E38027[I];
if E384075[I]=6 and E35600[I] le 0 then OTHPAY5[I]=-3;
if E384075[I]=6 and -4<E35600[I]<0 then OTHPAY5[I]=E35600[I];
end;

/** case iii. only one compensation at the beginning,same no. of hours. ***/
if E38329[I]=1 and E20700[I]=1 then do;

/* for tips*/
if E212002[I]=1 then do;
if E38102[I] ne 1 and E38102[I] ne 3 then do; /*with overtime at the beginning*/
/*report hourly wage at the beginning*/
if E23901[I]>0 and E38329B[I]=1 and E38329D[I]>=0 then OTHPAY1[I]=E38329D[I];
if E23901[I]>0 and E38329B[I]=2 and E34430[I]>0 and E38329D[I]>=0 then
    OTHPAY1[I]=(E38329D[I]*E34430[I])/E23901[I];
if E23901[I]>0 and E38329B[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E38329D[I]>=0 then
    OTHPAY1[I]=E38329D[I]/E23901[I];
if E23901[I]>0 and E38329B[I]=4 and E38329D[I]>=0 then OTHPAY1[I]=E38329D[I]/(2*E23901[I]);
if E23901[I]>0 and E38329B[I]=5 and E38329D[I]>=0 then OTHPAY1[I]=E38329D[I]/(4.3*E23901[I]);
if E23901[I]>0 and E38329B[I]=6 and E38329D[I]>=0 and E35600[I]>0 then
    OTHPAY1[I]=E38329D[I]/(E35600[I]*E23901[I]);
if E23901[I]>0 and E38329B[I]=8 and E38329D[I]>=0 then OTHPAY1[I]=E38329D[I]/(2.15*E23901[I]);
if E23901[I]>0 and E38329B[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF1[I]=OTHPF1[I]+1;
/*missing value*/;
if E38329D[I]>=0 and -4<E23901[I]<0 then OTHPAY1[I]=E23901[I];
if E38329D[I]>=0 and E23901[I]=0 then OTHPAY1[I]=-3;
if -4<E38329D[I]<0 then OTHPAY1[I]=E38329D[I];

/*report nonhourly wage*/
if E34428[I]>0 and E38329B[I]=1 and E38329D[I]>=0 then OTHPAY1[I]=E38329D[I];
if E34428[I]>0 and E38329B[I]=2 and E34430[I]>0 and E38329D[I]>=0 then
    OTHPAY1[I]=(E38329D[I]*E34430[I])/E34428[I];
if E34428[I]>0 and E38329B[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E38329D[I]>=0 then
    OTHPAY1[I]=E38329D[I]/E34428[I];
if E34428[I]>0 and E38329B[I]=4 and E38329D[I]>=0 then OTHPAY1[I]=E38329D[I]/(2*E34428[I]);
if E34428[I]>0 and E38329B[I]=5 and E38329D[I]>=0 then OTHPAY1[I]=E38329D[I]/(4.3*E34428[I]);
if E34428[I]>0 and E38329B[I]=6 and E38329D[I]>=0 and E35600[I]>0 then
    OTHPAY1[I]=E38329D[I]/(E35600[I]*E34428[I]);
if E34428[I]>0 and E38329B[I]=8 and E38329D[I]>=0 then OTHPAY1[I]=E38329D[I]/(2.15*E34428[I]);
if E34428[I]>0 and E38329B[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF1[I]=OTHPF1[I]+1;

```

```

/*missing value*/;
if E38329D[I]>=0 and -4<E34428[I]<0 then OTHPAY1[I]=E34428[I];
if E38329D[I]>=0 and E34428[I]=0 then OTHPAY1[I]=-3;
if -4<E38329D[I]<0 then OTHPAY1[I]=E38329D[I];
if E38329B[I]=2 and E34430[I] le 0 then OTHPAY1[I]=-3;
if E38329B[I]=2 and -4<E34430[I]<0 then OTHPAY1[I]=E34430[I];
if E38329B[I]=6 and E35600[I] le 0 then OTHPAY1[I]=-3;
if E38329B[I]=6 and -4<E35600[I]<0 then OTHPAY1[I]=E35600[I];
end;

if E3800B[I]=1 then do; /*without overtime at the beginning*/
/*report hourly wage at the beginning*/
if E23901[I]>0 and E38329B[I]=1 and E38329D[I]>=0 then OTHPAY1[I]=E38329D[I];
if E23901[I]>0 and E38329B[I]=2 and E34402B[I]>0 and E38329D[I]>=0 then
    OTHPAY1[I]=(E38329D[I]*E34402B[I])/E23901[I];
if E23901[I]>0 and E38329B[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E38329D[I]>=0 then
    OTHPAY1[I]=E38329D[I]/E23901[I];
if E23901[I]>0 and E38329B[I]=4 and E38329D[I]>=0 then OTHPAY1[I]=E38329D[I]/(2*E23901[I]);
if E23901[I]>0 and E38329B[I]=5 and E38329D[I]>=0 then OTHPAY1[I]=E38329D[I]/(4.3*E23901[I]);
if E23901[I]>0 and E38329B[I]=6 and E38329D[I]>=0 and E35600[I]>0 then
    OTHPAY1[I]=E38329D[I]/(E35600[I]*E23901[I]);
if E23901[I]>0 and E38329B[I]=8 and E38329D[I]>=0 then OTHPAY1[I]=E38329D[I]/(2.15*E23901[I]);
if E23901[I]>0 and E38329B[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF1[I]=OTHPF1[I]+1;
/*missing value*/;
if E38329D[I]>=0 and -4<E23901[I]<0 then OTHPAY1[I]=E23901[I];
if E38329D[I]>=0 and E23901[I]=0 then OTHPAY1[I]=-3;
if -4<E38329D[I]<0 then OTHPAY1[I]=E38329D[I];

/*report non-hourly wage at the beginning*/
if E34402[I]>0 and E38329B[I]=1 and E38329D[I]>=0 then OTHPAY1[I]=E38329D[I];
if E34402[I]>0 and E38329B[I]=2 and E34402B[I]>0 and E38329D[I]>=0 then
    OTHPAY1[I]=(E38329D[I]*E34402B[I])/E34402[I];
if E34402[I]>0 and E38329B[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E38329D[I]>=0 then
    OTHPAY1[I]=E38329D[I]/E34402[I];
if E34402[I]>0 and E38329B[I]=4 and E38329D[I]>=0 then OTHPAY1[I]=E38329D[I]/(2*E34402[I]);
if E34402[I]>0 and E38329B[I]=5 and E38329D[I]>=0 then OTHPAY1[I]=E38329D[I]/(4.3*E34402[I]);
if E34402[I]>0 and E38329B[I]=6 and E38329D[I]>=0 and E35600[I]>0 then
    OTHPAY1[I]=E38329D[I]/(E35600[I]*E34402[I]);
if E34402[I]>0 and E38329B[I]=8 and E38329D[I]>=0 then OTHPAY1[I]=E38329D[I]/(2.15*E34402[I]);
if E34402[I]>0 and E38329B[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF1[I]=OTHPF1[I]+1;
if E38329B[I] in (9,14) then OTHPAY1[I]=0;
/*missing value*/;
if E38329D[I]>=0 and -4<E34402[I]<0 then OTHPAY1[I]=E34402[I];
if E38329D[I]>=0 and E34402[I]=0 then OTHPAY1[I]=-3;
if E38329B[I]=2 and E34402B[I] le 0 then OTHPAY1[I]=-3;
if E38329B[I]=2 and -4<E34402B[I]<0 then OTHPAY1[I]=E34402B[I];
if E38329B[I]=6 and E35600[I] le 0 then OTHPAY1[I]=-3;
if E38329B[I]=6 and -4<E35600[I]<0 then OTHPAY1[I]=E35600[I];
end;
end;

/*for commissions*/
if E212003[I]=1 then do;
if E38102[I] ne 1 and E38102[I] ne 3 then do; /*with overtime at the beginning*/
/*report hourly wage at the beginning*/

```

Appendix 2: Employment Variable Creation

```
if E23901[I]>0 and E38329B[I]=1 and E38329D[I]>=0 then OTHPAY2[I]=E38329D[I];
if E23901[I]>0 and E38329B[I]=2 and E34430[I]>0 and E38329D[I]>=0 then
    OTHPAY2[I]=(E38329D[I]*E34430[I])/E23901[I];
if E23901[I]>0 and E38329B[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E38329D[I]>=0 then
    OTHPAY2[I]=E38329D[I]/E23901[I];
if E23901[I]>0 and E38329B[I]=4 and E38329D[I]>=0 then OTHPAY2[I]=E38329D[I]/(2*E23901[I]);
if E23901[I]>0 and E38329B[I]=5 and E38329D[I]>=0 then OTHPAY2[I]=E38329D[I]/(4.3*E23901[I]);
if E23901[I]>0 and E38329B[I]=6 and E38329D[I]>=0 and E35600[I]>0 then
    OTHPAY2[I]=E38329D[I]/(E35600[I]*E23901[I]);
if E23901[I]>0 and E38329B[I]=8 and E38329D[I]>=0 then OTHPAY2[I]=E38329D[I]/(2.15*E23901[I]);
if E23901[I]>0 and E38329B[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF1[I]=OTHPF1[I]+1;
/*missing value*/;
if E38329D[I]>=0 and -4<E23901[I]<0 then OTHPAY2[I]=E23901[I];
if E38329D[I]>=0 and E23901[I]=0 then OTHPAY2[I]=-3;
if -4<E38329D[I]<0 then OTHPAY2[I]=E38329D[I];

/*report nonhourly wage*/
if E34428[I]>0 and E38329B[I]=1 and E38329D[I]>=0 then OTHPAY2[I]=E38329D[I];
if E34428[I]>0 and E38329B[I]=2 and E34430[I]>0 and E38329D[I]>=0 then
    OTHPAY2[I]=(E38329D[I]*E34430[I])/E34428[I];
if E34428[I]>0 and E38329B[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E38329D[I]>=0 then
    OTHPAY2[I]=E38329D[I]/E34428[I];
if E34428[I]>0 and E38329B[I]=4 and E38329D[I]>=0 then OTHPAY2[I]=E38329D[I]/(2*E34428[I]);
if E34428[I]>0 and E38329B[I]=5 and E38329D[I]>=0 then OTHPAY2[I]=E38329D[I]/(4.3*E34428[I]);
if E34428[I]>0 and E38329B[I]=6 and E38329D[I]>=0 and E35600[I]>0 then
    OTHPAY2[I]=E38329D[I]/(E35600[I]*E34428[I]);
if E34428[I]>0 and E38329B[I]=8 and E38329D[I]>=0 then OTHPAY2[I]=E38329D[I]/(2.15*E34428[I]);
if E34428[I]>0 and E38329B[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF2[I]=OTHPF2[I]+1;
/*missing value*/;
if E38329D[I]>=0 and -4<E34428[I]<0 then OTHPAY2[I]=E34428[I];
if E38329D[I]>=0 and E34428[I]=0 then OTHPAY2[I]=-3;
if -4<E38329D[I]<0 then OTHPAY2[I]=E38329D[I];
if E38329B[I]=2 and E34430[I] le 0 then OTHPAY2[I]=-3;
if E38329B[I]=2 and -4<E34430[I]<0 then OTHPAY2[I]=E34430[I];
if E38329B[I]=6 and E35600[I] le 0 then OTHPAY2[I]=-3;
if E38329B[I]=6 and -4<E35600[I]<0 then OTHPAY2[I]=E35600[I];
end;

if E3800B[I]=1 then do; /*without overtime at the beginning*/
/*report hourly wage at the beginning*/
if E23901[I]>0 and E38329B[I]=1 and E38329D[I]>=0 then OTHPAY2[I]=E38329D[I];
if E23901[I]>0 and E38329B[I]=2 and E34402B[I]>0 and E38329D[I]>=0 then
    OTHPAY2[I]=(E38329D[I]*E34402B[I])/E23901[I];
if E23901[I]>0 and E38329B[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E38329D[I]>=0 then
    OTHPAY2[I]=E38329D[I]/E23901[I];
if E23901[I]>0 and E38329B[I]=4 and E38329D[I]>=0 then OTHPAY2[I]=E38329D[I]/(2*E23901[I]);
if E23901[I]>0 and E38329B[I]=5 and E38329D[I]>=0 then OTHPAY2[I]=E38329D[I]/(4.3*E23901[I]);
if E23901[I]>0 and E38329B[I]=6 and E38329D[I]>=0 and E35600[I]>0 then
    OTHPAY2[I]=E38329D[I]/(E35600[I]*E23901[I]);
if E23901[I]>0 and E38329B[I]=8 and E38329D[I]>=0 then OTHPAY2[I]=E38329D[I]/(2.15*E23901[I]);
if E23901[I]>0 and E38329B[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF1[I]=OTHPF1[I]+1;
/*missing value*/;
if E38329D[I]>=0 and -4<E23901[I]<0 then OTHPAY2[I]=E23901[I];
if E38329D[I]>=0 and E23901[I]=0 then OTHPAY2[I]=-3;
```

```

if -4<E38329D[I]<0 then OTHPAY2[I]=E38329D[I];

/*report nonhourly wage*/
if E34402[I]>0 and E38329B[I]=1 and E38329D[I]>=0 then OTHPAY2[I]=E38329D[I];
if E34402[I]>0 and E38329B[I]=2 and E34402B[I]>0 and E38329D[I]>=0 then
    OTHPAY2[I]=(E38329D[I]*E34402B[I])/E34402[I];
if E34402[I]>0 and E38329B[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E38329D[I]>=0 then
    OTHPAY2[I]=E38329D[I]/E34402[I];
if E34402[I]>0 and E38329B[I]=4 and E38329D[I]>=0 then OTHPAY2[I]=E38329D[I]/(2*E34402[I]);
if E34402[I]>0 and E38329B[I]=5 and E38329D[I]>=0 then OTHPAY2[I]=E38329D[I]/(4.3*E34402[I]);
if E34402[I]>0 and E38329B[I]=6 and E38329D[I]>=0 and E35600[I]>0 then
    OTHPAY2[I]=E38329D[I]/(E35600[I]*E34402[I]);
if E34402[I]>0 and E38329B[I]=8 and E38329D[I]>=0 then OTHPAY2[I]=E38329D[I]/(2.15*E34402[I]);
if E34402[I]>0 and E38329B[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF2[I]=OTHPF2[I]+1;
if E38329B[I] in (9,14) then OTHPAY2[I]=0;
/*missing value*/;

if E38329D[I]>=0 and -4<E34402[I]<0 then OTHPAY2[I]=E34402[I];
if E38329D[I]>=0 and E34402[I]=0 then OTHPAY2[I]=-3;
if E38329B[I]=2 and E34402B[I] le 0 then OTHPAY2[I]=-3;
if E38329B[I]=2 and -4<E34402B[I]<0 then OTHPAY2[I]=E34402B[I];
if E38329B[I]=6 and E35600[I] le 0 then OTHPAY2[I]=-3;
if E38329B[I]=6 and -4<E35600[I]<0 then OTHPAY2[I]=E35600[I];
end;
end;

/*for bonuses*/
if E212004[I]=1 then do;
if E38102[I] ne 1 and E38102[I] ne 3 then do; /*with overtime at the beginning*/
/* report hourly wage at the beginning*/
if E23901[I]>0 and E38329B[I]=1 and E38329D[I]>=0 then OTHPAY3[I]=E38329D[I];
if E23901[I]>0 and E38329B[I]=2 and E34430[I]>0 and E38329D[I]>=0 then
    OTHPAY3[I]=(E38329D[I]*E34430[I])/E23901[I];
if E23901[I]>0 and E38329B[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E38329D[I]>=0 then
    OTHPAY3[I]=E38329D[I]/E23901[I];
if E23901[I]>0 and E38329B[I]=4 and E38329D[I]>=0 then OTHPAY3[I]=E38329D[I]/(2*E23901[I]);
if E23901[I]>0 and E38329B[I]=5 and E38329D[I]>=0 then OTHPAY3[I]=E38329D[I]/(4.3*E23901[I]);
if E23901[I]>0 and E38329B[I]=6 and E38329D[I]>=0 and E35600[I]>0 then
    OTHPAY3[I]=E38329D[I]/(E35600[I]*E23901[I]);
if E23901[I]>0 and E38329B[I]=8 and E38329D[I]>=0 then OTHPAY3[I]=E38329D[I]/(2.15*E23901[I]);
if E23901[I]>0 and E38329B[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF3[I]=OTHPF3[I]+1;
/*missing value*/;

if E38329D[I]>=0 and -4<E23901[I]<0 then OTHPAY3[I]=E23901[I];
if E38329D[I]>=0 and E23901[I]=0 then OTHPAY3[I]=-3;
if -4<E38329D[I]<0 then OTHPAY3[I]=E38329D[I];

/*report non-hourly wage*/
if E34428[I]>0 and E38329B[I]=1 and E38329D[I]>=0 then OTHPAY3[I]=E38329D[I];
if E34428[I]>0 and E38329B[I]=2 and E34430[I]>0 and E38329D[I]>=0 then
    OTHPAY3[I]=(E38329D[I]*E34430[I])/E34428[I];
if E34428[I]>0 and E38329B[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E38329D[I]>=0 then
    OTHPAY3[I]=E38329D[I]/E34428[I];
if E34428[I]>0 and E38329B[I]=4 and E38329D[I]>=0 then OTHPAY3[I]=E38329D[I]/(2*E34428[I]);
if E34428[I]>0 and E38329B[I]=5 and E38329D[I]>=0 then OTHPAY3[I]=E38329D[I]/(4.3*E34428[I]);
if E34428[I]>0 and E38329B[I]=6 and E38329D[I]>=0 and E35600[I]>0 then
    OTHPAY3[I]=E38329D[I]/(E35600[I]*E34428[I]);

```

Appendix 2: Employment Variable Creation

```

if E34428[I]>0 and E38329B[I]=8 and E38329D[I]>=0 then OTHPAY3[I]=E38329D[I]/(2.15*E34428[I]);
if E34428[I]>0 and E38329B[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF3[I]=OTHPF3[I]+1;
/*missing value*/;
if E38329D[I]>=0 and -4<E34428[I]<0 then OTHPAY3[I]=E34428[I];
if E38329D[I]>=0 and E34428[I]=0 then OTHPAY3[I]=-3;
if -4<E38329D[I]<0 then OTHPAY3[I]=E38329D[I];
if E38329B[I]=2 and E34430[I] le 0 then OTHPAY3[I]=-3;
if E38329B[I]=2 and -4<E34430[I]<0 then OTHPAY3[I]=E34430[I];
if E38329B[I]=6 and E35600[I] le 0 then OTHPAY3[I]=-3;
if E38329B[I]=6 and -4<E35600[I]<0 then OTHPAY3[I]=E35600[I];
end;

if E3800B[I]=1 then do; /*without overtime at the beginning*/
/* report hourly wage at the beginning*/
if E23901[I]>0 and E38329B[I]=1 and E38329D[I]>=0 then OTHPAY3[I]=E38329D[I];
if E23901[I]>0 and E38329B[I]=2 and E34402B[I]>0 and E38329D[I]>=0 then
    OTHPAY3[I]=(E38329D[I]*E34402B[I])/E23901[I];
if E23901[I]>0 and E38329B[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E38329D[I]>=0 then
    OTHPAY3[I]=E38329D[I]/E23901[I];
if E23901[I]>0 and E38329B[I]=4 and E38329D[I]>=0 then OTHPAY3[I]=E38329D[I]/(2*E23901[I]);
if E23901[I]>0 and E38329B[I]=5 and E38329D[I]>=0 then OTHPAY3[I]=E38329D[I]/(4.3*E23901[I]);
if E23901[I]>0 and E38329B[I]=6 and E38329D[I]>=0 and E35600[I]>0 then
    OTHPAY3[I]=E38329D[I]/(E35600[I]*E23901[I]);
if E23901[I]>0 and E38329B[I]=8 and E38329D[I]>=0 then OTHPAY3[I]=E38329D[I]/(2.15*E23901[I]);
if E23901[I]>0 and E38329B[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF3[I]=OTHPF3[I]+1;
/*missing value*/;
if E38329D[I]>=0 and -4<E23901[I]<0 then OTHPAY3[I]=E23901[I];
if E38329D[I]>=0 and E23901[I]=0 then OTHPAY3[I]=-3;
if -4<E38329D[I]<0 then OTHPAY3[I]=E38329D[I];

/*report non-hourly wage*/
if E34402[I]>0 and E38329B[I]=1 and E38329D[I]>=0 then OTHPAY3[I]=E38329D[I];
if E34402[I]>0 and E38329B[I]=2 and E34402B[I]>0 and E38329D[I]>=0 then
    OTHPAY3[I]=(E38329D[I]*E34402B[I])/E34402[I];
if E34402[I]>0 and E38329B[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E38329D[I]>=0 then
    OTHPAY3[I]=E38329D[I]/E34402[I];
if E34402[I]>0 and E38329B[I]=4 and E38329D[I]>=0 then OTHPAY3[I]=E38329D[I]/(2*E34402[I]);
if E34402[I]>0 and E38329B[I]=5 and E38329D[I]>=0 then OTHPAY3[I]=E38329D[I]/(4.3*E34402[I]);
if E34402[I]>0 and E38329B[I]=6 and E38329D[I]>=0 and E35600[I]>0 then
    OTHPAY3[I]=E38329D[I]/(E35600[I]*E34402[I]);
if E34402[I]>0 and E38329B[I]=8 and E38329D[I]>=0 then OTHPAY3[I]=E38329D[I]/(2.15*E34402[I]);
if E34402[I]>0 and E38329B[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF3[I]=OTHPF3[I]+1;
if E38329B[I] in (9,14) then OTHPAY3[I]=0;
/*missing value*/;
if E38329D[I]>=0 and -4<E34402[I]<0 then OTHPAY3[I]=E34402[I];
if E38329D[I]>=0 and E34402[I]=0 then OTHPAY3[I]=-3;
if E38329B[I]=2 and E34402B[I] le 0 then OTHPAY3[I]=-3;
if E38329B[I]=2 and -4<E34402B[I]<0 then OTHPAY3[I]=E34402B[I];
if E38329B[I]=6 and E35600[I] le 0 then OTHPAY3[I]=-3;
if E38329B[I]=6 and -4<E35600[I]<0 then OTHPAY3[I]=E35600[I];
end;
end;

/*for incentive pay*/

```

```

if E212005[I]=1 then do;
  if E38102[I] ne 1 and E38102[I] ne 3 then do; /*with overtime at the beginning*/
    /* report hourly wage at the beginning*/
    if E23901[I]>0 and E38329B[I]=1 and E38329D[I]>=0 then OTHPAY4[I]=E38329D[I];
    if E23901[I]>0 and E38329B[I]=2 and E34430[I]>0 and E38329D[I]>=0 then
      OTHPAY4[I]=(E38329D[I]*E34430[I])/E23901[I];
    if E23901[I]>0 and E38329B[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E38329D[I]>=0 then
      OTHPAY4[I]=E38329D[I]/E23901[I];
    if E23901[I]>0 and E38329B[I]=4 and E38329D[I]>=0 then OTHPAY4[I]=E38329D[I]/(2*E23901[I]);
    if E23901[I]>0 and E38329B[I]=5 and E38329D[I]>=0 then OTHPAY4[I]=E38329D[I]/(4.3*E23901[I]);
    if E23901[I]>0 and E38329B[I]=6 and E38329D[I]>=0 and E35600[I]>0 then
      OTHPAY4[I]=E38329D[I]/(E35600[I]*E23901[I]);
    if E23901[I]>0 and E38329B[I]=8 and E38329D[I]>=0 then OTHPAY4[I]=E38329D[I]/(2.15*E23901[I]);
    if E23901[I]>0 and E38329B[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
      OTHPF4[I]=OTHPF4[I]+1;
    *missing value*/;
    if E38329D[I]>=0 and -4<E23901[I]<0 then OTHPAY4[I]=E23901[I];
    if E38329D[I]>=0 and E23901[I]=0 then OTHPAY4[I]=-3;
    if -4<E38329D[I]<0 then OTHPAY4[I]=E38329D[I];
  end;
  /* report nonhourly wage*/
  if E34428[I]>0 and E38329B[I]=1 and E38329D[I]>=0 then OTHPAY4[I]=E38329D[I];
  if E34428[I]>0 and E38329B[I]=2 and E34430[I]>0 and E38329D[I]>=0 then
    OTHPAY4[I]=(E38329D[I]*E34430[I])/E34428[I];
  if E34428[I]>0 and E38329B[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E38329D[I]>=0 then
    OTHPAY4[I]=E38329D[I]/E34428[I];
  if E34428[I]>0 and E38329B[I]=4 and E38329D[I]>=0 then OTHPAY4[I]=E38329D[I]/(2*E34428[I]);
  if E34428[I]>0 and E38329B[I]=5 and E38329D[I]>=0 then OTHPAY4[I]=E38329D[I]/(4.3*E34428[I]);
  if E34428[I]>0 and E38329B[I]=6 and E38329D[I]>=0 and E35600[I]>0 then
    OTHPAY4[I]=E38329D[I]/(E35600[I]*E34428[I]);
  if E34428[I]>0 and E38329B[I]=8 and E38329D[I]>=0 then OTHPAY4[I]=E38329D[I]/(2.15*E34428[I]);
  if E34428[I]>0 and E38329B[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF4[I]=OTHPF4[I]+1;
  *missing value*/;
  if E38329D[I]>=0 and -4<E34428[I]<0 then OTHPAY4[I]=E34428[I];
  if E38329D[I]>=0 and E34428[I]=0 then OTHPAY4[I]=-3;
  if -4<E38329D[I]<0 then OTHPAY4[I]=E38329D[I];
  if E38329B[I]=2 and E34430[I] le 0 then OTHPAY4[I]=-3;
  if E38329B[I]=2 and -4<E34430[I]<0 then OTHPAY4[I]=E34430[I];
  if E38329B[I]=6 and E35600[I] le 0 then OTHPAY4[I]=-3;
  if E38329B[I]=6 and -4<E35600[I]<0 then OTHPAY4[I]=E35600[I];
end;

if E3800B[I]=1 then do; /*without overtime at the beginning*/
  /* report hourly wage at the beginning*/
  if E23901[I]>0 and E38329B[I]=1 and E38329D[I]>=0 then OTHPAY4[I]=E38329D[I];
  if E23901[I]>0 and E38329B[I]=2 and E34402B[I]>0 and E38329D[I]>=0 then
    OTHPAY4[I]=(E38329D[I]*E34402B[I])/E23901[I];
  if E23901[I]>0 and E38329B[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E38329D[I]>=0 then
    OTHPAY4[I]=E38329D[I]/E23901[I];
  if E23901[I]>0 and E38329B[I]=4 and E38329D[I]>=0 then OTHPAY4[I]=E38329D[I]/(2*E23901[I]);
  if E23901[I]>0 and E38329B[I]=5 and E38329D[I]>=0 then OTHPAY4[I]=E38329D[I]/(4.3*E23901[I]);
  if E23901[I]>0 and E38329B[I]=6 and E38329D[I]>=0 and E35600[I]>0 then
    OTHPAY4[I]=E38329D[I]/(E35600[I]*E23901[I]);
  if E23901[I]>0 and E38329B[I]=8 and E38329D[I]>=0 then OTHPAY4[I]=E38329D[I]/(2.15*E23901[I]);
  if E23901[I]>0 and E38329B[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF4[I]=OTHPF4[I]+1;

```

```

/*missing value*/;
if E38329D[I]>=0 and -4<E23901[I]<0 then OTHPAY4[I]=E23901[I];
if E38329D[I]>=0 and E23901[I]=0 then OTHPAY4[I]=-3;
if -4<E38329D[I]<0 then OTHPAY4[I]=E38329D[I];

/* report nonhourly wage*/
if E34402[I]>0 and E38329B[I]=1 and E38329D[I]>=0 then OTHPAY4[I]=E38329D[I];
if E34402[I]>0 and E38329B[I]=2 and E34402B[I]>0 and E38329D[I]>=0 then
    OTHPAY4[I]=(E38329D[I]*E34402B[I])/E34402[I];
if E34402[I]>0 and E38329B[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E38329D[I]>=0 then
    OTHPAY4[I]=E38329D[I]/E34402[I];
if E34402[I]>0 and E38329B[I]=4 and E38329D[I]>=0 then OTHPAY4[I]=E38329D[I]/(2*E34402[I]);
if E34402[I]>0 and E38329B[I]=5 and E38329D[I]>=0 then OTHPAY4[I]=E38329D[I]/(4.3*E34402[I]);
if E34402[I]>0 and E38329B[I]=6 and E38329D[I]>=0 and E35600[I]>0 then
    OTHPAY4[I]=E38329D[I]/(E35600[I]*E34402[I]);
if E34402[I]>0 and E38329B[I]=8 and E38329D[I]>=0 then OTHPAY4[I]=E38329D[I]/(2.15*E34402[I]);
if E34402[I]>0 and E38329B[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF4[I]=OTHPF4[I]+1;
if E38329B[I] in (9,14) then OTHPAY4[I]=0;
*missing value*/;
if E38329D[I]>=0 and -4<E34402[I]<0 then OTHPAY4[I]=E34402[I];
if E38329D[I]>=0 and E34402[I]=0 then OTHPAY4[I]=-3;
if E38329B[I]=2 and E34402B[I] le 0 then OTHPAY4[I]=-3;
if E38329B[I]=2 and -4<E34402B[I]<0 then OTHPAY4[I]=E34402B[I];
if E38329B[I]=6 and E35600[I] le 0 then OTHPAY4[I]=-3;
if E38329B[I]=6 and -4<E35600[I]<0 then OTHPAY4[I]=E35600[I];
end;
end;

/*for others*/
if E212006[I]=1 then do;
if E38102[I] ne 1 and E38102[I] ne 3 then do; /*with overtime at the beginning*/
/* report hourly wage at the beginning*/
if E23901[I]>0 and E38329B[I]=1 and E38329D[I]>=0 then OTHPAY5[I]=E38329D[I];
if E23901[I]>0 and E38329B[I]=2 and E34430[I]>0 and E38329D[I]>=0 then
    OTHPAY5[I]=(E38329D[I]*E34430[I])/E23901[I];
if E23901[I]>0 and E38329B[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E38329D[I]>=0 then
    OTHPAY5[I]=E38329D[I]/E23901[I];
if E23901[I]>0 and E38329B[I]=4 and E38329D[I]>=0 then OTHPAY5[I]=E38329D[I]/(2*E23901[I]);
if E23901[I]>0 and E38329B[I]=5 and E38329D[I]>=0 then OTHPAY5[I]=E38329D[I]/(4.3*E23901[I]);
if E23901[I]>0 and E38329B[I]=6 and E38329D[I]>=0 and E35600[I]>0 then
    OTHPAY5[I]=E38329D[I]/(E35600[I]*E23901[I]);
if E23901[I]>0 and E38329B[I]=8 and E38329D[I]>=0 then OTHPAY5[I]=E38329D[I]/(2.15*E23901[I]);
if E23901[I]>0 and E38329B[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF5[I]=OTHPF5[I]+1;
*missing value*/;
if E38329D[I]>=0 and -4<E23901[I]<0 then OTHPAY5[I]=E23901[I];
if E38329D[I]>=0 and E23901[I]=0 then OTHPAY5[I]=-3;
if -4<E38329D[I]<0 then OTHPAY5[I]=E38329D[I];

/*report non-hourly wage*/
if E34428[I]>0 and E38329B[I]=1 and E38329D[I]>=0 then OTHPAY5[I]=E38329D[I];
if E34428[I]>0 and E38329B[I]=2 and E34430[I]>0 and E38329D[I]>=0 then
    OTHPAY5[I]=(E38329D[I]*E34430[I])/E34428[I];
if E34428[I]>0 and E38329B[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E38329D[I]>=0 then
    OTHPAY5[I]=E38329D[I]/E34428[I];
if E34428[I]>0 and E38329B[I]=4 and E38329D[I]>=0 then OTHPAY5[I]=E38329D[I]/(2*E34428[I]);

```

```

if E34428[I]>0 and E38329B[I]=5 and E38329D[I]>=0 then OTHPAY5[I]=E38329D[I]/(4.3*E34428[I]);
if E34428[I]>0 and E38329B[I]=6 and E38329D[I]>=0 and E35600[I]>0 then
    OTHPAY5[I]=E38329D[I]/(E35600[I]*E34428[I]);
if E34428[I]>0 and E38329B[I]=8 and E38329D[I]>=0 then OTHPAY5[I]=E38329D[I]/(2.15*E34428[I]);
if E34428[I]>0 and E38329B[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF5[I]=OTHPF5[I]+1;
*missing value*/;
if E38329D[I]>=0 and -4<E34428[I]<0 then OTHPAY5[I]=E34428[I];
if E38329D[I]>=0 and E34428[I]=0 then OTHPAY5[I]=-3;
if -4<E38329D[I]<0 then OTHPAY5[I]=E38329D[I];
if E38329B[I]=2 and E34430[I] le 0 then OTHPAY5[I]=-3;
if E38329B[I]=2 and -4<E34430[I]<0 then OTHPAY5[I]=E34430[I];
if E38329B[I]=6 and E35600[I] le 0 then OTHPAY5[I]=-3;
if E38329B[I]=6 and -4<E35600[I]<0 then OTHPAY5[I]=E35600[I];
end;

if E3800B[I]=1 then do; /*without overtime at the beginning*/
/* report hourly wage at the beginning*/
if E23901[I]>0 and E38329B[I]=1 and E38329D[I]>=0 then OTHPAY5[I]=E38329D[I];
if E23901[I]>0 and E38329B[I]=2 and E34402B[I]>0 and E38329D[I]>=0 then
    OTHPAY5[I]=(E38329D[I]*E34402B[I])/E23901[I];
if E23901[I]>0 and E38329B[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E38329D[I]>=0 then
    OTHPAY5[I]=E38329D[I]/E23901[I];
if E23901[I]>0 and E38329B[I]=4 and E38329D[I]>=0 then OTHPAY5[I]=E38329D[I]/(2*E23901[I]);
if E23901[I]>0 and E38329B[I]=5 and E38329D[I]>=0 then OTHPAY5[I]=E38329D[I]/(4.3*E23901[I]);
if E23901[I]>0 and E38329B[I]=6 and E38329D[I]>=0 and E35600[I]>0 then
    OTHPAY5[I]=E38329D[I]/(E35600[I]*E23901[I]);
if E23901[I]>0 and E38329B[I]=8 and E38329D[I]>=0 then OTHPAY5[I]=E38329D[I]/(2.15*E23901[I]);
if E23901[I]>0 and E38329B[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF5[I]=OTHPF5[I]+1;
*missing value*/;
if E38329D[I]>=0 and -4<E23901[I]<0 then OTHPAY5[I]=E23901[I];
if E38329D[I]>=0 and E23901[I]=0 then OTHPAY5[I]=-3;
if -4<E38329D[I]<0 then OTHPAY5[I]=E38329D[I];

/*report non-hourly wage*/
if E34402[I]>0 and E38329B[I]=1 and E38329D[I]>=0 then OTHPAY5[I]=E38329D[I];
if E34402[I]>0 and E38329B[I]=2 and E34402B[I]>0 and E38329D[I]>=0 then
    OTHPAY5[I]=(E38329D[I]*E34402B[I])/E34402[I];
if E34402[I]>0 and E38329B[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E38329D[I]>=0 then
    OTHPAY5[I]=E38329D[I]/E34402[I];
if E34402[I]>0 and E38329B[I]=4 and E38329D[I]>=0 then OTHPAY5[I]=E38329D[I]/(2*E34402[I]);
if E34402[I]>0 and E38329B[I]=5 and E38329D[I]>=0 then OTHPAY5[I]=E38329D[I]/(4.3*E34402[I]);
if E34402[I]>0 and E38329B[I]=6 and E38329D[I]>=0 then OTHPAY5[I]=E38329D[I]/(52*E34402[I]);
if E34402[I]>0 and E38329B[I]=8 and E38329D[I]>=0 then OTHPAY5[I]=E38329D[I]/(2.15*E34402[I]);
if E34402[I]>0 and E38329B[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF5[I]=OTHPF5[I]+1;
if E38329B[I] in (9,14) then OTHPAY5[I]=0;
*missing value*/;
if E38329D[I]>=0 and -4<E34402[I]<0 then OTHPAY5[I]=E34402[I];
if E38329D[I]>=0 and E34402[I]=0 then OTHPAY5[I]=-3;
if E38329B[I]=2 and E34402B[I] le 0 then OTHPAY5[I]=-3;
if E38329B[I]=2 and -4<E34402B[I]<0 then OTHPAY5[I]=E34402B[I];
if E38329B[I]=6 and E35600[I] le 0 then OTHPAY5[I]=-3;
if E38329B[I]=6 and -4<E35600[I]<0 then OTHPAY5[I]=E35600[I];
end;
end;

```

end;

```

/** case iv. one compensation at the beginning, diff no. of hours. ***/
if E38329[I]=1 and E20700[I]=1 then do;

/* for tips*/
if E212002[I]=1 then do;
  if E3800B[I]=0 then do; /* without overtime at the beginning*/
    if E3800F[I]>0 and E38329B[I]=1 and E38329D[I]>=0 then OTHPAY1[I]=E38329D[I];
    if E3800F[I]>0 and E38329B[I]=2 and E38116B[I]>0 and E38329D[I]>=0 then
      OTHPAY1[I]=(E38329D[I]*E38116B[I])/E3800F[I];
    if E3800F[I]>0 and E38329B[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E38329D[I]>=0 then
      OTHPAY1[I]=E38329D[I]/E3800F[I];
    if E3800F[I]>0 and E38329B[I]=4 and E38329D[I]>=0 then OTHPAY1[I]=E38329D[I]/(2*E3800F[I]);
    if E3800F[I]>0 and E38329B[I]=5 and E38329D[I]>=0 then OTHPAY1[I]=E38329D[I]/(4.3*E3800F[I]);
    if E3800F[I]>0 and E38329B[I]=6 and E38329D[I]>=0 and E35600[I]>0 then
      OTHPAY1[I]=E38329D[I]/(E35600[I]*E3800F[I]);
    if E3800F[I]>0 and E38329B[I]=8 and E38329D[I]>=0 then OTHPAY1[I]=E38329D[I]/(2.15*E3800F[I]);
    if E3800F[I]>0 and E38329B[I] in (0,7,12,13,14,15,16,17,18,99,21,22,23,24,25,26,28) then
      OTHPF1[I]=OTHPF1[I]+1;
    if E38329B[I] in (9,14) then OTHPAY1[I]=0;
    /*missing value*/;
    if E38329D[I]>=0 and -4<E3800F[I]<0 then OTHPAY1[I]=E3800F[I];
    if E38329D[I]>=0 and E3800F[I]=0 then OTHPAY1[I]=-3;
    if -4<E38329D[I]<0 then OTHPAY1[I]=E38329D[I];
    if E38329B[I]=2 and E38116B[I]=0 then OTHPAY1[I]=-3;
    if E38329B[I]=2 and -4<E38116B[I]<0 then OTHPAY1[I]=E38116B[I];
    if E38329B[I]=6 and E35600[I] le 0 then OTHPAY1[I]=-3;
    if E38329B[I]=6 and -4<E35600[I]<0 then OTHPAY1[I]=E35600[I];
  end;

if E38102[I]=1 or E38102[I]=3 then do; /* with overtime at the beginning*/
if E38103[I]>0 and E38329B[I]=1 and E38329D[I]>=0 then OTHPAY1[I]=E38329D[I];
if E38103[I]>0 and E38329B[I]=2 and E38116B[I]>0 and E38329D[I]>=0 then
  OTHPAY1[I]=(E38329D[I]*E38116B[I])/E38103[I];
if E38103[I]>0 and E38329B[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E38329D[I]>=0 then
  OTHPAY1[I]=E38329D[I]/E38103[I];
if E38103[I]>0 and E38329B[I]=4 and E38329D[I]>=0 then OTHPAY1[I]=E38329D[I]/(2*E38103[I]);
if E38103[I]>0 and E38329B[I]=5 and E38329D[I]>=0 then OTHPAY1[I]=E38329D[I]/(4.3*E38103[I]);
if E38103[I]>0 and E38329B[I]=6 and E38329D[I]>=0 and E35600[I]>0 then
  OTHPAY1[I]=E38329D[I]/(E35600[I]*E38103[I]);
if E38103[I]>0 and E38329B[I]=8 and E38329D[I]>=0 then OTHPAY1[I]=E38329D[I]/(2.15*E38103[I]);
if E38103[I]>0 and E38329B[I] in (0,7,12,13,14,15,16,17,18,99,21,22,23,24,25,26,28) then
  OTHPF1[I]=OTHPF1[I]+1;
if E38329B[I] in (9,14) then OTHPAY1[I]=0;
/*missing value*/;
if E38329D[I]>=0 and -4<E38103[I]<0 then OTHPAY1[I]=E38103[I];
if E38329D[I]>=0 and E38103[I]=0 then OTHPAY1[I]=-3;
if -4<E38329D[I]<0 then OTHPAY1[I]=E38329D[I];
if E38329B[I]=2 and E38116B[I]=0 then OTHPAY1[I]=-3;
if E38329B[I]=2 and -4<E38116B[I]<0 then OTHPAY1[I]=E38116B[I];
if E38329B[I]=6 and E35600[I] le 0 then OTHPAY1[I]=-3;
if E38329B[I]=6 and -4<E35600[I]<0 then OTHPAY1[I]=E35600[I];
end;
end;

/*for commissions*/

```

```

if E212003[I]=1 then do;
/*without overtime at the beginning*/
if E3800F[I]>0 and E38329B[I]=1 and E38329D[I]>=0 then OTHPAY2[I]=E38329D[I];
if E3800F[I]>0 and E38329B[I]=2 and E38116B[I]>0 and E38329D[I]>=0 then
    OTHPAY2[I]=(E38329D[I]*E38116B[I])/E3800F[I];
if E3800F[I]>0 and E38329B[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E38329D[I]>=0 then
    OTHPAY2[I]=E38329D[I]/E3800F[I];
if E3800F[I]>0 and E38329B[I]=4 and E38329D[I]>=0 then OTHPAY2[I]=E38329D[I]/(2*E3800F[I]);
if E3800F[I]>0 and E38329B[I]=5 and E38329D[I]>=0 then OTHPAY2[I]=E38329D[I]/(4.3*E3800F[I]);
if E3800F[I]>0 and E38329B[I]=6 and E38329D[I]>=0 and E35600[I]>0 then
    OTHPAY2[I]=E38329D[I]/(E35600[I]*E3800F[I]);
if E3800F[I]>0 and E38329B[I]=8 and E38329D[I]>=0 then OTHPAY2[I]=E38329D[I]/(2.15*E3800F[I]);
if E3800F[I]>0 and E38329B[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF2[I]=OTHPF2[I]+1;
if E38329B[I] in (9,14) then OTHPAY2[I]=0;
*missing value*/;
if E38329D[I]>=0 and -4<E3800F[I]<0 then OTHPAY2[I]=E3800F[I];
if E38329D[I]>=0 and E3800F[I]=0 then OTHPAY2[I]=-3;
if -4<E38329D[I]<0 then OTHPAY2[I]=E38329D[I];
if E38329B[I]=2 and E38116B[I]=0 then OTHPAY2[I]=-3;
if E38329B[I]=2 and -4<E38116B[I]<0 then OTHPAY2[I]=E38116B[I];
if E38329B[I]=6 and E35600[I] le 0 then OTHPAY2[I]=-3;
if E38329B[I]=6 and -4<E35600[I]<0 then OTHPAY2[I]=E35600[I];

/* with overtime at the beginning*/
if E38102[I]=1 or E38102[I]=3 then do;
if E38103[I]>0 and E38329B[I]=1 and E38329D[I]>=0 then OTHPAY2[I]=E38329D[I];
if E38103[I]>0 and E38329B[I]=2 and E38116B[I]>0 and E38329D[I]>=0 then
    OTHPAY2[I]=(E38329D[I]*E38116B[I])/E38103[I];
if E38103[I]>0 and E38329B[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E38329D[I]>=0 then
    OTHPAY2[I]=E38329D[I]/E38103[I];
if E38103[I]>0 and E38329B[I]=4 and E38329D[I]>=0 then OTHPAY2[I]=E38329D[I]/(2*E38103[I]);
if E38103[I]>0 and E38329B[I]=5 and E38329D[I]>=0 then OTHPAY2[I]=E38329D[I]/(4.3*E38103[I]);
if E38103[I]>0 and E38329B[I]=6 and E38329D[I]>=0 and E35600[I]>0 then
    OTHPAY2[I]=E38329D[I]/(E35600[I]*E38103[I]);
if E38103[I]>0 and E38329B[I]=8 and E38329D[I]>=0 then OTHPAY2[I]=E38329D[I]/(2.15*E38103[I]);
if E38103[I]>0 and E38329B[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF2[I]=OTHPF2[I]+1;
if E38329B[I] in (9,14) then OTHPAY2[I]=0;
*missing value*/;
if E38329D[I]>=0 and -4<E38103[I]<0 then OTHPAY2[I]=E38103[I];
if E38329D[I]>=0 and E38103[I]=0 then OTHPAY2[I]=-3;
if -4<E38329D[I]<0 then OTHPAY2[I]=E38329D[I];
if E38329B[I]=2 and E38116B[I]=0 then OTHPAY2[I]=-3;
if E38329B[I]=2 and -4<E38116B[I]<0 then OTHPAY2[I]=E38116B[I];
if E38329B[I]=6 and E35600[I] le 0 then OTHPAY2[I]=-3;
if E38329B[I]=6 and -4<E35600[I]<0 then OTHPAY2[I]=E35600[I];
end;
end;

/*for bonuses*/
if E212004[I]=1 then do;
/* without overtime at the beginning*/
if E3800F[I]>0 and E38329B[I]=1 and E38329D[I]>=0 then OTHPAY3[I]=E38329D[I];
if E3800F[I]>0 and E38329B[I]=2 and E38116B[I]>0 and E38329D[I]>=0 then
    OTHPAY3[I]=(E38329D[I]*E38116B[I])/E3800F[I];

```

Appendix 2: Employment Variable Creation

```

if E3800F[I]>0 and E38329B[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E38329D[I]>=0 then
    OTHPAY3[I]=E38329D[I]/E3800F[I];
if E3800F[I]>0 and E38329B[I]=4 and E38329D[I]>=0 then OTHPAY3[I]=E38329D[I]/(2*E3800F[I]);
if E3800F[I]>0 and E38329B[I]=5 and E38329D[I]>=0 then OTHPAY3[I]=E38329D[I]/(4.3*E3800F[I]);
if E3800F[I]>0 and E38329B[I]=6 and E38329D[I]>=0 and E35600[I]>0 then
    OTHPAY3[I]=E38329D[I]/(E35600[I]*E3800F[I]);
if E3800F[I]>0 and E38329B[I]=8 and E38329D[I]>=0 then OTHPAY3[I]=E38329D[I]/(2.15*E3800F[I]);
if E3800F[I]>0 and E38329B[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF3[I]=OTHPF3[I]+1;
if E38329B[I] in (9,14) then OTHPAY3[I]=0;
/*missing value*/;
if E38329D[I]>=0 and -4<E3800F[I]<0 then OTHPAY3[I]=E3800F[I];
if E38329D[I]>=0 and E3800F[I]=0 then OTHPAY3[I]=-3;
if -4<E38329D[I]<0 then OTHPAY3[I]=E38329D[I];
if E38329B[I]=2 and E38116B[I]=0 then OTHPAY3[I]=-3;
if E38329B[I]=2 and -4<E38116B[I]<0 then OTHPAY3[I]=E38116B[I];
if E38329B[I]=6 and E35600[I] le 0 then OTHPAY3[I]=-3;
if E38329B[I]=6 and -4<E35600[I]<0 then OTHPAY3[I]=E35600[I];

if E38102[I]=1 or E38102[I]=3 then do; /*with overtime at the beginning*/
if E38103[I]>0 and E38329B[I]=1 and E38329D[I]>=0 then OTHPAY3[I]=E38329D[I];
if E38103[I]>0 and E38329B[I]=2 and E38116B[I]>0 and E38329D[I]>=0 then
    OTHPAY3[I]=(E38329D[I]*E38116B[I])/E38103[I];
if E38103[I]>0 and E38329B[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E38329D[I]>=0 then
    OTHPAY3[I]=E38329D[I]/E38103[I];
if E38103[I]>0 and E38329B[I]=4 and E38329D[I]>=0 then OTHPAY3[I]=E38329D[I]/(2*E38103[I]);
if E38103[I]>0 and E38329B[I]=5 and E38329D[I]>=0 then OTHPAY3[I]=E38329D[I]/(4.3*E38103[I]);
if E38103[I]>0 and E38329B[I]=6 and E38329D[I]>=0 and E35600[I]>0 then
    OTHPAY3[I]=E38329D[I]/(E35600[I]*E38103[I]);
if E38103[I]>0 and E38329B[I]=8 and E38329D[I]>=0 then OTHPAY3[I]=E38329D[I]/(2.15*E38103[I]);
if E38103[I]>0 and E38329B[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF3[I]=OTHPF3[I]+1;
if E38329B[I] in (9,14) then OTHPAY3[I]=0;
/*missing value*/;
if E38329D[I]>=0 and -4<E38103[I]<0 then OTHPAY3[I]=E38103[I];
if E38329D[I]>=0 and E38103[I]=0 then OTHPAY3[I]=-3;
if -4<E38329D[I]<0 then OTHPAY3[I]=E38329D[I];
if E38329B[I]=2 and E38116B[I]=0 then OTHPAY3[I]=-3;
if E38329B[I]=2 and -4<E38116B[I]<0 then OTHPAY3[I]=E38116B[I];
if E38329B[I]=6 and E35600[I] le 0 then OTHPAY3[I]=-3;
if E38329B[I]=6 and -4<E35600[I]<0 then OTHPAY3[I]=E35600[I];
end;
end;

/*for incentive pay*/
if E212005[I]=1 then do;
/*without overtime at the beginning*/
if E3800F[I]>0 and E38329B[I]=1 and E38329D[I]>=0 then OTHPAY4[I]=E38329D[I];
if E3800F[I]>0 and E38329B[I]=2 and E38116B[I]>0 and E38329D[I]>=0 then
    OTHPAY4[I]=(E38329D[I]*E38116B[I])/E3800F[I];
if E3800F[I]>0 and E38329B[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E38329D[I]>=0 then
    OTHPAY4[I]=E38329D[I]/E3800F[I];
if E3800F[I]>0 and E38329B[I]=4 and E38329D[I]>=0 then OTHPAY4[I]=E38329D[I]/(2*E3800F[I]);
if E3800F[I]>0 and E38329B[I]=5 and E38329D[I]>=0 then OTHPAY4[I]=E38329D[I]/(4.3*E3800F[I]);
if E3800F[I]>0 and E38329B[I]=6 and E38329D[I]>=0 and E35600[I]>0 then
    OTHPAY4[I]=E38329D[I]/(E35600[I]*E3800F[I]);
if E3800F[I]>0 and E38329B[I]=8 and E38329D[I]>=0 then OTHPAY4[I]=E38329D[I]/(2.15*E3800F[I]);

```

```

if E3800F[I]>0 and E38329B[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF4[I]=OTHPF4[I]+1;
if E38329B[I] in (9,14) then OTHPAY4[I]=0;
/*missing value*/;
if E38329D[I]>0 and -4<E3800F[I]<0 then OTHPAY4[I]=E3800F[I];
if E38329D[I]>=0 and E3800F[I]=0 then OTHPAY4[I]=-3;
if -4<E38329D[I]<0 then OTHPAY4[I]=E38329D[I];
if E38329B[I]=2 and E38116B[I]=0 then OTHPAY4[I]=-3;
if E38329B[I]=2 and -4<E38116B[I]<0 then OTHPAY4[I]=E38116B[I];
if E38329B[I]=6 and E35600[I] le 0 then OTHPAY4[I]=-3;
if E38329B[I]=6 and -4<E35600[I]<0 then OTHPAY4[I]=E35600[I];

if E38102[I]=1 or E38102[I]=3 then do; /* with overtime at the beginning*/
if E38103[I]>0 and E38329B[I]=1 and E38329D[I]>=0 then OTHPAY4[I]=E38329D[I];
if E38103[I]>0 and E38329B[I]=2 and E38116B[I]>0 and E38329D[I]>=0 then
    OTHPAY4[I]=(E38329D[I]*E38116B[I])/E38103[I];
if E38103[I]>0 and E38329B[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E38329D[I]>=0 then
    OTHPAY4[I]=E38329D[I]/E38103[I];
if E38103[I]>0 and E38329B[I]=4 and E38329D[I]>=0 then OTHPAY4[I]=E38329D[I]/(2*E38103[I]);
if E38103[I]>0 and E38329B[I]=5 and E38329D[I]>=0 then OTHPAY4[I]=E38329D[I]/(4.3*E38103[I]);
if E38103[I]>0 and E38329B[I]=6 and E38329D[I]>=0 and E35600[I]>0 then
    OTHPAY4[I]=E38329D[I]/(E35600[I]*E38103[I]);
if E38103[I]>0 and E38329B[I]=8 and E38329D[I]>=0 then OTHPAY4[I]=E38329D[I]/(2.15*E38103[I]);
if E38103[I]>0 and E38329B[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF4[I]=OTHPF4[I]+1;
if E38329B[I] in (9,14) then OTHPAY4[I]=0;
/*missing value*/;
if E38329D[I]>0 and -4<E38103[I]<0 then OTHPAY4[I]=E38103[I];
if E38329D[I]>=0 and E38103[I]=0 then OTHPAY4[I]=-3;
if -4<E38329D[I]<0 then OTHPAY4[I]=E38329D[I];
if E38329B[I]=2 and E38116B[I]=0 then OTHPAY4[I]=-3;
if E38329B[I]=2 and -4<E38116B[I]<0 then OTHPAY4[I]=E38116B[I];
if E38329B[I]=6 and E35600[I] le 0 then OTHPAY4[I]=-3;
if E38329B[I]=6 and -4<E35600[I]<0 then OTHPAY4[I]=E35600[I];
end;
end;

/*for others*/
if E212006[I]=1 then do;
/* without overtime at the beginning*/
if E3800F[I]>0 and E38329B[I]=1 and E38329D[I]>=0 then OTHPAY5[I]=E38329D[I];
if E3800F[I]>0 and E38329B[I]=2 and E38116B[I]>0 and E38329D[I]>=0 then
    OTHPAY5[I]=(E38329D[I]*E38116B[I])/E3800F[I];
if E3800F[I]>0 and E38329B[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E38329D[I]>=0 then
    OTHPAY5[I]=E38329D[I]/E3800F[I];
if E3800F[I]>0 and E38329B[I]=4 and E38329D[I]>=0 then OTHPAY5[I]=E38329D[I]/(2*E3800F[I]);
if E3800F[I]>0 and E38329B[I]=5 and E38329D[I]>=0 then OTHPAY5[I]=E38329D[I]/(4.3*E3800F[I]);
if E3800F[I]>0 and E38329B[I]=6 and E38329D[I]>=0 and E35600[I]>0 then
    OTHPAY5[I]=E38329D[I]/(E35600[I]*E3800F[I]);
if E3800F[I]>0 and E38329B[I]=8 and E38329D[I]>=0 then OTHPAY5[I]=E38329D[I]/(2.15*E3800F[I]);
if E3800F[I]>0 and E38329B[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF5[I]=OTHPF5[I]+1;
if E38329B[I] in (9,14) then OTHPAY5[I]=0;
/*missing value*/;
if E38329D[I]>=0 and -4<E3800F[I]<0 then OTHPAY5[I]=E3800F[I];
if E38329D[I]>=0 and E3800F[I]=0 then OTHPAY5[I]=-3;
if -4<E38329D[I]<0 then OTHPAY5[I]=E38329D[I];

```

```

if E38329B[I]=2 and E38116B[I]=0 then OTHPAY5[I]=-3;
if E38329B[I]=2 and -4<E38116B[I]<0 then OTHPAY5[I]=E38116B[I];
if E38329B[I]=6 and E35600[I] le 0 then OTHPAY5[I]=-3;
if E38329B[I]=6 and -4<E35600[I]<0 then OTHPAY5[I]=E35600[I];

if E38102[I]=1 or E38102[I]=3 then do; /* with overtime at the beginning*/
if E38103[I]>0 and E38329B[I]=1 and E38329D[I]>=0 then OTHPAY5[I]=E38329D[I];
if E38103[I]>0 and E38329B[I]=2 and E38116B[I]>0 and E38329D[I]>=0 then
    OTHPAY5[I]=(E38329D[I]*E38116B[I])/E38103[I];
if E38103[I]>0 and E38329B[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E38329D[I]>=0 then
    OTHPAY5[I]=E38329D[I]/E38103[I];
if E38103[I]>0 and E38329B[I]=4 and E38329D[I]>=0 then OTHPAY5[I]=E38329D[I]/(2*E38103[I]);
if E38103[I]>0 and E38329B[I]=5 and E38329D[I]>=0 then OTHPAY5[I]=E38329D[I]/(4.3*E38103[I]);
if E38103[I]>0 and E38329B[I]=6 and E38329D[I]>=0 and E35600[I]>0 then
    OTHPAY5[I]=E38329D[I]/(E35600[I]*E38103[I]);
if E38103[I]>0 and E38329B[I]=8 and E38329D[I]>=0 then OTHPAY5[I]=E38329D[I]/(2.15*E38103[I]);
if E38103[I]>0 and E38329B[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF5[I]=OTHPF5[I]+1;
if E38329B[I] in (9,14) then OTHPAY5[I]=0;
/*missing value*/;
if E38329D[I]>=0 and -4<E38103[I]<0 then OTHPAY5[I]=E38103[I];
if E38329D[I]>=0 and E38103[I]=0 then OTHPAY5[I]=-3;
if -4<E38329D[I]<0 then OTHPAY5[I]=E38329D[I];
if E38329B[I]=2 and E38116B[I]=0 then OTHPAY5[I]=-3;
if E38329B[I]=2 and -4<E38116B[I]<0 then OTHPAY5[I]=E38116B[I];
if E38329B[I]=6 and E35600[I] le 0 then OTHPAY5[I]=-3;
if E38329B[I]=6 and -4<E35600[I]<0 then OTHPAY5[I]=E35600[I];
end;
end;

end;

/** case v. more than one compensation at the beginning, same no. of hours. ***/
if E38330[I]=1 and E20700[I]=1 then do;

/* for tips*/
if E38102[I] ne 1 and E38102[I] ne 3 then do; /*with overtime at the beginning*/
/*report hourly wage at the beginning*/
if E23901[I]>0 and E384071[I]=1 and E384161[I]>=0 then OTHPAY1[I]=E384161[I];
if E23901[I]>0 and E384071[I]=2 and E34430[I]>0 and E384161[I]>=0 then
    OTHPAY1[I]=(E384161[I]*E34430[I])/E23901[I];
if E23901[I]>0 and E384071[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E384161[I]>=0 then
    OTHPAY1[I]=E384161[I]/E23901[I];
if E23901[I]>0 and E384071[I]=4 and E384161[I]>=0 then OTHPAY1[I]=E384161[I]/(2*E23901[I]);
if E23901[I]>0 and E384071[I]=5 and E384161[I]>=0 then OTHPAY1[I]=E384161[I]/(4.3*E23901[I]);
if E23901[I]>0 and E384071[I]=6 and E384161[I]>=0 and E35600[I]>0 then
    OTHPAY1[I]=E384161[I]/(E35600[I]*E23901[I]);
if E23901[I]>0 and E384071[I]=8 and E384161[I]>=0 then OTHPAY1[I]=E384161[I]/(2.15*E23901[I]);
if E23901[I]>0 and E384071[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF1[I]=OTHPF1[I]+1;
/*missing value*/;
if E384161[I]>=0 and -4<E23901[I]<0 then OTHPAY1[I]=E23901[I];
if E384161[I]>=0 and E23901[I]=0 then OTHPAY1[I]=-3;
if -4<E384161[I]<0 then OTHPAY1[I]=E384161[I];

/*report non-hourly wage*/
if E34428[I]>0 and E384071[I]=1 and E384161[I]>=0 then OTHPAY1[I]=E384161[I];

```

Appendix 2: Employment Variable Creation

```
if E34428[I]>0 and E384071[I]=2 and E34430[I]>0 and E384161[I]>=0 then
    OTHPAY1[I]=(E384161[I]*E34430[I])/E34428[I];
if E34428[I]>0 and E384071[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E384161[I]>=0 then
    OTHPAY1[I]=E384161[I]/E34428[I];
if E34428[I]>0 and E384071[I]=4 and E384161[I]>=0 then OTHPAY1[I]=E384161[I]/(2*E34428[I]);
if E34428[I]>0 and E384071[I]=5 and E384161[I]>=0 then OTHPAY1[I]=E384161[I]/(4.3*E34428[I]);
if E34428[I]>0 and E384071[I]=6 and E384161[I]>=0 and E35600[I]>0 then
    OTHPAY1[I]=E384161[I]/(E35600[I]*E34428[I]);
if E34428[I]>0 and E384071[I]=8 and E384161[I]>=0 then OTHPAY1[I]=E384161[I]/(2.15*E34428[I]);
if E34428[I]>0 and E384071[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF1[I]=OTHPF1[I]+1;
/*missing value*/;
if E384161[I]>=0 and -4<E34428[I]<0 then OTHPAY1[I]=E34428[I];
if E384161[I]>=0 and E34428[I]=0 then OTHPAY1[I]=-3;
if -4<E384161[I]<0 then OTHPAY1[I]=E384161[I];
if E384071[I]=2 and E34430[I] le 0 then OTHPAY1[I]=-3;
if E384071[I]=2 and -4<E34430[I]<0 THNE OTHPAY1[I]=E34430[I];
if E384071[I]=6 and E35600[I] le 0 then OTHPAY1[I]=-3;
if E384071[I]=6 and -4<E35600[I]<0 then OTHPAY1[I]=E35600[I];
end;

if E3800B[I]=1 then do; /*without overtime at the beginning*/
/*report hourly wage at the beginning*/
if E23901[I]>0 and E384071[I]=1 and E384161[I]>=0 then OTHPAY1[I]=E384161[I];
if E23901[I]>0 and E384071[I]=2 and E34402B[I]>0 and E384161[I]>=0 then
    OTHPAY1[I]=(E384161[I]*E34402B[I])/E23901[I];
if E23901[I]>0 and E384071[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E384161[I]>=0 then
    OTHPAY1[I]=E384161[I]/E23901[I];
if E23901[I]>0 and E384071[I]=4 and E384161[I]>=0 then OTHPAY1[I]=E384161[I]/(2*E23901[I]);
if E23901[I]>0 and E384071[I]=5 and E384161[I]>=0 then OTHPAY1[I]=E384161[I]/(4.3*E23901[I]);
if E23901[I]>0 and E384071[I]=6 and E384161[I]>=0 and E35600[I]>0 then
    OTHPAY1[I]=E384161[I]/(E35600[I]*E23901[I]);
if E23901[I]>0 and E384071[I]=8 and E384161[I]>=0 then OTHPAY1[I]=E384161[I]/(2.15*E23901[I]);
if E23901[I]>0 and E384071[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF1[I]=OTHPF1[I]+1;
/*missing value*/;
if E384161[I]>=0 and -4<E23901[I]<0 then OTHPAY1[I]=E23901[I];
if E384161[I]>=0 and E23901[I]=0 then OTHPAY1[I]=-3;
if -4<E384161[I]<0 then OTHPAY1[I]=E384161[I];

/*report non-hourly wage*/
if E34402[I]>0 and E384071[I]=1 and E384161[I]>=0 then OTHPAY1[I]=E384161[I];
if E34402[I]>0 and E384071[I]=2 and E34402B[I]>0 and E384161[I]>=0 then
    OTHPAY1[I]=(E384161[I]*E34402B[I])/E34402[I];
if E34402[I]>0 and E384071[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E384161[I]>=0 then
    OTHPAY1[I]=E384161[I]/E34402[I];
if E34402[I]>0 and E384071[I]=4 and E384161[I]>=0 then OTHPAY1[I]=E384161[I]/(2*E34402[I]);
if E34402[I]>0 and E384071[I]=5 and E384161[I]>=0 then OTHPAY1[I]=E384161[I]/(4.3*E34402[I]);
if E34402[I]>0 and E384071[I]=6 and E384161[I]>=0 and E35600[I]>0 then
    OTHPAY1[I]=E384161[I]/(E35600[I]*E34402[I]);
if E34402[I]>0 and E384071[I]=8 and E384161[I]>=0 then OTHPAY1[I]=E384161[I]/(2.15*E34402[I]);
if E34402[I]>0 and E384071[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF1[I]=OTHPF1[I]+1;
if E384071[I] in (9,14) then OTHPAY1[I]=0;
/*missing value*/;
if E384161[I]>=0 and -4<E34402[I]<0 then OTHPAY1[I]=E34402[I];
if E384161[I]>=0 and E34402[I]=0 then OTHPAY1[I]=-3;
```

Appendix 2: Employment Variable Creation

```

if E384071[I]=2 and E34402B[I] le 0 then OTHPAY1[I]=-3;
if E384071[I]=2 and -4<E34402B[I]<0 THNE OTHPAY1[I]=E34402B[I];
if E384071[I]=6 and E35600[I] le 0 then OTHPAY1[I]=-3;
if E384071[I]=6 and -4<E35600[I]<0 then OTHPAY1[I]=E35600[I];
end;

/*for commissions*/
if E38102[I] ne 1 and E38102[I] ne 3 then do; /*with overtime at the beginning*/
/* report hourly wage at the beginning*/
if E23901[I]>0 and E384072[I]=1 and E384162[I]>=0 then OTHPAY2[I]=E384162[I];
if E23901[I]>0 and E384072[I]=2 and E34430[I]>0 and E384162[I]>=0 then
    OTHPAY2[I]=(E384162[I]*E34430[I])/E23901[I];
if E23901[I]>0 and E384072[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E384162[I]>=0 then
    OTHPAY2[I]=E384162[I]/E23901[I];
if E23901[I]>0 and E384072[I]=4 and E384162[I]>=0 then OTHPAY2[I]=E384162[I]/(2*E23901[I]);
if E23901[I]>0 and E384072[I]=5 and E384162[I]>=0 then OTHPAY2[I]=E384162[I]/(4.3*E23901[I]);
if E23901[I]>0 and E384072[I]=6 and E384162[I]>=0 and E35600[I]>0 then
    OTHPAY2[I]=E384162[I]/(E35600[I]*E23901[I]);
if E23901[I]>0 and E384072[I]=8 and E384162[I]>=0 then OTHPAY2[I]=E384162[I]/(2.15*E23901[I]);
if E23901[I]>0 and E384072[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF2[I]=OTHPF2[I]+1;
*missing value*;
if E384162[I]>=0 and -4<E23901[I]<0 then OTHPAY2[I]=E23901[I];
if E384162[I]>=0 and E23901[I]=0 then OTHPAY2[I]=-3;
if -4<E384162[I]<0 then OTHPAY2[I]=E384162[I];

/* report non-hourly wage*/
if E34428[I]>0 and E384072[I]=1 and E384162[I]>=0 then OTHPAY2[I]=E384162[I];
if E34428[I]>0 and E384072[I]=2 and E34430[I]>0 and E384162[I]>=0 then
    OTHPAY2[I]=(E384162[I]*E34430[I])/E34428[I];
if E34428[I]>0 and E384072[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E384162[I]>=0 then
    OTHPAY2[I]=E384162[I]/E34428[I];
if E34428[I]>0 and E384072[I]=4 and E384162[I]>=0 then OTHPAY2[I]=E384162[I]/(2*E34428[I]);
if E34428[I]>0 and E384072[I]=5 and E384162[I]>=0 then OTHPAY2[I]=E384162[I]/(4.3*E34428[I]);
if E34428[I]>0 and E384072[I]=6 and E384162[I]>=0 and E35600[I]>0 then
    OTHPAY2[I]=E384162[I]/(E35600[I]*E34428[I]);
if E34428[I]>0 and E384072[I]=8 and E384162[I]>=0 then OTHPAY2[I]=E384162[I]/(2.15*E34428[I]);
if E34428[I]>0 and E384072[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF2[I]=OTHPF2[I]+1;
*missing value*;
if E384162[I]>=0 and -4<E34428[I]<0 then OTHPAY2[I]=E34428[I];
if E384162[I]>=0 and E34428[I]=0 then OTHPAY2[I]=-3;
if -4<E384162[I]<0 then OTHPAY2[I]=E384162[I];
if E384072[I]=2 and E34430[I] le 0 then OTHPAY2[I]=-3;
if E384072[I]=2 and -4<E34430[I]<0 THNE OTHPAY2[I]=E34430[I];
if E384072[I]=6 and E35600[I] le 0 then OTHPAY2[I]=-3;
if E384072[I]=6 and -4<E35600[I]<0 then OTHPAY2[I]=E35600[I];
end;

if E3800B[I]=1 then do; /*without overtime at the beginning*/
/* report hourly wage at the beginning*/
if E23901[I]>0 and E384072[I]=1 and E384162[I]>=0 then OTHPAY2[I]=E384162[I];
if E23901[I]>0 and E384072[I]=2 and E34402B[I]>0 and E384162[I]>=0 then
    OTHPAY2[I]=(E384162[I]*E34402B[I])/E23901[I];
if E23901[I]>0 and E384072[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E384162[I]>=0 then
    OTHPAY2[I]=E384162[I]/E23901[I];
if E23901[I]>0 and E384072[I]=4 and E384162[I]>=0 then OTHPAY2[I]=E384162[I]/(2*E23901[I]);

```

Appendix 2: Employment Variable Creation

```
if E23901[I]>0 and E384072[I]=5 and E384162[I]>=0 then OTHPAY2[I]=E384162[I]/(4.3*E23901[I]);
if E23901[I]>0 and E384072[I]=6 and E384162[I]>=0 and E35600[I]>0 then
    OTHPAY2[I]=E384162[I]/(E35600[I]*E23901[I]);
if E23901[I]>0 and E384072[I]=8 and E384162[I]>=0 then OTHPAY2[I]=E384162[I]/(2.15*E23901[I]);
if E23901[I]>0 and E384072[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF2[I]=OTHPF2[I]+1;
*missing value*/;
if E384162[I]>=0 and -4<E23901[I]<0 then OTHPAY2[I]=E23901[I];
if E384162[I]>=0 and E23901[I]=0 then OTHPAY2[I]=-3;
if -4<E384162[I]<0 then OTHPAY2[I]=E384162[I];

/* report non-hourly wage*/
if E34402[I]>0 and E384072[I]=1 and E384162[I]>=0 then OTHPAY2[I]=E384162[I];
if E34402[I]>0 and E384072[I]=2 and E34402B[I]>0 and E384162[I]>=0 then
    OTHPAY2[I]=(E384162[I]*E34402B[I])/E34402[I];
if E34402[I]>0 and E384072[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E384162[I]>=0 then
    OTHPAY2[I]=E384162[I]/E34402[I];
if E34402[I]>0 and E384072[I]=4 and E384162[I]>=0 then OTHPAY2[I]=E384162[I]/(2*E34402[I]);
if E34402[I]>0 and E384072[I]=5 and E384162[I]>=0 then OTHPAY2[I]=E384162[I]/(4.3*E34402[I]);
if E34402[I]>0 and E384072[I]=6 and E384162[I]>=0 and E35600[I]>0 then
    OTHPAY2[I]=E384162[I]/(E35600[I]*E34402[I]);
if E34402[I]>0 and E384072[I]=8 and E384162[I]>=0 then OTHPAY2[I]=E384162[I]/(2.15*E34402[I]);
if E34402[I]>0 and E384072[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF2[I]=OTHPF2[I]+1;
if E384072[I] in (9,14) then OTHPAY2[I]=0;
*missing value*/;
if E384162[I]>=0 and -4<E34402[I]<0 then OTHPAY2[I]=E34402[I];
if E384162[I]>=0 and E34402[I]=0 then OTHPAY2[I]=-3;
if E384072[I]=2 and E34402B[I] le 0 then OTHPAY2[I]=-3;
if E384072[I]=2 and -4<E34402B[I]<0 THNE OTHPAY2[I]=E34402B[I];
if E384072[I]=6 and E35600[I] le 0 then OTHPAY2[I]=-3;
if E384072[I]=6 and -4<E35600[I]<0 then OTHPAY2[I]=E35600[I];
end;

/*for bonuses*/
if E38102[I] ne 1 and E38102[I] ne 3 then do; /*with overtime at the beginning*/
/* report hourly wage at the beginning*/
if E23901[I]>0 and E384073[I]=1 and E384163[I]>=0 then OTHPAY3[I]=E384163[I];
if E23901[I]>0 and E384073[I]=2 and E34430[I]>0 and E384163[I]>=0 then
    OTHPAY3[I]=(E384163[I]*E34430[I])/E23901[I];
if E23901[I]>0 and E384073[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E384163[I]>=0 then
    OTHPAY3[I]=E384163[I]/E23901[I];
if E23901[I]>0 and E384073[I]=4 and E384163[I]>=0 then OTHPAY3[I]=E384163[I]/(2*E23901[I]);
if E23901[I]>0 and E384073[I]=5 and E384163[I]>=0 then OTHPAY3[I]=E384163[I]/(4.3*E23901[I]);
if E23901[I]>0 and E384073[I]=6 and E384163[I]>=0 and E35600[I]>0 then
    OTHPAY3[I]=E384163[I]/(E35600[I]*E23901[I]);
if E23901[I]>0 and E384073[I]=8 and E384163[I]>=0 then OTHPAY3[I]=E384163[I]/(2.15*E23901[I]);
if E23901[I]>0 and E384073[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF3[I]=OTHPF3[I]+1;
*missing value*/;
if E384163[I]>=0 and -4<E23901[I]<0 then OTHPAY3[I]=E23901[I];
if E384163[I]>=0 and E23901[I]=0 then OTHPAY3[I]=-3;
if -4<E384163[I]<0 then OTHPAY3[I]=E384163[I];

/*report non-hourly wage*/
if E34428[I]>0 and E384073[I]=1 and E384163[I]>=0 then OTHPAY3[I]=E384163[I];
```

```

if E34428[I]>0 and E384073[I]=2 and E34430[I]>0 and E384163[I]>=0 then
    OTHPAY3[I]=(E384163[I]*E34430[I])/E34428[I];
if E34428[I]>0 and E384073[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E384163[I]>=0 then
    OTHPAY3[I]=E384163[I]/E34428[I];
if E34428[I]>0 and E384073[I]=4 and E384163[I]>=0 then OTHPAY3[I]=E384163[I]/(2*E34428[I]);
if E34428[I]>0 and E384073[I]=5 and E384163[I]>=0 then OTHPAY3[I]=E384163[I]/(4.3*E34428[I]);
if E34428[I]>0 and E384073[I]=6 and E384163[I]>=0 and E35600[I]>0 then
    OTHPAY3[I]=E384163[I]/(E35600[I]*E34428[I]);
if E34428[I]>0 and E384073[I]=8 and E384163[I]>=0 then OTHPAY3[I]=E384163[I]/(2.15*E34428[I]);
if E34428[I]>0 and E384073[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF3[I]=OTHPF3[I]+1;
/*missing value*/;
if E384163[I]>=0 and -4<E34428[I]<0 then OTHPAY3[I]=E34428[I];
if E384163[I]>=0 and E34428[I]=0 then OTHPAY3[I]=-3;
if -4<E384163[I]<0 then OTHPAY3[I]=E384163[I];
if E384073[I]=2 and E34430[I]<0 then OTHPAY3[I]=-3;
if E384073[I]=2 and -4<E34430[I]<0 THNE OTHPAY3[I]=E34430[I];
if E384073[I]=6 and E35600[I]<0 then OTHPAY3[I]=-3;
if E384073[I]=6 and -4<E35600[I]<0 then OTHPAY3[I]=E35600[I];
end;

if E3800B[I]=1 then do; /*without overtime at the beginning*/
/* report hourly wage at the beginning*/
if E23901[I]>0 and E384073[I]=1 and E384163[I]>=0 then OTHPAY3[I]=E384163[I];
if E23901[I]>0 and E384073[I]=2 and E34402B[I]>0 and E384163[I]>=0 then
    OTHPAY3[I]=(E384163[I]*E34402B[I])/E23901[I];
if E23901[I]>0 and E384073[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E384163[I]>=0 then
    OTHPAY3[I]=E384163[I]/E23901[I];
if E23901[I]>0 and E384073[I]=4 and E384163[I]>=0 then OTHPAY3[I]=E384163[I]/(2*E23901[I]);
if E23901[I]>0 and E384073[I]=5 and E384163[I]>=0 then OTHPAY3[I]=E384163[I]/(4.3*E23901[I]);
if E23901[I]>0 and E384073[I]=6 and E384163[I]>=0 and E35600[I]>0 then
    OTHPAY3[I]=E384163[I]/(E35600[I]*E23901[I]);
if E23901[I]>0 and E384073[I]=8 and E384163[I]>=0 then OTHPAY3[I]=E384163[I]/(2.15*E23901[I]);
if E23901[I]>0 and E384073[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF3[I]=OTHPF3[I]+1;
/*missing value*/;
if E384163[I]>=0 and -4<E23901[I]<0 then OTHPAY3[I]=E23901[I];
if E384163[I]>=0 and E23901[I]=0 then OTHPAY3[I]=-3;
if -4<E384163[I]<0 then OTHPAY3[I]=E384163[I];

/*report non-hourly wage*/
if E34402[I]>0 and E384073[I]=1 and E384163[I]>=0 then OTHPAY3[I]=E384163[I];
if E34402[I]>0 and E384073[I]=2 and E34402B[I]>0 and E384163[I]>=0 then
    OTHPAY3[I]=(E384163[I]*E34402B[I])/E34402[I];
if E34402[I]>0 and E384073[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E384163[I]>=0 then
    OTHPAY3[I]=E384163[I]/E34402[I];
if E34402[I]>0 and E384073[I]=4 and E384163[I]>=0 then OTHPAY3[I]=E384163[I]/(2*E34402[I]);
if E34402[I]>0 and E384073[I]=5 and E384163[I]>=0 then OTHPAY3[I]=E384163[I]/(4.3*E34402[I]);
if E34402[I]>0 and E384073[I]=6 and E384163[I]>=0 and E35600[I]>0 then
    OTHPAY3[I]=E384163[I]/(E35600[I]*E34402[I]);
if E34402[I]>0 and E384073[I]=8 and E384163[I]>=0 then OTHPAY3[I]=E384163[I]/(2.15*E34402[I]);
if E34402[I]>0 and E384073[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF3[I]=OTHPF3[I]+1;
if E384073[I] in (9,14) then OTHPAY3[I]=0;
/*missing value*/;
if E384163[I]>=0 and -4<E34402[I]<0 then OTHPAY3[I]=E34402[I];
if E384163[I]>=0 and E34402[I]=0 then OTHPAY3[I]=-3;

```

```

if E384073[I]=2 and E34402B[I] le 0 then OTHPAY3[I]=-3;
if E384073[I]=2 and -4<E34402B[I]<0 THNE OTHPAY3[I]=E34402B[I];
if E384073[I]=6 and E35600[I] le 0 then OTHPAY3[I]=-3;
if E384073[I]=6 and -4<E35600[I]<0 then OTHPAY3[I]=E35600[I];
end;

/*for incentive pay*/
if E38102[I] ne 1 and E38102[I] ne 3 then do; /*with overtime at the beginning*/
  /*report hourly wage at the beginning*/
  if E23901[I]>0 and E384074[I]=1 and E384164[I]>=0 then OTHPAY4[I]=E384164[I];
  if E23901[I]>0 and E384074[I]=2 and E34430[I]>0 and E384164[I]>=0 then
    OTHPAY4[I]=(E384164[I]*E34430[I])/E23901[I];
  if E23901[I]>0 and E384074[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E384164[I]>=0 then
    OTHPAY4[I]=E384164[I]/E23901[I];
  if E23901[I]>0 and E384074[I]=4 and E384164[I]>=0 then OTHPAY4[I]=E384164[I]/(2*E23901[I]);
  if E23901[I]>0 and E384074[I]=5 and E384164[I]>=0 then OTHPAY4[I]=E384164[I]/(4.3*E23901[I]);
  if E23901[I]>0 and E384074[I]=6 and E384164[I]>=0 and E35600[I]>0 then
    OTHPAY4[I]=E384164[I]/(E35600[I]*E23901[I]);
  if E23901[I]>0 and E384074[I]=8 and E384164[I]>=0 then OTHPAY4[I]=E384164[I]/(2.15*E23901[I]);
  if E23901[I]>0 and E384074[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF4[I]=OTHPF4[I]+1;
  *missing value*;
  if E384164[I]>=0 and -4<E23901[I]<0 then OTHPAY4[I]=E23901[I];
  if E384164[I]>=0 and E23901[I]=0 then OTHPAY4[I]=-3;
  if -4<E384164[I]<0 then OTHPAY4[I]=E384164[I];

/*report non-hourly wage*/
if E34428[I]>0 and E384074[I]=1 and E384164[I]>=0 then OTHPAY4[I]=E384164[I];
if E34428[I]>0 and E384074[I]=2 and E34430[I]>0 and E384164[I]>=0 then
  OTHPAY4[I]=(E384164[I]*E34430[I])/E34428[I];
if E34428[I]>0 and E384074[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E384164[I]>=0 then
  OTHPAY4[I]=E384164[I]/E34428[I];
if E34428[I]>0 and E384074[I]=4 and E384164[I]>=0 then OTHPAY4[I]=E384164[I]/(2*E34428[I]);
if E34428[I]>0 and E384074[I]=5 and E384164[I]>=0 then OTHPAY4[I]=E384164[I]/(4.3*E34428[I]);
if E34428[I]>0 and E384074[I]=6 and E384164[I]>=0 and E35600[I]>0 then
  OTHPAY4[I]=E384164[I]/(E35600[I]*E34428[I]);
if E34428[I]>0 and E384074[I]=8 and E384164[I]>=0 then OTHPAY4[I]=E384164[I]/(2.15*E34428[I]);
if E34428[I]>0 and E384074[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
  OTHPF4[I]=OTHPF4[I]+1;
  *missing value*;
  if E384164[I]>=0 and -4<E34428[I]<0 then OTHPAY4[I]=E34428[I];
  if E384164[I]>=0 and E34428[I]=0 then OTHPAY4[I]=-3;
  if -4<E384164[I]<0 then OTHPAY4[I]=E384164[I];
  if E384074[I]=2 and E34430[I] le 0 then OTHPAY4[I]=-3;
  if E384074[I]=2 and -4<E34430[I]<0 THNE OTHPAY4[I]=E34430[I];
  if E384074[I]=6 and E35600[I] le 0 then OTHPAY4[I]=-3;
  if E384074[I]=6 and -4<E35600[I]<0 then OTHPAY4[I]=E35600[I];
end;

if E3800B[I]=1 then do; /*without overtime at the beginning*/
  /*report hourly wage at the beginning*/
  if E23901[I]>0 and E384074[I]=1 and E384164[I]>=0 then OTHPAY4[I]=E384164[I];
  if E23901[I]>0 and E384074[I]=2 and E34402B[I]>0 and E384164[I]>=0 then
    OTHPAY4[I]=(E384164[I]*E34402B[I])/E23901[I];
  if E23901[I]>0 and E384074[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E384164[I]>=0 then
    OTHPAY4[I]=E384164[I]/E23901[I];
  if E23901[I]>0 and E384074[I]=4 and E384164[I]>=0 then OTHPAY4[I]=E384164[I]/(2*E23901[I]);

```

Appendix 2: Employment Variable Creation

```

if E23901[I]>0 and E384074[I]=5 and E384164[I]>=0 then OTHPAY4[I]=E384164[I]/(4.3*E23901[I]);
if E23901[I]>0 and E384074[I]=6 and E384164[I]>=0 and E35600[I]>0 then
    OTHPAY4[I]=E384164[I]/(E35600[I]*E23901[I]);
if E23901[I]>0 and E384074[I]=8 and E384164[I]>=0 then OTHPAY4[I]=E384164[I]/(2.15*E23901[I]);
if E23901[I]>0 and E384074[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF4[I]=OTHPF4[I]+1;
*missing value*;
if E384164[I]>=0 and -4<E23901[I]<0 then OTHPAY4[I]=E23901[I];
if E384164[I]>=0 and E23901[I]=0 then OTHPAY4[I]=-3;
if -4<E384164[I]<0 then OTHPAY4[I]=E384164[I];

/*report non-hourly wage*/
if E34402[I]>0 and E384074[I]=1 and E384164[I]>=0 then OTHPAY4[I]=E384164[I];
if E34402[I]>0 and E384074[I]=2 and E34402B[I]>0 and E384164[I]>=0 then
    OTHPAY4[I]=(E384164[I]*E34402B[I])/E34402[I];
if E34402[I]>0 and E384074[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E384164[I]>=0 then
    OTHPAY4[I]=E384164[I]/E34402[I];
if E34402[I]>0 and E384074[I]=4 and E384164[I]>=0 then OTHPAY4[I]=E384164[I]/(2*E34402[I]);
if E34402[I]>0 and E384074[I]=5 and E384164[I]>=0 then OTHPAY4[I]=E384164[I]/(4.3*E34402[I]);
if E34402[I]>0 and E384074[I]=6 and E384164[I]>=0 and E35600[I]>0 then
    OTHPAY4[I]=E384164[I]/(E35600[I]*E34402[I]);
if E34402[I]>0 and E384074[I]=8 and E384164[I]>=0 then OTHPAY4[I]=E384164[I]/(2.15*E34402[I]);
if E34402[I]>0 and E384074[I] in (0,7,12,13,114,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF4[I]=OTHPF4[I]+1;
if E384074[I] in (9,14) then OTHPAY4[I]=0;
*missing value*;
if E384164[I]>=0 and -4<E34402[I]<0 then OTHPAY4[I]=E34402[I];
if E384164[I]>=0 and E34402[I]=0 then OTHPAY4[I]=-3;
if E384074[I]=2 and E34402B[I] le 0 then OTHPAY4[I]=-3;
if E384074[I]=2 and -4<E34402B[I]<0 THNE OTHPAY4[I]=E34402B[I];
if E384074[I]=6 and E35600[I] le 0 then OTHPAY4[I]=-3;
if E384074[I]=6 and -4<E35600[I]<0 then OTHPAY4[I]=E35600[I];
end;

/*for others*/
if E38102[I] ne 1 and E38102[I] ne 3 then do; /*with overtime at the beginning*/
/* report hourly wage at the beginning*/
if E23901[I]>0 and E384075[I]=1 and E384165[I]>=0 then OTHPAY5[I]=E384165[I];
if E23901[I]>0 and E384075[I]=2 and E34430[I]>0 and E384165[I]>=0 then
    OTHPAY5[I]=(E384165[I]*E34430[I])/E23901[I];
if E23901[I]>0 and E384075[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E384165[I]>=0 then
    OTHPAY5[I]=E384165[I]/E23901[I];
if E23901[I]>0 and E384075[I]=4 and E384165[I]>=0 then OTHPAY5[I]=E384165[I]/(2*E23901[I]);
if E23901[I]>0 and E384075[I]=5 and E384165[I]>=0 then OTHPAY5[I]=E384165[I]/(4.3*E23901[I]);
if E23901[I]>0 and E384075[I]=6 and E384165[I]>=0 and E35600[I]>0 then
    OTHPAY5[I]=E384165[I]/(E35600[I]*E23901[I]);
if E23901[I]>0 and E384075[I]=8 and E384165[I]>=0 then OTHPAY5[I]=E384165[I]/(2.15*E23901[I]);
if E23901[I]>0 and E384075[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF5[I]=OTHPF5[I]+1;
*missing value*;
if E384165[I]>=0 and -4<E23901[I]<0 then OTHPAY5[I]=E23901[I];
if E384165[I]>=0 and E23901[I]=0 then OTHPAY5[I]=-3;
if -4<E384165[I]<0 then OTHPAY5[I]=E384165[I];

/*report non-hourly wage*/
if E34428[I]>0 and E384075[I]=1 and E384165[I]>=0 then OTHPAY5[I]=E384165[I];

```

Appendix 2: Employment Variable Creation

```

if E34428[I]>0 and E384075[I]=2 and E34430[I]>0 and E384165[I]>=0 then
    OTHPAY5[I]=(E384165[I]*E34430[I])/E34428[I];
if E34428[I]>0 and E384075[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E384165[I]>=0 then
    OTHPAY5[I]=E384165[I]/E34428[I];
if E34428[I]>0 and E384075[I]=4 and E384165[I]>=0 then OTHPAY5[I]=E384165[I]/(2*E34428[I]);
if E34428[I]>0 and E384075[I]=5 and E384165[I]>=0 then OTHPAY5[I]=E384165[I]/(4.3*E34428[I]);
if E34428[I]>0 and E384075[I]=6 and E384165[I]>=0 and E35600[I]>0 then
    OTHPAY5[I]=E384165[I]/(E35600[I]*E34428[I]);
if E34428[I]>0 and E384075[I]=8 and E384165[I]>=0 then OTHPAY5[I]=E384165[I]/(2.15*E34428[I]);
if E34428[I]>0 and E384075[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF5[I]=OTHPF5[I]+1;
*missing value*/;
if E384165[I]>=0 and -4<E34428[I]<0 then OTHPAY5[I]=E34428[I];
if E384165[I]>=0 and E34428[I]=0 then OTHPAY5[I]=-3;
if -4<E384165[I]<0 then OTHPAY5[I]=E384165[I];
if E384075[I]=2 and E34430[I] le 0 then OTHPAY5[I]=-3;
if E384075[I]=2 and -4<E34430[I]<0 THNE OTHPAY5[I]=E34430[I];
if E384075[I]=6 and E35600[I] le 0 then OTHPAY5[I]=-3;
if E384075[I]=6 and -4<E35600[I]<0 then OTHPAY5[I]=E35600[I];
end;

if E3800B[I]=1 then do; /*without overtime at the beginning*/
/* report hourly wage at the beginning*/
if E23901[I]>0 and E384075[I]=1 and E384165[I]>=0 then OTHPAY5[I]=E384165[I];
if E23901[I]>0 and E384075[I]=2 and E34402B[I]>0 and E384165[I]>=0 then
    OTHPAY5[I]=(E384165[I]*E34402B[I])/E23901[I];
if E23901[I]>0 and E384075[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E384165[I]>=0 then
    OTHPAY5[I]=E384165[I]/E23901[I];
if E23901[I]>0 and E384075[I]=4 and E384165[I]>=0 then OTHPAY5[I]=E384165[I]/(2*E23901[I]);
if E23901[I]>0 and E384075[I]=5 and E384165[I]>=0 then OTHPAY5[I]=E384165[I]/(4.3*E23901[I]);
if E23901[I]>0 and E384075[I]=6 and E384165[I]>=0 and E35600[I]>0 then
    OTHPAY5[I]=E384165[I]/(E35600[I]*E23901[I]);
if E23901[I]>0 and E384075[I]=8 and E384165[I]>=0 then OTHPAY5[I]=E384165[I]/(2.15*E23901[I]);
if E23901[I]>0 and E384075[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF5[I]=OTHPF5[I]+1;
*missing value*/;
if E384165[I]>=0 and -4<E23901[I]<0 then OTHPAY5[I]=E23901[I];
if E384165[I]>=0 and E23901[I]=0 then OTHPAY5[I]=-3;
if -4<E384165[I]<0 then OTHPAY5[I]=E384165[I];

/*report non-hourly wage*/
if E34402[I]>0 and E384075[I]=1 and E384165[I]>=0 then OTHPAY5[I]=E384165[I];
if E34402[I]>0 and E384075[I]=2 and E34402B[I]>0 and E384165[I]>=0 then
    OTHPAY5[I]=(E384165[I]*E34402B[I])/E34402[I];
if E34402[I]>0 and E384075[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E384165[I]>=0 then
    OTHPAY5[I]=E384165[I]/E34402[I];
if E34402[I]>0 and E384075[I]=4 and E384165[I]>=0 then OTHPAY5[I]=E384165[I]/(2*E34402[I]);
if E34402[I]>0 and E384075[I]=5 and E384165[I]>=0 then OTHPAY5[I]=E384165[I]/(4.3*E34402[I]);
if E34402[I]>0 and E384075[I]=6 and E384165[I]>=0 and E35600[I]>0 then
    OTHPAY5[I]=E384165[I]/(E35600[I]*E34402[I]);
if E34402[I]>0 and E384075[I]=8 and E384165[I]>=0 then OTHPAY5[I]=E384165[I]/(2.15*E34402[I]);
if E34402[I]>0 and E384075[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF5[I]=OTHPF5[I]+1;
if E384075[I] in (9,14) then OTHPAY5[I]=0;
*missing value*/;
if E384165[I]>=0 and -4<E34402[I]<0 then OTHPAY5[I]=E34402[I];
if E384165[I]>=0 and E34402[I]=0 then OTHPAY5[I]=-3;

```

Appendix 2: Employment Variable Creation

```
if E384075[I]=2 and E34402B[I] le 0 then OTHPAY5[I]=-3;
if E384075[I]=2 and -4<E34402B[I]<0 THNE OTHPAY5[I]=E34402B[I];
if E384075[I]=6 and E35600[I] le 0 then OTHPAY5[I]=-3;
if E384075[I]=6 and -4<E35600[I]<0 then OTHPAY5[I]=E35600[I];
end;
end;

/** case vi. more then one compensation at the beginning, diff no. of hours. ***/
if E38330[I]=1 and E20700[I]=1 and (E38102[I]=1 or E38102[I]=3) then do;

/* for tips*/
if E38103[I]>0 and E384071[I]=1 and E384161[I]>=0 then OTHPAY1[I]=E384161[I];
if E38103[I]>0 and E384071[I]=2 and E38116B[I]>0 and E384161[I]>=0 then
    OTHPAY1[I]=(E384161[I]*E38116B[I])/E38103[I];
if E38103[I]>0 and E384071[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E384161[I]>=0 then
    OTHPAY1[I]=E384161[I]/E38103[I];
if E38103[I]>0 and E384071[I]=4 and E384161[I]>=0 then OTHPAY1[I]=E384161[I]/(2*E38103[I]);
if E38103[I]>0 and E384071[I]=5 and E384161[I]>=0 then OTHPAY1[I]=E384161[I]/(4.3*E38103[I]);
if E38103[I]>0 and E384071[I]=6 and E384161[I]>=0 and E35600[I]>0 then
    OTHPAY1[I]=E384161[I]/(E35600[I]*E38103[I]);
if E38103[I]>0 and E384071[I]=8 and E384161[I]>=0 then OTHPAY1[I]=E384161[I]/(2.15*E38103[I]);
if E38103[I]>0 and E384071[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF1[I]=OTHPF1[I]+1;
if E384071[I] in (9,14) then OTHPAY1[I]=0;
/*missing value*/;

if E384161[I]>=0 and -4<E38103[I]<0 then OTHPAY1[I]=E38103[I];
if E384161[I]>=0 and E38103[I]=0 then OTHPAY1[I]=-3;
if -4<E384161[I]<0 then OTHPAY1[I]=E384161[I];
if E384071[I]=2 and E38116B[I]=0 then OTHPAY1[I]=-3;
if E384071[I]=2 and -4<E38116B[I]<0 then OTHPAY1[I]=E38116B[I];
if E384071[I]=6 and E35600[I] le 0 then OTHPAY1[I]=-3;
if E384071[I]=6 and -4<E35600[I]<0 then OTHPAY1[I]=E35600[I];

/*for commissions*/
if E38103[I]>0 and E384072[I]=1 and E384162[I]>=0 then OTHPAY2[I]=E384162[I];
if E38103[I]>0 and E384072[I]=2 and E38116B[I]>0 and E384162[I]>=0 then
    OTHPAY2[I]=(E384162[I]*E38116B[I])/E38103[I];
if E38103[I]>0 and E384072[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E384162[I]>=0 then
    OTHPAY2[I]=E384162[I]/E38103[I];
if E38103[I]>0 and E384072[I]=4 and E384162[I]>=0 then OTHPAY2[I]=E384162[I]/(2*E38103[I]);
if E38103[I]>0 and E384072[I]=5 and E384162[I]>=0 then OTHPAY2[I]=E384162[I]/(4.3*E38103[I]);
if E38103[I]>0 and E384072[I]=6 and E384162[I]>=0 and E35600[I]>0 then
    OTHPAY2[I]=E384162[I]/(E35600[I]*E38103[I]);
if E38103[I]>0 and E384072[I]=8 and E384162[I]>=0 then OTHPAY2[I]=E384162[I]/(2.15*E38103[I]);
if E38103[I]>0 and E384072[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF2[I]=OTHPF2[I]+1;
if E384072[I] in (9,14) then OTHPAY2[I]=0;
/*missing value*/;

if E384162[I]>=0 and -4<E38103[I]<0 then OTHPAY2[I]=E38103[I];
if E384162[I]>=0 and E38103[I]=0 then OTHPAY2[I]=-3;
if -4<E384162[I]<0 then OTHPAY2[I]=E384162[I];
if E384072[I]=2 and E38116B[I]=0 then OTHPAY2[I]=-3;
if E384072[I]=2 and -4<E38116B[I]<0 then OTHPAY2[I]=E38116B[I];
if E384072[I]=6 and E35600[I] le 0 then OTHPAY2[I]=-3;
if E384072[I]=6 and -4<E35600[I]<0 then OTHPAY2[I]=E35600[I];

/*for bonuses*/

```

Appendix 2: Employment Variable Creation

```
if E38103[I]>0 and E384073[I]=1 and E384163[I]>=0 then OTHPAY3[I]=E384163[I];
if E38103[I]>0 and E384073[I]=2 and E38116B[I]>0 and E384163[I]>=0 then
    OTHPAY3[I]=(E384163[I]*E38116B[I])/E38103[I];
if E38103[I]>0 and E384073[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E384163[I]>=0 then
    OTHPAY3[I]=E384163[I]/E38103[I];
if E38103[I]>0 and E384073[I]=4 and E384163[I]>=0 then OTHPAY3[I]=E384163[I]/(2*E38103[I]);
if E38103[I]>0 and E384073[I]=5 and E384163[I]>=0 then OTHPAY3[I]=E384163[I]/(4.3*E38103[I]);
if E38103[I]>0 and E384073[I]=6 and E384163[I]>=0 and E35600[I]>0 then
    OTHPAY3[I]=E384163[I]/(E35600[I]*E38103[I]);
if E38103[I]>0 and E384073[I]=8 and E384163[I]>=0 then OTHPAY3[I]=E384163[I]/(2.15*E38103[I]);
if E38103[I]>0 and E384073[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF3[I]=OTHPF3[I]+1;
if E384073[I] in (9,14) then OTHPAY3[I]=0;
*missing value*/;
if E384163[I]>=0 and -4<E38103[I]<0 then OTHPAY3[I]=E38103[I];
if E384163[I]>=0 and E38103[I]=0 then OTHPAY3[I]=-3;
if -4<E384163[I]<0 then OTHPAY3[I]=E384163[I];
if E384073[I]=2 and E38116B[I]=0 then OTHPAY3[I]=-3;
if E384073[I]=2 and -4<E38116B[I]<0 then OTHPAY3[I]=E38116B[I];
if E384073[I]=6 and E35600[I] le 0 then OTHPAY3[I]=-3;
if E384073[I]=6 and -4<E35600[I]<0 then OTHPAY3[I]=E35600[I];

/*for incentive pay*/
if E38103[I]>0 and E384074[I]=1 and E384164[I]>=0 then OTHPAY4[I]=E384164[I];
if E38103[I]>0 and E384074[I]=2 and E38116B[I]>0 and E384164[I]>=0 then
    OTHPAY4[I]=(E384164[I]*E38116B[I])/E38103[I];
if E38103[I]>0 and E384074[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E384164[I]>=0 then
    OTHPAY4[I]=E384164[I]/E38103[I];
if E38103[I]>0 and E384074[I]=4 and E384164[I]>=0 then OTHPAY4[I]=E384164[I]/(2*E38103[I]);
if E38103[I]>0 and E384074[I]=5 and E384164[I]>=0 then OTHPAY4[I]=E384164[I]/(4.3*E38103[I]);
if E38103[I]>0 and E384074[I]=6 and E384164[I]>=0 and E35600[I]>0 then
    OTHPAY4[I]=E384164[I]/(E35600[I]*E38103[I]);
if E38103[I]>0 and E384074[I]=8 and E384164[I]>=0 then OTHPAY4[I]=E384164[I]/(2.15*E38103[I]);
if E38103[I]>0 and E384074[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF4[I]=OTHPF4[I]+1;
if E384074[I] in (9,14) then OTHPAY4[I]=0;
*missing value*/;
if E384164[I]>=0 and -4<E38103[I]<0 then OTHPAY4[I]=E38103[I];
if E384164[I]>=0 and E38103[I]=0 then OTHPAY4[I]=-3;
if -4<E384164[I]<0 then OTHPAY4[I]=E384164[I];
if E384074[I]=2 and E38116B[I]=0 then OTHPAY4[I]=-3;
if E384074[I]=2 and -4<E38116B[I]<0 then OTHPAY4[I]=E38116B[I];
if E384074[I]=6 and E35600[I] le 0 then OTHPAY4[I]=-3;
if E384074[I]=6 and -4<E35600[I]<0 then OTHPAY4[I]=E35600[I];

/*for others*/
if E38103[I]>0 and E384075[I]=1 and E384165[I]>=0 then OTHPAY5[I]=E384165[I];
if E38103[I]>0 and E384075[I]=2 and E38116B[I]>0 and E384165[I]>=0 then
    OTHPAY5[I]=(E384165[I]*E38116B[I])/E38103[I];
if E38103[I]>0 and E384075[I] in (3,0,7,12,13,15,16,17,99,21,22,23,24,25,26,28) and E384165[I]>=0 then
    OTHPAY5[I]=E384165[I]/E38103[I];
if E38103[I]>0 and E384075[I]=4 and E384165[I]>=0 then OTHPAY5[I]=E384165[I]/(2*E38103[I]);
if E38103[I]>0 and E384075[I]=5 and E384165[I]>=0 then OTHPAY5[I]=E384165[I]/(4.3*E38103[I]);
if E38103[I]>0 and E384075[I]=6 and E384165[I]>=0 and E35600[I]>0 then
    OTHPAY5[I]=E384165[I]/(E35600[I]*E38103[I]);
if E38103[I]>0 and E384075[I]=8 and E384165[I]>=0 then OTHPAY5[I]=E384165[I]/(2.15*E38103[I]);
```

Appendix 2: Employment Variable Creation

```
if E38103[I]>0 and E384075[I] in (0,7,12,13,14,15,16,17,99,21,22,23,24,25,26,28) then
    OTHPF5[I]=OTHPF5[I]+1;
if E384075[I] in (9,14) then OTHPAY5[I]=0;
*missing value*/;
if E384165[I]>=0 and -4<E38103[I]<0 then OTHPAY5[I]=E38103[I];
if E384165[I]>=0 and E38103[I]=0 then OTHPAY5[I]=-3;
if -4<E384165[I]<0 then OTHPAY5[I]=E384165[I];
if E384075[I]=2 and E38116B[I]=0 then OTHPAY5[I]=-3;
if E384075[I]=2 and -4<E38116B[I]<0 then OTHPAY5[I]=E38116B[I];
if E384075[I]=6 and E35600[I] le 0 then OTHPAY5[I]=-3;
if E384075[I]=6 and -4<E35600[I]<0 then OTHPAY5[I]=E35600[I];
end;
end;
end;
```

***** Part 4: Overall End Hourly Compensation *****

```
do I=1 to 9;
if E37901B[I]=1 or E59900[I]=1 then do;
    HRCOMP[I]=0;
    if HRWG[I] ge 0 then HRCOMP[I]=HRCOMP[I]+HRWG[I];
    if OT[I] ge 0 then HRCOMP[I]=HRCOMP[I]+OT[I];
    if OTHPAY1[I] ge 0 then HRCOMP[I]=HRCOMP[I]+OTHPAY1[I];
    if OTHPAY2[I] ge 0 then HRCOMP[I]=HRCOMP[I]+OTHPAY2[I];
    if OTHPAY3[I] ge 0 then HRCOMP[I]=HRCOMP[I]+OTHPAY3[I];
    if OTHPAY4[I] ge 0 then HRCOMP[I]=HRCOMP[I]+OTHPAY4[I];
    if OTHPAY5[I] ge 0 then HRCOMP[I]=HRCOMP[I]+OTHPAY5[I];
    if -4<HRWG[I]<0 or -4<OT[I]<0 or -4<OTHPAY1[I]<0 or -4<OTHPAY2[I]<0 or -4<OTHPAY3[I]<0 or -
        4<OTHPAY4[I]<0 or -4<OTHPAY5[I]<0 then HRCOMP[I]=-3;
    if HRWG[I]=-4 then HRCOMP[I]=-4;
end;
end;
```

***** Section 3: Job lasts 13 weeks or less *****

```
JLENG01=-4;           JLENG02=-4;           JLENG03=-4;
JLENG04=-4;           JLENG05=-4;           JLENG06=-4;
JLENG07=-4;           JLENG08=-4;           JLENG09=-4;
```

```
array JLENG JLENG01 JLENG02 JLENG03 JLENG04 JLENG05 JLENG06 JLENG07 JLENG08 JLENG09
    JLENG10;
```

```
do I=1 to 9;
if E37701[I]=0 then JLENG[I]=0;
if E58201[I]=0 then JLENG[I]=0;
if E37901B[I]>-1 or E59900[I]>-1 or E58401[I]>-1 then do;
    if E37901B[I]=1 or E59900[I]=1 or E58401[I]=0 then JLENG[I]=1;
    else JLENG[I]=0;
end;
end;
```

```
do I=1 to 9;
if JLENG[I]>-4 and HRWG[I]=-4 then HRWG[I]=-3;
if JLENG[I]>-4 and HRCOMP[I]=-4 then HRCOMP[I]=-3;
end;
```

```
array HRWGR HRWGR01 HRWGR02 HRWGR03 HRWGR04 HRWGR05 HRWGR06 HRWGR07  
    HRWGR08 HRWGR09 HRWGR10;  
  
HRWGR01=-4;          HRWGR02=-4;          HRWGR03=-4;  
HRWGR04=-4;          HRWGR05=-4;          HRWGR06=-4;  
HRWGR07=-4;          HRWGR08=-4;          HRWGR09=-4;  
  
do I=1 to 9;  
    if HRWG[I] ne -4 then do;  
        HRWGR[I]=ROUND(HRWG[I],1);  
    end;  
end;  
  
array HRCOMR HRCOMR01 HRCOMR02 HRCOMR03 HRCOMR04 HRCOMR05 HRCOMR06  
    HRCOMR07 HRCOMR08 HRCOMR09;  
  
HRCOMR01=-4;          HRCOMR02=-4;          HRCOMR03=-4;  
HRCOMR04=-4;          HRCOMR05=-4;          HRCOMR06=-4;  
HRCOMR07=-4;          HRCOMR08=-4;          HRCOMR09=-4;  
  
array HRCOMP HRCOMP01 HRCOMP02 HRCOMP03 HRCOMP04 HRCOMP05 HRCOMP06  
    HRCOMP07 HRCOMP08 HRCOMP09;  
  
do I=1 to 9;  
    if HRCOMP[I] ne -4 then do;  
        HRCOMR[I]=ROUND(HRCOMP[I],1);  
    end;  
end;  
  
endsas;
```

NUMBER OF WEEKS WORKED DURING 19XX

Variables Created: CV_WKSWK_YR.80 – CV_WKSWK_YR.99

Programs Used

This program uses **bdate1.sas** and **emp_begin.sas** as input (see the first page of this appendix for details).

This program counts the number of weeks each respondent worked at any employee-type job for each year of potential work activity (1980-99). Respondents not working in a given year are given a default value of zero (0) weeks worked. Otherwise, the variable indicates the actual cumulative weeks worked on all jobs in that year.

```
/* Section 1: Create variables for # of weeks worked at
any job during a given year based on round 2 data. */
```

```
array job1wks (i) wk1_1-wk1_1044;
array job2wks (i) wk2_1-wk2_1044;
array job3wks (i) wk3_1-wk3_1044;
array job4wks (i) wk4_1-wk4_1044;
array job5wks (i) wk5_1-wk5_1044;
array job6wks (i) wk6_1-wk6_1044;
array job7wks (i) wk7_1-wk7_1044;
array job8wks (i) wk8_1-wk8_1044;
array job9wks (i) wk9_1-wk9_1044;
array alljobs (i) wks1-wks1044;
array starw (i) starw1-starw9;
array stopw (i) stopw1-stopw9;
array uid (i) uid1-uid9;
```

```
do over alljobs; alljobs=0; end;
```

```
/** Overlay multiple jobs over JOB 1 work weeks */
/* "Alljobs" is the event history array that accounts for
all jobs worked over 1980-1999. */
```

```
do over job1wks; alljobs=job1wks; end;
```

```
do over alljobs;
  if job2wks=1 then do; alljobs=job2wks; end;
  if job2wks=-3 and alljobs=0 then do;
    alljobs=job2wks; end;
end;
```

```
/* The program repeats the "do over all jobs" command
for jobs 3-9. The loops are not shown here; contact
NLS User Services for more information. */
```

```
/* Calculate cumulative wks on all jobs for each year */
/* 1980 */
wksw80=0;
do i=1 to 52;
  if alljobs=1 then do; wksw80=wksw80+1; end;
end;
do i=1 to 52;
  if alljobs=-3 then do; wksw80=-3; end;
end;
```

```
*****At this point the program loops through the
same code used above for 1980 for each year 1981-98,
creating the variables wks81, wks82, wks83, and so on
through wks99. These loops are deleted due to space
considerations; users who need to see the entire code
should contact NLS User Services. The week numbers
for the "do i" statement for each year are as follows:
```

| | | | |
|------|---------|------|----------|
| 1981 | 53-104 | 1991 | 575-626 |
| 1982 | 105-156 | 1992 | 627-678 |
| 1983 | 157-209 | 1993 | 679-730 |
| 1984 | 210-261 | 1994 | 731-783 |
| 1985 | 262-313 | 1995 | 784-835 |
| 1986 | 314-365 | 1996 | 836-887 |
| 1987 | 366-417 | 1997 | 888-939 |
| 1988 | 418-470 | 1998 | 940-991 |
| 1989 | 471-522 | 1999 | 992-1044 |
| 1990 | 523-574 | | ***** |

```
do i=1 to 7;
/* start date invalid */
if starw<0 and starw>-4 then do;
```

```
  if stopw>1 then do; wksw80=-3; end;
  if stopw>52 then do; wksw81=-3; end;
  if stopw>104 then do; wksw82=-3; end;
  if stopw>156 then do; wksw83=-3; end;
  if stopw>209 then do; wksw84=-3; end;
  if stopw>261 then do; wksw85=-3; end;
  if stopw>313 then do; wksw86=-3; end;
  if stopw>365 then do; wksw87=-3; end;
  if stopw>417 then do; wksw88=-3; end;
  if stopw>470 then do; wksw89=-3; end;
  if stopw>522 then do; wksw90=-3; end;
  if stopw>574 then do; wksw91=-3; end;
  if stopw>626 then do; wksw92=-3; end;
  if stopw>678 then do; wksw93=-3; end;
  if stopw>730 then do; wksw94=-3; end;
  if stopw>783 then do; wksw95=-3; end;
  if stopw>834 then do; wksw96=-3; end;
  if stopw>887 then do; wksw97=-3; end;
  if stopw>939 then do; wksw98=-3; end;
  if stopw>991 then do; wksw99=-3; end;
end;
```

```
/* stop date invalid */
if stopw<0 and stopw>-4 then do;
```

```

if starw<53 then do; wksw80=-3; end;
if starw<105 then do; wksw81=-3; end;
if starw<157 then do; wksw82=-3; end;
if starw<210 then do; wksw83=-3; end;
if starw<262 then do; wksw84=-3; end;
if starw<314 then do; wksw85=-3; end;
if starw<366 then do; wksw86=-3; end;
if starw<418 then do; wksw87=-3; end;
if starw<471 then do; wksw88=-3; end;
if starw<523 then do; wksw89=-3; end;
if starw<575 then do; wksw90=-3; end;
if starw<627 then do; wksw91=-3; end;
if starw<679 then do; wksw92=-3; end;
if starw<731 then do; wksw93=-3; end;
if starw<784 then do; wksw94=-3; end;
if starw<836 then do; wksw95=-3; end;
if starw<888 then do; wksw96=-3; end;
if starw<940 then do; wksw97=-3; end;
if starw<992 then do; wksw98=-3; end;
if starw<1044 then do; wksw99=-3; end;
end;

/*** Include valid skips ***/
if e200=0 then do;
  wksw80=-4; wksw81=-4; wksw82=-4; wksw83=-4;
  wksw84=-4; wksw85=-4; wksw86=-4; wksw87=-4;
  wksw88=-4; wksw89=-4; wksw90=-4; wksw91=-4;
  wksw92=-4; wksw93=-4; wksw94=-4; wksw95=-4;
  wksw96=-4; wksw97=-4; wksw98=-4; wksw99=-4;
end;
if e200=-3 then do;
  wksw80=-3; wksw81=-3; wksw82=-3; wksw83=-3;
  wksw84=-3; wksw85=-3; wksw86=-3; wksw87=-3;
  wksw88=-3; wksw89=-3; wksw90=-3; wksw91=-3;
  wksw92=-3; wksw93=-3; wksw94=-3; wksw95=-3;
  wksw96=-3; wksw97=-3; wksw98=-3; wksw99=-3;
end;
if e200=-5 then do;
  wksw80=-5; wksw81=-5; wksw82=-5; wksw83=-5;
  wksw84=-5; wksw85=-5; wksw86=-5; wksw87=-5;
  wksw88=-5; wksw89=-5; wksw90=-5; wksw91=-5;
  wksw92=-5; wksw93=-5; wksw94=-5; wksw95=-5;
  wksw96=-5; wksw97=-5; wksw98=-5; wksw99=-5;
end;

check=0;
do over starw;
  if starw>0 then do; if starw<dliwk then check=1; end;
end;

/* Section 2: Take the created variables for total weeks
   worked in any job during a given year for Round 1 and
   Round 2 and add them. One variable is created for
   each year 1980-1999. The default is set to zero. */

/* Define r1ten99 as zero to match up array lengths */
r1ten99=0;

array r1ten (20) r1ten80-r1ten99; /* R1 created var. */
array wksw (20) wksw80-wksw99; /* R2 created var. */
array twks (20) twks80-twks99;

/* Initialize weeks worked in a given year to be zero */
do i=1 to 20; twks(i)=0; end;

/* Begin by splitting up periods where R1 and R2
   exclusively collect weeks worked information. Define
   the R1 interview year as the split. Any weeks worked
   information collected before the R1 interview year
   should be independent of data collected in R2. */
do i=1 to 20;
  if r1ten(i)>0 then do; twks(i)=r1ten(i); end;
  if wksw(i)>0 then do; twks(i)=wksw(i); end;
  if r1ten(i)>0 and wksw(i)>0 then do;
    twks(i)=r1ten(i)+wksw(i); end;
end;

/* If either created variable from R1 or R2 is a -1, -2 or
   -3, then the overall created variable will be -1, -2 or -3,
   respectively. */
do i=1 to 20;
  if -4<r1ten(i)<0 then do; twks(i)=r1ten(i); end;
  if wksw(i)<0 then do; twks(i)=wksw(i); end;
end;

/* Special hand edit case. This respondent reported was
   interviewed in R1 and started a new job later that week.
   The UID code begins with "98" but the start week
   must be updated by one week. Otherwise, the program
   has the respondent working 53 weeks in 1998. */
if pubid=525 then twks98=52;

oops=0;
do i=16 to 20; if twks(i)>53 then do; oops=1; end; end;

/* The following hand edit cases result from
   respondents reporting jobs during the R2 interview that
   began before the R1 interview date. */
if pubid=32 then twks97=52;
if pubid=2398 then do; twks96=52; twks97=50; end;
if pubid=2698 then do; twks96=52; twks97=48; end;
if pubid=3222 then twks97=52;
if pubid=3499 then twks97=40;
if pubid=4505 then twks97=52;
if pubid=4921 then do; twks94=10; twks95=11;
  twks96=11; end;
if pubid=5178 then do; twks96=38; twks97=45; end;
if pubid=5497 then twks97=49;
if pubid=5658 then twks97=36;
if pubid=6324 then do; twks94=13; twks95=14;
  twks96=14; twks97=41; twks98=33; end;
if pubid=6949 then twks97=11;
if pubid=3222 then do; twks96=52; twks97=52; end;

endsas;

```

NUMBER OF WEEKS WORKED SINCE LAST INTERVIEW

Variables Created: CV_WKSWK_DLI

Programs Used

This program uses **bdate1.sas** and **emp_begin.sas** as input (see the first page of this appendix for details).

Created for each individual, this program counts the number of weeks the respondent worked at any employee-type job since the last interview. Respondents not working in a given year are given a default value of zero.

```

array job1wks (i) wk1_1-wk1_1044;
array job2wks (i) wk2_1-wk2_1044;
array job3wks (i) wk3_1-wk3_1044;
array job4wks (i) wk4_1-wk4_1044;
array job5wks (i) wk5_1-wk5_1044;
array job6wks (i) wk6_1-wk6_1044;
array job7wks (i) wk7_1-wk7_1044;
array job8wks (i) wk8_1-wk8_1044;
array job9wks (i) wk9_1-wk9_1044;
array alljobs (i) wks1-wks1044;
array starw (i) starw1-starw9;
array stopw (i) stopw1-stopw9;

do over alljobs; alljobs=0; end;

/* We update the startdate to the R1 interview date
when the stardate is before the interview date and the
stop date is after the interview date. */
alert=0;
do over starw;
  if starw<dliwk and starw>0 and stopw>dliwk then do;
    starw=dliwk; alert=1; end;
  end;

/* Jobs that begin and end before the Round1 interview
date will not be counted in this program. */
array switch (i) switch1-switch9;
do over switch; switch=0; end;
whoa=0;
do over starw;
  if starw<dliwk and stopw<dliwk and starw>0 and
    stopw>0 then do; switch=1; whoa=1; end; end;

do i=1 to 1044;
  if switch1=1 then do; job1wks=0; end;
  if switch2=1 then do; job2wks=0; end;
  if switch3=1 then do; job3wks=0; end;
  if switch4=1 then do; job4wks=0; end;
  if switch5=1 then do; job5wks=0; end;
  if switch6=1 then do; job6wks=0; end;
  if switch7=1 then do; job7wks=0; end;
  if switch8=1 then do; job8wks=0; end;
  if switch9=1 then do; job9wks=0; end;
end;

/** Overlay multiple jobs over JOB 1 work weeks **/
```

```

do over job1wks; alljobs=job1wks; end;

do over alljobs;
  if job2wks=1 then do;
    alljobs=job2wks; end;
  if job2wks=-3 and alljobs=0 then do;
    alljobs=job2wks; end;
end;
/* The program repeats the "do over all jobs" command
for jobs 3-9. The loops are not shown here; contact
NLS User Services for more information. */

/* Calculate cumulative weeks on all jobs since birth */
allwks=0;
if birthwk>0 then do;
  do i=birthwk to 1044;
    if alljobs=1 then do; allwks=allwks+1; end;
  end;
  do i=birthwk to 1044;
    if alljobs=-3 then do; allwks=-3; end;
  end;
  if starw1<0 and starw1>-4 then do; allwks=-3; end;
  if starw2<0 and starw2>-4 then do; allwks=-3; end;
  if starw3<0 and starw3>-4 then do; allwks=-3; end;
  if starw4<0 and starw4>-4 then do; allwks=-3; end;
  if starw5<0 and starw5>-4 then do; allwks=-3; end;
  if starw6<0 and starw6>-4 then do; allwks=-3; end;
  if starw7<0 and starw7>-4 then do; allwks=-3; end;
  if starw8<0 and starw8>-4 then do; allwks=-3; end;
  if starw9<0 and starw9>-4 then do; allwks=-3; end;
  if stopw1<0 and stopw1>-4 then do; allwks=-3; end;
  if stopw2<0 and stopw2>-4 then do; allwks=-3; end;
  if stopw3<0 and stopw3>-4 then do; allwks=-3; end;
  if stopw4<0 and stopw4>-4 then do; allwks=-3; end;
  if stopw5<0 and stopw5>-4 then do; allwks=-3; end;
  if stopw6<0 and stopw6>-4 then do; allwks=-3; end;
  if stopw7<0 and stopw7>-4 then do; allwks=-3; end;
  if stopw8<0 and stopw8>-4 then do; allwks=-3; end;
  if stopw9<0 and stopw9>-4 then do; allwks=-3; end;
end;

if e200=0 then do; allwks=-4; end;
if e200=-3 then do; allwks=-3; end;
if e200=-5 then do; allwks=-5; end;

endsas;
```

NUMBER OF WEEKS WORKED SINCE AGE 14

Variables Created: CV_WKSWK_EVER

Programs Used

This program uses **bdate1.sas** and **emp_begin.sas** as input (see the first page of this appendix for details).

For each respondent, this program creates a variable counting the number of weeks worked at any employee-type job since the age of 14. Respondents not working are given a default value of zero.

```

/* Section 1: Creates a variable for number of weeks
   worked at any job since age 14 using R2 data. */

array job1wks (i) wk1_1-wk1_1044;
array job2wks (i) wk2_1-wk2_1044;
array job3wks (i) wk3_1-wk3_1044;
array job4wks (i) wk4_1-wk4_1044;
array job5wks (i) wk5_1-wk5_1044;
array job6wks (i) wk6_1-wk6_1044;
array job7wks (i) wk7_1-wk7_1044;
array job8wks (i) wk8_1-wk8_1044;
array job9wks (i) wk9_1-wk9_1044;
array alljobs (i) wks1-wks1044;
array starw (i) starw1-starw9;
array stopw (i) stopw1-stopw9;

/** Overlay multiple jobs over JOB 1 work weeks **/ 

do over job1wks; alljobs=job1wks; end;

do over alljobs;
  if job2wks=1 then do;
    alljobs=job2wks; end;
  if job2wks=-3 and alljobs=0 then do;
    alljobs=job2wks; end;
end;

/* The program repeats the "do over all jobs" command
   for jobs 3-9. The loops are not shown here; contact
   NLS User Services for more information. */

/* Calculate cumulative wks on all jobs since age 14 */
wks14w=0;

if age14wk>0 then do;
  do i=age14wk to 1044;
    if alljobs=1 then do; wks14w=wks14w+1; end;
    if alljobs=-3 then do; wks14w=-3; end;
  end;
end;

/* Start date invalid */
do i=1 to 9;
if starw<0 and starw>-4 then do;
  if age14wk<53 and stopw>1 then do;
    wks14w=-3; end;

```

```

if age14wk<105 and stopw>52 then do;
  wks14w=-3; end;
if age14wk<157 and stopw>104 then do;
  wks14w=-3; end;
if age14wk<210 and stopw>156 then do;
  wks14w=-3; end;
if age14wk<262 and stopw>209 then do;
  wks14w=-3; end;
if age14wk<314 and stopw>261 then do;
  wks14w=-3; end;
if age14wk<366 and stopw>313 then do;
  wks14w=-3; end;
if age14wk<418 and stopw>365 then do;
  wks14w=-3; end;
if age14wk<471 and stopw>417 then do;
  wks14w=-3; end;
if age14wk<523 and stopw>470 then do;
  wks14w=-3; end;
if age14wk<575 and stopw>522 then do;
  wks14w=-3; end;
if age14wk<627 and stopw>574 then do;
  wks14w=-3; end;
if age14wk<679 and stopw>626 then do;
  wks14w=-3; end;
if age14wk<731 and stopw>678 then do;
  wks14w=-3; end;
if age14wk<784 stopw>730 then do;
  wks14w=-3; end;
if age14wk<836 and stopw>783 then do;
  wks14w=-3; end;
if age14wk<888 and stopw>835 then do;
  wks14w=-3; end;
if age14wk<940 and stopw>887 then do;
  wks14w=-3; end;
if age14wk<991 and stopw>939 then do;
  wks14w=-3; end;
if age14wk<1044 and stopw>991 then do;
  wks14w=-3; end;
end;

/* Stop date invalid */
if stopw<0 and stopw>-4 then do;
  if age14wk<53 and starw<53 then do;
    wks14w=-3; end;
  if age14wk<105 and starw<105 then do;
    wks14w=-3; end;

```

Appendix 2: Employment Variable Creation

```
if age14wk<157 and starw<157 then do;
  wks14w=-3; end;
if age14wk<210 and starw<210 then do;
  wks14w=-3; end;
if age14wk<262 and starw<262 then do;
  wks14w=-3; end;
if age14wk<314 and starw<314 then do;
  wks14w=-3; end;
if age14wk<366 and starw<366 then do;
  wks14w=-3; end;
if age14wk<418 and starw<418 then do;
  wks14w=-3; end;
if age14wk<471 and starw<471 then do;
  wks14w=-3; end;
if age14wk<523 and starw<523 then do;
  wks14w=-3; end;
if age14wk<575 and starw<575 then do;
  wks14w=-3; end;
if age14wk<627 and starw<627 then do;
  wks14w=-3; end;
if age14wk<679 and starw<679 then do;
  wks14w=-3; end;
if age14wk<731 and starw<731 then do;
  wks14w=-3; end;
if age14wk<784 and starw<784 then do;
  wks14w=-3; end;
if age14wk<836 and starw<836 then do;
  wks14w=-3; end;
if age14wk<888 and starw<888 then do;
  wks14w=-3; end;
if age14wk<940 and starw<940 then do;
  wks14w=-3; end;
if age14wk<991 and starw<991 then do;
  wks14w=-3; end;
if age14wk<1044 and starw<1044 then do;
  wks14w=-3; end;
end;
end;

/*** Include valid skips ***/
if e200=0 then do; wks14w=-4; end;
if e200=-3 then do; wks14w=-3; end;
if e200=-5 then do; wks14w=-5; end;

/* Section 2: Calculate the total number of weeks
   worked since age 14 by adding together the created
   variables from R1 and R2.

/* Initialize created variable for both rounds to zero.*/
allwks14=0;

/* If both the R1 and R2 variables are positive, we can
   simply add them together. If one is positive and one is
   not, then the negative value will be the total created
   variable for both rounds. If neither is positive, then the
   total created variable for both rounds will be zero. */

/* Both rounds positive*/
if r1wks14>0 then do; allwks14=r1wks14; end;
if wks14w>0 then do; allwks14=wks14w; end;
if r1wks14>0 and wks14w>0 then do;
  allwks14=r1wks14+wks14w;
end;

/* If one variable is negative.*/
if (r1wks14=-2 or r1wks14=-3) then
  allwks14=r1wks14;
if (wks14w=-2 or wks14w=-3 or wks14w=-5) then
  allwks14=wks14w;

/* The following hand edit cases result from
   respondents reporting jobs during the R2 interview that
   began and ended before the R1 interview date.*/
if pubid=32 then allwks14=187;
if pubid=5497 then allwks14=137;
if pubid=2398 then allwks14=95;
if pubid=5658 then allwks14=132;
if pubid=2615 then allwks14=78;
if pubid=6324 then allwks14=91;
if pubid=2698 then allwks14=174;
if pubid=6949 then allwks14=29;
if pubid=4921 then allwks14=39;
if pubid=3222 then allwks14=190;
if pubid=5178 then allwks14=85;
if pubid=3003 then allwks14=122;

endsas;
```

WEEKS WORKED AT EMPLOYEE JOB #X DURING 19XX

Variables Created: CV_WKSWK_JOB_YR.01.80 – CV_WKSWK_JOB_YR.01.99
CV_WKSWK_JOB_YR.02.80 – CV_WKSWK_JOB_YR.02.99 etc. through job #9

Programs Used

This program uses **emp_begin.sas** as input (see the first page of this appendix for details).

This program creates variables for each of the respondent's jobs counting the number of weeks worked in each calendar year. A variable is created for each potential job even if the respondent has worked no jobs in a given year with the default value set to zero (0). The most jobs held by any respondent as of round 2 was nine, so variables are created for nine jobs for each respondent.

| | |
|---|--|
| <pre> /* Section 1: Create the Round2 variable for number of weeks worked at a given job in a given year. */ array starw (i) starw1-starw9; array job1wks (i) wk1_1-wk1_1044; array job2wks (i) wk2_1-wk2_1044; array job3wks (i) wk3_1-wk3_1044; array job4wks (i) wk4_1-wk4_1044; array job5wks (i) wk5_1-wk5_1044; array job6wks (i) wk6_1-wk6_1044; array job7wks (i) wk7_1-wk7_1044; array job8wks (i) wk8_1-wk8_1044; array job9wks (i) wk9_1-wk9_1044; /** Calculate cumulative weeks on individual jobs for each year **/</pre> | <pre> if job9wks=-3 then do; wksw809=-3; end; end; /** The same three sets of commands applied to the 1980 weeks variables are used for each subsequent year. However, this code is not shown here due to space considerations. Researchers needing the complete code should contact NLS User Services. The variables and "do i" statements for each year are as follows:</pre> |
| <pre> /* 1980 */ wksw801=0; wksw802=0; wksw803=0; wksw804=0; wksw805=0; wksw806=0; wksw807=0; wksw808=0; wksw809=0;</pre> | <pre> 1981 wksw811 wksw819 do i=53 to 104 1982 wksw821 wksw829 do i=105 to 156 1983 wksw831 wksw839 do i=157 to 209 1984 wksw841 wksw849 do i=210 to 261 1985 wksw851 wksw859 do i=262 to 313 1986 wksw861 wksw869 do i=314 to 365 1987 wksw871 wksw879 do i=366 to 417 1988 wksw881 wksw889 do i=418 to 470 1989 wksw891 wksw899 do i=471 to 522 1990 wksw901 wksw909 do i=523 to 574 1991 wksw911 wksw919 do i=575 to 626 1992 wksw921 wksw929 do i=627 to 678 1993 wksw931 wksw939 do i=679 to 730 1994 wksw941 wksw949 do i=731 to 783 1995 wksw951 wksw959 do i=784 to 835 1996 wksw961 wksw969 do i=836 to 887 1997 wksw971 wksw979 do i=888 to 939 1998 wksw981 wksw989 do i=940 to 991 1999 wksw991 wksw999 do i=992 to 1044 **/ <pre> /** Insert valid skips **/ if e200=0 then do; wksw801=-4; /* and so on through wksw809=-4 */ /* and so on through wksw991 through wksw999 */ end;</pre> </pre> |
| <pre> do i=1 to 52; if job1wks=-3 then do; wksw801=-3; end; if job2wks=-3 then do; wksw802=-3; end; if job3wks=-3 then do; wksw803=-3; end; if job4wks=-3 then do; wksw804=-3; end; if job5wks=-3 then do; wksw805=-3; end; if job6wks=-3 then do; wksw806=-3; end; if job7wks=-3 then do; wksw807=-3; end; if job8wks=-3 then do; wksw808=-3; end; if job9wks=-3 then do; wksw809=-3; end; end;</pre> | <pre> if e200=-3 then do; wksw801=-3; /* and so on through wksw809=-3 */ /* and so on through wksw991 through wksw999 */ end;</pre> |
| <pre> do i=1 to 52; if job1wks=-3 then do; wksw801=-3; end; if job2wks=-3 then do; wksw802=-3; end; if job3wks=-3 then do; wksw803=-3; end; if job4wks=-3 then do; wksw804=-3; end; if job5wks=-3 then do; wksw805=-3; end; if job6wks=-3 then do; wksw806=-3; end; if job7wks=-3 then do; wksw807=-3; end; if job8wks=-3 then do; wksw808=-3; end; end;</pre> | <pre> if e200=-5 then do; wksw801=-5; /* and so on through wksw809=-5 */ /* and so on through wksw991 through wksw999 */ end;</pre> |

```

/* Start date invalid */
if starw1<0 and starw1>-4 then do;
  if stopw1>1 then do; wksw801=-3; end;
  if stopw1>52 then do; wksw811=-3; end;
  if stopw1>104 then do; wksw821=-3; end;
  if stopw1>156 then do; wksw831=-3; end;
  if stopw1>209 then do; wksw841=-3; end;
  if stopw1>261 then do; wksw851=-3; end;
  if stopw1>313 then do; wksw861=-3; end;
  if stopw1>365 then do; wksw871=-3; end;
  if stopw1>417 then do; wksw881=-3; end;
  if stopw1>470 then do; wksw891=-3; end;
  if stopw1>522 then do; wksw901=-3; end;
  if stopw1>574 then do; wksw911=-3; end;
  if stopw1>626 then do; wksw921=-3; end;
  if stopw1>678 then do; wksw931=-3; end;
  if stopw1>730 then do; wksw941=-3; end;
  if stopw1>783 then do; wksw951=-3; end;
  if stopw1>835 then do; wksw961=-3; end;
  if stopw1>887 then do; wksw971=-3; end;
  if stopw1>939 then do; wksw981=-3; end;
  if stopw1>991 then do; wksw991=-3; end;
end;

/* Stop date invalid */
if stopw1<0 and stopw1>-4 then do;
  if starw1<53 then do; wksw801=-3; end;
  if starw1<105 then do; wksw811=-3; end;
  if starw1<157 then do; wksw821=-3; end;
  if starw1<210 then do; wksw831=-3; end;
  if starw1<262 then do; wksw841=-3; end;
  if starw1<314 then do; wksw851=-3; end;
  if starw1<366 then do; wksw861=-3; end;
  if starw1<418 then do; wksw871=-3; end;
  if starw1<471 then do; wksw881=-3; end;
  if starw1<523 then do; wksw891=-3; end;
  if starw1<575 then do; wksw901=-3; end;
  if starw1<627 then do; wksw911=-3; end;
  if starw1<679 then do; wksw921=-3; end;
  if starw1<731 then do; wksw931=-3; end;
  if starw1<784 then do; wksw941=-3; end;
  if starw1<836 then do; wksw951=-3; end;
  if starw1<888 then do; wksw961=-3; end;
  if starw1<940 then do; wksw971=-3; end;
  if starw1<991 then do; wksw981=-3; end;
  if starw1<1044 then do; wksw991=-3; end;
end;

/* At this point the program loops through the same
code invalid start/stop date for each job (2-9), using the
same week numbers. Researchers who need to see the
complete code should contact NLS User Services. */

/* Section 2: Combine the Round 1 and Round 2
created variables for number of weeks worked at job x
during year. */

/* WKSJ945 is the Round1 created variable for 5th job
(i.e. 5th UID code) in 1994. WKS945 is the Round2
created variable of the same description. TWKS594 is
the variable that combines these two created variables
(of the same description). */

/* 7 employers for R1, 9 employers for R2 */
array r1uid (i) r1uid1-r1uid7;
array uid (i) uid1-uid9;

/* Arrange the jobs by year for each round */
array wksj80 (i) wks801-wks807; /*R1*/
array wksw80 (i) wksw801-wksw809; /*R2*/
/* and so on through wksj98 and wksw98 */
array wksw99 (i) wksw991-wksw999; /* R2 only */

/* Arrange the jobs by year for total created value */
array twks80 (i) twks801-twks809;
array twks81 (i) twks811-twks819;
/* and so on through: */
array twks99 (i) twks991-twks999;

/* Initialize total weeks count to zero */
do i=1 to 9;
  twks80=0; /* and so on through twks99=0 */
end;

/* Accounts for Round2 non-interview cases */
if e200=-5 then do;
  wksw801=-5; /* and so on through wksw809=-5 */
  /* and so on through wksw991 through wksw999 */
end;

/* Begin by updating the total created variable if the R2
variable is positive. We are only worried about R2 jobs
and R1 jobs that were worked in Round2. If a negative
value is given (don't know, refusal, or non-interview)
then total created variable will be that negative value. */

/* 1980 */
do i=1 to 9;
  if wksw80>0 then do; twks80=wksw80; end;
  if wksw80<0 then do; twks80=wksw80; end;
end;

/* At this point the program repeats the 1980 pattern
for each year through 1996 (wksw96 and twks96) */

/* 1997 */
/* Since the R1 interview occurred in 1997-98, need to
account for both R1 and R2 created variables. Begin
with negative conditions. */

do i=1 to 9;
  if wksw97<0 then do; twks97=wksw97; end;
  /* Update if Round2 created variable is positive */

```

```

if wksw97>0 then do; twks97=wksw97; end;
end;

/* When matching jobs from R1 to R2, we only need
to worry about jobs in R2 with a UID beginning with
"97" since only those jobs could have been worked in
both rounds. */

array match1 (i) match11-match19;
/* and so on through match7 */

/* Define match14 (for example) as a dummy variable
that equals one when the first job on the Round1 UID
roster and the fourth job on the Round2 roster have the
same UID. */
do i=1 to 9;
    match1=0; match2=0; match3=0; match4=0;
    match5=0; match6=0; match7=0;
end;

/* Determine if any UID from the first position in R1
matches with any UID in Round2 */
do over uid;
    if r1uid1>0 and uid>0 then do;
        if r1uid1=uid then do; match1=1; end;
        end;
    end;

/* UID from position 2, 3, etc. in R1 matches R2 */
do over uid;
    if r1uid2>0 and uid>0 then do; /* repeat for r1uid3-
riuid7*/
        if r1uid2=uid then do; /* repeat for r1uid3-riuid7*/
            match2=1; /* repeat for match3-match7*/
            end;
        end;
    end;

/* Now reassign total created variable if there is a job
was worked in both rounds. Using the match variable, a
job worked in both rounds will update the total created
variable. Need to add positivity condition on the weeks
worked values since some of the values equal -3. */

if dli_y=1997 then do;
    do over match1;
        if match1=1 then do;
            twks80=wks801;
            twks81=wks811;
            /* and so on through twks96 and wks961 */
        end;
    end;

```

```

do over match1;
if match1=1 and wksw97 ge 0 and wks971 ge 0
    then do; twks97=wks971+wksw97; end;
if match1=1 and (wksw97<0 or wks971<0) then
    do; twks97=-3; end;
end;

do over match2; /*repeat for match3-match7*/
/* At this point the program loops through the
above code using match2 and wksXX2, match3 and
wksXX3, and so on through match7 and wksXX7.
Users who need the complete code should contact
NLS User Services. */
end;

/* 1998 */
do i=1 to 9;
    if wksw98<0 then do; twks98=wksw98; end;
    /* Update if Round2 created variable is positive */
    if wksw98>0 then do; twks98=wksw98; end;
end;

if dli_y=1998 then do;
    do over match1;
        if match1=1 then do;
            twks80=wks801;
            twks81=wks811;
            /* and so on through twks97 and wks971 */
        end;
    end;

do over match1;
if match1=1 and wksw98=>0 and wks981=>0 then
    do; twks98=wks981+wksw98; end;
if match1=1 and (wksw98<0 or wks981<0) then
    do; twks98=-3; end;
end;

do over match2; /*repeat for match3-match7*/
/* At this point the program loops through the
above code using match2 and wksXX2, match3 and
wksXX3, and so on through match7 and wksXX7.
Users who need the complete code should contact
NLS User Services. */
end;

/* 1999: can only use Round2 created variables */
do i=1 to 9;
    if wksw99<0 then do; twks99=wksw99; end;
    if wksw99>0 then do; twks99=wksw99; end;
end;

```

TOTAL TENURE AT JOB #X AS OF THE SURVEY DATE

Variables Created: CV_WKSWK_JOB_DLI.01 – CV_WKSWK_JOB_DLI.09

Programs Used

This program uses **emp_begin.sas** as input (see the first page of this appendix for details).

This program creates a variable for each job calculating the total length of job tenure in weeks (excluding within-job gaps) since the respondent's 14th birthday. A variable is created for each potential job even if the respondent has no data for that job, with the default value set to zero (0). The most jobs held by any respondent as of round 2 was nine, so variables are created for nine jobs for each respondent.

| | |
|--|---|
| <pre>/* Section 1: Create a variable for total tenure at each job between the R1 and R2 interview dates. */ array starw (i) starw1-starw9; array job1wks (i) wk1_1-wk1_1044; array job2wks (i) wk2_1-wk2_1044; array job3wks (i) wk3_1-wk3_1044; array job4wks (i) wk4_1-wk4_1044; array job5wks (i) wk5_1-wk5_1044; array job6wks (i) wk6_1-wk6_1044; array job7wks (i) wk7_1-wk7_1044; array job8wks (i) wk8_1-wk8_1044; array job9wks (i) wk9_1-wk9_1044; /** Calculate cumulative weeks on individual jobs for all years */ tenure1=0; tenure2=0; tenure3=0; tenure4=0; tenure5=0; tenure6=0; tenure7=0; tenure8=0; tenure9=0; do i=1 to 1044; if job1wks=1 then do; tenure1=tenure1+1; end; if job2wks=1 then do; tenure2=tenure2+1; end; if job3wks=1 then do; tenure3=tenure3+1; end; if job4wks=1 then do; tenure4=tenure4+1; end; if job5wks=1 then do; tenure5=tenure5+1; end; if job6wks=1 then do; tenure6=tenure6+1; end; if job7wks=1 then do; tenure7=tenure7+1; end; if job8wks=1 then do; tenure8=tenure8+1; end; if job9wks=1 then do; tenure9=tenure9+1; end; end; flag=0; /* Invalid information for a certain week leads to invalid created variables.*/ do i=1 to 1044; if job1wks=-3 then do; tenure1=-3; flag=1; end; if job2wks=-3 then do; tenure2=-3; flag=2; end; if job3wks=-3 then do; tenure3=-3; flag=3; end; if job4wks=-3 then do; tenure4=-3; flag=4; end; if job5wks=-3 then do; tenure5=-3; flag=5; end; if job6wks=-3 then do; tenure6=-3; flag=6; end; if job7wks=-3 then do; tenure7=-3; flag=7; end;</pre> | <pre>if job8wks=-3 then do; tenure8=-3; flag=8; end; if job9wks=-3 then do; tenure9=-3; flag=9; end; end; do i=1 to 1044; /* Start date invalid */ if starw1<0 and starw1>-4 then do; tenure1=-3; end; if starw2<0 and starw2>-4 then do; tenure2=-3; end; if starw3<0 and starw3>-4 then do; tenure3=-3; end; if starw4<0 and starw4>-4 then do; tenure4=-3; end; if starw5<0 and starw5>-4 then do; tenure5=-3; end; if starw6<0 and starw6>-4 then do; tenure6=-3; end; if starw7<0 and starw7>-4 then do; tenure7=-3; end; if starw8<0 and starw8>-4 then do; tenure8=-3; end; if starw9<0 and starw9>-4 then do; tenure9=-3; end; /* Stop date invalid */ if stopw1<0 and stopw1>-4 then do; tenure1=-3; end; if stopw2<0 and stopw2>-4 then do; tenure2=-3; end; if stopw3<0 and stopw3>-4 then do; tenure3=-3; end; if stopw4<0 and stopw4>-4 then do; tenure4=-3; end; if stopw5<0 and stopw5>-4 then do; tenure5=-3; end; if stopw6<0 and stopw6>-4 then do; tenure6=-3; end; if stopw7<0 and stopw7>-4 then do; tenure7=-3; end; if stopw8<0 and stopw8>-4 then do; tenure8=-3; end; if stopw9<0 and stopw9>-4 then do; tenure9=-3; end; end; if e200=0 then do; tenure1=-4; tenure2=-4; tenure3=-4; tenure4=-4; tenure5=-4; tenure6=-4; tenure7=-4; tenure8=-4; tenure9=-4; end; if e200=-3 then do; tenure1=-3; tenure2=-3; tenure3=-3; tenure4=-3; tenure5=-3; tenure6=-3; tenure7=-3; tenure8=-3; tenure9=-3; end; if e200=-5 then do; tenure1=-5; tenure2=-5; tenure3=-5; tenure4=-5; tenure5=-5; tenure6=-5; tenure7=-5; tenure8=-5; tenure9=-5; end;</pre> |
|--|---|

```

/* Section 2: Calculate the total job tenure by
matching up the UID's from Round1 and Round2. */

/* When matching jobs from Round1 to Round2, we
only need to examine jobs in Round2 with a UID
beginning with "97" since only those jobs that could
have been worked in both rounds. For example,
match14 is a dummy variable that equals one when the
first job on the Round1 UID roster and the fourth job
on the Round2 roster have the same UID */

array r1uid (i) r1uid1-r1uid7;
array uid (i) uid1-uid9;
array match1 (i) match11-match19;
array match2 (i) match21-match29;
array match3 (i) match31-match39;
array match4 (i) match41-match49;
array match5 (i) match51-match59;
array match6 (i) match61-match69;
array match7 (i) match71-match79;

do i=1 to 9;
    match1=0; match2=0; match3=0; match4=0;
    match5=0; match6=0; match7=0;
end;

/* Determine if any UID from the first position in
Round1 matches with any UID in Round2 */
do over uid;
    if r1uid1>0 and uid>0 then do;
        if r1uid1=uid then do; match1=1; end;
        end;
    end;

/* UID from position 2, 3, etc. in R1 matches R2 */

```

```

do over uid;
    if r1uid2>0 and uid>0 then do; /* repeat for r1uid3-
r1uid7 */
        if r1uid2=uid then do; /*repeat for r1uid3-r1uid7*/
            match2=1; /* repeat for match3-match7 */
            end;
        end;
    end;

/* Define totten as the total tenure of a job in weeks.
This includes jobs worked in both Round1 and
Round2 or jobs worked in only Round2. We are
considering jobs worked only in Round1 in this
program. */
array totten (i) totten1-totten9;
array tenure (i) tenure1-tenure9;
array r1ten (i) r1ten1-r1ten7;

do over totten; totten=0; end;

do over tenure; totten=tenure; end;

/* Now reassign total tenure if there is a job was worked
in both rounds. Using the match variable, a job worked
in both rounds will update the total tenure variable.
Need to add positivity condition on the tenure values
since some of the values equal -3. */

do over match1; /* repeat for match2-match7 */
    if match1=1 and tenure=>0 and r1ten1=>0 then do;
        totten=r1ten1+tenure; end;
    if match1=1 and -4<r1ten1<0 then do;
        totten=r1ten1; end;
    end;

endsas;

```

TOTAL HOURS WORKED IN 19XX

Variables Created: CV_HOURS_WK_YR.80 – CV_HOURS_WK_YR.99

Programs Used

This program uses **bdate1.sas** and **emp_begin.sas** as input (see the first page of this appendix for details).

Variables Used

| Name in Program | Question Name on CD | Name in Program | Question Name on CD |
|-------------------|-------------------------|-------------------|---------------------|
| E200 | YEMP-200 | E38000F1-E38000F6 | YEMP-38000F.01 |
| E239011-E239019 | YEMP-23901.01-.09 | E381031, E381032 | YEMP-38103.01, .02 |
| E245011-E245016 | YEMP-24501.01-.06 | E381051, E381052 | YEMP-38105.01, .02 |
| E344021-E344027 | YEMP-34402.01-.07 | E599011-E599017 | YEMP-59901.01-.07 |
| E344031-E344035 | YEMP-34403.01-.05 | E880001-E880006 | YEMP-88000.01-.06 |
| E344281-E344285 | YEMP-34428.01-.05 | E885011-E885015 | YEMP-88501.01-.05 |
| E2261012-E2261032 | YEMP-22610.01.02-.03.02 | E984021-E984027 | YEMP-98402.01-.07 |
| E379021-E379028 | YEMP-37902.01-.08 | E984031-E984035 | YEMP-98403.01-.05 |
| E379041-E379047 | YEMP-37904.01-.07 | E984291-E984295 | YEMP-98429.01-.05 |
| E380001-E380009 | YEMP-38000.01-.09 | | |

This program calculates the number of hours worked by the respondent at all **employee-type** jobs in each calendar year. A variable is created for each respondent even if the respondent has worked no jobs in a given year with the default value set to zero (0). Note that when both "starting hours" and "current hours" are reported, the latter are used to construct these measures.

```

/* Section 1: Creates a variable for total annual hours
   worked at employee-type job during a given year using
   Round 2 data. */

/* Organize hours for each job */
array starw (k) starw1-starw9;
array stopw (k) stopw1-stopw9;

/* shrs1 - starting hours    ehrs1 - ending hours */
array shrs (k) shrs1-shrs9;
array ehrs (k) ehrs1-ehrs9;
array hours (k) hours1-hours9;
array hrck (k) hrck1-hrck9;

/* To the right of the array statements are the true
   ranges of the pulled variables. Values of variables
   outside the ranges will be represented by dots. */
array e23901 (k) e239011-e239019;      /* 1-9 */
array e24501 (k) e245011-e245019;      /* 1-6 */
array e34402 (k) e344021-e344029;      /* 1-7 */
array e34403 (k) e344031-e344039;      /* 1,2,3,5 */
array e34428 (k) e344281-e344289;      /* 1,2,3,5 */
array e37904 (k) e379041-e379049;      /* 1-5,7 */
array e38000f (k) e38000f1-e38000f9;    /* 1-6 */
array e38103 (k) e381031-e381039;      /* 1,2 */
array e38105 (k) e381051-e381059;      /* 1,2 */
array e59901 (k) e599011-e599019;      /* 1-7 */
array e88000 (k) e880001-e880009;      /* 1-6 */
array e88501 (k) e885011-e885019;      /* 1-5 */
array e98402 (k) e984021-e984209;      /* 1-7 */
array e98403 (k) e984031-e984039;      /* 1,2,3,5 */

array e98429 (k) e984291-e984299;      /* 1,2,3,5 */
array e100256 (k) e1002561-e1002569;    /* 1,2,5 */

/* Define the number of hours per week worked at the
   start date and the end date of a job. Question E59901
   decides whether the job was listed on the Round 1
   roster, for jobs under 13 weeks. Question E88000 also
   decides whether the job was listed on the Round 1
   roster, but only for jobs that last longer than 13 weeks.
   The code below defines the starting hours and ending
   hours per week at those jobs, including overtime. */

do k=1 to 9;
/* start hours jobs< 13 weeks */
  if e23901>-4 then do; shrs=e23901; end;
  if e23901>-4 and e24501>-4 then do;
    shrs=e23901+e24501; end;
  if e34402>-4 then do; shrs=e34402; end;
  if e34403>-4 then do; shrs=e34403; end;
  if e34403>-4 and e34428>-4 then do;
    shrs=e34403+e34428; end;
/* end hours jobs< 13 weeks */
  if e37904>-4 then do; ehrs=e37904; end;
/* end hours for jobs > 13 weeks */
  if e38000f>-4 then do; ehrs=e38000f; end;
  if e38103>-4 then do; ehrs=e38103; end;
  if e38103>-4 and e38105>-4 then do;
    ehrs=e38103+e38105; end;
/* jobs from dli that are less than 13 weeks long */
  if e59901>-4 then do; shrs=e59901; end;
/* jobs from dli that are longer than 13 weeks */

```

```

if e88000>-4 then do; shrs=e88000; end;
/* E88501 is an overtime question. */
if e88501>-4 and e88000>-4 then do;
    shrs=e88501+e88000; end;
if e98402>-4 then do; shrs=e98402; end;
if e98429>-4 then do; shrs=e98429; end;
if e98429>-4 and e98403>-4 then do;
    shrs=e98402+e98429; end;
if e100256>-4 then do; shrs=e100256; end;
end;

/* The code below decides which hours per week total
(starting or ending) will be used in determining hours
worked per year. We prefer using ending hours as the
measure, hours per week starts with a default number,
is updated if starting hours are available, and is updated
once more if ending hours are available. */

* Set default hours to zero;
do over hours; hours=0; hrck=0; end;

* Take starting hours if reported (eliminates -4's);
do over shrs;
    if shrs=>0 then do; hours=shrs; end;
/* hrck marks when invalid answers are given to the
hrs/wk questions */
    if -4<shrs<0 then do; hrck=shrs; end;
end;

* Write over if end hours reported;
do over ehrs;
    if ehrs=>0 then do; hours=ehrs; end;
    if -4<ehrs<0 then do; hrck=ehrs; end;
end;

array job1wks (i) wk1_1-wk1_1044;
array job2wks (i) wk2_1-wk2_1044;
array job3wks (i) wk3_1-wk3_1044;
array job4wks (i) wk4_1-wk4_1044;
array job5wks (i) wk5_1-wk5_1044;
array job6wks (i) wk6_1-wk6_1044;
array job7wks (i) wk7_1-wk7_1044;
array job8wks (i) wk8_1-wk8_1044;
array job9wks (i) wk9_1-wk9_1044;

/** Calculate cumulative weeks on individual jobs for
each year **/ 

/* 1980 */ /* beginning here code loops for each year */
wks801=0; wks802=0; wks803=0; wks804=0;
wks805=0; wks806=0; wks807=0; wks808=0;
wks809=0;
ah801=0; ah802=0; ah803=0; ah804=0; ah805=0;
ah806=0; ah807=0; ah808=0; ah809=0;

/* Update hours counter when respondent reports
employment in weeks 1 to 52. */

do i=1 to 52;
    if job1wks=1 then do; wks801=wks801+1; end;
    if job2wks=1 then do; wks802=wks802+1; end;
    if job3wks=1 then do; wks803=wks803+1; end;
    if job4wks=1 then do; wks804=wks804+1; end;
    if job5wks=1 then do; wks805=wks805+1; end;
    if job6wks=1 then do; wks806=wks806+1; end;
    if job7wks=1 then do; wks807=wks807+1; end;
    if job8wks=1 then do; wks808=wks808+1; end;
    if job9wks=1 then do; wks809=wks809+1; end;
end;

do i=1 to 52;
    if job1wks ne -3 and job2wks ne -3 and job3wks ne -
3 and job4wks ne -3 and job5wks ne -3 and
job6wks ne -3 and job7wks ne -3 and job8wks
ne -3 and job9wks ne -3 then do;
    if hours1>0 then ah801=hours1*wks801;
    if hours2>0 then ah802=hours2*wks802;
    if hours3>0 then ah803=hours3*wks803;
    if hours4>0 then ah804=hours4*wks804;
    if hours5>0 then ah805=hours5*wks805;
    if hours6>0 then ah806=hours6*wks806;
    if hours7>0 then ah807=hours7*wks807;
    if hours8>0 then ah808=hours8*wks808;
    if hours9>0 then ah809=hours9*wks809;
end;
if job1wks=-3 or job2wks=-3 or job3wks=-3 or
job4wks=-3 or job5wks=-3 or job6wks=-3 or
job7wks=-3 or job8wks=-3 or job9wks=-3 then
do;
    ah801=-3;    ah802=-3;    ah803=-3;
    ah804=-3;    ah805=-3;    ah806=-3;
    ah807=-3;    ah808=-3;    ah809=-3;
    goto exit1;
end;
exit1:
if ah801=>0 and ah802=>0 and ah803=>0 and
ah804=>0 and ah805=>0 and ah806=>0 and
ah807=>0 and ah808=>0 and ah809=>0 then
do;
    tohrs80=ah801+ah802+ah803+ah804+ah805
        +ah806+ah807+ah808+ah809;
end;

if ah801=-3 or ah802=-3 or ah803=-3 or ah804=-3 or
ah805=-3 or ah806=-3 or ah807=-3 or ah808=-3
or ah809=-3 then do;
    tohrs80=-3;
end;

if wks801>0 and hrck1<0 then tohrs80=hrck1;
/* and so on through wks809>0 and hrck9<0 */

/* At this point the program loops through the above
code for each year through 1999. This code is not

```

shown here due to space constraints. Users who need the complete code should contact NLS User Services. */

```
/* Fill in -3's for cases where start/stop date is
unknown/refused. */


```

```
do k=1 to 9;
/* start date invalid */
if starw<0 and starw>-4 then do;
  if stopw>1 then do; tothrs80=-3; end;
  if stopw>52 then do; tothrs81=-3; end;
  if stopw>104 then do; tothrs82=-3; end;
  if stopw>156 then do; tothrs83=-3; end;
  if stopw>209 then do; tothrs84=-3; end;
  if stopw>261 then do; tothrs85=-3; end;
  if stopw>313 then do; tothrs86=-3; end;
  if stopw>365 then do; tothrs87=-3; end;
  if stopw>417 then do; tothrs88=-3; end;
  if stopw>470 then do; tothrs89=-3; end;
  if stopw>522 then do; tothrs90=-3; end;
  if stopw>574 then do; tothrs91=-3; end;
  if stopw>626 then do; tothrs92=-3; end;
  if stopw>678 then do; tothrs93=-3; end;
  if stopw>730 then do; tothrs94=-3; end;
  if stopw>783 then do; tothrs95=-3; end;
  if stopw>835 then do; tothrs96=-3; end;
  if stopw>887 then do; tothrs97=-3; end;
  if stopw>939 then do; tothrs98=-3; end;
  if stopw>991 then do; tothrs99=-3; end;
end;
```

```
/* stop date invalid */
if stopw<0 and stopw>-4 then do;
  if starw<53 then do; tothrs80=-3; end;
  if starw<105 then do; tothrs81=-3; end;
  if starw<157 then do; tothrs82=-3; end;
  if starw<210 then do; tothrs83=-3; end;
  if starw<262 then do; tothrs84=-3; end;
  if starw<314 then do; tothrs85=-3; end;
  if starw<366 then do; tothrs86=-3; end;
  if starw<418 then do; tothrs87=-3; end;
  if starw<471 then do; tothrs88=-3; end;
  if starw<523 then do; tothrs89=-3; end;
  if starw<575 then do; tothrs90=-3; end;
  if starw<627 then do; tothrs91=-3; end;
  if starw<679 then do; tothrs92=-3; end;
  if starw<731 then do; tothrs93=-3; end;
  if starw<784 then do; tothrs94=-3; end;
  if starw<836 then do; tothrs95=-3; end;
  if starw<888 then do; tothrs96=-3; end;
  if starw<940 then do; tothrs97=-3; end;
  if starw<991 then do; tothrs98=-3; end;
  if starw<1044 then do; tothrs99=-3; end;
end;
end;
```

```
if e200=0 then do;
```

```
tothrs80=-4; tothrs81=-4; tothrs82=-4; tothrs83=-4;
tothrs84=-4; tothrs85=-4; tothrs86=-4; tothrs87=-4;
tothrs88=-4; tothrs89=-4; tothrs90=-4; tothrs91=-4;
tothrs92=-4; tothrs93=-4; tothrs94=-4; tothrs95=-4;
tothrs96=-4; tothrs97=-4; tothrs98=-4; tothrs99=-4;
end;
```

```
if e200=-3 then do;
  tothrs80=-3; tothrs81=-3; tothrs82=-3; tothrs83=-3;
  tothrs84=-3; tothrs85=-3; tothrs86=-3; tothrs87=-3;
  tothrs88=-3; tothrs89=-3; tothrs90=-3; tothrs91=-3;
  tothrs92=-3; tothrs93=-3; tothrs94=-3; tothrs95=-3;
  tothrs96=-3; tothrs97=-3; tothrs98=-3; tothrs99=-3;
end;
```

```
if e200=-5 then do;
  tothrs80=-5; tothrs81=-5; tothrs82=-5; tothrs83=-5;
  tothrs84=-5; tothrs85=-5; tothrs86=-5; tothrs87=-5;
  tothrs88=-5; tothrs89=-5; tothrs90=-5; tothrs91=-5;
  tothrs92=-5; tothrs93=-5; tothrs94=-5; tothrs95=-5;
  tothrs96=-5; tothrs97=-5; tothrs98=-5; tothrs99=-5;
end;
```

```
/* Section 2: Sum the created variables for total hours
worked in any job during a given year for R1 and R2 */


```

```
array tothrs (20) tothrs80-tothrs99; /* R2 variable */
array r1hrs (20) r1hrs80-r1hrs99; /* R1 variable */
array allhrs (20) allhrs80-allhrs99; /* Total from both
rounds created variable */


```

```
/* Initialize the total created variable to be zero. */
do i=1 to 20; allhrs(i)=0; end;
```

```
/* Begin by splitting up periods where R1 and R2
exclusively collect hours worked information. Define
the R1 interview year as the split. Any hours worked
data collected before the R1 interview year should be
independent of information collected in R2. */


```

```
do i=1 to 20;
  if r1hrs(i)>0 then do; allhrs(i)=r1hrs(i); end;
  if tothrs(i)>0 then do; allhrs(i)=tothrs(i); end;
  if r1hrs(i)>0 and tothrs(i)>0 then do;
    allhrs(i)=r1hrs(i)+tothrs(i); end;
end;
```

```
/* Define negative values for the total created var. */
do i=1 to 20;
  if -4<r1hrs(i)<0 then do; allhrs(i)=r1hrs(i); end;
  if tothrs(i)=-1 or tothrs(i)=-2 or tothrs(i)=-3 or
    tothrs(i)=-5 then do;
    allhrs(i)=tothrs(i); end;
end;
endsas;
```

TOTAL HOURS WORKED SINCE AGE 14

Variables Created: CV_HOURS_WK_EVER

Programs Used

This program uses **bdate1.sas** and **emp_begin.sas** as input (see the first page of this appendix for details).

Variables Used

This program uses the same variables as the previous program, "Total Hours Worked in 19xx."

This program calculates the number of hours worked by the respondent at all **employee-type** jobs since turning 14. A variable is created for each respondent even if the respondent has worked no jobs in a given year with the default value set to zero (0). Note that when both "starting hours" and "current hours" are reported, the latter are used to construct these measures

```
/* Section 1: Create a variable for total hours worked at
employee-type jobs since age 14 using round 2 data. */

/* Organize hours for each job */
array starw (i) starw1-starw9;
array stopw (i) stopw1-stopw9;

/* shrs1 - starting hours    ehrs1 - ending hours */
array shrs (i) shrs1-shrs9;
array ehrs (i) ehrs1-ehrs9;
array hours (i) hours1-hours9;
array hrck (i) hrck1-hrck9;

/* To the right of the array statements are the true
ranges of the pulled variables. Values of variables
outside the ranges will be represented by dots. */
array e23901 (i) e239011-e239019; /* 1-9 */
array e24501 (i) e245011-e245019; /* 1-6 */
array e34402 (i) e344021-e344029; /* 1-7 */
array e34403 (i) e344031-e344039; /* 1,2,3,5 */
array e34428 (i) e344281-e344289; /* 1,2,3,5 */
array e37904 (i) e379041-e379049; /* 1-5,7 */
array e38000f (i) e38000f1-e38000f9; /* 1-6 */
array e38103 (i) e381031-e381039; /* 1,2 */
array e38105 (i) e381051-e381059; /* 1,2 */
array e59901 (i) e599011-e599019; /* 1-7 */
array e88000 (i) e880001-e880009; /* 1-6 */
array e88501 (i) e885011-e885019; /* 1-5 */
array e98402 (i) e984021-e984209; /* 1-7 */
array e98403 (i) e984031-e984039; /* 1,2,3,5 */
array e98429 (i) e984291-e984299; /* 1,2,3,5 */
array e100256 (i) e1002561-e1002569; /* 1-3 */

/* Define the number of hours per week worked at the
start date and the end date of a job. Question E59901
decides whether the job was listed on the Round 1
roster, for jobs under 13 weeks. Question E88000 also
decides whether the job was listed on the Round 1
roster, but only for jobs that last longer than 13 weeks.
```

The code below defines the starting hours and ending hours per week at those jobs, including overtime. */

```
do i=1 to 9;
/* starting hours for jobs shorter than 13 weeks */
  if e23901>-4 then do; shrs=e23901; end;
  if e23901>-4 and e24501>-4 then do;
    shrs=e23901+e24501; end;
  if e34402>-4 then do; shrs=e34402; end;
  if e34403>-4 then do; shrs=e34403; end;
  if e34403>-4 and e34428>-4 then do;
    shrs=e34403+e34428; end;
/* ending hours for jobs less than 13 weeks long */
  if e37904>-4 then do; ehrs=e37904; end;
/* ending hours for jobs longer than 13 weeks */
  if e38000f>-4 then do; ehrs=e38000f; end;
  if e38103>-4 then do; ehrs=e38103; end;
  if e38103>-4 and e38105>-4 then do;
    ehrs=e38103+e38105; end;
/* jobs from dli that are less than 13 weeks long */
  if e59901>-4 then do; shrs=e59901; end;
/* jobs from dli that are longer than 13 weeks */
  if e88000>-4 then do; shrs=e88000; end;
/* E88501 is an overtime question. */
  if e88501>-4 and e88000>-4 then do;
    shrs=e88501+e88000; end;
  if e98402>-4 then do; shrs=e98402; end;
  if e98429>-4 then do; shrs=e98429; end;
  if e98429>-4 and e98403>-4 then do;
    shrs=e98402+e98429; end;
  if e100256>-4 then do; shrs=e100256; end;
end;

/* The code below decides which hours per week total
(starting or ending) will be used in determining hours
worked per year. We prefer using ending hours as the
measure, hours per week starts with a default number (-
16), is updating if starting hours are available, and is
updated once more if ending hours are available. */
```

Appendix 2: Employment Variable Creation

```

* Set default hours to zero (-16);
do over hours;
  hours=0; hrck=0;
end;

/* Take starting hours if reported (eliminates -4's); hrck
marks when invalid answers are given to the hrs/wk
questions */
do over shrs;
  if shrs=>0 then do; hours=shrs; end;
  if -4<shrs<0 then do; hrck=shrs; end;
end;

/* Write over if end hours reported */
do over ehrs;
  if ehrs=>0 then do; hours=ehrs; end;
  if -4<ehrs<0 then do; hrck=ehrs; end;
end;

array job1wks (i) wk1_1-wk1_1044;
array job2wks (i) wk2_1-wk2_1044;
array job3wks (i) wk3_1-wk3_1044;
array job4wks (i) wk4_1-wk4_1044;
array job5wks (i) wk5_1-wk5_1044;
array job6wks (i) wk6_1-wk6_1044;
array job7wks (i) wk7_1-wk7_1044;
array job8wks (i) wk8_1-wk8_1044;
array job9wks (i) wk9_1-wk9_1044;

/** Calculate cumulative weeks on individual jobs for
each year **/


wks1=0; wks2=0; wks3=0; wks4=0; wks5=0; wks6=0;
  wks7=0; wks8=0; wks9=0;
thrs1=0; thrs2=0; thrs3=0; thrs4=0; thrs5=0; thrs6=0;
  thrs7=0; thrs8=0; thrs9=0;

if 0<age14wk<1044 then do;
  do i=age14wk to 1044;
    if job1wks=1 then do; wks1=wks1+1; end;
    if job2wks=1 then do; wks2=wks2+1; end;
    if job3wks=1 then do; wks3=wks3+1; end;
    if job4wks=1 then do; wks4=wks4+1; end;
    if job5wks=1 then do; wks5=wks5+1; end;
    if job6wks=1 then do; wks6=wks6+1; end;
    if job7wks=1 then do; wks7=wks7+1; end;
    if job8wks=1 then do; wks8=wks8+1; end;
    if job9wks=1 then do; wks9=wks9+1; end;
  end;
end;

if 0<age14wk<1044 then do;
  do i=age14wk to 1044;
    if job1wks ne -3 and job2wks ne -3 and job3wks ne -
      3 and job4wks ne -3 and job5wks ne -3 and
      job6wks ne -3 and job7wks ne -3 and
      job8wks ne -3 and job9wks ne -3 then do;

```

```

thrs1=hours1*wks1;      thrs2=hours2*wks2;
thrs3=hours3*wks3;      thrs4=hours4*wks4;
thrs5=hours5*wks5;      thrs6=hours6*wks6;
thrs7=hours7*wks7;      thrs8=hours8*wks8;
thrs9=hours9*wks9;
end;
if job1wks=-3 or job2wks=-3 or job3wks=-3 or
  job4wks=-3 or job5wks=-3 or job6wks=-3 or
  job7wks=-3 or job8wks=-3 or
  job9wks=-3 then do;
  thrs1=-3; r2hrs14=-3;
  goto exit1;
end;
end; exit1:

if thrs1=>0 and thrs2=>0 and thrs3=>0 and thrs4=>0 and
  and thrs5=>0 and thrs6=>0 and thrs7=>0 and
  thrs8=>0 and
thrs9=>0 then do;
  r2hrs14=thrs1+thrs2+thrs3+thrs4+thrs5+thrs6+
    thrs7+thrs8+thrs9;
end;

if thrs1=-3 or thrs2=-3 or thrs3=-3 or thrs4=-3 or
  thrs5=-3 or thrs6=-3 or thrs7=-3 or thrs8=-3
  or thrs9=-3 then do;
  r2hrs14=-3;
end;

/* hrck used when invalid hrs./wk answers are given */
if wks1>0 and hrck1<0 and r2hrs14>0 then
r2hrs14=hrck1;
/* and so on through wks9 and hrck9 */

/* Remove invalid start/stop dates */
do i=1 to 9;
if starw<0 and starw>-4 then do;
  if age14wk<53 and stopw>1 then r2hrs14=-3;
  if age14wk<105 and stopw>52 then r2hrs14=-3;
  if age14wk<157 and stopw>104 then r2hrs14=-3;
  if age14wk<210 and stopw>156 then r2hrs14=-3;
  if age14wk<262 and stopw>209 then r2hrs14=-3;
  if age14wk<314 and stopw>261 then r2hrs14=-3;
  if age14wk<366 and stopw>313 then r2hrs14=-3;
  if age14wk<418 and stopw>365 then r2hrs14=-3;
  if age14wk<471 and stopw>417 then r2hrs14=-3;
  if age14wk<523 and stopw>470 then r2hrs14=-3;
  if age14wk<575 and stopw>522 then r2hrs14=-3;
  if age14wk<627 and stopw>574 then r2hrs14=-3;
  if age14wk<679 and stopw>626 then r2hrs14=-3;
  if age14wk<731 and stopw>678 then r2hrs14=-3;
  if age14wk<784 and stopw>730 then r2hrs14=-3;
  if age14wk<836 and stopw>783 then r2hrs14=-3;
  if age14wk<888 and stopw>835 then r2hrs14=-3;
  if age14wk<940 and stopw>887 then r2hrs14=-3;
  if age14wk<991 and stopw>939 then r2hrs14=-3;

```

```

if age14wk<1044 and stopw>991 then r2hrs14=-3;
end;

if stopw<0 and stopw>-4 then do;
  if age14wk<53 and starw<53 then r2hrs14=-3;
  if age14wk<105 and starw<105 then r2hrs14=-3;
  if age14wk<157 and starw<157 then r2hrs14=-3;
  if age14wk<210 and starw<210 then r2hrs14=-3;
  if age14wk<262 and starw<262 then r2hrs14=-3;
  if age14wk<314 and starw<314 then r2hrs14=-3;
  if age14wk<366 and starw<366 then r2hrs14=-3;
  if age14wk<418 and starw<418 then r2hrs14=-3;
  if age14wk<471 and starw<471 then r2hrs14=-3;
  if age14wk<523 and starw<523 then r2hrs14=-3;
  if age14wk<575 and starw<575 then r2hrs14=-3;
  if age14wk<627 and starw<627 then r2hrs14=-3;
  if age14wk<679 and starw<679 then r2hrs14=-3;
  if age14wk<731 and starw<731 then r2hrs14=-3;
  if age14wk<784 and starw<784 then r2hrs14=-3;
  if age14wk<836 and starw<836 then r2hrs14=-3;
  if age14wk<888 and starw<888 then r2hrs14=-3;
  if age14wk<940 and starw<940 then r2hrs14=-3;
  if age14wk<991 and starw<991 then r2hrs14=-3;
  if age14wk<1044 and starw<1044 then r2hrs14=-3;
end;
end;

if e200=-5 then r2hrs14=-5;
if e200=0 then r2hrs14=-4;
if e200=-3 then r2hrs14=-3;

```

```

/* Section 2: Combine variables for R1 and R2 */

/* Initialize created variable for both rounds to zero. */
allhrs14=0;

/* Account for non-interview cases */
if e200=-5 then r2hrs14=-5;

/* By the construction of the Round2 created variable,
we can simply add the two created variables from
Round1 and Round2 if they are both positive. If one is
positive and one is not, then the negative value will be
the total created variable for both rounds. If neither is
positive, then the total created variable for both rounds
will be zero. */

if tothrs14>0 then do; allhrs14=tothrs14; end;
if r2hrs14>0 then do; allhrs14=r2hrs14; end;

if tothrs14>0 and r2hrs14>0 then do;
  allhrs14=tothrs14+r2hrs14; end;

/* Define negative values for the created variables. */
if -4<tothrs14<0 then allhrs14=tothrs14;
if r2hrs14=-1 or r2hrs14=-2 or r2hrs14=-3 or r2hrs14=-5
then allhrs14=r2hrs14;

endsas;

```

NUMBER OF JOBS HELD DURING 19XX

Variables Created: CV_TTL_JOBS_YR.80 – CV_TTL_JOBS_YR.99

Programs Used

This program uses **emp_begin.sas** as input (see the first page of this appendix for details).

This program calculates the number of employee-type jobs the respondent held during each calendar year. This variable is created only for respondents who have worked at least one week since age 14.

/* Section 1: Create a variable for each year for number of jobs held in that year using R2 information. */

```
array job1wks (i) wk1_1-wk1_1044;
array job2wks (i) wk2_1-wk2_1044;
array job3wks (i) wk3_1-wk3_1044;
array job4wks (i) wk4_1-wk4_1044;
array job5wks (i) wk5_1-wk5_1044;
array job6wks (i) wk6_1-wk6_1044;
array job7wks (i) wk7_1-wk7_1044;
array job8wks (i) wk8_1-wk8_1044;
array job9wks (i) wk9_1-wk9_1044;
array starw (i) starw1-starw9;
array stopw (i) stopw1-stopw9;

/** Indicate if worked at least one week on a job in a given year **/
```

```
/* 1980 */
job801=0;      job802=0;      job803=0;
job804=0;      job805=0;      job806=0;
job807=0;      job808=0;      job809=0;
```

```
do i=1 to 52;
  if job1wks=-3 then do; job801=-3; end;
  /* and so on through job9wks and job809 */
end;
```

```
do i=1 to 52;
  if job1wks=1 then do; job801=1; end;
  /* and so on through job9wks and job809 */
end;
```

```
if job801 ne -3 and job802 ne -3 and job803 ne -3 and job804 ne -3 and job805 ne -3 and job806 ne -3 and job807 ne -3 and job808 ne -3 and job809 ne -3 then do;
  njobs80=sum(job801,job802,job803,job804,job805,
              job806,job807,job808,job809);
end;
if job801=-3 or job802=-3 or job803=-3 or job804=-3
or job805=-3 or job806=-3 or job807=-3 or job808=-3
or job809=-3 then do;
  njobs80=-3;
end;
```

/* At this point the program loops through and creates a series of job variables for each year (1981–1999). This code is not shown here due to space limitations. Users needing the entire program should contact NLS User Services. The variables and “do i” statements for each year are as follows:

| | | |
|------|------------------------|------------------|
| 1981 | job811-job819; njobs81 | do i=53 to 104 |
| 1982 | job821-job829; njobs82 | do i=105 to 156 |
| 1983 | job831-job839; njobs83 | do i=157 to 209 |
| 1984 | job841-job849; njobs84 | do i=210 to 261 |
| 1985 | job851-job859; njobs85 | do i=262 to 313 |
| 1986 | job861-job869; njobs86 | do i=314 to 365 |
| 1987 | job871-job879; njobs87 | do i=366 to 417 |
| 1988 | job881-job889; njobs88 | do i=418 to 470 |
| 1989 | job891-job899; njobs89 | do i=471 to 522 |
| 1990 | job901-job909; njobs90 | do i=523 to 574 |
| 1991 | job911-job919; njobs91 | do i=575 to 626 |
| 1992 | job921-job929; njobs92 | do i=627 to 679 |
| 1993 | job931-job939; njobs93 | do i=679 to 730 |
| 1994 | job941-job949; njobs94 | do i=731 to 783 |
| 1995 | job951-job959; njobs95 | do i=784 to 835 |
| 1996 | job961-job969; njobs96 | do i=836 to 887 |
| 1997 | job971-job979; njobs97 | do i=888 to 939 |
| 1998 | job981-job989; njobs98 | do i=940 to 991 |
| 1999 | job991-job999; njobs99 | do i=992 to 1044 |

```
do i=1 to 9;
/* start date invalid */
if starw<0 and starw>-4 then do;
  if stopw>1 then do; njobs80=njobs80+1; end;
  if stopw>52 then do; njobs81=njobs81+1; end;
  if stopw>104 then do; njobs82=njobs82+1; end;
  if stopw>156 then do; njobs83=njobs83+1; end;
  if stopw>209 then do; njobs84=njobs84+1; end;
  if stopw>261 then do; njobs85=njobs85+1; end;
  if stopw>313 then do; njobs86=njobs86+1; end;
  if stopw>365 then do; njobs87=njobs87+1; end;
  if stopw>417 then do; njobs88=njobs88+1; end;
  if stopw>470 then do; njobs89=njobs89+1; end;
  if stopw>522 then do; njobs90=njobs90+1; end;
  if stopw>574 then do; njobs91=njobs91+1; end;
  if stopw>626 then do; njobs92=njobs92+1; end;
  if stopw>678 then do; njobs93=njobs93+1; end;
  if stopw>730 then do; njobs94=njobs94+1; end;
  if stopw>783 then do; njobs95=njobs95+1; end;
  if stopw>835 then do; njobs96=njobs96+1; end;
```

```

if stopw>887 then do; njobs97=njobs97+1; end;
if stopw>939 and intwk>939 then do;
    njobs98=njobs98+1; end;
if stopw>991 and intwk>991 then do;
    njobs99=njobs99+1; end;
end;

/*stop date invalid*/
if stopw<0 and stopw>-4 then do;
    if starw<53 then do; njobs80=njobs80+1; end;
    if starw<105 then do; njobs81=njobs81+1; end;
    if starw<157 then do; njobs82=njobs82+1; end;
    if starw<210 then do; njobs83=njobs83+1; end;
    if starw<262 then do; njobs84=njobs84+1; end;
    if starw<314 then do; njobs85=njobs85+1; end;
    if starw<366 then do; njobs86=njobs86+1; end;
    if starw<418 then do; njobs87=njobs87+1; end;
    if starw<471 then do; njobs88=njobs88+1; end;
    if starw<523 then do; njobs89=njobs89+1; end;
    if starw<575 then do; njobs90=njobs90+1; end;
    if starw<627 then do; njobs91=njobs91+1; end;
    if starw<679 then do; njobs92=njobs92+1; end;
    if starw<731 then do; njobs93=njobs93+1; end;
    if starw<784 then do; njobs94=njobs94+1; end;
    if starw<836 then do; njobs95=njobs95+1; end;
    if starw<888 then do; njobs96=njobs96+1; end;
    if starw<940 then do; njobs97=njobs97+1; end;
    if starw<991 and intwk>939 then do;
        njobs98=njobs98+1; end;
    if starw<1044 and intwk>991 then do;
        njobs99=njobs99+1; end;
end;
end;

*** Include valid skips;

if e200=0 then do;
    njobs80=-4; njobs81=-4; njobs82=-4; njobs83=-4;
    njobs84=-4; njobs85=-4; njobs86=-4; njobs87=-4;
    njobs88=-4; njobs89=-4; njobs90=-4; njobs91=-4;
    njobs92=-4; njobs93=-4; njobs94=-4; njobs95=-4;
    njobs96=-4; njobs97=-4; njobs98=-4; njobs99=-4;
end;

if e200=-3 then do;
    njobs80=-3; njobs81=-3; njobs82=-3; njobs83=-3;
    njobs84=-3; njobs85=-3; njobs86=-3; njobs87=-3;
    njobs88=-3; njobs89=-3; njobs90=-3; njobs91=-3;
    njobs92=-3; njobs93=-3; njobs94=-3; njobs95=-3;
    njobs96=-3; njobs97=-3; njobs98=-3; njobs99=-3;
end;

if e200=-5 then do;
    njobs80=-5; njobs81=-5; njobs82=-5; njobs83=-5;
    njobs84=-5; njobs85=-5; njobs86=-5; njobs87=-5;
    njobs88=-5; njobs89=-5; njobs90=-5; njobs91=-5;

```

```

njobs92=-5; njobs93=-5; njobs94=-5; njobs95=-5;
njobs96=-5; njobs97=-5; njobs98=-5; njobs99=-5;
end;

/* Section 2: Calculate the total number of jobs in a
given year by matching up the UID's from Round1
and Round2.*/

array njobs (20) njobs80-njobs99; /* R2 variable */
array r1job (20) r1job80-r1job99; /* R1 variable */
array alljob (20) alljob80-alljob99; /* Total from both
rounds created variable */

/* Initialize created variable */
do i=1 to 20; alljob(i)=0; end;

/* Begin by adding together the created variables from
both rounds. Jobs that are double counted will be
subtracted off later in the program. */
do i=1 to 20;
    if r1job(i)>0 then do; alljob(i)=r1job(i); end;
    if njobs(i)>0 then do; alljob(i)=njobs(i); end;
    if r1job(i)>0 and njobs(i)>0 then do;
        alljob(i)=r1job(i)+njobs(i); end;
end;

/* Define negative values for total created variable. */
do i=1 to 20;
    if -4<r1job(i)<0 then alljob(i)=-3;
    if njobs(i)=-1 or njobs(i)=-2 or njobs(i)=-3 or
        njobs(i)=-5 then do;
        alljob(i)=njobs(i); end;
end;

/* Jobs that are worked in 1997 or 1998 have the
possibility of being double counted since they are
counted in each Round's created variable programs.
Here we compare the UID's from each round to see
which jobs are double counted. For example, match14
is the dummy variable that equals one when the first
job on the R1 UID roster and the fourth job on the R2
roster have the same UID */

array r1uid (i) r1uid1-r1uid7;
array uid (i) uid1-uid9;
array match1 (i) match11-match19;
array match2 (i) match21-match29;
array match3 (i) match31-match39;
array match4 (i) match41-match49;
array match5 (i) match51-match59;
array match6 (i) match61-match69;
array match7 (i) match71-match79;

do i=1 to 9;
    match1=0; match2=0; match3=0; match4=0;
    match5=0; match6=0; match7=0; end;

```

```

/* Determine if any UID from the first position in
Round1 matches with any UID in Round2 */
do over uid;
  if r1uid1>0 and uid>0 then do;
    if r1uid1=uid then do; match1=1; end;
    end;
  end;

/* Any UID from position 2, 3, etc. matches */
do over uid;
  if r1uid2>0 and uid>0 then do; /* repeat for r1uid3-
    r1uid7 */
    if r1uid2=uid then do; /* repeat for r1uid3-r1uid7 */
      match2=1; /* repeat for match3-match7 */
      end;
    end;
  end;

/* Define "same97" as a counted variable that adds up
how many matching UIDs are in the respondent's UID
roster. This will be subtracted from the total number of
jobs created variable to avoid double counting the same
job. Begin by initializing "same" variables to zero. */
same97=0; same98=0;

```

```

array stopyr (i) stopyr1-stopyr9;
/* Consider respondents with a R1 int. date in 1997. */
if dli_y=1997 then do;
  do over match1;
    if match1=1 then do; same97=same97+1; end;
    end;
  /* and so on through match7 */
end;

/* Consider respondents with a R1 int. date in 1998. */
if dli_y=1998 then do;
  do over match1;
    if match1=1 then do; same98=same98+1; end;
    end;
  /* and so on through match7 */
end;

/* Now subtract the "same" count variable from the
total created variable computed above.*/
if alljob97=>0 then do; alljob97=alljob97-same97; end;
if alljob98=>0 then do; alljob98=alljob98-same98; end;

endsas;

```

TOTAL NUMBER OF JOBS HELD SINCE AGE 14

Variables Created: CV_TTL_JOBS_EVER

Programs Used

This program uses **bdate1.sas** and **emp_begin.sas** as input (see the first page of this appendix for details).

This program calculates the total number of employee-type jobs held by the respondent since age 14. It is only created for respondents who have worked at least one week since age 14.

```

/* Section 1: Create a variable for number of jobs held since age 14 using R2 data. The default value is set to zero. */

array job1wks (i) wk1_1-wk1_1044; array job2wks (i) wk2_1-wk2_1044; array job3wks (i) wk3_1-wk3_1044;
array job4wks (i) wk4_1-wk4_1044; array job5wks (i) wk5_1-wk5_1044; array job6wks (i) wk6_1-wk6_1044;
array job7wks (i) wk7_1-wk7_1044; array job8wks (i) wk8_1-wk8_1044; array job9wks (i) wk9_1-wk9_1044;

/** Indicate if worked at least one week on a given job since age 14 */
job1=0;           job2=0;           job3=0;
job4=0;           job5=0;           job6=0;
job7=0;           job8=0;           job9=0;

if age14wk>0 then do;
do i=age14wk to 1044;
  if job1wks=1 then do; job1=1; end;  if job2wks=1 then do; job2=1; end;
  if job4wks=1 then do; job4=1; end;  if job5wks=1 then do; job5=1; end;
  if job7wks=1 then do; job7=1; end;  if job8wks=1 then do; job8=1; end;
end;
end;

njobs14=sum(job1,job2,job3,job4,job5,job6,job7,job8,job9);

/* Invalid start dates */
if starw1>-4 and starw1<0 and stopw1>0 and stopw1>age14wk then do; njobs14=njobs14+1; end;
if starw2>-4 and starw2<0 and stopw2>0 and stopw2>age14wk then do; njobs14=njobs14+1; end;
if starw3>-4 and starw3<0 and stopw3>0 and stopw3>age14wk then do; njobs14=njobs14+1; end;
if starw4>-4 and starw4<0 and stopw4>0 and stopw4>age14wk then do; njobs14=njobs14+1; end;
if starw5>-4 and starw5<0 and stopw5>0 and stopw5>age14wk then do; njobs14=njobs14+1; end;
if starw6>-4 and starw6<0 and stopw6>0 and stopw6>age14wk then do; njobs14=njobs14+1; end;
if starw7>-4 and starw7<0 and stopw7>0 and stopw7>age14wk then do; njobs14=njobs14+1; end;
if starw8>-4 and starw8<0 and stopw8>0 and stopw8>age14wk then do; njobs14=njobs14+1; end;
if starw9>-4 and starw9<0 and stopw9>0 and stopw9>age14wk then do; njobs14=njobs14+1; end;

if stopw1>-4 and stopw1<0 and starw1=>age14wk then do; njobs14=njobs14+1; end;
if stopw2>-4 and stopw2<0 and starw2=>age14wk then do; njobs14=njobs14+1; end;
if stopw3>-4 and stopw3<0 and starw3=>age14wk then do; njobs14=njobs14+1; end;
if stopw4>-4 and stopw4<0 and starw4=>age14wk then do; njobs14=njobs14+1; end;
if stopw5>-4 and stopw5<0 and starw5=>age14wk then do; njobs14=njobs14+1; end;
if stopw6>-4 and stopw6<0 and starw6=>age14wk then do; njobs14=njobs14+1; end;
if stopw7>-4 and stopw7<0 and starw7=>age14wk then do; njobs14=njobs14+1; end;
if stopw8>-4 and stopw8<0 and starw8=>age14wk then do; njobs14=njobs14+1; end;
if stopw9>-4 and stopw9<0 and starw9=>age14wk then do; njobs14=njobs14+1; end;

/* Invalid stop dates */
if stopw1>-4 and stopw1<0 and starw1>0 and starw1<age14wk then do; njobs14=-3; end;
if stopw2>-4 and stopw2<0 and starw2>0 and starw2<age14wk then do; njobs14=-3; end;
if stopw3>-4 and stopw3<0 and starw3>0 and starw3<age14wk then do; njobs14=-3; end;

```

Appendix 2: Employment Variable Creation

```
if stopw4>-4 and stopw4<0 and starw4>0 and starw4<age14wk then do; njobs14=-3; end;
if stopw5>-4 and stopw5<0 and starw5>0 and starw5<age14wk then do; njobs14=-3; end;
if stopw6>-4 and stopw6<0 and starw6>0 and starw6<age14wk then do; njobs14=-3; end;
if stopw7>-4 and stopw7<0 and starw7>0 and starw7<age14wk then do; njobs14=-3; end;
if stopw8>-4 and stopw8<0 and starw8>0 and starw8<age14wk then do; njobs14=-3; end;
if stopw9>-4 and stopw9<0 and starw9>0 and starw9<age14wk then do; njobs14=-3; end;

if e200=0 then do; njobs14=-4; end;
if e200=-3 then do; njobs14=-3; end;
if e200=-5 then do; njobs14=-5; end;

/* Section 2: Calculate the total number of jobs since age 14 by matching up the UID's from R1 and R2. */

alljob14=0;      /* Initialize created variable */

/* Begin by adding together the created variables from both rounds. Jobs that are double counted will be subtracted
off later in the program. */
if r1jobs14>0 then do; alljob14=r1jobs14; end;
if njobs14>0 then do; alljob14=njobs14; end;
if r1jobs14>0 and njobs14>0 then do; alljob14=r1jobs14+njobs14; end;

/* Define negative values for the total created variable. */
if -4<r1jobs14<0 then alljob14=-3;
if njobs14=-2 or njobs14=-3 or njobs14=-5 then do; alljob14=njobs14; end;

/* Jobs that are worked in 1997 or 1998 have the possibility of being double counted. Here we will compare the
UID's to see which jobs are double counted. For example, match14 is a dummy variable that equals one when the
first job on the Round1 UID roster and the fourth job on the Round2 roster have the same UID. */
array r1uid (i) r1uid1-r1uid7;          array uid (i) uid1-uid9;
array match1 (i) match11-match19;        array match2 (i) match21-match29;
array match3 (i) match31-match39;        array match4 (i) match41-match49;
array match5 (i) match51-match59;        array match6 (i) match61-match69;
array match7 (i) match71-match79;
do i=1 to 9; match1=0; match2=0; match3=0; match4=0; match5=0; match6=0; match7=0; end;

do over uid;           /* If any UID from the first position in Round1 matches with any UID in Round2 */
  if r1uid1>0 and uid>0 then do; /* and so on for r1uid2-r1uid7 and match2-match7 */
    if r1uid1=uid then do; match1=1; end;
  end;
end;

/* Define "same" as a count variable adding the # of matching UIDs are in the respondent's roster and subtract from
the total # of jobs variable to avoid double counting the same job. Begin by initializing "same" variables to zero. */
same=0;

do over match1; if match1=1 then do; same=same+1; end; end;
do over match2; if match2=1 then do; same=same+1; end; end;
do over match3; if match3=1 then do; same=same+1; end; end;
do over match4; if match4=1 then do; same=same+1; end; end;
do over match5; if match5=1 then do; same=same+1; end; end;
do over match6; if match6=1 then do; same=same+1; end; end;
do over match7; if match7=1 then do; same=same+1; end; end;

/* Now subtract the "same" count variable from the total created variable computed above. */
if alljob14=>0 then do; alljob14=alljob14-same; end;

endsas;
```

NLSY97 Appendix 3: Family Background and Formation Variable Creation

HOUSEHOLD SIZE AS OF THE SURVEY DATE

Variables Created:

| |
|----------------|
| CV_HH_SIZE |
| CV_HH_UNDER_6 |
| CV_HH_UNDER_18 |

Variables Used

| Name in Program | Question Name on CD |
|-------------------|---------------------|
| PUBID | PUBID |
| HAGE01-HAGE14 | HHI_AGE.01-.14 |
| HUID01-HUID14 | HHI_UID.01-.14 |
| RSAGE | SYMBOL!KEY!AGE |
| H5070001-H5070010 | YHHI-50700.01-.10 |

This program creates several variables describing the composition of the respondent's household: the total number of residents, the number of residents under age 6, and the number of residents under age 18.

```

array hage hage01-hage14;
array huid huid01-huid14;

/* Create dummy variables hhdum[i] (i=1 TO 14)
   that equals one if the ith member has a member ID,
   and zero if the ith member does not have a member
   ID (i.e.hhid0i=-4). Also create dummies for
   members under age 6 and under age 18.*/

array hhdum hhdum01-hhdum14;
array dum6 dum601-dum614;
array dum18 dum1801-dum1814;

do i=1 to 14;
  hhdum[i]=0;
  dum6[i]=0;
  dum18[i]=0;
end;

do i=1 to 14;
  if huid[i]>-1 then hhdum[i]=1;
  if -1<hage[i]<6 then dum6[i]=1;
  if -1<hage[i]<18 then dum18[i]=1;
end;

/* Create the dummy for the respondent's age */
rdum6=0;      rdum18=0;

if -1<rsage<6 then rdum6=1;
if -1<rsage<18 then rdum18=1;

/* Create household size hysize by adding up the
   dummies hhdum[i] and also add one for the
   respondent. Similarly, create variables under6 and
   under18 by adding up the other two dummies.*/
hysize=1;
under6=rdum6;
under18=rdum18;

do i=1 to 14;
  hsize=hsize+hhdum[i];
  under6=under6+dum6[i];
  under18=under18+dum18[i];
end;

if huid01=-5 then do;
  hsize=-5;
  under6=-5;
  under18=-5;
end;

if -4<huid01<0 or -4<huid02<0 or -4<huid03<0 or
-4<huid04<0 or -4<huid05<0 or -4<huid06<0 or
-4<huid07<0 or -4<huid08<0 or -4<huid09<0 or
-4<huid10<0 or -4<huid11<0 or -4<huid12<0 or
-4<huid13<0 or -4<huid14<0 then hsize=-3;

if -4<hage01<0 or -4<hage02<0 or -4<hage03<0 or
-4<hage04<0 or -4<hage05<0 or -4<hage06<0 or
-4<hage07<0 or -4<hage08<0 or -4<hage09<0 or
-4<hage10<0 or -4<hage11<0 or -4<hage12<0 or
-4<hage13<0 or -4<hage14<0 or -4<rsage<0 then do;
  under6=-3;
  under18=-3;
end;

/*There are four households with one member with
an estimated age under 18. Add 1 in each case.*/
if h5070001 in (1,2) or h5070002 in (1,2) or
h5070003 in (1,2) or h5070004 in (1,2) or
h5070005 in (1,2) or h5070006 in (1,2) or
h5070007 in (1,2) or h5070008 in (1,2) or
h5070009 in (1,2) or h5070010 in (1,2) then
under18=under18+1;

endsas;

```

YOUTH'S RELATIONSHIP TO HOUSEHOLD PARENT FIGURE(S)

Variables Created: CV_YTH_REL_HH_CURRENT

Variables Used

| Name in Program | Question Name on CD | Name in Program | Question Name on CD |
|-------------------|-----------------------------|-------------------|---------------------|
| PUBID | PUBID | HH2UID01-HH2UID14 | HHI_UID.01-14 |
| YID | YOUTH_ID.01 | MARRY_1-MARRY_14 | HHI_MARSTAT.01-14 |
| SH931-SH935 | SH-93.01-.05 (round 1) | REL_1-REL_14 | HHI_RELY.01-14 |
| HHI2_101-HHI2_116 | HHI2_HHI1D.01-.16 (round 1) | GEND_1-GEND_14 | HHI_SEX.01-14 |
| HH1UID01-HH1UID16 | HHI2_UID.01-.16 (round 1) | AGE | CV AGE INT DATE |
| HHAGE01-HHAGE14 | HHI AGE.01-14 | | |

Codes for Created Variable

- | | |
|------------------------------------|---------------------------------|
| 1 = Both biological parents | 6 = Adoptive parent(s) |
| 2 = Two parents, biological mother | 7 = Foster parent(s) |
| 3 = Two parents, biological father | 8 = No parents, grandparents |
| 4 = Biological mother only | 9 = No parents, other relatives |
| 5 = Biological father only | 10 = Anything else |

This program creates a variable identifying the youth's relationship to the primary adults in the household. In round 2 there is no information collected on the legal guardian of the youth, so it is not possible to determine whether respondents are living with non-parent relatives because they are guardians or because the living situation is better (e.g., closer to school, no rent). For this reason, youths above the age 18 and above are considered independent and, if they are not living with an identified parent or parent-figure (legal guardian), are put into the anything else category. Youths below the age of 18 who are not living with an identified parent or parent-figure are put into the category that most closely matches their household situation.

```

array rel_a (i) rel_1-rel_14;
array marry_a (i) marry_1-marry_14;
array gend_a (i) gend_1-gend_14;
array age_a (i) hhage01-hhage14;

drop hh2_101-hh2_116;

*this part determines legal guardians so that we can determine whether any guardians are present from the R1 interview;
array hh1uid (i) hh1uid01-hh1uid14;           *round 1 variable;
array hh2uid (i) hh2uid01-hh2uid14;           *round 2 variable;

do i=1 to 14;
  if yid=1 and sh931>0 and sh931+100=hh1uid then do;
    uid=hh1uid; end;
  if yid=2 and sh932>0 and sh932+100=hh1uid then do;
    uid=hh1uid; end;
  if yid=3 and sh933>0 and sh933+100=hh1uid then do;
    uid=hh1uid; end;
  if yid=4 and sh934>0 and sh934+100=hh1uid then do;
    uid=hh1uid; end;
  if yid=5 and sh935>0 and sh935+100=hh1uid then do;
    uid=hh1uid; end;
  end;

do i=1 to 14;
  if (uid ne . and uid=hh2uid) then line=i;
  if (uid ne . and uid=hh2uid) then line=i;

```

```

if (uid ne . and uid=hh2uid) then line=i;
if (uid ne . and uid=hh2uid) then line=i;
if (uid ne . and uid=hh2uid) then line=i;
end;
legal=0;
do i=1 to 14;
  if line=i then legal=rel_a;
end;

momid=0;      domid=0;      adopdad=0;
admom=0;      fostma=0;      fostda=0;
stepma=0;      stepda=0;      husb=0;
wife=0;        grand=0;       relat=0;
nonrel=0;      indep=0;      spouse=0;
**legal;
do i=1 to 14;
  if (legal>28 and legal<37) then do; grand=1; end;
  *spouse;
  if legal=1 or legal=2 then do; spouse=1; end;
  *brother/sister;
  if (legal>12 and legal<19) and age_a>20 then do; relat=1; end;
  *aunt or uncle and other relatives;
  if (legal>69 and legal<85) and age_a>20 then do; relat=1; end;
  *lover, roommate, other non-relative, mom's or dad's partner;
  if legal=69 or legal=68 or legal=85 or legal=88 or legal=89 then do; nonrel=1; end;
  if legal=-1 or legal=-2 or legal=-3 then do; invalid=1; end;
  if rel_1=-4 and rel_2=-4 and rel_3=-4 and rel_4=-4 and rel_5=-4 and rel_6=-4 and rel_7=-4 and
    rel_8=-4 and rel_9=-4 and rel_10=-4 and rel_11=-4 and rel_12=-4 and rel_13=-4 and rel_14=-4 then do;
    indep=1;
  end;
end;
**not legal;
do i=1 to 14;
  if (rel_a>28 and rel_a<37) then do; nlgrand=1; end;
  *spouse;
  if rel_a=1 or rel_a=2 then do; nlspouse=1; end;
  *brother/sister;
  if (rel_a>12 and rel_a<19) and age_a>20 then do; nlrelat=1; end;
  *aunt or uncle and other relatives;
  if (rel_a>69 and rel_a<85) and age_a>20 then do; relat=1; end;
  *lover, roommate, other non-relative, mom's or dad's partner;
  if rel_a=69 or rel_a=68 or rel_a=85 or rel_a=88 or rel_a=89 then do; nlnnrl=1; end;
  if rel_a=-1 or rel_a=-2 or rel_a=-3 then do; nlinv=1; end;
end;

if age<18 then do;
  if nlgrand ne . and legal=0 then grand=nlgrand;
  if nlspouse ne . and legal=0 then spouse=nlspouse;
  if nlrelat ne . and legal=0 then relat=nlrelat;
  if nlnnrl ne . and legal=0 then nonrel=nlnnrl;
  if nlinv ne . and legal=0 then invalid=nlinv;
end;

*for all youths;
if nlspouse ne . then spouse=nlspouse;

do i=1 to 14;

```

```
if rel_1=3 then momid=1;
/*and so on through rel_14=3 and momid=14*/
end;

do i=1 to 14;
  if rel_1=4 then domid=1;
/*and so on through rel_14=4 and domid=14*/
end;

do i=1 to 14;
  if rel_1=5 then admom=1;
/*and so on through rel_14=5 and admom=14*/
end;

do i=1 to 14;
  if rel_1=6 then adopdad=1;
/*and so on through rel_14=6 and adopdad=14*/
end;

do i=1 to 14;
  if rel_1=7 then stepma=1;
/*and so on through rel_14=7 and stepma=14*/
end;

do i=1 to 14;
  if rel_1=8 then stepda=1;
/*and so on through rel_14=8 and stepda=14*/
end;

do i=1 to 14;
  if rel_1=9 then fostma=1;
/*and so on through rel_14=9 and fostma=14*/
end;

do i=1 to 14;
  if rel_1=10 then fostda=1;
/*and so on through rel_14=10 and fostda=14*/
end;

/*used to determine whether single parent households contain another parent. If so hand edits are done below*/

do i=1 to 14;
  if momid>0 then do;
    if momid=1 and marry_1=1 then husb=1;
/*and so on through momid=14, marry_14=1, and husb=14*/
    end;
  end;

do i=1 to 14;
  if domid>0 then do;
    if domid=1 and marry_1=1 then wife=1;
/*and so on through domid=14, marry_14=1, and wife=14*/
    end;
  end;

rel=-16;
if age>17 then do; rel=10; end;
```

```
if marry_1=-5 then do; rel=-5; end;
if indep=1 then do; rel=10; end;
if invalid=1 then do; rel=-3; end;
if nonrel>0 then do; rel=10; end;
if relat>0 then do; rel=9; end;
if grand>0 and momid=0 and domid=0 then do; rel=8; end;
if spouse=1 then do; rel=10; end;
if fostda>0 or fostma>0 then do; rel=7; end;
if admom>0 or adopdad>0 then do; rel=6; end;
if stepda>0 or stepma>0 then do; rel=10; end;
if domid>0 and momid=0 then do; rel=5; end;
if momid>0 and domid=0 then do; rel=4; end;
if domid>0 and momid=0 then do; if admom>0 or stepma>0 then rel=3; end;
if momid>0 and domid=0 then do; if adopdad>0 or stepda>0 then rel=2; end;
if momid>0 and domid>0 then do; both=1; rel=1; end;

/*hand edits from husb and wife variables*/    if pubid=504 then rel=2;
                                                if pubid=2261 then rel=2;
                                                if pubid=1099 then rel=10;
/*additional hand edits*/                      if pubid=5636 then rel=9;
*round 1 legal guardians not captured by the above;      if pubid=7236 then rel=9;

if rel=-16 then rel=10;

endsas;
```

YOUTH'S MARITAL STATUS AND MARITAL/COHABITATION HISTORY

| | | |
|---------------------------|--|--|
| Variables Created: | CV_MARSTAT CV_FIRST_COHAB_DATE_M CV_FIRST_MARRY_DATE_M CV_FIRST_COHAB_MONTH CV_COHAB_TTL | CV_MARSTAT_COLLAPSED CV_FIRST_COHAB_DATE_Y CV_FIRST_MARRY_DATE_Y CV_FIRST_COHAB_MONTH CV_MARRIAGES_TTL |
|---------------------------|--|--|

Variables Used

| Name in Program | Question Name on CD | Name in Program | Question Name on CD |
|---------------------------|----------------------|---------------------------|-----------------------------|
| Round 1 variables: | | | |
| M45001_1, M45002_1 | YMAR-4500.01, .02 | BDATE_D, BDATE_M, BDATE_Y | KEY!BDATE_D, _M, _Y |
| M54011_1, M54012_1 | YMAR-5400.01.01, .02 | FCOH_M, FCOH_Y | CV_FIRST_COHAB_DATE_M, _Y |
| PUBID | PUBID | FCOHM | CV_FIRST_COHAB_MONTH |
| MARSTAT1 | CV_MARSTAT | FMAR_M, FMAR_Y | CV_FIRST_MARRY_DATE_M, _Y |
| CMARSTA1 | CV_MARSTAT_COLLAPSED | FMARM | CV_FIRST_MARRY_MONTH |
| COH_TTL | CV_COHAB_TTL | LINT_D, LINT_M, LINT_Y | CV_INTERVIEW_DATE_D, _M, _Y |
| MAR_TTL | CV_MARRIAGES_TTL | | |
| Round 2 variables: | | | |
| M620 | YMAR-620 | M48002M, M48002Y | YMAR-4800.02~M, ~Y |
| M650 | YMAR-650 | M48003M, M48003Y | YMAR-4800.03~M, ~Y |
| M700 | YMAR-700 | M540011, M540012 | YMAR-5400.01.01, .02 |
| M710 | YMAR-710 | M560012M, M560012Y | YMAR-5600.01.02~M, ~Y |
| M712 | YMAR-712 | M570011M, M570011Y | YMAR-5700.01.01~M, ~Y |
| M714 | YMAR-714 | M570012M, M570012Y | YMAR-5700.01.02~M, ~Y |
| M716 | YMAR-716 | M63001-M63003 | YMAR-6300.01-.03 |
| M718 | YMAR-718 | M700011M, M700011Y | YMAR-7000.01.01~M, ~Y |
| M729D | YMAR-729D | M727011, M727021 | YMAR-7270.01.01, .02.01 |
| M730 | YMAR-730 | M730011, M730021 | YMAR-7300.01.01, .02.01 |
| M740 | YMAR-740 | M7900111-M7900113 | YMAR-7900.01.01.01-.03 |
| M760 | YMAR-760 | M7900211 | YMAR-7900.02.01.01 |
| M1000 | YMAR-1000 | M810111M, M810111Y | YMAR-8100.01.01.01~M, ~Y |
| M1500 | YMAR-1500 | M810112M, M810112Y | YMAR-8100.01.01.02~M, ~Y |
| M30501-M30503 | YMAR-3050.01-.03 | M820111M, M820111Y | YMAR-8200.01.01.01~M, ~Y |
| M31001M, M31001Y | YMAR-3100.01~M, ~Y | M820211M, M820211Y | YMAR-8200.02.01.01~M |
| M31002M, M31002Y | YMAR-3100.02~M, ~Y | M910011 | YMAR-9100.01.01 |
| M31003M, M31003Y | YMAR-3100.03~M, ~Y | M920011M, M920011Y | YMAR-9200.01.01~M, ~Y |
| M45001-M45003 | YMAR-4500.01-.03 | M9800111 | YMAR-9800.01.01.01 |
| M46501-M46503 | YMAR-4650.01-.03 | M101111M, M101111Y | YMAR-10100.01.01.01~M, ~Y |
| M46701, M46702 | YMAR-4670.01, .02 | M114001-M114003 | YMAR-11400.01-.03 |
| M47001-M47003 | YMAR-4700.01-.03 | INT_D, INT_M, INT_Y | CV_INTERVIEW_DATE~D, ~M, ~Y |
| M48001M, M48001Y | YMAR-4800.01~M, ~Y | | |

Codes for Created Variable

Marital/Cohabitation Status

- 1 = never married, cohabiting
- 2 = never married, not cohabiting
- 3 = married, spouse present
- 4 = married, spouse absent
- 5 = separated, cohabiting
- 6 = separated, not cohabiting
- 7 = divorced, cohabiting
- 8 = divorced, not cohabiting
- 9 = widowed, cohabiting
- 10 = widowed, not cohabiting

Collapsed Marital Status

- 0 = never married
- 1 = married
- 2 = separated
- 3 = divorced
- 4 = widowed

This program creates two variables that describe marital status/cohabitation status as of the interview date for respondents age 16 and older. Other respondents are valid skips (-4). Note that later partners take precedence over earlier partners.

The program also creates variables which provide the dates of the youth's first marriage and/or cohabitation in both a continuous month scheme and as actual dates (for more information on the continuous month scheme, see appendix 7 in this document). Summary variables count the total number of marriages and cohabitations for each youth. Note that these variables are available only for youths age 16 and older as of 12/31/96. If a respondent is cohabiting and then marries it is considered both a cohabitation and a marriage. If someone refuses or doesn't know the full date of their marriage or cohabitation, then the spell is counted in the total variables and the date variables are coded -1 or -2 as applicable.

```

if pubid>0;
if m650 ge 0 then m620=m650;

iym=int_y*100+int_m;
dliym=lint_y*100+lint_m;
dlicm=(lint_y-1980)*12+lint_m;
doicm=(int_y-1980)*12+int_m;

/*set up arrays*/
array m    (l)  m001-m232;
array coh  (l)  coha001-coha232;
array mars (l)  mars001-mars232;

/*initialize values to 0 for all who go through section*/
if m700>-4 then do;
marstat=2;
ttlm=0;
ttlc=0;

do l=1 to 232;
  if dlicm le L le doicm then do;
    m=0; coh=0; end;
  end;

array ysca (t) M31001Y M700011Y;
array yscb (t) M31002Y yscb2;
array yscc (t) M31003Y yscc2;
array msca (t) M31001M M700011M;
array mscb (t) M31002M mscb2;
array mscc (t) M31003M mscc2;
array yeca (t) M48001Y M920011Y;
array yecb (t) M48002Y yecb2;
array yecc (t) M48003Y yecc2;
array meca (t) M48001M M920011M;
array mech (t) M48002M mech2;
array mecc (t) M48003M mecc2;
array ymsca (t) ymsca1  ymsca2;
array ymscb (t) ymscb1  ymscb2;
array ymscc (t) ymscc1  ymscc2;
array ymeca (t) ymeca1  ymeca2;
array ymecb (t) ymecb1  ymecb2;
array ymecc (t) ymecc1  ymecc2;
array csmca (t) csmca1  csmca2;
array csmcb (t) csmcb1  csmcb2;
array csmcc (t) csmcc1  csmcc2;

array cemca (t) cemca1  cemca2;
array cemcb (t) cemcb1  cemcb2;
array cemcc (t) cemcc1  cemcc2;
array cdliia (t) M46701 cdliia2;
array cdlib (t) M46702 cdlib2;
array cdlic (t) cdlic1  cdlic2;
array cbega (t) M47001 cbega2;
array cbegb (t) M47002 cbegb2;
array cbegc (t) M47003 cbegc2;
array cbeg2a (t) M910011 cbeg2a2;
array cbeg2b (t) cbeg2b1 cbeg2b2;
array cbeg2c (t) cbeg2c1 cbeg2c2;
array mbega (t) M45001 mbega2;
array mbegb (t) M45002 mbegb2;
array mbegc (t) M45003 mbegc2;
array ymsma (t) ymsma1-ymsma2;
array ymsmb (t) ymsmb1-ymsmb2;
array ymsmc (t) ymsmc1-ymsmc2;
array csmma (t) csmma1-csmma2;
array csmbb (t) csmbb1-csmbb2;
array csmmc (t) csmmc1-csmmc2;
array csmsa (t) csmsa1-csmsa2;
array csmsb (t) csmsb1-csmsb2;
array csmsc (t) csmsc1-csmsc2;
array ymssa (t) ymssa1-ymssa2;
array ymssb (t) ymssb1-ymssb2;
array ymssc (t) ymssc1-ymssc2;
array mdlia (t) M45001_1 mdlia2;
array mdlib (t) M45002_1 mdlib2;
array mdlic (t) mdlic1 mdlic2;
array cemma (t) cemma1-cemma2;
array cemmb (t) cemmb1-cemmb2;
array cemmc (t) cemmc1-cemmc2;
array howal (t) M540011 M9800111;
array howa2 (t) M540012 howa22;
array hymal (t) M570011Y M101111Y;
array hmma1 (t) M570011M M101111M;
array hyma2 (t) M570012Y hyma22;
array hmma2 (t) M570012M hmma22;
array hysa2 (t) M560012Y hysa22;
array hmsa2 (t) M560012M hmsa22;
array howa3 (t) M7900111 howa32;
array howb3 (t) M7900211 howb32;
array hyma3 (t) M820111Y hyma32;
array hymb3 (t) M820211Y hymb32;
```

```

array hmma3 (t) M820111M hmma32;
array hmmb3 (t) M820211M hmmb32;
array hysa3 (t) M810111Y hysa32;
array hmsa3 (t) M810111M hmsa32;
array howa4 (t) M7900112 howa42;
array hysa4 (t) M810112Y hysa42;
array hmsa4 (t) M810112M hmsa42;
array howa5 (t) M7900113 howa52;
array cmarsa (t) cmarsa1 cmarsa2;
array cmarsb (t) cmarsb1 cmarsb2;
array cmarsc (t) cmarsc1 cmarsc2;
array nuca (t) nuca1 nuca2;
array nucb (t) nucb1 nucb2;
array nucc (t) nucc1 nucc2;
array numa (t) numa1 numa2;
array numb (t) numb1 numb2;
array numc (t) numc1 numc2;
array fixmc1 (t) fixmc11 fixmc12;
array fixmc2 (t) fixmc21 fixmc22;
array fixmc3 (t) fixmc31 fixmc32;
array fixmm1 (t) fixmm11 fixmm12;
array fixmm2 (t) fixmm21 fixmm22;
array fixmm3 (t) fixmm31 fixmm32;
array fixyc1 (t) fixyc11 fixyc12;
array fixyc2 (t) fixyc21 fixyc22;
array fixyc3 (t) fixyc31 fixyc32;
array fixym1 (t) fixym11 fixym12;
array fixym2 (t) fixym21 fixym22;
array fixym3 (t) fixym31 fixym32;
array ysc (p) ysca yscb ysc;
array msc (p) msca msca mscc;
array yec (p) yeca yecb yecc;
array mec (p) meca mecb mecc;
array ymsc (p) ymsca ymscb ymscc;
array ymec (p) ymeca ymecb ymec;
array csmc (p) csmca csmcb csmcc;
array cemc (p) cemca cemcb cemcc;
array cdli (p) cdli cdlib cdlic;
array cbeg (p) cbega cbegb cbegc;
array cbeg2 (p) cbeg2a cbeg2b cbeg2c;
array mbeg (p) mbega mbegb mbegc;
array cmars (p) cmarsa cmarsb cmarsc;
array ymsm (p) ymsma ymsmb ymsmc;
array csmm (p) csmma csmb csmmc;
array csms (p) csmsa csmsb csmsc;
array ymss (p) ymssa ymsss ymssc;
array cemm (p) cemma cemmb cemmc;
array cems (p) cemsa cemsb cemsc;
array how1 (p) howa1 howb1 howc1;
array how2 (p) howa2 howb2 howc2;
array how3 (p) howa3 howb3 howc3;
array how4 (p) howa4 howb4 howc4;
array how5 (p) howa5 howb5 howc5;
array hym1 (p) hyma1 hymb1 hymc1;
array hmm1 (p) hmma1 hmmb1 hmmc1;
array hym2 (p) hyma2 hymb2 hymc2;
array hmm2 (p) hmma2 hmmb2 hmmc2;

array hym3 (p) hyma3 hymb3 hymc3;
array hmm3 (p) hmma3 hmmb3 hmmc3;
array hym4 (p) hyma4 hymb4 hymc4;
array hmm4 (p) hmma4 hmmb4 hmmc4;
array hys1 (p) hysa1 hysb1 hysc1;
array hms1 (p) hmsa1 hmsb1 hmsc1;
array hys2 (p) hysa2 hysb2 hysc2;
array hms2 (p) hmsa2 hmsb2 hmsc2;
array hys3 (p) hysa3 hysb3 hysc3;
array hms3 (p) hmsa3 hmsb3 hmsc3;
array hys4 (p) hysa4 hysb4 hysc4;
array hms4 (p) hmsa4 hmsb4 hmsc4;
array mdli (p) mdlia mdlib mdlic;
array nuc (p) nuca nucb nucc;
array num (p) numa numb numc;
array fixmc (p) fixmc1 fixmc2 fixmc3;
array fixmm (p) fixmm1 fixmm2 fixmm3;
array fixyc (p) fixyc1 fixyc2 fixyc3;
array fixym (p) fixym1 fixym2 fixym3;

if M54011_1>-4 then M45001_1=M54011_1;
if M54012_1>-4 then M45002_1=M54012_1;

do p=1 to 3;
do t=1 to 2;
  if cdli=1 then ymsc=dliy;
  if mdli=1 then do; cmars=1; ymsm=dliy; end;

/*hand edit*/
  if mdli=3 and pubid ne 4303 then do;
    cmars=2; ymss=dliy; end;
    if -2 le mdli le -1 then cmars=mdli;
    if ysc>0 and msc>0 then ymsc=(ysc*100)+msc;
    if -3 le ysc le 0 then do; fixyc=1; end;
    if -3 le msc le 0 then do; fixmc=1; end;
    if ysc>0 and msc>0 then ymsc=(ysc*100)+msc;
    if -3 le ysc le 0 or -3 le msc le 0 then do;
      ymsc=dliy; end;

/*need to add stop information for arrays*/
  if yec>0 and mec>0 then ymec=(yec*100)+mec;
  if -3 le yec le 0 or -3 le mec le 0 and cbeg ne 1 and
    cdli ne 1 and cbeg2 ne 1 then do;
    ymec=iym; end;
  if -3 le yec le 0 or -3 le mec le 0 and (cbeg=1 or
    cdli=1 or cbeg2=1) then do;
    ymec=iym-1; end;

  csmc=(round(ymsc,100)-198000)*.12+(ymsc-
    round(ymsc,100));
  cemc=(round(ymec,100)-198000)*.12+(ymec-
    round(ymec,100));
  if cbeg=1 or cdli=1 or cbeg2=1 then cemc=doicm;

  if mbeg=1 then do; cmars=1; ymsm=ymsc; ymsc=.;
    if fixmc=1 then fixmm=1;
    if fixyc=1 then fixym=1;

```

```

end;

if how1=1 and hym1>0 and hmm1>0 then do;
  cmars=1; ymsm=(hym1*100)+hmm1; end;
if -2 le how1 le -1 then cmars=how1;

if how2=1 and hym2>0 and hmm2>0 then do;
  cmars=1; ymss=(hym2*100)+hmm2; end;
if how2=3 and hys2>0 and hms2>0 then do;
  cmars=2; ymss=(hys2*100)+hms2; end;
if how2=4 and hys2>0 and hms2>0 then do;
  cmars=3; ymss=(hys2*100)+hms2; end;
if how2=5 and hys2>0 and hms2>0 then do;
  cmars=0; ymss=(hys2*100)+hms2; end;
if -2 le how2 le -1 then cmars=how2;

if how3=1 and hym3>0 and hmm3>0 then do;
  cmars=1; ymsm=(hym3*100)+hmm3; end;
if how3=3 and hys3>0 and hms3>0 then do;
  cmars=2; ymss=(hys3*100)+hms3; end;
if -2 le how3 le -1 then cmars=how3;

if how4=1 and hym4>0 and hmm4>0 then do;
  cmars=1; ymsm=(hym4*100)+hmm4; end;
if how4=3 and hys4>0 and hms4>0 then do;
  cmars=2; ymss=(hys4*100)+hms4; end;
if -2 le how4 le -1 then cmars=how4;

if how5=1 and hym5>0 and hmm5>0 then do;
  cmars=1; ymsm=(hym5*100)+hmm5; end;
if how5=3 and hys5>0 and hms5>0 then do;
  cmars=2; ymss=(hys5*100)+hms5; end;
if -2 le how5 le -1 then cmars=how5;

csmm=(round(ymsm,100)-198000)*.12+(ymsm-
  round(ymsm,100));
csms=(round(ymss,100)-198000)*.12+(ymss-
  round(ymss,100));
if csms>0 then cems=doicm;
if csmm>0 then cemm=doicm;

if csmc>0 then do;
  ttlc=ttlc+1; nuc=ttlc;
  if nuc=1 then cohcm=csmc;
end;

if csmm>0 then do;
  ttlm=ttlm+1; num=ttlm;
  if num=1 then marcm=csmm;
end;

corrc=0; corrm=0;
if ttlc>-1 or ttlm>-1 then do;
  if csmca2>0 and csmca2 ne . then do;
    corrc=corrc+1; end;
  if csmcb2>0 and csmcb2 ne . then do;
    corrc=corrc+1; end;

```

```

if csmcc2>0 and csmcc2 ne . then do;
  corrc=corrc+1; end;
if csmma2>0 and csmma2 ne . then do;
  corrm=corrm+1; end;
if csmb2>0 and csmb2 ne . then do;
  corrm=corrm+1; end;
if csmmc2>0 and csmmc2 ne . then do;
  corrm=corrm+1; end;
end;

ttlcnew=ttlc-corrc; ttlmnew=ttlm-corrm;

C=0;
do L=1 to 232;
  C=C+1;
  if csmc>0 and cemc>0 and csmc LE C LE cemc
    then coh=1;
  if cmars=1 and csmm>0 and cemm>0 and csmm
    le c le cemm then m=1;
  if 2 le cmars le 3 and csms>0 and cems>0 and
    csms le c le cems then m=cmars;
  if -2 le cmars le -1 then m=cmars;
  if c=doicm then do;
    if -2 le m le -1 then mars=m;
    if m=0 and coh=1 then mars=1;
    if m=0 and coh=0 then mars=2;
    if m=1 and coh=1 then mars=3;
    if m=1 and coh=0 then mars=4;
    if m=2 and coh=1 then mars=5;
    if m=2 and coh=0 then mars=6;
    if m=3 and coh=1 then mars=7;
    if m=3 and coh=0 then mars=8;
    if m=4 and coh=1 then mars=9;
    if m=4 and coh=0 then mars=10;
    marstat=mars;
  end;
end;
end;
end;
end;

if 1 le marstat le 2 then cmarstat=0;
if 3 le marstat le 4 then cmarstat=1;
if 5 le marstat le 6 then cmarstat=2;
if 7 le marstat le 8 then cmarstat=3;
if 9 le marstat le 10 then cmarstat=4;
if -2 le marstat le -1 then cmarstat=marstat;

array cvcm cohcm marcm;
array cvy cohy mary;
array cvm cohm marm;

do over cvcm;
  if 229 le cvcm le 232 then do;
    cvy=1999; cvm=cvcm-228; end;
  if 217 le cvcm le 228 then do;
    cvy=1998; cvm=cvcm-216; end;

```

Appendix 3: Family Background and Formation Variable Creation

```

if 205 le cvcm le 216 then do;
  cvy=1997; cvm=cvcm-204; end;
if 193 le cvcm le 204 then do;
  cvy=1996; cvm=cvcm-192; end;
if 181 le cvcm le 192 then do;
  cvy=1995; cvm=cvcm-180; end;
if 169 le cvcm le 180 then do;
  cvy=1994; cvm=cvcm-168; end;
if cvcm=. then do; cvcm=-4; cvy=-4; cvm=-4; end;
end;

/*hand edits for those with start date prior to 1994*/
if pubid=3927 then do; cohlm=1; cohyl=1981; end;
if pubid=5848 then do; cohlm=12; cohyl=1980; end;
if pubid=7891 then do;
  cohlm=12; cohyl=1981; marm=12; mary=1981; end;
if pubid=8522 then do; cohlm=9; cohyl=1981; end;

/*to correct for those who don't know the start month
for cohabiting*/
if fixmc11=1 and (m31001m<0 and m31001m>-4)
then do;
  cohym=m31001y; cohcm=-3; cohlm=-3; end;
/*to correct for those who don't know the start year for
cohabiting*/
if fixyc11=1 and (m31001y<0 and m31001y>-4) then
do;
  cohym=m31001m; cohcm=-3; cohyl=-3; end;
/*to correct for those who don't know either the start
year or month for cohabiting*/
if fixyc11=1 and fixmc11=1 then do;
  cohym=-3; cohcm=-3; cohyl=-3; end;

/*correct for those who don't know start month for
marriage*/
if fixmm11=1 and (m31001y<0 and m31001y>-4)
then do;
  mary=m31001y; marcm=-3; marm=-3; end;
/*correct for those who don't know start year for
marriage*/
if fixym11=1 and (m31001m<0 and m31001m>-4)
then do;
  marm=m31001m; marcm=-3; mary=-3; end;
/*to correct for those who don't know either the start
year or month for marriage*/
if fixym11=1 and fixmm11=1 then do;
  marm=-3; marcm=-3; mary=-3; end;

/*if old date present and don't deny then use old date*/
if fcohym ne -4 and (m712=1 or m714=1 or (m712=0
and m718=1)) then do;
  cohcm=fcohym; cohyl=fcoh_y; cohlm=fcoh_m; end;
if fmarm ne -4 and (m712=1 or m714=1) then do;
  marcm=fmarm; mary=fmar_y; marm=fmar_m; end;

/*correct total marriages & cohabs to reflect previous*/
/*previously in relationship, none since interview*/
if m712=1 and m740=0 then do;
  ttlc=coh_ttl;
  if mar_ttl ne 0 then ttlmnew=mar_ttl;
end;
/*previously in relationship, in more since interview*/
if m712=1 and m740=1 then do;
  ttlmnew=ttlmnew+mar_ttl;
  ttlcnew=ttlcnew+coh_ttl;
end;
/*in relationship at int date, no more since interview*/
if (m714=1 or m716=1) and m730=0 then do;
  if mar_ttl ne 0 then ttlmnew=mar_ttl;
  ttlcnew=coh_ttl;
end;
/*in relationship at int date, in more since interview*/
if (m714=1 or m716=1) and m730=1 then do;
  ttlmnew=ttlmnew+mar_ttl;
  ttlcnew=ttlcnew+coh_ttl;
end;
/*hand edits for separation without marriage*/
if pubid=3717 or pubid=5848 then do;
  marstat=1; cmarstat=0;
end;

if m700=-4 then do;
  marstat=-4; cmarstat=-4; ttlmnew=-4;
  ttlcnew=-4; cohcm=-4; marcm=-4;
  cohyl=-4; mary=-4; cohlm=-4; marm=-4;
end;

if m700=-5 then do;
  marstat=-5; cmarstat=-5; ttlmnew=-5;
  ttlcnew=-5; cohcm=-5; marcm=-5;
  cohyl=-5; mary=-5; cohlm=-5; marm=-5;
end;

/*hand edits based on interviewer remarks*/
if pubid=39 or pubid=6025 then do;
  marstat=2; cmarstat=0; ttlmnew=0;
  ttlcnew=0; cohcm=-4; marcm=-4;
  cohyl=-4; mary=-4; cohlm=-4; marm=-4;
end;

value fcmar 0 ='never married' 1 ='married'
2 ='separated' 3 ='divorced' 4 ='widowed';
value fcmmp 169-192='94-5' 193-216='96-7' 217-
232='98-9';

array m norcid cohym cohyl cohcm marm mary marcm
marstat cmarstat ttlcnew ttlmnew;
array X X01-X11;

do over X; X=m; if X=. then X=-4; end;
endsas;

```

YOUTH'S FERTILITY AND CHILD STATUS

| | | |
|---------------------------|--|--|
| Variables Created: | CV_CHILD_BIRTH_DATE.xx_M CV_CHILD_DEATH_DATE.xx_M CV_CHILD_BIRTH_MONTH.xx CV_CHILD_STATUS.xx CV_BIO_CHILD_HH | CV_CHILD_BIRTH_DATE.xx_Y CV_CHILD_DEATH_DATE.xx_Y CV_CHILD_DEATH_MONTH.xx CV_BIO_CHILD_NR |
|---------------------------|--|--|

Variables Used

| Name in Program | Question Name on CD | Name in Program | Question Name on CD |
|--------------------|------------------------------|-----------------|-------------------------|
| BDEAD971, BDEAD972 | BIOCHILD_DEAD.01, .02 | F59001, F59002 | YFER-5900.01, .02 |
| CVBIRM1, CVBIRY1 | CV_CHILD_BIRTH_DATE.01_M, _Y | F6000M, F6000Y | YFER-6000.01~M, ~Y |
| CVBIRM2, CVBIRY2 | CV_CHILD_BIRTH_DATE.02_M, _Y | BDAYM1, BDAYY1 | BIOCHILD_BDATE.01~M, ~Y |
| CVDEAM1, CVDEAY1 | CV_CHILD_DEATH_DATE.01_M, _Y | BDAYM2, BDAYY2 | BIOCHILD_BDATE.02~M, ~Y |
| CVSTAT1, CVSTAT2 | CV_CHILD_STATUS.01, .02 | BDAYM3, BDAYY3 | BIOCHILD_BDATE.03~M, ~Y |
| F400 | YFER-400 | BDEAD1-BDEAD3 | BIOCHILD_DEAD.01~.03 |
| F56001M, F56001Y | YFER-5600.01~M, ~Y | RES1-RES3 | BIOCHILD_RESIDE.01~.03 |
| F56002M, F56002Y | YFER-5600.02~M, ~Y | PUBID | PUBID |

Codes for Created Variables

Date of birth and death variables

Date variables are presented as both the actual month and year and the month number in a continuous month scheme.

Status variables

- 1 Adopted
- 2 Deceased
- 3 Non-resident, foster care
- 4 Non-resident, not adopted or in foster care
- 5 Resident

This program creates a number of variables describing the youth's fertility and the current status of the youth's children. For more information on the continuous month system, see appendix 7 in this document.

/* first, we create a variable indicating the dobm(i) and doby(i) for each biological child. Information is taken from the fertility roster(BIOC) and the fertility section of the youth survey (YFER). */

```
array bdaym[3] bdaym1-bdaym3;
array bdayy[3] bdayy1-bdayy3;
array dobm[3] dobm1-dobm3 ;
array doby[3] doby1-doby3 ;
```

```
do i=1 to 3;
  dobm(i)=-4;
  if bdaym(i) eq -3 then dobm(i)=-3;
  if bdaym(i) gt 0 then dobm(i)=bdaym(i);
end;
```

```
do i=1 to 3;
  doby(i)=-4;
  if bdayy(i) eq -3 then doby(i)=-3;
  if bdayy(i) gt 0 then doby(i)=bdayy(i);
end;
```

/* second, create a continuous month scheme variable for the month of birth of the children using the formula: (12*(doby(i)-1980)+dobm(i)) */

```
array mob[3] mob1-mob3 ;

do i=1 to 3;
  mob(i)=-4;
  if dobm(i) eq -3 or doby(i) eq -3 then mob(i)=-3;
  if dobm(i) gt 0 and doby(i) ge 1980 then
    mob(i)=12*(doby(i)-1980)+dobm(i);
end;
```

/* third, create an actual date variable for the date of death of the youth's children */

```
array bdead[3] bdead1-bdead3;
array dodm[3] dodm1-dodm3;
array dody[3] dody1-dody3;
```

```
do i=1 to 3;
  dodm(i)=-4;
  dody(i)=-4;
end;
```

```
if cvdeam1 gt 0 then dodm1=cvdeam1;
if cvdeam1=-3 then dodm1=-3;
if f6000m gt 0 then dodm1=f6000m;
```

```
if cvdeay1 gt 0 then dody1=cvdeay1;
```

Appendix 3: Family Background and Formation Variable Creation

```

if cvdeay1=-3 then dody1=-3;
if f6000y gt 0 then dody1=f6000y;

/* fourth, create a continuous month scheme variable
for the month of death of the children using the
formula: (12*(dody(i)-1980)+dodm(i)) */

array mod[3] mod1-mod3;

do i=1 to 3;
  if dodm(i)=-4 then mod(i)=-4;
  if dodm(i)=-3 then mod(i)=-3;
  if dodm(i) gt 0 and dody(i) ge 1980 then
    mod(i)=12*(dody(i)-1980)+dodm(i);
end;

/* taking care of two special cases, where the listed
children were determined not to exist.*/
if pubid=1714 or pubid=6734 then do;
  do i=1 to 3;
    dobm(i)=-4; doby(i)=-4; mob(i)=-4;
    dodm(i)=-4; dody(i)=-4; mod(i)=-4;
  end;
end;

/* fifth, create a variable indicating the status of youth's
first (second, third) child: */

array res[3] res1-res3;
array status[3] status1-status3 ;

/*initialize the status variable and determine which kids
live at home */
do i=1 to 3;
  status(i)=-4;
  if mob(i) eq -4 or res(i) eq -4 or res(i)=-5 then
    status(i)=-4;
  if res(i) eq -3 then status(i)=-3;
  if res(i) eq 1 then status(i)=5;
end;

/* if they are dead */
do i=1 to 3;
  if dodm(i) gt 0 then status(i)=2;
end;

/* res(i)=0: children's status was not updated in some
cases. F5900 variables are used to determine the status
for some of those children. */

/* adjusted by f59001 variable*/
do i=1 to 3;
  if res(i)=0 and dodm(i)=-4 then do;
    status(i)=-3;
  end;

if bdayy(i) = f56001y and bdaym(i) = f56001m then
  do;
    if f59001 gt 0 then status(i)=4;
    if f59001=1 then status(i)=5;
    if f59001=3 then status(i)=1;
    if f59001=4 then status(i)=3;
  end;
end;

/* adjusted by f59002 variable*/
do i=1 to 3;
  if bdayy(i) = f56002y and bdaym(i) = f56002m then
    do;
      if f59002 gt 0 then status(i)=4;
      if f59002=1 then status(i)=5;
      if f59002=3 then status(i)=1;
      if f59002=4 then status(i)=3;
    end;
end;

/* sixth, the number of children ever born and residing
in the household (tbiores) */

array biores[3] bires1-bires3;

/*initialize the bires variable and create tbiores */
do i=1 to 3;
  bires(i)=0;
  if status(i) eq 5 then bires(i)=1;
end;

tbiores=bires1+bires2+bires3;
if mob1=-4 then tbiores=-4;

/* seventh, the number of children ever born and not
residing in the household (tbionres) */

array bionres[3] bionres1-bionres3;

do i=1 to 3;
  bionres(i)=0;
  if (status(i) eq 1 or status(i) eq 3 or status(i) eq 4 or
      res(i)=0) and status(i) ne 2 then bionres(i)=1;
end;

tbionres=bionres1+bionres2+bionres3;
if mob1=-4 then tbionres=-4;

/* sort created variables by birthdays, so that the first
child listed is the oldest child. */
rmob1=0;
rmob2=0;
rmob3=0;

array rmob[3] rmob1 rmob2 rmob3;
array m[3] m1 m2 m3;

```

```

do i=1 to 3;
  rmob(i)=mob(i);
end;

do i=1 to 3;
  if rmob(i)=-3 then mob(i)=1000;
end;

m1=-4; m2=-4; m3=-4;

/* consider the families with one child */
if (mob1>-4 and mob2=-4 and mob3=-4) or (mob1=-4
   and mob2>-4 and mob3=-4) or (mob1=-4 and
   mob2=-4 and mob3>-4) then do;
  if mob1>-4 then m1=mob1;
  if mob2>-4 then m1=mob2;
  if mob3>-4 then m1=mob3;
end;

/* consider families with 2 children */
if (mob1>-4 and mob2>-4 and mob3=-4) then do;
  if mob1 ge mob2 then do;
    m1=mob2; m2=mob1;
  end;
  if mob1<mob2 then do;
    m1=mob1; m2=mob2;
  end;
end;

if (mob1=-4 and mob2>-4 and mob3>-4) then do;
  if mob2 ge mob3 then do;
    m1=mob3; m2=mob2;
  end;
  if mob2<mob3 then do;
    m1=mob2; m2=mob3;
  end;
end;

if (mob1>-4 and mob2=-4 and mob3>-4) then do;
  if mob1 ge mob3 then do;
    m1=mob3; m2=mob1;
  end;
  if mob1<mob3 then do;
    m1=mob1; m2=mob3;
  end;
end;

/* consider families with three children */
if (mob1>-4 and mob2>-4 and mob3>-4) then do;
  m1=min(mob1, mob2, mob3);
  m3=max(mob1, mob2, mob3);
end;
do i=1 to 3;
  if mob(i) ne m1 and mob(i) ne m3 then m2=mob(i);
end;

array cvbm[3] cvbm1-cvbm3;
array cvby[3] cvby1-cvby3;
array cvmob[3] cvmob1-cvmob3;
array cvdm[3] cvdm1-cvdm3;
array cvdy[3] cvdy1-cvdy3;
array cvmod[3] cvmod1-cvmod3;
array cvstat[3] cvstat1-cvstat3;

do i=1 to 3;
  if m1=mob(i) then do;
    cvbm1=dobm(i); cvby1=doby(i); cvmob1=rmob(i);
    cvdm1=dodm(i); cvdy1=dody(i); cvmod1=mod(i);
    cvstat1=status(i);
  end;
end;

do i=1 to 3;
  if m2=mob(i) then do;
    cvbm2=dobm(i); cvby2=doby(i); cvmob2=rmob(i);
    cvdm2=dodm(i); cvdy2=dody(i); cvmod2=mod(i);
    cvstat2=status(i);
  end;
end;

do i=1 to 3;
  if m3=mob(i) then do;
    cvbm3=dobm(i); cvby3=doby(i); cvmob3=rmob(i);
    cvdm3=dodm(i); cvdy3=dody(i); cvmod3=mod(i);
    cvstat3=status(i);
  end;
end;

/* non-interview cases*/
if f400=-5 then do;
  do i=1 to 3;
    cvbm[i]=-5; cvby[i]=-5; cvmob[i]=-5;
    cvdm[i]=-5; cvdy[i]=-5; cvmod[i]=-5;
    cvstat[i]=-5;
  end;
  tbiores=-5; tbionres=-5;
end;

/* special case—misidentification for one baby */
if pubid=8729 then do;
  cvbm1=11; cvby1=1992; cvmob1=155;
  cvdm1=11; cvdy1=1996; cvmod1=203;
  cvstat1=2;
  cvbm2=2; cvby2=1996; cvmob2=194;
  cvdm2=-4; cvdy2=-4; cvmod2=-4;
  cvstat2=5;
  cvbm3=9; cvby3=1998; cvmob3=225;
  cvdm3=-4; cvdy3=-4; cvmod3=-4;
  cvstat3=5;
  tbiores=2; tbionres=0;
end;

endsas;

```

NUMBER OF RESIDENCES SINCE AGE 12

Variables Created: CV_TTL_RESIDENCES

Variables Used

| Name in Program | Question Name on CD |
|-----------------|---------------------|
| CV | CV_TTL_RESIDENCES |
| Y3500 | YHHI-3500 |
| Y3600 | YHHI-3600 |

This program calculates the number of residences in which youth has lived since age 12. In round 1, the variable was created with information from the parent interview. In round 2, information collected from the respondent is combined with the round 1 variable to update the previous information.

```
/* Initialize each case to -4 */

RESID=-4;

/* Refusals in either round1 or round2 */
if cv=-1 or y3600=-1 then resid=-1;

/* The answer is don't know in either round1 or round2 */
if cv=-2 or y3600=-2 then resid=-2;

/* Non-negative answers in both round, then add up the numbers of residences in two rounds */
if cv ge 0 and y3600 ge 0 then resid=cv+y3600;
if cv ge 0 and y3500=0 then resid=cv;

/* If parents were not interviewed in round1*/
if cv=-4 then resid=-3;

/* If respondents were not interviewed in round2 */
if Y3600=-5 then resid=-5;

endsas;
```

NLSY97 Appendix 4:
Geographic Variable Creation

Several variables in the main data set provide information about the respondent's area of residence. These variables permit researchers to identify key characteristics of the area without needing access to the geocode CD-ROM. Geographic variables were created using a software program called Matchmaker for Windows, V2.5; therefore, no programming code is provided for these variables. Instead, this document offers a brief general description of the methods used to generate these variables. For more information about the process of classifying a respondent's metropolitan area or about the geographic variables in general, refer to the introduction to the *Geocode Codebook Supplement* or contact NLS User Services.

CENSUS REGION OF RESIDENCE AT SURVEY DATE

Variable Created: CV_CENSUS_REGION

This variable classifies respondents as residing in one of four regions defined by the U.S. Bureau of the Census. These regions are as follows:

| Census Division | States |
|-----------------|--|
| Northeast | Connecticut, Maine, Massachusetts, New Hampshire, New Jersey, New York, Pennsylvania, Rhode Island, and Vermont |
| North Central | Illinois, Indiana, Iowa, Kansas, Michigan, Minnesota, Missouri, Nebraska, North Dakota, Ohio, South Dakota, and Wisconsin |
| South | Alabama, Arkansas, Delaware, District of Columbia, Florida, Georgia, Kentucky, Louisiana, Maryland, Mississippi, North Carolina, Oklahoma, South Carolina, Tennessee, Texas, Virginia, and West Virginia |
| West | Alaska, Arizona, California, Colorado, Hawaii, Idaho, Montana, Nevada, New Mexico, Oregon, Utah, Washington, Wyoming |

MSA STATUS AT SURVEY DATE

Variable Created: CV_MSA

This variable provides users with information about whether the respondent lived in the central city of the MSA, in another part of the MSA, or outside of an MSA. As defined by the Census Bureau, a central city is the major city lying within a Metropolitan Statistical Area (MSA). Initially, two variables were created using the TIGER/Line files (a database developed by the Census Bureau) to classify respondent residences: one identifying the respondent's MSA status and one identifying the respondent's central city status. The variables were then combined to produce a single MSA/central city variable. For this dataset, respondents are coded as follows:

- 1 not in MSA
- 2 in MSA, not central city
- 3 in MSA, central city
- 4 in MSA, not known
- 5 not in country

RURAL VS. URBAN

Variable Created: CV_URBAN_RURAL

As defined by the geocode software, urban places are “closely settled, named, communities that generally contain a mixture of residential, commercial, and retail areas, and have a population greater than 2,500.” These criteria are based on those used by the Census Bureau to identify urban areas. All other places are considered rural. Based on the TIGER/Line files, respondents residing in urban areas are coded 1 and those residing in rural areas are coded 0. Census Bureau information on urban and rural places can be retrieved from the following internet site: <http://www.census.gov/geo/www/tiger/index.html>.

COLLAPSED UNEMPLOYMENT RATE

Variable Created: UNEMPRATE-COL

To provide a measure of the economic situation in the respondent’s area of residence, the dataset includes a variable indicating the unemployment rate. The round 1 NLSY97 unemployment rate variable was constructed using state and metropolitan area labor force data from the May 1998 publication of *Employment and Earnings* for the month of March 1998; the round 2 data were taken from the May 1999 publication for March 1999. *Employment and Earnings*, published by the U.S. Department of Labor, Bureau of Labor Statistics, lists the size of the civilian labor force and number of unemployed persons for every state and most metropolitan areas. The variable is created as follows:

1. If the respondent lives in a metropolitan area that is listed in *Employment and Earnings*, then the unemployment rate in the NLSY97 variable is the unemployment rate for that metropolitan area. This rate is calculated by dividing the number of unemployed persons by the number of people in the civilian labor force as reported by BLS.
2. If the respondent does not reside in a metropolitan area listed in *Employment and Earnings*, he or she is assigned a “balance of state” unemployment rate. In these cases, the figures provided for the state and its metropolitan areas are used to compute the unemployment rate for the portion of the state that is not represented in any metropolitan statistical area. (Because the *Employment and Earnings* numbers are based on an older set of MSA codes than the NLSY97 geographic variables, there are a few cases in which NLSY97 metropolitan areas do not match those used in the BLS publication. Researchers who need more exact information should contact BLS or NLS User Services about completing a confidentiality agreement and obtaining the NLSY97 Geocode CD-ROM.)

After the MSA or balance-of-state unemployment rate is calculated for each respondent, the variable for the main file CD-ROM is collapsed into ranges (less than 3.0%, 3.0–5.9%, 6.0–8.9%, 9.0–11.9%, 12.0–14.9%, and 15.0% or higher). This collapsed variable protects the privacy and confidentiality of respondents.

NLSY97 Appendix 5:
Income and Assets Variable Creation

HOUSEHOLD INCOME AND ASSETS

Variables Created: CV_HH_NET_WORTH_Y
CV_INCOME_GROSS_YR
CV_HH_POV_RATIO

Variables Used

| Name in Program | Question Name on CD | Name in Program | Question Name on CD |
|--------------------|---------------------------------|--------------------|----------------------------|
| YAS50 | YAST-050 | YAS4790-YAS4816 | YAST-4790-YAST-4816 |
| YAS200, YAS210 | YAST-200, YAST-210 | YAS4814L, YAS4814U | YAST-4814~000001, ~000002 |
| YAS600 | YAST-600 | YAS4840-YAS4846 | YAST-4840-YAST-4846 |
| YAS900 | YAST-900 | YAS4844L, YAS4844U | YAST-4844~000001, ~000002 |
| YAS1400, YAS1500 | YAST-1400, YAST-1500 | YAS4870-YAS4906 | YAST-4870-YAST-4906 |
| YAS1610 | YAST-1610 | YAS4904L, YAS4904U | YAST-4904~000001, ~000002 |
| YAS1860-YAS1866 | YAST-1860, YAST-1862, YAST-1866 | YAS5040 | YAST-5040 |
| YAS2520 | YAST-2520 | YAS5060, YAS5062 | YAST-5060, YAST-5062 |
| YAS2550-YAS2556 | YAST-2550, YAST-2552, YAST-2556 | A506601-A506620 | YAST-5066.01-.20 |
| YAS2760, YAS2766 | YAST-2760, YAST-2766 | A507401-A507420 | YAST-5074.01-.20 |
| YAS3380-YAS3386 | YAST-3380, YAST-3382, YAST-3386 | A508001-A508020 | YAST-5080.01-.20 |
| YAS3384L, YAS3384U | YAST-3384~000001, ~000002 | A508201-A508220 | YAST-5082.01-.20 |
| YAS3740- YAS3756 | YAST-3740-YAST-3756 | A508601, A508619 | YAST-5086.01, .19 |
| YAS3754L, YAS3754U | YAST-3754~000001, ~000002 | YAS5150, YAS5152 | YAST-5150, YAST-5152 |
| YAS3780 | YAST-3780 | A516401-A516420 | YAST-5164.01-.20 |
| YAS3790 | YAST-3790 | A517001-A517020 | YAST-5170.01-.20 |
| YAS3810, YAS3812 | YAST-3810, YAST-3812 | A517201-A517220 | YAST-5172.01-.20 |
| YAS3840-YAS3882 | YAST-3840-YAST-3882 | A517601 | YAST-5176.01 |
| YAS3910-YAS3922 | YAST-3910-YAST-3922 | YAS5210-YAS5226 | YAST-5210-YAST-5226 |
| YAS3924L, YAS3924U | YAST-3924~000001, ~000002 | YAS5224L, YAS5224U | YAST-5224~000001, ~000002 |
| YAS3950-YAS3966 | YAST-3950-YAST-3966 | HIRELY01-HIRELY14 | HHI_RELY.01-.14 |
| YAS3964L, YAS3964U | YAST-3964~000001, ~000002 | YI_300 | YINC-300 |
| YAS4010-YAS4036 | YAST-4010-YAST-4036 | YI_400 | YINC-400 |
| YAS4034L, YAS4034U | YAST-4034~000001, ~000002 | YI_1400-YI_3100 | YINC-1400-YINC-3100 |
| YAS4140-YAS4162 | YAST-4140-YAST-4162 | YI_3300-YI_5300 | YINC-3300-YINC-5300 |
| YAS4270-YAS4296 | YAST-4270-YAST-4296 | YI_55001-YI_55007 | YINC-5500~000000 - ~000006 |
| YAS4294L, YAS4294U | YAST-4294~000001, ~000002 | YI_5600-YI_10800 | YINC-5600-YINC-10800 |
| YAS4400-YAS4426 | YAST-4400-YAST-4426 | I1110001-I1110014 | YINC-11100.01-.14 |
| YAS4424L, YAS4424U | YAST-4424~000001, ~000002 | I1160001-I1160013 | YINC-11600.01-.13 |
| YAS4530-YAS4556 | YAST-4530-YAST-4556 | I1170001-I1170013 | YINC-11700.01-.13 |
| YAS4554L, YAS4554U | YAST-4554~000001, ~000002 | PUBID | PUBID |
| YAS4660-YAS4686 | YAST-4660-YAST-4686 | hhsize | CV_HH_SIZE |
| YAS4684L, YAS4684U | YAST-4684~000001, ~000002 | under18 | CV_HH_UNDER_18 |

This program creates the household net worth and gross household income variables. The household net worth variable is an actual number that results from adding the values of all assets and subtracting liabilities of the household. The gross household income variable includes total annual cash receipts before taxes from all sources. The program then creates a ratio comparing the household's total income to federal poverty guidelines based on the number of household residents and the number of members under age 18.

Researchers should note that, like many income and asset variables in the data set, these three variables are topcoded to protect respondent privacy. More information about topcoding is available in the *NLSY97 User's Guide*.

******* SECTION 1: GROSS HOUSEHOLD INCOME *******

```

flag=0;
/*First, create a variable indicating net receipts from non-farm employment earned by youth (Y), such as wages
(nfarmwgY). */
nfarmwgY=-4;
flag=0;
if (YI_1400=1 and YI_1700 ne -1 and YI_1700 ne -2 and YI_1700 ne -3)

```

Appendix 5: Income and Assets Variable Creation

```

or (YI_1400=-1 and YI_1600=1 and YI_1700 ne -1 and YI_1700 ne -2 and YI_1700 ne -3)
or (YI_1400=-2 and YI_1500=1 and YI_1700 ne -1 and YI_1700 ne -2 and YI_1700 ne -3)
or (YI_1400=-2 and YI_1500=-1 and YI_1600=1 and YI_1700 ne -1 and YI_1700 ne -2 and YI_1700 ne -3)
    then nfarmwgY=YI_1700;

if (YI_1400=1 or (YI_1400=-1 and YI_1600=1) or (YI_1400=-2 and YI_1500=1) or (YI_1400=-2 and YI_1500=-1
and YI_1600=1) or (YI_1400=-3 and YI_1600=1) or (YI_1400=-3 and YI_1500=1) or (YI_1400=-2 and YI_1500=-3
and YI_1600=1)) and (YI_1700 eq -1 or YI_1700 eq -2 or YI_1700 eq -3) then do;
    if YI_1800=1 then do; nfarmwgY=2500; flag=1; end;
    if YI_1800=2 then do; nfarmwgY=7500; flag=1; end;
    if YI_1800=3 then do; nfarmwgY=17500; flag=1; end;
    if YI_1800=4 then do; nfarmwgY=37500; flag=1; end;
    if YI_1800=5 then do; nfarmwgY=75000; flag=1; end;
    if YI_1800=6 then do; nfarmwgY=175000; flag=1; end;
    if YI_1800=7 then do; nfarmwgY=250001; flag=1; end;
end;
if YI_1400=0 or YI_1500=0 or YI_1600=0 then nfarmwgY=0;
if YI_1600=-1 or YI_1800=-1 then nfarmwgY=-1;
if YI_1500=-2 or YI_1800=-2 then nfarmwgY=-2;
if YI_1600=-3 or YI_1500=-3 or YI_1800=-3 then nfarmwgY=-3;

/* For all the questions below, the youth must be INDEPENDENT (YI_1900=1) */

/*Second, create a variable indicating net receipts from farm self-employment earned by the Y (farmwgY)*/
farmwgY=-4;
if YI_2000=1 and YI_2100 ne -1 and YI_2100 ne -2 and YI_2100 ne -3 then farmwgY=YI_2100;
if YI_2000=1 and (YI_2100 eq -1 or YI_2100 eq -2 or YI_2100 eq -3) then do;
    if YI_2200=1 then do; farmwgY=-2; flag=1; end;
    if YI_2200=2 then do; farmwgY=2500; flag=1; end;
    if YI_2200=3 then do; farmwgY=7500; flag=1; end;
    if YI_2200=4 then do; farmwgY=17500; flag=1; end;
    if YI_2200=5 then do; farmwgY=37500; flag=1; end;
    if YI_2200=6 then do; farmwgY=75000; flag=1; end;
    if YI_2200=7 then do; farmwgY=175000; flag=1; end;
    if YI_2200=8 then do; farmwgY=250001; flag=1; end;
end;
if YI_2000=0 then farmwgY=0;
if YI_2000=-1 or YI_2200=-1 then farmwgY=-1;
if YI_2000=-2 or YI_2200=-2 then farmwgY=-2;
if YI_2000=-3 or YI_2200=-3 then farmwgY=-3;

/*Third, create the above variables for the event the Y has a spouse/partner: nfarmwgP and farmwgP respectively */

nfarmwgP=-4;
if (YI_2300=1 and YI_2400=1 and YI_2600 ne -1 and YI_2600 ne -2 and YI_2600 ne -3) or
(YI_2300=1 and YI_2400=-1 and YI_2500=1 and YI_2600 ne -1 and YI_2600 ne -2 and YI_2600 ne -3)
    then nfarmwgP=YI_2600;
if (YI_2300=1 and YI_2400=1 and (YI_2600 eq -1 or YI_2600 eq -2 or YI_2600 eq -3)) or
(YI_2300=1 and YI_2400=-1 and YI_2500=1 and (YI_2600 eq -1 or YI_2600 eq -2 or YI_2600 eq -3)) then do;
    if YI_2700=1 then do; nfarmwgP=2500; flag=1; end;
    if YI_2700=2 then do; nfarmwgP=7500; flag=1; end;
    if YI_2700=3 then do; nfarmwgP=17500; flag=1; end;
    if YI_2700=4 then do; nfarmwgP=37500; flag=1; end;
    if YI_2700=5 then do; nfarmwgP=75000; flag=1; end;
    if YI_2700=6 then do; nfarmwgP=175000; flag=1; end;
    if YI_2700=7 then do; nfarmwgP=250001; flag=1; end;
end;

```

```

if YI_2300=0 or YI_2400=0 or YI_2500=0 then nfarmwgP=0;
if YI_2300=-1 or YI_2500=-1 or YI_2700=-1 then nfarmwgP=-1;
if YI_2300=-2 or YI_2400=-2 or YI_2500=-2 or YI_2700=-2 then nfarmwgP=-2;
if YI_2300=-3 or YI_2400=-3 or YI_2500=-3 or YI_2700=-3 then nfarmwgP=-3;

/* For any farm-related income from the spouse or partner */
farmwgP=-4;
if YI_2300=1 and YI_2900=1 and YI_3000 ne -1 and YI_3000 ne -2 and YI_3000 ne -3 then farmwgP=YI_3000;
if YI_2300=1 and YI_2900=1 and (YI_3000 eq -1 or YI_3000 eq -2 or YI_3000 eq -3) then do;
    if YI_3100=1 then do; farmwgP=-2; flag=1; end;
    if YI_3100=2 then do; farmwgP=2500; flag=1; end;
    if YI_3100=3 then do; farmwgP=7500; flag=1; end;
    if YI_3100=4 then do; farmwgP=17500; flag=1; end;
    if YI_3100=5 then do; farmwgP=37500; flag=1; end;
    if YI_3100=6 then do; farmwgP=75000; flag=1; end;
    if YI_3100=7 then do; farmwgP=175000; flag=1; end;
    if YI_3100=8 then do; farmwgP=250001; flag=1; end;
end;
if YI_2300=0 or YI_2900=0 then farmwgP=0;
if YI_2300=-1 or YI_2900=-1 or YI_3100=-1 then farmwgP=-1;
if YI_2300=-2 or YI_2900=-2 or YI_3100=-2 then farmwgP=-2;
if YI_2300=-3 or YI_2900=-3 or YI_3100=-3 then farmwgP=-3;

/* create a variable indicating whether they collected any child support (childsuY) */

childsuY=-4;
if YI_3300=1 and YI_3900=1 and YI_4000=1 and YI_4100 ne -1 and YI_4100 ne -2 and YI_4100 ne -3
    then childsuY=YI_4100;
if YI_3300=1 and YI_3900=1 and YI_4000=1 and (YI_4100 eq -1 or YI_4100 eq -2 or YI_4100 eq -3) then do;
    if YI_4200=1 then do; childsuY=500; flag=1; end;
    if YI_4200=2 then do; childsuY=1750; flag=1; end;
    if YI_4200=3 then do; childsuY=3750; flag=1; end;
    if YI_4200=4 then do; childsuY=7500; flag=1; end;
    if YI_4200=5 then do; childsuY=17500; flag=1; end;
    if YI_4200=6 then do; childsuY=37500; flag=1; end;
    if YI_4200=7 then do; childsuY=50001; flag=1; end;
end;
if YI_3300=0 or YI_3900=0 or YI_4000=0 then childsuY=0;
if YI_3300=-1 or YI_3900=-1 or YI_4000=-1 or YI_4200=-1 then childsuY=-1;
if YI_3300=-2 or YI_3900=-2 or YI_4000=-2 or YI_4200=-2 then childsuY=-2;
if YI_3300=-1 or YI_3900=-1 or YI_4000=-3 or YI_4100=-3 or YI_4200=-3 then childsuY=-3;

/* create a variable indicating the amount of interest received by the youth (Y) and his/her partner/spouse */

interesY=-4;
if YI_4300=1 and YI_4400 ne -1 and YI_4400 ne -2 and YI_4400 ne -3 then interesY=YI_4400;
if YI_4300=1 and (YI_4400 eq -1 or YI_4400 eq -2 or YI_4400 eq -3) then do;
    if YI_4500=1 then do; interesY=250; flag=1; end;
    if YI_4500=2 then do; interesY=750; flag=1; end;
    if YI_4500=3 then do; interesY=1750; flag=1; end;
    if YI_4500=4 then do; interesY=3750; flag=1; end;
    if YI_4500=5 then do; interesY=6250; flag=1; end;
    if YI_4500=6 then do; interesY=8750; flag=1; end;
    if YI_4500=7 then do; interesY=10001; flag=1; end;
end;
if YI_4300=0 then interesY=0;
if YI_4300=-1 or YI_4500=-1 then interesY=-1;

```

```

if YI_4300=-2 or YI_4500=-2 then interesY=-2;
if YI_4300=-3 or YI_4400=-3 or YI_4500=-3 then interesY=-3;

/* create a variable indicating whether they collected any dividends from stocks and mutual funds */
dividend=-4;
if YI_4600=1 and YI_4700 ne -1 and YI_4700 ne -2 and YI_4700 ne -3 then dividend=YI_4700;
if YI_4600=1 and (YI_4700 eq -1 or YI_4700 eq -2 or YI_4700 eq -3) then do;
    if YI_4800=1 then do; dividend=250; flag=1; end;
    if YI_4800=2 then do; dividend=750; flag=1; end;
    if YI_4800=3 then do; dividend=1750; flag=1; end;
    if YI_4800=4 then do; dividend=3750; flag=1; end;
    if YI_4800=5 then do; dividend=6250; flag=1; end;
    if YI_4800=6 then do; dividend=8750; flag=1; end;
    if YI_4800=7 then do; dividend=10001; flag=1; end;
end;
if YI_4600=0 then dividend=0;
if YI_4600=-1 or YI_4800=-1 then dividend=-1;
if YI_4600=-2 or YI_4800=-2 then dividend=-2;
if YI_4600=-3 or YI_4700=-3 or YI_4800=-3 then dividend=-3;

/* create a variable indicating any rental income */
rentalIY=-4;
if YI_4900=1 and YI_5000 ne -1 and YI_5000 ne -2 and YI_5000 ne -3 then rentalIY=YI_5000;
if YI_4900=1 and (YI_5000 eq -1 or YI_5000 eq -2 or YI_5000 eq -3) then do;
    if YI_5100=1 then do; rentalIY=500; flag=1; end;
    if YI_5100=2 then do; rentalIY=1750; flag=1; end;
    if YI_5100=3 then do; rentalIY=3750; flag=1; end;
    if YI_5100=4 then do; rentalIY=7500; flag=1; end;
    if YI_5100=5 then do; rentalIY=17500; flag=1; end;
    if YI_5100=6 then do; rentalIY=37500; flag=1; end;
    if YI_5100=7 then do; rentalIY=50001; flag=1; end;
end;
if YI_4900=0 then rentalIY=0;
if YI_4900=-1 or YI_5100=-1 then rentalIY=-1;
if YI_4900=-2 or YI_5100=-2 then rentalIY=-2;
if YI_4900=-3 or YI_5000=-3 or YI_5100=-3 then rentalIY=-3;

/* create a variable indicating whether they received any property or money from estates, trusts, annuities or
inheritances */

estatesY=-4;
if YI_5200=1 and YI_5300 ne -1 and YI_5300 ne -2 and YI_5300 ne -3 then estatesY=YI_5300;
if YI_5200=1 and (YI_5300 eq -1 or YI_5300 eq -2 or YI_5300 eq -3) then do;
    if YI_5400=1 then do; estatesY=2500; flag=1; end;
    if YI_5400=2 then do; estatesY=7500; flag=1; end;
    if YI_5400=3 then do; estatesY=17500; flag=1; end;
    if YI_5400=4 then do; estatesY=37500; flag=1; end;
    if YI_5400=5 then do; estatesY=75000; flag=1; end;
    if YI_5400=6 then do; estatesY=175000; flag=1; end;
    if YI_5400=7 then do; estatesY=250001; flag=1; end;
end;
if YI_5200=0 then estatesY=0;
if YI_5200=-1 or YI_5400=-1 then estatesY=-1;
if YI_5200=-2 or YI_5400=-2 then estatesY=-2;
if YI_5200=-3 or YI_5300=-3 or YI_5400=-3 then estatesY=-3;

```

```

/* create a variable indicating any allowances parents might have given the youth if he/she still lives with them (NOT INCLUDED IN THE YOUTH INCOME) */
allowpar=-4;
if YI_5600=1 then do;
if YI_5700=1 and YI_5800 ne -1 and YI_5800 ne -2 and YI_5800 ne -3 then allowpar=YI_5800;
if YI_5700=1 and (YI_5800 eq -1 or YI_5800 eq -2 or YI_5800 eq -3) then do;
    if YI_5900=1 then do; allowpar=250; flag=1; end;
    if YI_5900=2 then do; allowpar=750; flag=1; end;
    if YI_5900=3 then do; allowpar=1750; flag=1; end;
    if YI_5900=4 then do; allowpar=3750; flag=1; end;
    if YI_5900=5 then do; allowpar=6250; flag=1; end;
    if YI_5900=6 then do; allowpar=8750; flag=1; end;
    if YI_5900=7 then do; allowpar=10001; flag=1; end;
end;
if YI_5700=0 then allowpar=0;
if YI_5700=-1 or YI_5900=-1 then allowpar=-1;
if YI_5700=-2 or YI_5900=-2 then allowpar=-2;
if YI_5700=-3 or YI_5800=-3 or YI_5900=-3 then allowpar=-3;
end;

/* If living with your mother or female guardian, did you receive any money from her? (NOT INCLUDED IN THE YOUTH INCOME DEFINITION) */
allowmot=-4;
if (HIRELY01 ne 3 and HIRELY02 ne 3 and HIRELY03 ne 3 and HIRELY04 ne 3 and HIRELY05 ne 3 and HIRELY06 ne 3 and HIRELY07 ne 3 and HIRELY08 ne 3 and HIRELY09 ne 3 and HIRELY10 ne 3 and HIRELY11 ne 3 and HIRELY12 ne 3 and HIRELY13 ne 3 and HIRELY14 ne 3) then do;
    if YI_6500=1 and YI_6600 ne -1 and YI_6600 ne -2 and YI_6600 ne -3 then allowmot=YI_6600;
    if YI_6500=1 and (YI_6600 eq -1 or YI_6600 eq -2 or YI_6600 eq -3) then do;
        if YI_6700=1 then do; allowmot=250; flag=1; end;
        if YI_6700=2 then do; allowmot=750; flag=1; end;
        if YI_6700=3 then do; allowmot=1750; flag=1; end;
        if YI_6700=4 then do; allowmot=3750; flag=1; end;
        if YI_6700=5 then do; allowmot=6250; flag=1; end;
        if YI_6700=6 then do; allowmot=8750; flag=1; end;
        if YI_6700=7 then do; allowmot=10001; flag=1; end;
    end;
if YI_6500=0 then allowmot=0;
if YI_6500=-1 or YI_6700=-1 then allowmot=-1;
if YI_6500=-2 or YI_6700=-2 then allowmot=-2;
if YI_6500=-3 or YI_6600=-3 or YI_6700=-3 then allowmot=-3;
end;

/* If the youth lived with the father or male guardian and received any allowances from him (NOT INCLUDED IN THE YOUTH INCOME DEFINITION) */
allowfat=-4;
if (HIRELY01 ne 4 and HIRELY02 ne 4 and HIRELY03 ne 4 and HIRELY04 ne 4 and HIRELY05 ne 4 and HIRELY06 ne 4 and HIRELY07 ne 4 and HIRELY08 ne 4 and HIRELY09 ne 4 and HIRELY10 ne 4 and HIRELY11 ne 4 and HIRELY12 ne 4 and HIRELY13 ne 4 and HIRELY14 ne 4) then do;
    if YI_7100=1 and YI_7200 ne -1 and YI_7200 ne -2 and YI_7200 ne -3 then allowfat=YI_7200;
    if YI_7100=1 and (YI_7200 eq -1 or YI_7200 eq -2 or YI_7200 eq -3) then do;
        if YI_7300=1 then do; allowfat=250; flag=1; end;
        if YI_7300=2 then do; allowfat=750; flag=1; end;
        if YI_7300=3 then do; allowfat=1750; flag=1; end;
        if YI_7300=4 then do; allowfat=3750; flag=1; end;
        if YI_7300=5 then do; allowfat=6250; flag=1; end;
        if YI_7300=6 then do; allowfat=8750; flag=1; end;
        if YI_7300=7 then do; allowfat=10001; flag=1; end;
    end;

```

```

    end;
if YI_7100=0 then allowfat=0;
if YI_7100=-1 or YI_7300=-1 then allowfat=-1;
if YI_7100=-2 or YI_7300=-2 then allowfat=-2;
if YI_7100=-3 or YI_7200=-3 or YI_7300=-3 then allowfat=-3;
end;

/* Income received by the youth from other sources: SS payments, pension and retirement income, alimony,
payments from insurance policies, etc... */
pensionY=-4;
if YI_7600=1 and YI_7700 ne -1 and YI_7700 ne -2 and YI_7700 ne -3 then pensionY=YI_7700;
if YI_7600=1 and (YI_7700 eq -1 or YI_7700 eq -2 or YI_7700 eq -3) then do;
    if YI_7800=1 then do; pensionY=500; flag=1; end;
    if YI_7800=2 then do; pensionY=1750; flag=1; end;
    if YI_7800=3 then do; pensionY=3750; flag=1; end;
    if YI_7800=4 then do; pensionY=7500; flag=1; end;
    if YI_7800=5 then do; pensionY=17500; flag=1; end;
    if YI_7800=6 then do; pensionY=37500; flag=1; end;
    if YI_7800=7 then do; pensionY=50001; flag=1; end;
end;
if YI_7600=0 then pensionY=0;
if YI_7600=-1 or YI_7800=-1 then pensionY=-1;
if YI_7600=-2 or YI_7800=-2 then pensionY=-2;
if YI_7600=-3 or YI_7700=-3 or YI_7800=-3 then pensionY=-3;

/* For everyone, DEPENDENT OR INDEPENDENT, allowances received by the youth from his/her family
(NOT INCLUDED IN THE INCOME DEFINITION OF THE YOUTH) */
yfaallow=-4;
if YI_8100=1 and YI_8200 ne -1 and YI_8200 ne -2 and YI_8200 ne -3 and YI_8300=1 then yfaallow=YI_8200*52;
if YI_8100=1 and YI_8200 ne -1 and YI_8200 ne -2 and YI_8200 ne -3 and YI_8300=2 then yfaallow=YI_8200*12;
if YI_8100=1 and YI_8200 ne -1 and YI_8200 ne -2 and YI_8200 ne -3 and YI_8300=3 then do;
    yfaallow=-3; flag=1; end;
if YI_8100=0 then yfaallow=0;
if YI_8100=-1 or YI_8200=-1 or YI_8300=-1 then yfaallow=-1;
if YI_8100=-2 or YI_8200=-2 or YI_8300=-2 then yfaallow=-2;
if YI_8100=-3 or YI_8200=-3 or YI_8300=-3 then yfaallow=-3;

/* Now a few more questions regarding income if the individual is INDEPENDENT (YI_8500=1) or 14 YEARS OF
AGE OR OLDER (YI_8400=1)*

/* If the youth lived with the father, the father's income */
faincome=-4;
/* identifying the father is in the household */
if (HIRELY01=4 or HIRELY02=4 or HIRELY03=4 or HIRELY04=4 or HIRELY05=4 or HIRELY06=4 or
HIRELY07=4 or HIRELY08=4 or HIRELY09=4 or HIRELY10=4 or HIRELY11=4 or HIRELY12=4 or
HIRELY13=4 or HIRELY14=4) then do;
    if YI_8700=1 and YI_8800 ne -1 and YI_8800 ne -2 and YI_8800 ne -3 then faincome=YI_8800;
    if YI_8700=1 and (YI_8800 eq -1 or YI_8800 eq -2 or YI_8800 eq -3) then do;
        if YI_8900=1 then do; faincome=2500; flag=1; end;
        if YI_8900=2 then do; faincome=7500; flag=1; end;
        if YI_8900=3 then do; faincome=17500; flag=1; end;
        if YI_8900=4 then do; faincome=37500; flag=1; end;
        if YI_8900=5 then do; faincome=75000; flag=1; end;
        if YI_8900=6 then do; faincome=175000; flag=1; end;
        if YI_8900=7 then do; faincome=250001; flag=1; end;
    end;
end;

```

```

if YI_8700=0 then faincome=0;
if YI_8700=-1 or YI_8900=-1 then faincome=-1;
if YI_8700=-2 or YI_8900=-2 then faincome=-2;
if YI_8700=-3 or YI_8800=-3 or YI_8900=-3 then faincome=-3;

/* If the youth lived with the mother, the mother's income */
maincome=-4;
/* identify the mother is in the household */
if (HIRELY01=3 or HIRELY02=3 or HIRELY03=3 or HIRELY04=3 or HIRELY05=3 or HIRELY06=3 or
HIRELY07=3 or HIRELY08=3 or HIRELY09=3 or HIRELY10=3 or HIRELY11=3 or HIRELY12=3 or
HIRELY13=3 or HIRELY14=3) then do;
  if YI_9200=1 and YI_9300 ne -1 and YI_9300 ne -2 and YI_9300 ne -3 then maincome=YI_9300;
  if YI_9200=1 and (YI_9300 eq -1 or YI_9300 eq -2 or YI_9300 eq -3) then do;
    if YI_9400=1 then do; maincome=2500; flag=1; end;
    if YI_9400=2 then do; maincome=7500; flag=1; end;
    if YI_9400=3 then do; maincome=17500; flag=1; end;
    if YI_9400=4 then do; maincome=37500; flag=1; end;
    if YI_9400=5 then do; maincome=75000; flag=1; end;
    if YI_9400=6 then do; maincome=175000; flag=1; end;
    if YI_9400=7 then do; maincome=250001; flag=1; end;
  end;
end;
if YI_9200=0 then maincome=0;
if YI_9200=-1 or YI_9400=-1 then maincome=-1;
if YI_9200=-2 or YI_9400=-2 then maincome=-2;
if YI_9200=-3 or YI_9400=-3 then maincome=-3;

/* If the youth lives with the male guardian */
mgincome=-4;
if YI_9600=1 and YI_9700=1 and YI_9800 ne -1 and YI_9800 ne -2 and YI_9800 ne -3 then mgincome=YI_9800;
if YI_9600=1 and YI_9700=1 and (YI_9800 eq -1 or YI_9800 eq -2 or YI_9800 eq -3) then do;
  if YI_9900=1 then do; mgincome=2500; flag=1; end;
  if YI_9900=2 then do; mgincome=7500; flag=1; end;
  if YI_9900=3 then do; mgincome=17500; flag=1; end;
  if YI_9900=4 then do; mgincome=37500; flag=1; end;
  if YI_9900=5 then do; mgincome=75000; flag=1; end;
  if YI_9900=6 then do; mgincome=175000; flag=1; end;
  if YI_9900=7 then do; mgincome=250001; flag=1; end;
end;
if YI_9700=0 then mgincome=0;
if YI_9600=-1 or YI_9700=-1 or YI_9900=-1 then mgincome=-1;
if YI_9600=-2 or YI_9700=-2 or YI_9900=-2 then mgincome=-2;
if YI_9600=-3 or YI_9700=-3 or YI_9800=-3 or YI_9900=-3 then mgincome=-3;

/* If the youth lives with female guardian */
fgincome=-4;
if YI_10100=1 and YI_10200=1 and YI_10300 ne -1 and YI_10300 ne -2 and YI_10300 ne -3
  then fgincome=YI_10300;
if YI_10100=1 and YI_10200=1 and (YI_10300 eq -1 or YI_10300 eq -2 or YI_10300 eq -3) then do;
  if YI_10400=1 then do; fgincome=2500; flag=1; end;
  if YI_10400=2 then do; fgincome=7500; flag=1; end;
  if YI_10400=3 then do; fgincome=17500; flag=1; end;
  if YI_10400=4 then do; fgincome=37500; flag=1; end;
  if YI_10400=5 then do; fgincome=75000; flag=1; end;
  if YI_10400=6 then do; fgincome=175000; flag=1; end;
  if YI_10400=7 then do; fgincome=250001; flag=1; end;
end;

```

```

if YI_10200=0 then fgincome=0;
if YI_10100=-1 or YI_10200=-1 or YI_10400=-1 then fgincome=-1;
if YI_10100=-2 or YI_10200=-2 or YI_10400=-2 then fgincome=-2;
if YI_10100=-3 or YI_10200=-3 or YI_10300=-3 or YI_10400=-3 then fgincome=-3;

/* Check for income from any household member 14 years of age or older and who hasn't been asked before */

array otfamI otfamI01-otfamI14;
array I11100 I1110001-I1110014;
array I11600 I1160001-I1160014;
array I11700 I1170001-I1170014;

do I=1 to 14;
  otfamI(I)=-4;
  if YI_10800=1 then do;
    if I11100(I)=0 and I11600(I) ne -1 and I11600(I) ne -2 and I11600(I) ne -3 and I11600(I) ne -4
      then otfamI(I)=I11600(I);
    if I11100(I)=0 and (I11600(I) eq -1 or I11600(I) eq -2 or I11600(I) eq -3) then do;
      if I11700(I)=1 then do; otfamI(I)=2500; flag=1; end;
      if I11700(I)=2 then do; otfamI(I)=7500; flag=1; end;
      if I11700(I)=3 then do; otfamI(I)=17500; flag=1; end;
      if I11700(I)=4 then do; otfamI(I)=37500; flag=1; end;
      if I11700(I)=5 then do; otfamI(I)=75000; flag=1; end;
      if I11700(I)=6 then do; otfamI(I)=175000; flag=1; end;
      if I11700(I)=7 then do; otfamI(I)=250001; flag=1; end;
    end;
    if I11100(I)=-1 or I11700(I)=-1 then otfamI(I)=-1;
    if I11100(I)=-2 or I11700(I)=-2 then otfamI(I)=-2;
    if I11100(I)=-3 or I11600(I)=-3 or I11700(I)=-3 then otfamI(I)=-3;
  end;
end;

/* We now create gross hh income according to the youth*/

groshhIY=0;
if YI_1900=1 then do;
  if nfarmwgY not in (-1,-2,-3,-4) then groshhIY=groshhIY+nfarmwgY;
  if farmwgY not in (-1,-2,-3,-4) then groshhIY=groshhIY+farmwgY;
  if nfarmwgP not in (-1,-2,-3,-4) then groshhIY=groshhIY+nfarmwgP;
  if farmwgP not in (-1,-2,-3,-4) then groshhIY=groshhIY+farmwgP;
  if childsuY not in (-1,-2,-3,-4) then groshhIY=groshhIY+childsuY;
  if interesY not in (-1,-2,-3,-4) then groshhIY=groshhIY+interesY;
  if dividend not in (-1,-2,-3,-4) then groshhIY=groshhIY+dividend;
  if rentalIY not in (-1,-2,-3,-4) then groshhIY=groshhIY+rentalIY;
  if estatesY not in (-1,-2,-3,-4) then groshhIY=groshhIY+estatesY;
  if pensionY not in (-1,-2,-3,-4) then groshhIY=groshhIY+pensionY;
  if faincome not in (-1,-2,-3,-4) then groshhIY=groshhIY+faincome;
  if maincome not in (-1,-2,-3,-4) then groshhIY=groshhIY+maincome;
  if mgincome not in (-1,-2,-3,-4) then groshhIY=groshhIY+mgincome;
  if fgincome not in (-1,-2,-3,-4) then groshhIY=groshhIY+fgincome;
  if afdcy not in (-1,-2,-3,-4) then groshhIY=groshhIY+afdcy;
  if ssiy not in (-1,-2,-3,-4) then groshhIY=groshhIY+ssiy;
  if othe not in (-1,-2,-3,-4) then groshhIY=groshhIY+othe;
  if prgamt not in (-1,-2,-3,-4) then groshhIY=groshhIY+prgamt;
  do I=1 to 14;
    if otfamI[I] not in (-1,-2,-3,-4) then groshhIY=groshhIY+otfamI[I];
  end;

```

```

end;

if YI_1900=-1 or YI_8500=-1 or (nfarmwgY=-1 or farmwgY=-1 or nfarmwgP=-1 or farmwgP=-1 or childsuY=-1 or
    interesY=-1 or dividend=-1 or rentalIY=-1 or estatesY=-1 or pensionY=-1 or faincome=-1 or maincome=-1 or
    mgincome=-1 or fgincome=-1 or of famI01=-1 or of famI02=-1 or of famI03=-1 or of famI04=-1 or of famI04=-1
    or of famI05=-1 or of famI06=-1 or of famI07=-1 or of famI08=-1 or of famI09=-1 or of famI10=-1 or of famI11=-1
    or of famI12=-1 or of famI13=-1 or of famI14=-1 or afdcy=-1 or ssiy=-1 or othe=-1 or prgamt=-1) then
    groshhIY=-1;
if YI_1900=-2 or YI_8500=-2 or (nfarmwgY=-2 or farmwgY=-2 or nfarmwgP=-2 or farmwgP=-2 or childsuY=-2 or
    interesY=-2 or dividend=-2 or rentalIY=-2 or pensionY=-2 or estatesY=-2 or faincome=-2 or maincome=-2 or
    mgincome=-2 or fgincome=-2 or of famI01=-2 or of famI02=-2 or of famI03=-2 or of famI04=-2 or of famI04=-2
    or of famI05=-2 or of famI06=-2 or of famI07=-2 or of famI08=-2 or of famI09=-2 or of famI10=-2 or of famI11=-2
    or of famI12=-2 or of famI13=-2 or of famI14=-2 or afdcy=-2 or ssiy=-2 or othe=-2 or prgamt=-2 ) then
    groshhIY=-2;
if YI_1900=-3 or YI_8500=-3 or (nfarmwgY=-3 or farmwgY=-3 or nfarmwgP=-3 or farmwgP=-3 or childsuY=-3 or
    interesY=-3 or dividend=-3 or rentalIY=-3 or pensionY=-3 or estatesY=-3 or faincome=-3 or maincome=-3 or
    mgincome=-3 or fgincome=-3 or of famI01=-3 or of famI02=-3 or of famI03=-3 or of famI04=-3 or of famI04=-3
    or of famI05=-3 or of famI06=-3 or of famI07=-3 or of famI08=-3 or of famI09=-3 or of famI10=-3 or of famI11=-3
    or of famI12=-3 or of famI13=-3 or of famI14=-3 or afdcy=-3 or ssiy=-3 or othe=-3 or prgamt=-3 ) then
    groshhIY=-3;
if YI_1900=0 or YI_1900=-4 then groshhIY=-4;
if YAS50=-5 then groshhIY=-5; /*give the people who were not interviewed in round 2 value -5.*/

```

***** SECTION 2: HOUSEHOLD NET WORTH *****

if YAS50=1 then do;

/* If the youth OWNS some land, ITS PRESENT VALUE. */

```

pvranch=0;
if YAS1400=1 then do;
    if YAS1610=1 or YAS1610=2 then do;
        if YAS1860=1 then pvranch=YAS1862;
        if YAS1860=2 then pvranch=YAS1864L+(YAS1864U-YAS1864L)/2;
        if ((YAS1860 ne 1 and YAS1860 ne 2) or (YAS1862=-1 or YAS1862=-2)) then do;
            if YAS1866=1 then do; pvranch=12500; flag=1; end;
            if YAS1866=2 then do; pvranch=37500; flag=1; end;
            if YAS1866=3 then do; pvranch=75000; flag=1; end;
            if YAS1866=4 then do; pvranch=175000; flag=1; end;
            if YAS1866=5 then do; pvranch=375000; flag=1; end;
            if YAS1866=6 then do; pvranch=750000; flag=1; end;
            if YAS1866=7 then do; pvranch=1000001; flag=1; end;
        end;
    end;
if (YAS1610=3 or YAS1610=4) and YAS2120=100 then do; /* If the youth reported owning only part of it */
    if YAS2140=1 then pvranch=YAS2142;
    if YAS2140=2 then pvranch=(YAS2144L+(YAS2144U-YAS2144L)/2);
    if ((YAS2140 NE 1 AND YAS2140 NE 2) OR (YAS2142=-1 OR YAS2142=-2)) then do;
        if YAS2146=1 then do; pvranch=12500; flag=1; end;
        if YAS2146=2 then do; pvranch=37500; flag=1; end;
        if YAS2146=3 then do; pvranch=75000; flag=1; end;
        if YAS2146=4 then do; pvranch=175000; flag=1; end;
        if YAS2146=5 then do; pvranch=375000; flag=1; end;
        if YAS2146=6 then do; pvranch=750000; flag=1; end;
        if YAS2146=7 then do; pvranch=1000001; flag=1; end;

```

```

    end;
end;
if (YAS1610=3 or YAS1610=4) and YAS2120 ne 100 then do;
    if YAS2170=1 then pvranch=YAS2172;
    if YAS2170=2 then pvranch=(YAS2174L+(YAS2174U-YAS2174L)/2);
    if ((YAS2170 ne 1 and YAS2170 ne 2) or (YAS2172=-1 or YAS2172=-2)) then do;
        if YAS2176=1 then do; pvranch=12500; flag=1; end;
        if YAS2176=2 then do; pvranch=37500; flag=1; end;
        if YAS2176=3 then do; pvranch=75000; flag=1; end;
        if YAS2176=4 then do; pvranch=175000; flag=1; end;
        if YAS2176=5 then do; pvranch=375000; flag=1; end;
        if YAS2176=6 then do; pvranch=750000; flag=1; end;
        if YAS2176=7 then do; pvranch=1000001; flag=1; end;
    end;
end;
end;

if YAS1400=-1 or YAS1610=-1 or YAS1864L=-1 or YAS1864U=-1 or YAS1866=-1 or YAS2144L=-1 or
    YAS2144U=-1 or YAS2146=-1 or YAS2174L=-1 or YAS2174U=-1 or YAS2176=-1 then pvranch=-1;
if YAS1400=-2 or YAS1610=-2 or YAS1864L=-2 or YAS1864U=-2 or YAS1866=-2 or YAS2144L=-2 or
    YAS2144U=-2 or YAS2146=-2 or YAS2174L=-2 or YAS2174U=-2 or YAS2176=-2 then pvranch=-2;
if YAS1400=-3 or YAS1610=-3 or YAS1864L=-3 or YAS1864U=-3 or YAS1866=-3 or YAS2144L=-3 or
    YAS2144U=-3 or YAS2146=-3 or YAS2174=-3 or YAS2176=-3 then pvranch=-3;

/* PRESENT VALUE of mobile home/site. */

pvmbst=0; /* If the youth OWNS the mobile home and the site */
if YAS1500=1 then do;
    if (YAS2520=1 or YAS2520=2) and YAS2550=1 then pvmbst=YAS2552;
    if (YAS2520=1 or YAS2520=2) and YAS2550=2 then pvmbst=YAS2554L+(YAS2554U-YAS2554L)/2;
    if (YAS2520=1 or YAS2520=2) and ((YAS2550 ne 1 and YAS2550 ne 2) or (YAS2552=-1 or YAS2552=-2))
then do;
        if YAS2556=1 then do; pvmbst=500; flag=1; end;
        if YAS2556=2 then do; pvmbst=1750; flag=1; end;
        if YAS2556=3 then do; pvmbst=3750; flag=1; end;
        if YAS2556=4 then do; pvmbst=7500; flag=1; end;
        if YAS2556=5 then do; pvmbst=17500; flag=1; end;
        if YAS2556=6 then do; pvmbst=37500; flag=1; end;
        if YAS2556=7 then do; pvmbst=50001; flag=1; end;
    end;
end;
if YAS1500=-1 or YAS2520=-1 or YAS2554L=-1 or YAS2554U=-1 or YAS2556=-1 then pvmbst=-1;
if YAS1500=-2 or YAS2520=-2 or YAS2554L=-2 or YAS2554U=-2 or YAS2556=-2 then pvmbst=-2;
if YAS1500=-3 or YAS2520=-3 or YAS2552=-3 or YAS2554L=-3 or YAS2554U=-3 or YAS2556=-3 then pvmbst=-3;

pvmb=0; /* If the youth owns only the mobile home */
if YAS1500=1 then do;
    if (YAS2520=3 or YAS2520=4) and YAS2760=1 then pvmb=YAS2762;
    if (YAS2520=3 or YAS2520=4) and YAS2760=2 then pvmb=(YAS2764L+(YAS2764U-YAS2764L)/2);
    if (YAS2520=3 or YAS2520=4) and ((YAS2760 ne 1 and YAS2760 ne 2) or (YAS2762=-1 or YAS2762=-2))
then do;
        if YAS2766=1 then do; pvmb=12500; flag=1; end;
        if YAS2766=2 then do; pvmb=37500; flag=1; end;
        if YAS2766=3 then do; pvmb=75000; flag=1; end;
        if YAS2766=4 then do; pvmb=175000; flag=1; end;
        if YAS2766=5 then do; pvmb=375000; flag=1; end;
        if YAS2766=6 then do; pvmb=750000; flag=1; end;
    end;
end;

```

```

        if YAS2766=7 then do; pvmb=1000001; flag=1; end;
      end;
    end;

if YAS1500=-1 or YAS2520=-1 or YAS2764L=-1 or YAS2764U=-1 or YAS2766=-1 then pvmb=-1;
if YAS1500=-2 or YAS2520=-2 or YAS2764L=-2 or YAS2764U=-2 or YAS2766=-2 then pvmb=-2;
if YAS1500=-3 or YAS2520=-3 or YAS2762=-3 or YAS2764L=-3 or YAS2764U=-3 or YAS2766=-3 then pvmb=-3;

pvst=0; /* If the youth owns only the mobile home site */
if YAS1500=1 then do;
  if (YAS2520=5 or YAS2520=6) and YAS3010=1 then pvst=YAS3012;
  if (YAS2520=5 or YAS2520=6) and YAS3010=2 then pvst=(YAS3014L+(YAS3014U-YAS3014L)/2);
  if (YAS2520=5 or YAS2520=6) and ((YAS3010 ne 1 and YAS3010 NE 2) or (YAS3012=-1 or YAS3012=-2))
    then do;
      if YAS3016=1 then do; pvst=12500; flag=1; end;
      if YAS3016=2 then do; pvst=57500; flag=1; end;
      if YAS3016=3 then do; pvst=75000; flag=1; end;
      if YAS3016=4 then do; pvst=175000; flag=1; end;
      if YAS3016=5 then do; pvst=575000; flag=1; end;
      if YAS3016=6 then do; pvst=750000; flag=1; end;
      if YAS3016=7 then do; pvst=1000001; flag=1; end;
    end;
  end;
end;

if YAS2520=-1 or YAS3014L=-1 or YAS3014U=-1 or YAS3016=-1 or YAS1500=-1 then pvst=-1;
if YAS2520=-2 or YAS3014L=-2 or YAS3014U=-2 or YAS3016=-2 or YAS1500=-2 then pvst=-2;
if YAS2520=-3 or YAS3012=-3 or YAS3014L=-3 or YAS3014U=-3 or YAS3016=-3 or YAS1500=-3 then pvst=-3;

/* If the respondent owns the apartment, its value */

pvapthm=0;
if YAS3310=1 or YAS3310=2 then do;
  if YAS3380=1 then pvapthm=YAS3382;
  if YAS3380=2 then pvapthm=(YAS3384L+(YAS3384U-YAS3384L)/2);
  if ((YAS3380 ne 1 and YAS3380 ne 2) or (YAS3382=-1 or YAS3382=-2)) then do;
    if YAS3386=1 then do; pvapthm=500; flag=1; end;
    if YAS3386=2 then do; pvapthm=1750; flag=1; end;
    if YAS3386=3 then do; pvapthm=3750; flag=1; end;
    if YAS3386=4 then do; pvapthm=7500; flag=1; end;
    if YAS3386=5 then do; pvapthm=17500; flag=1; end;
    if YAS3386=6 then do; pvapthm=37500; flag=1; end;
    if YAS3386=7 then do; pvapthm=50001; flag=1; end;
  end;
end;

if YAS3310=-1 or YAS3384L=-1 or YAS3384U=-1 or YAS3386=-1 then pvapthm=-1;
if YAS3310=-2 or YAS3384L=-2 or YAS3384U=-2 or YAS3386=-2 then pvapthm=-2;
if YAS3310=-3 or YAS3382=-3 or YAS3384L=-3 or YAS3384U=-3 or YAS3386=-3 then pvapthm=-3;

/* Any mortgage or land contract on land or property */

mortgagY=0;
if YAS3740=1 or YAS3740=2 then do;
  if YAS3750=1 and YAS3752 ge 0 then mortgagY=YAS3752;
  if YAS3750=2 and YAS3754U ge 0 and YAS3754L ge 0 then mortgagY=(YAS3754L+(YAS3754U-YAS3754L)/2);
  if ((YAS3750 ne 1 and YAS3750 NE 2) or (YAS3752=-1 or YAS3752=-2)) then do;

```

```

if YAS3756=1 then do; mortgagY=500; flag=1; end;
if YAS3756=2 then do; mortgagY=1750; flag=1; end;
if YAS3756=3 then do; mortgagY=3750; flag=1; end;
if YAS3756=4 then do; mortgagY=7500; flag=1; end;
if YAS3756=5 then do; mortgagY=17500; flag=1; end;
if YAS3756=6 then do; mortgagY=37500; flag=1; end;
if YAS3756=7 then do; mortgagY=50001; flag=1; end;
end;
end;

if YAS3740=-1 or YAS3754L=-1 or YAS3754U=-1 or YAS3756=-1 then mortgagY=-1;
if YAS3740=-2 or YAS3754L=-2 or YAS3754U=-2 or YAS3756=-2 then mortgagY=-2;
if YAS3740=-3 or YAS3752=-3 or YAS3754L=-3 or YAS3754U=-3 or YAS3756=-3 then mortgagY=-3;

/* Any loans to remodel this residence obtained in this second round */

loanowed=0;
if YAS3790=1 then do;
  if YAS3810=1 and YAS3812 ge 0 then loanowed=YAS3812;
  if YAS3810=2 and YAS3814U ge 0 and YAS3814L ge 0 then loanowed=(YAS3814L+(YAS3814U-
  YAS3814L)/2);
  if ((YAS3810 ne 1 and YAS3810 ne 2) or (YAS3812=-1 or YAS3812=-2)) then do;
    if YAS3816=1 then do; loanowed=12500; flag=1; end;
    if YAS3816=2 then do; loanowed=37500; flag=1; end;
    if YAS3816=3 then do; loanowed=75000; flag=1; end;
    if YAS3816=4 then do; loanowed=175000; flag=1; end;
    if YAS3816=5 then do; loanowed=375000; flag=1; end;
    if YAS3816=6 then do; loanowed=750000; flag=1; end;
    if YAS3816=7 then do; loanowed=1000001; flag=1; end;
  end;
end;

if YAS3780=-1 or YAS3785=-1 or YAS3790=-1 or YAS3814L=-1 or YAS3814U=-1 or YAS3816=-1 then
loanowed=-1;
if YAS3780=-2 or YAS3785=-2 or YAS3790=-2 or YAS3814L=-2 or YAS3814U=-2 or YAS3816=-2 then
loanowed=-2;
if YAS3780=-3 or YAS3785=-3 or YAS3790=-3 or YAS3812=-3 or YAS3814L=-3 or YAS3814U=-3 or YAS3816=-3
then loanowed=-3;

/* Any remaining loans from round one still unpaid, total amount owed */

stilowed=0;
if (YAS3840=1 and YAS3850=1) or YAS3860=1 then do;
  if YAS3880=1 and YAS3882 ge 0 then stilowed=YAS3882;
  if YAS3880=2 and YAS3884U ge 0 and YAS3884L ge 0 then stilowed=(YAS3884L+(YAS3884U-
  YAS3884L)/2);
  if ((YAS3880 ne 1 and YAS3880 ne 2) or (YAS3882=-1 or YAS3882=-2)) then do;
    if YAS3886=1 then do; stilowed=500; flag=1; end;
    if YAS3886=2 then do; stilowed=1750; flag=1; end;
    if YAS3886=3 then do; stilowed=3750; flag=1; end;
    if YAS3886=4 then do; stilowed=7500; flag=1; end;
    if YAS3886=5 then do; stilowed=17500; flag=1; end;
    if YAS3886=6 then do; stilowed=37500; flag=1; end;
    if YAS3886=7 then do; stilowed=50001; flag=1; end;
  end;
end;

```

```

if YAS3860=-1 or YAS3884L=-1 or YAS3884U=-1 or YAS3886=-1 then stilowed=-1;
if YAS3860=-2 or YAS3884L=-2 or YAS3884U=-2 or YAS3886=-2 then stilowed=-2;
if YAS3860=-3 or YAS3882=-3 or YAS3884L=-3 or YAS3884U=-3 or YAS3886=-3 then stilowed=-3;

```

```
/* Any second mortgages */
```

```

secmortY=0;
if YAS3910=1 then do;
  if YAS3920=1 and YAS3922 ge 0 then secmortY=YAS3922;
  if YAS3920=2 and YAS3924U ge 0 and YAS3924L ge 0 then secmortY=(YAS3924L+(YAS3924U-
  YAS3924L)/2);
  if ((YAS3920 ne 1 and YAS3920 ne 2) or (YAS3922=-1 or YAS3922=-2)) then do;
    if YAS3926=1 then do; secmortY=500; flag=1; end;
    if YAS3926=2 then do; secmortY=1750; flag=1; end;
    if YAS3926=3 then do; secmortY=3750; flag=1; end;
    if YAS3926=4 then do; secmortY=7500; flag=1; end;
    if YAS3926=5 then do; secmortY=17500; flag=1; end;
    if YAS3926=6 then do; secmortY=37500; flag=1; end;
    if YAS3926=7 then do; secmortY=50001; flag=1; end;
  end;
end;

if YAS3910=-1 or YAS3924L=-1 or YAS3924U=-1 or YAS3926=-1 then secmortY=-1;
if YAS3910=-2 or YAS3924L=-2 or YAS3924U=-2 or YAS3926=-2 then secmortY=-2;
if YAS3910=-3 or YAS3922=-3 or YAS3924L=-3 or YAS3924U=-3 or YAS3926=-3 then secmortY=-3;

```

```
/* Any taxes on the property to be paid */
```

```

proptaxY=0;
if YAS3950=1 then do;
  if YAS3960=1 and YAS3962 ge 0 then proptaxY=YAS3962;
  if YAS3960=2 and YAS3964U ge 0 and YAS3964L ge 0 then proptaxY=(YAS3964L+(YAS3964U-
  YAS3964L)/2);
  if ((YAS3960 ne 1 and YAS3960 ne 2) or (YAS3962=-1 or YAS3962=-2)) then do;
    if YAS3966=1 then do; proptaxY=500; flag=1; end;
    if YAS3966=2 then do; proptaxY=1750; flag=1; end;
    if YAS3966=3 then do; proptaxY=3750; flag=1; end;
    if YAS3966=4 then do; proptaxY=7500; flag=1; end;
    if YAS3966=5 then do; proptaxY=17500; flag=1; end;
    if YAS3966=6 then do; proptaxY=37500; flag=1; end;
    if YAS3966=7 then do; proptaxY=50001; flag=1; end;
  end;
end;

if YAS3950=-1 or YAS3964L=-1 or YAS3964U=-1 or YAS3966=-1 then proptaxY=-1;
if YAS3950=-2 or YAS3964L=-2 or YAS3964U=-2 or YAS3966=-2 then proptaxY=-2;
if YAS3950=-3 or YAS3962=-3 or YAS3964L=-3 or YAS3964U=-3 or YAS3966=-3 then proptaxY=-3;

```

```
/* Own a business, partnership or professional practice */
```

```

pvbussY=0;
if YAS4010=1 or YAS4010=2 then do;
  if YAS4030=1 then pbuss=YAS4032;
  if YAS4030=2 then pbuss=(YAS4034L+(YAS4034U-YAS4034L)/2);
  if ((YAS4030 ne 1 and YAS4030 ne 2) or (YAS4032=-1 or YAS4032=-2)) then do;
    if YAS4036=1 then do; pbuss=12500; flag=1; end;
    if YAS4036=2 then do; pbuss=37500; flag=1; end;

```

```

if YAS4036=3 then do; pbuss=75000; flag=1; end;
if YAS4036=4 then do; pbuss=175000; flag=1; end;
if YAS4036=5 then do; pbuss=375000; flag=1; end;
if YAS4036=6 then do; pbuss=750000; flag=1; end;
if YAS4036=7 then do; pbuss=1000001; flag=1; end;
end;
end;

if YAS4010=-1 or YAS4034L=-1 or YAS4034U=-1 or YAS4036=-1 then pbuss=-1;
if YAS4010=-2 or YAS4034L=-2 or YAS4034U=-2 or YAS4036=-2 then pbuss=-2;
if YAS4010=-3 or YAS4032=-3 or YAS4034L=-3 or YAS4034U=-3 or YAS4036=-3 then pbuss=-3;

/* Second real estate owned */

secrestY=0;
if YAS4140=1 or YAS4140=2 then do;
  if YAS4160=1 then secrestY=YAS4162;
  if YAS4160=2 then secrestY=(YAS4164L+(YAS4164U-YAS4164L)/2);
  if ((YAS4160 ne 1 and YAS4160 ne 2) or (YAS4162=-1 or YAS4162=-2)) then do;
    if YAS4166=1 then do; secrestY=12500; flag=1; end;
    if YAS4166=2 then do; secrestY=37500; flag=1; end;
    if YAS4166=3 then do; secrestY=75000; flag=1; end;
    if YAS4166=4 then do; secrestY=175000; flag=1; end;
    if YAS4166=5 then do; secrestY=375000; flag=1; end;
    if YAS4166=6 then do; secrestY=750000; flag=1; end;
    if YAS4166=7 then do; secrestY=1000001; flag=1; end;
  end;
end;

if YAS4140=-1 or YAS4164L=-1 or YAS4164U=-1 or YAS4166=-1 then secrestY=-1;
if YAS4140=-2 or YAS4164L=-2 or YAS4164U=-2 or YAS4166=-2 then secrestY=-2;
if YAS4140=-3 or YAS4162=-3 or YAS4164L=-3 or YAS4164U=-3 or YAS4166=-3 then secrestY=-3;

/* Any retirement plans or pensions */

retireY=0;
if YAS4270=1 or YAS4270=2 or YAS4270=3 then do;
  if YAS4290=1 and YAS4292 ge 0 then retireY=YAS4292;
  if YAS4290=2 and YAS4294U ge 0 and YAS4294L ge 0 then retireY=(YAS4294L+(YAS4294U-YAS4294L)/2);
  if ((YAS4290 ne 1 and YAS4290 ne 2) or (YAS4292=-1 or YAS4292=-2)) then do;
    if YAS4296=1 then do; retireY=2500; flag=1; end;
    if YAS4296=2 then do; retireY=7500; flag=1; end;
    if YAS4296=3 then do; retireY=17500; flag=1; end;
    if YAS4296=4 then do; retireY=37500; flag=1; end;
    if YAS4296=5 then do; retireY=75000; flag=1; end;
    if YAS4296=6 then do; retireY=175000; flag=1; end;
    if YAS4296=7 then do; retireY=250001; flag=1; end;
  end;
end;

if YAS4270=-1 or YAS4294L=-1 or YAS4294U=-1 or YAS4296=-1 then retireY=-1;
if YAS4270=-2 or YAS4294L=-2 or YAS4294U=-2 or YAS4296=-2 then retireY=-2;
if YAS4270=-3 or YAS4292=-3 or YAS4294L=-3 or YAS4294U=-3 or YAS4296=-3 then retireY=-3;

/* Any savings in saving accounts, money market, funds, trusts,...*/

savingsY=0;

```

```

if YAS4400=1 or YAS4400=2 or YAS4400=3 then do;
    if YAS4420=1 and YAS4422 ge 0 then savingsY=YAS4422;
    if YAS4420=2 and YAS4424U ge 0 and YAS4424L ge 0 then savingsY=(YAS4424L+(YAS4424U-
YAS4424L)/2);
    if ((YAS4420 ne 1 and YAS4420 ne 2) or (YAS4422=-1 or YAS4422=-2)) then do;
        if YAS4426=1 then do; savingsY=500; flag=1; end;
        if YAS4426=2 then do; savingsY=1750; flag=1; end;
        if YAS4426=3 then do; savingsY=3750; flag=1; end;
        if YAS4426=4 then do; savingsY=7500; flag=1; end;
        if YAS4426=5 then do; savingsY=17500; flag=1; end;
        if YAS4426=6 then do; savingsY=37500; flag=1; end;
        if YAS4426=7 then do; savingsY=50001; flag=1; end;
    end;
end;

if YAS4400=-1 or YAS4424L=-1 or YAS4424U=-2 or YAS4426=-1 then savingsY=-1;
if YAS4400=-2 or YAS4424L=-2 or YAS4424U=-2 or YAS4426=-2 then savingsY=-2;
if YAS4400=-3 or YAS4422=-3 or YAS4424L=-3 or YAS4424U=-3 or YAS4426=-3 then savingsY=-3;

/* Any other savings in bonds or CDs */

othsavY=0;
if YAS4530=1 or YAS4530=2 or YAS4530=3 then do;
    if YAS4550=1 and YAS4552 ge 0 then othsavY=YAS4552;
    if YAS4550=2 and YAS4554U ge 0 and YAS4554L ge 0 then othsavY=(YAS4554L+(YAS4554U-
YAS4554L)/2);
    if ((YAS4550 ne 1 and YAS4550 ne 2) or (YAS4552=-1 or YAS4552=-2)) then do;
        if YAS4556=1 then do; othsavY=500; flag=1; end;
        if YAS4556=2 then do; othsavY=1750; flag=1; end;
        if YAS4556=3 then do; othsavY=3750; flag=1; end;
        if YAS4556=4 then do; othsavY=7500; flag=1; end;
        if YAS4556=5 then do; othsavY=17500; flag=1; end;
        if YAS4556=6 then do; othsavY=37500; flag=1; end;
        if YAS4556=7 then do; othsavY=50001; flag=1; end;
    end;
end;

if YAS4530=-1 or YAS4554L=-1 or YAS4554U=-1 or YAS4556=-1 then othsavY=-1;
if YAS4530=-2 or YAS4554L=-2 or YAS4554U=-2 or YAS4556=-2 then othsavY=-2;
if YAS4530=-3 or YAS4552=-3 or YAS4554L=-3 or YAS4554U=-3 or YAS4556=-3 then othsavY=-3;

/* Any stocks in corporations, mutual funds */

stockY=0;
if YAS4660=1 or YAS4660=2 or YAS4660=3 then do;
    if YAS4680=1 then stockY=YAS4682;
    if YAS4680=2 then stockY=(YAS4684L+(YAS4684U-YAS4684L)/2);
    if ((YAS4680 ne 1 and YAS4680 ne 2) or (YAS4682=-1 or YAS4682=-2)) then do;
        if YAS4686=1 then do; stockY=500; flag=1; end;
        if YAS4686=2 then do; stockY=1750; flag=1; end;
        if YAS4686=3 then do; stockY=3750; flag=1; end;
        if YAS4686=4 then do; stockY=7500; flag=1; end;
        if YAS4686=5 then do; stockY=17500; flag=1; end;
        if YAS4686=6 then do; stockY=37500; flag=1; end;
        if YAS4686=7 then do; stockY=50001; flag=1; end;
    end;
end;

```

```

if YAS4660=-1 or YAS4684L=-1 or YAS4684U=-1 or YAS4686=-1 then stockY=-1;
if YAS4660=-2 or YAS4684L=-2 or YAS4684U=-2 or YAS4686=-2 then stockY=-2;
if YAS4660=-3 or YAS4682=-3 or YAS4684L=-3 or YAS4684U=-3 or YAS4686=-3 then stockY=-3;

```

```

/* Present value of any vehicles owned */

```

```

pvcarsY=0;
if YAS4790=1 or YAS4790=2 or YAS4790=3 then do;
    if YAS4810=1 then pvcarsY=YAS4812;
    if YAS4810=2 then pvcarsY=(YAS4814L+(YAS4814U-YAS4814L)/2);
    if ((YAS4810 ne 1 and YAS4810 ne 2) or (YAS4812=-1 or YAS4812=-2)) then do;
        if YAS4816=1 then do; pvcarsY=2500; flag=1; end;
        if YAS4816=2 then do; pvcarsY=7500; flag=1; end;
        if YAS4816=3 then do; pvcarsY=17500; flag=1; end;
        if YAS4816=4 then do; pvcarsY=37500; flag=1; end;
        if YAS4816=5 then do; pvcarsY=75000; flag=1; end;
        if YAS4816=6 then do; pvcarsY=175000; flag=1; end;
        if YAS4816=7 then do; pvcarsY=250001; flag=1; end;
    end;
end;

```

```

if YAS4790=-1 or YAS4814L=-1 or YAS4814U=-1 or YAS4816=-1 then pvcarsY=-1;
if YAS4790=-2 or YAS4814L=-2 or YAS4814U=-2 or YAS4816=-2 then pvcarsY=-2;
if YAS4790=-3 or YAS4812=-3 or YAS4814L=-3 or YAS4814U=-3 or YAS4376=-3 then pvcarsY=-3;

```

```

/* Money still owed on these vehicles */

```

```

cardebt=0;
if YAS4840=1 then cardebt=YAS4842;
if YAS4840=2 then cardebt=(YAS4844U-(YAS4844U-YAS4844L)/2);
if ((YAS4840 ne 1 and YAS4840 ne 2) or (YAS4842=-1 or YAS4842=-2)) then do;
    if YAS4846=1 then do; cardebt=2500; flag=1; end;
    if YAS4846=2 then do; cardebt=7500; flag=1; end;
    if YAS4846=3 then do; cardebt=17500; flag=1; end;
    if YAS4846=4 then do; cardebt=37500; flag=1; end;
    if YAS4846=5 then do; cardebt=75000; flag=1; end;
    if YAS4846=6 then do; cardebt=175000; flag=1; end;
    if YAS4846=7 then do; cardebt=250001; flag=1; end;
end;

```

```

if YAS4790=-1 or YAS4844L=-1 or YAS4844U=-1 or YAS4846=-1 then cardebt=-1;
if YAS4790=-2 or YAS4844L=-2 or YAS4844U=-2 or YAS4846=-2 then cardebt=-2;
if YAS4790=-3 oR YAS4842=-3 or YAS4844L=-3 or YAS4844U=-3 or YAS4376=-3 then cardebt=-3;

```

```

/* Present value of owned furniture */

```

```

pvfurnty=0;
if YAS4870=1 then do; pvfurnty=2500; flag=1; end;
if YAS4870=2 then do; pvfurnty=7500; flag=1; end;
if YAS4870=3 then do; pvfurnty=17500; flag=1; end;
if YAS4870=4 then do; pvfurnty=37500; flag=1; end;
if YAS4870=5 then do; pvfurnty=75000; flag=1; end;
if YAS4870=6 then do; pvfurnty=175000; flag=1; end;
if YAS4870=7 then do; pvfurnty=250001; flag=1; end;

```

```

if YAS4870=-1 then pvfurnty=-1;

```

```

if YAS4870=-2 then pvfurnty=-2;
if YAS4870=-3 then pvfurnty=-3;

/* Any other assets not being mentioned before */

otassetY=0;
if YAS4880=1 or YAS4880=2 or YAS4880=3 then do;
  if YAS4900=1 then otassetY=YAS4902;
  if YAS4900=2 then otassetY=(YAS4904L+(YAS4904U-YAS4904L)/2);
  if ((YAS4900 ne 1 and YAS4900 ne 2) or (YAS4902=-1 or YAS4902=-2)) then do;
    if YAS4906=1 then do; otassetY=2500; flag=1; end;
    if YAS4906=2 then do; otassetY=7500; flag=1; end;
    if YAS4906=3 then do; otassetY=17500; flag=1; end;
    if YAS4906=4 then do; otassetY=37500; flag=1; end;
    if YAS4906=5 then do; otassetY=75000; flag=1; end;
    if YAS4906=6 then do; otassetY=175000; flag=1; end;
    if YAS4906=7 then do; otassetY=250001; flag=1; end;
  end;
end;

if YAS4880=-1 or YAS4904L=-1 or YAS4904U=-1 or YAS4906=-1 then otassetY=-1;
if YAS4880=-2 or YAS4904L=-2 or YAS4904U=-2 or YAS4906=-2 then otassetY=-2;
if YAS4880=-3 or YAS4902=-3 or YAS4904L=-3 or YAS4904U=-3 or YAS4376=-3 then otassetY=-3;

/* Any loans still owed to family or relatives */

array rloan rloan01 rloan02 rloan03 rloan04 rloan05 rloan06 rloan07 rloan08 rloan09 rloan10 rloan11 rloan12
      rloan13 rloan14 rloan15 rloan16 rloan17 rloan18 rloan19 rloan20;
array A5080 A508001 A508002 A508003 A508004 A508005 A508006 A508007 A508008 A508009 A508010
      A508011 A508012 A508013 A508014 A508015 A508016 A508017 A508018 A508019 A508020;
array A5082 A508201 A508202 A508203 A508204 A508205 A508206 A508207 A508208 A508209 A508210
      A508211 A508212 A508213 A508214 A508215 A508216 A508217 A508218 A508219 A508220;
array A5084L A5084L01 A5084L02 A5084L03 A5084L04 A5084L05 A5084L06 A5084L07 A5084L08 A5084L09
      A5084L10 A5084L11 A5084L12 A5084L13 A5084L14 A5084L15 A5084L16 A5084L17 A5084L18
      A5084L19 A5084L20;
array A5084U A5084U01 A5084U02 A5084U03 A5084U04 A5084U05 A5084U06 A5084U07 A5084U08
      A5084U09 A5084U10 A5084U11 A5084U12 A5084U13 A5084U14 A5084U15 A5084U16 A5084U17
      A5084U18 A5084U19 A5084U20;
array A5086 A508601 A508602 A508603 A508604 A508605 A508606 A508607 A508608 A508609 A508610
      A508611 A508612 A508613 A508614 A508615 A508616 A508617 A508618 A508619 A508620;
array A5170 A517001 A517002 A517003 A517004 A517005 A517006 A517007 A517008 A517009 A517010
      A517011 A517012 A517013 A517014 A517015 A517016 A517017 A517018 A517019 A517020;
array A5172 A517201 A517202 A517203 A517204 A517205 A517206 A517207 A517208 A517209 A517210
      A517211 A517212 A517213 A517214 A517215 A517216 A517217 A517218 A517219 A517220;
array A5174L A5174L01 A5174L02 A5174L03 A5174L04 A5174L05 A5174L06 A5174L07 A5174L08 A5174L09
      A5174L10 A5174L11 A5174L12 A5174L13 A5174L14 A5174L15 A5174L16 A5174L17 A5174L18
      A5174L19 A5174L20;
array A5174U A5174U01 A5174U02 A5174U03 A5174U04 A5174U05 A5174U06 A5174U07 A5174U08
      A5174U09 A5174U10 A5174U11 A5174U12 A5174U13 A5174U14 A5174U15 A5174U16 A5174U17
      A5174U18 A5174U19 A5174U20;
array A5176 A517601 A517602 A517603 A517604 A517605 A517606 A517607 A517608 A517609 A517610
      A517611 A517612 A517613 A517614 A517615 A517616 A517617 A517618 A517619 A517620;
array A5074 A507401-A507420;
array A5066 A506601-A506620;
array A5164 A516401-A516420;

do I=1 to 20;

```

```

rloan(I)=0;
if YAS5040=1 and YAS5060 ne 1 and YAS5062 ne 1 then do;
    if A5080(I)=1 then rloan(I)=A5082(I);
    if A5080(I)=2 then rloan(I)=(A5084L(I)+(A5084U(I)-A5084L(I))/2);
    if ((A5080(I) ne 1 and A5080(I) ne 2) or (A5082(I)=-1 or A5082(I)=-2)) then do;
        if A5086(I)=1 then do; rloan(I)=500; flag=1; end;
        if A5086(I)=2 then do; rloan(I)=1750; flag=1; end;
        if A5086(I)=3 then do; rloan(I)=3750; flag=1; end;
        if A5086(I)=4 then do; rloan(I)=7500; flag=1; end;
        if A5086(I)=5 then do; rloan(I)=17500; flag=1; end;
        if A5086(I)=6 then do; rloan(I)=37500; flag=1; end;
        if A5086(I)=7 then do; rloan(I)=50001; flag=1; end;
    end;
end;

if YAS5040=-1 or A5074(I)=-1 or A5086(I)=-1 or A5084L[I]=-1 or A5084U[I]=-1 then rloan(I)=-1;
if YAS5040=-2 OR A5074(I)=-2 or A5086(I)=-2 or A5084L[I]=-2 or A5084U[I]=-2 then rloan(I)=-2;
if YAS5040=-3 or A5082(I)=-3 or A5086(I)=-3 or A5084L[I]=-3 or A5084U[I]=-3 then rloan(I)=-3;

if YAS5130=1 and YAS5150 ne 1 and YAS5152 ne 1 then do;
    if A5170(I)=1 then rloan(I)=A5172(I);
    if A5170(I)=2 then rloan(I)=(A5174L(I)+(A5174U(I)-A5174L(I))/2);
    if ((A5170(I) ne 1 and A5170(I) ne 2) or (A5172(I)=-1 or A5172(I)=-2)) then do;
        if A5176(I)=1 then do; rloan(I)=500; flag=1; end;
        if A5176(I)=2 then do; rloan(I)=1750; flag=1; end;
        if A5176(I)=3 then do; rloan(I)=3750; flag=1; end;
        if A5176(I)=4 then do; rloan(I)=7500; flag=1; end;
        if A5176(I)=5 then do; rloan(I)=17500; flag=1; end;
        if A5176(I)=6 then do; rloan(I)=37500; flag=1; end;
        if A5176(I)=7 then do; rloan(I)=50001; flag=1; end;
    end;
end;

if YAS5130=-1 or A5164(I)=-1 or A5176(I)=-1 or A5174L[I]=-1 or A5174U[I]=-1 then rloan(I)=-1;
if YAS5130=-2 or A5164(I)=-2 or A5176(I)=-2 or A5174L[I]=-2 or A5174U[I]=-2 then rloan(I)=-2;
if YAS5130=-3 or A5172(I)=-3 or A5176(I)=-3 or A5174L[I]=-3 or A5174U[I]=-3 then rloan(I)=-3;

end;

/* Any other debts from loans, credit cards, etc...*/

othdebtY=0;
if YAS5210=1 then do;
    if YAS5220=1 then othdebtY=YAS5222;
    if YAS5220=2 then othdebtY=(YAS5224L+(YAS5224U-YAS5224L)/2);
    if ((YAS5220 ne 1 and YAS5220 ne 2) or (YAS5222=-1 or YAS5222=-2)) then do;
        if YAS5226=1 then do; othdebtY=500; flag=1; end;
        if YAS5226=2 then do; othdebtY=1750; flag=1; end;
        if YAS5226=3 then do; othdebtY=3750; flag=1; end;
        if YAS5226=4 then do; othdebtY=7500; flag=1; end;
        if YAS5226=5 then do; othdebtY=17500; flag=1; end;
        if YAS5226=6 then do; othdebtY=37500; flag=1; end;
        if YAS5226=7 then do; othdebtY=50001; flag=1; end;
    end;
end;

if YAS5210=-1 or YAS5224L=-1 or YAS5224U=-1 or YAS5226=-1 then othdebtY=-1;

```

```

if YAS5210=-2 or YAS5224L=-2 or YAS5224U=-2 or YAS5226=-2 then othdebtY=-2;
if YAS5210=-3 or YAS5222=-3 or YAS5224L=-3 or YAS5224U=-3 or YAS5226=-3 then othdebtY=-3;

/* We now calculate the household net worth according to the youth: hhworthY=assets-liabilities */

hhworthY=0;
if pvranch not in (-1, -2, -3) then hhworthY=hhworthY+pvranch;
if pvmbst not in (-1, -2, -3) then hhworthY=hhworthY+pvmbst;
if pvmb not in (-1, -2, -3) then hhworthY=hhworthY+pvmb;
if pvst not in (-1, -2, -3) then hhworthY=hhworthY+pvst;
if pvapthm not in (-1, -2, -3) then hhworthY=hhworthY+pvapthm;
if pvbussY not in (-1, -2, -3) then hhworthY=hhworthY+pvbussY;
if secrestY not in (-1, -2, -3) then hhworthY=hhworthY+secrestY;
if retireY not in (-1, -2, -3) then hhworthY=hhworthY+retireY;
if savingsY not in (-1, -2, -3) then hhworthY=hhworthY+savingsY;
if othsavY not in (-1, -2, -3) then hhworthY=hhworthY+othsavY;
if stockY not in (-1, -2, -3) then hhworthY=hhworthY+stockY;
if pvcarsY not in (-1, -2, -3) then hhworthY=hhworthY+pvcarsY;
if pfurnty not in (-1, -2, -3) then hhworthY=hhworthY+pfurnty;
if otassetY not in (-1, -2, -3) then hhworthY=hhworthY+otassetY;
if mortgagY not in (-1, -2, -3) then hhworthY=hhworthY-mortgagY;
if loanowed not in (-1, -2, -3) then hhworthY=hhworthY-loanowed;
if stilowed not in (-1, -2, -3) then hhworthY=hhworthY-stilowed;
if secmortY not in (-1, -2, -3) then hhworthY=hhworthY-secmortY;
if cardebt not in (-1, -2, -3) then hhworthY=hhworthY-cardebt;
if proptaxY not in (-1, -2, -3) then hhworthY=hhworthY-proptaxY;
if othdebtY not in (-1, -2, -3) then hhworthY=hhworthY-othdebtY;
do I=1 to 19;
    if rloan[I] not in (-1, -2, -3) then hhworthY=hhworthY-rloan[I];
end;
end; /* CORRESPONDING TO YAS50=1 */

if YAS50=0 or YAS50=-4 then hhworthY=-4;
if YAS50=-1 or (pvranch=-1 or pvmbast=-1 or pvmbhm=-1 or pvapthm=-1 or pvbussY=-1 or
    secrestY=-1 or retireY=-1 or savingsY=-1 or othsavY=-1 or stockY=-1 or pvcarsY=-1 or pfurnty=-1 or otassetY=-1 or
    mortgagY=-1 or loanowed=-1 or stilowed=-1 or secmortY=-1 or cardebt=-1 or proptaxY=-1 or rloan01=-1 or
    rloan02=-1 or rloan03=-1 or rloan04=-1 or rloan05=-1 or rloan06=-1 or rloan07=-1 or rloan08=-1 or
    rloan09=-1 or rloan10=-1 or rloan11=-1 or rloan12=-1 or rloan13=-1 or rloan14=-1 or rloan15=-1 or rloan16=-1 or
    rloan17=-1 or rloan18=-1 or rloan19=-1 or othdebtY=-1)
then hhworthY=-1;
if YAS50=-2 or (pvranch=-2 or pvmbast=-2 or pvmbhm=-2 or pvapthm=-2 or pvbussY=-2 or
    secrestY=-2 or retireY=-2 or savingsY=-2 or othsavY=-2 or stockY=-2 or pvcarsY=-2 or pfurnty=-2 or otassetY=-2 or
    mortgagY=-2 or loanowed=-2 or stilowed=-2 or secmortY=-2 or cardebt=-2 or proptaxY=-2 or rloan01=-2 or
    rloan02=-2 or rloan03=-2 or rloan04=-2 or rloan05=-2 or rloan06=-2 or rloan07=-2 or rloan08=-2 or
    rloan09=-2 or rloan10=-2 or rloan11=-2 or rloan12=-2 or rloan13=-2 or rloan14=-2 or rloan15=-2 or rloan16=-2 or
    rloan17=-2 or rloan18=-2 or rloan19=-2 or othdebtY=-2)
then hhworthY=-2;
if YAS50=-3 or (pvranch=-3 or pvmbast=-3 or pvmbhm=-3 or pvapthm=-3 or pvbussY=-3 or
    secrestY=-3 or retireY=-3 or savingsY=-3 or othsavY=-3 or stockY=-3 or pvcarsY=-3 or pfurnty=-3 or otassetY=-3 or
    mortgagY=-3 or loanowed=-3 or stilowed=-3 or secmortY=-3 or cardebt=-3 or proptaxY=-3 or rloan01=-3 or
    rloan02=-3 or rloan03=-3 or rloan04=-3 or rloan05=-3 or rloan06=-3 or rloan07=-3 or rloan08=-3 or
    rloan09=-3 or rloan10=-3 or rloan11=-3 or rloan12=-3 or rloan13=-3 or rloan14=-3 or rloan15=-3 or rloan16=-3 or
    rloan17=-3 or rloan18=-3 or rloan19=-3 or othdebtY=-3)
then hhworthY=-3;
if YAS50=-5 then hhworthY=-5; /*give the people who were not interviewed in round 2 value -5.*/

```

***** SECTION 3: HOUSEHOLD POVERTY STATUS *****

```

povert=-4;
if HHSIZE=-1 or under18=-1 then povert=-1;
if HHSIZE=-2 or under18=-2 then povert=-2;
if HHSIZE=-3 or under18=-3 then povert=-3;

if HHSIZE=1 then povert=8350;

if HHSIZE=2 then do;
    if under18=0 then povert=10748;
    if under18=1 then povert=11063;
end;

if HHSIZE=3 then do;
    if under18=0 then povert=12554;
    if under18=1 then povert=12919;
    if under18=2 then povert=12931;
end;

if HHSIZE=4 then do;
    if under18=0 then povert=16555;
    if under18=1 then povert=16825;
    if under18=2 then povert=16276;
    if under18=3 then povert=16333;
end;

if HHSIZE=5 then do;
    if under18=0 then povert=19964;
    if under18=1 then povert=20255;
    if under18=2 then povert=19634;
    if under18=3 then povert=19154;
    if under18=4 then povert=18861;
end;

if HHSIZE=6 then do;
    if under18=0 then povert=22962;
    if under18=1 then povert=23053;
    if under18=2 then povert=22578;
    if under18=3 then povert=22123;
    if under18=4 then povert=21446;
    if under18=5 then povert=21045;
end;

if HHSIZE=7 then do;
    if under18=0 then povert=26421;
    if under18=1 then povert=26586;
    if under18=2 then povert=26017;
    if under18=3 then povert=25621;
    if under18=4 then povert=24882;
    if under18=5 then povert=24021;
    if under18=6 then povert=23076;
end;

```

| | |
|---|---|
| <pre> if HHSIZE=8 then do; if under18=0 then povert=29550; if under18=1 then povert=29811; if under18=2 then povert=29274; if under18=3 then povert=28804; if under18=4 then povert=28137; if under18=5 then povert=27290; if under18=6 then povert=26409; if under18=7 then povert=26185; end; if HHSIZE>=9 then do; if under18=0 then povert=35546; if under18=1 then povert=35719; if under18=2 then povert=35244; if under18=3 then povert=34845; if under18=4 then povert=34190; if under18=5 then povert=33289; if under18=6 then povert=32474; if under18=7 then povert=32272; if under18=8 then povert=31029; end; </pre> | <pre> povthr=-4; if groshhIY ge 0 and povert gt 0 then povthr=groshhIY/povert; if groshhIY eq -1 or povert=-1 then povthr=-1; if groshhIY eq -2 or povert=-2 then povthr=-2; if groshhIY eq -3 or povert=-3 then povthr=-3; if groshhIY eq -4 then povthr=-4; povthr2=-4; if povthr ge 0 then do; povthr2=povthr*100; end; if povthr eq -1 then povthr2=-1; if povthr eq -2 then povthr2=-2; if povthr eq -3 then povthr2=-3; povthr3=-4; if povthr2 ge 0 then do; povthr3=round(povthr2, 1); end; if povthr2 eq -1 then povthr3=-1; if povthr2 eq -2 then povthr3=-2; if povthr2 eq -3 then povthr3=-3; /* Respondents not interviewed in round 2 */ if YAS50=-5 then povthr3=-5; endsas; </pre> |
|---|---|

PARTICIPATION IN GOVERNMENT PROGRAMS

Variables Created: CV_AMT_GOVNT_PGM_PCY.80 – CV_AMT_GOVNT_PGM_PCY.99
 CV_GOVNT_PGM_EVER
 CV_GOVNT_PGM_YR.80 – CV_GOVNT_PGM_YR.99

Variables Used

| Name in Program | Question Name on CD | Name in Program | Question Name on CD |
|--------------------|------------------------|----------------------|-----------------------------------|
| PUBID | PUBID | BDATE_D,_M,_Y | KEY!BDATE_D,_M,_Y (rd. 1) |
| P1210 | YPRG-1210 | LINT_D,LINT_M,LINT_Y | CV_INTERVIEW_DATE_D,_M,_Y (rd. 1) |
| P1895 | YPRG-1895_R2 | PS4001M, PS4001Y | YPRG-4000_R2.01~M, ~Y |
| P1905 | YPRG-1905_R2 | PS47001M, PS47001Y | YPRG-4700_R2.01~M, ~Y |
| P1920 | YPRG-1920_R2 | PS10201M, PS10201Y | YPRG-10200_R2.01~M, ~Y |
| P1930 | YPRG-1930_R2 | PS10901M, PS10901Y | YPRG-10900_R2.01~M, ~Y |
| P39011 | YPRG-3901_R2.01 | P16390M, P16390Y | YPRG-16390_R2~M, ~Y |
| PS42001 | YPRG-4200_R2.01 | P191001M, P191001Y | YPRG-19100_R2.01~M, ~Y |
| PS44001 | YPRG-4400_R2.01 | P191002M, P191002Y | YPRG-19100_R2.02~M, ~Y |
| PS49001 | YPRG-4900_R2.01 | PS20001M, PS20001Y | YPRG-20000_R2.01~M, ~Y |
| PS59001 | YPRG-5900_R2.01 | PS20002M, PS20002Y | YPRG-20000_R2.02~M, ~Y |
| PS61001 | YPRG-6100_R2.01 | PS209011-PS209015 | YPRG-20900_R2.01~000001 – ~000005 |
| P6200B1 | YPRG-6200B_R2.01 | P21410M, P21410Y | YPRG-21410_R2~M, ~Y |
| PS63001 | YPRG-6300_R2.01 | PS22001M, PS22001Y | YPRG-22000_R2.01~M, ~Y |
| P9865 | YPRG-9865_R2 | PS22002M, PS22002Y | YPRG-22000_R2.02~M, ~Y |
| P9867 | YPRG-9867_R2 | PS22601M, PS22601Y | YPRG-22600_R2.01~M, ~Y |
| P9890 | YPRG-9890_R2 | PS22602M, PS22602Y | YPRG-22600_R2.02~M, ~Y |
| P9900 | YPRG-9900_R2 | PS235011-PS235015 | YPRG-23500_R2.01~000001 – ~000005 |
| P10100A1 | YPRG-10100A_R2.01 | PS235199 | YPRG-23500_R2.01~000099 |
| PS104001 | YPRG-10400_R2.01 | PS235021-PS235025 | YPRG-23500_R2.02~000001 – ~000005 |
| PS106001 | YPRG-10600_R2.01 | P235299 | YPRG-23500_R2.02~000099 |
| PS111001 | YPRG-11100_R2.01 | P235031-P235035 | YPRG-23500_R2.03~000001 – ~000005 |
| PS121001 | YPRG-12100_R2.01 | P2350399 | YPRG-23500_R2.03~000099 |
| P12500A1 | YPRG-12500A_R2.01 | P358001M, P358001Y | YPRG-35800_R2.01~M, ~Y |
| PS126001 | YPRG-12600_R2.01 | P358002M, P358002Y | YPRG-35800_R2.02~M, ~Y |
| P16385 | YPRG-16385_R2 | P358003M, P358003Y | YPRG-35800_R2.03~M, ~Y |
| P16400 | YPRG-16400_R2 | P358401M, P358401Y | YPRG-35840_R2.01~M, ~Y |
| P16410 | YPRG-16410_R2 | P358402M, P358402Y | YPRG-35840_R2.02~M, ~Y |
| P19063A1, P19063A2 | YPRG-19063A_R2.01, .02 | P3592011-P3592015 | YPRG-35920_R2.01~000001 – ~000005 |
| PS202001 | YPRG-20200_R2.01 | P3592021-P3592025 | YPRG-35920_R2.02~000001 – ~000005 |
| PS207001, PS207002 | YPRG-20700_R2.01 | P3592031-P3592035 | YPRG-35920_R2.03~000001 – ~000005 |
| PS208001 | YPRG-20800_R2.01 | P36100M, P36100Y | YPRG-36100_R2~M, ~Y |
| P21100A1, P21100A2 | YPRG-21100A_R2.01, .02 | PS31001M, PS31001Y | YPRG-31000_R2.01~M, ~Y |
| P21397 | YPRG-21397_R2 | PS31002M, PS31002Y | YPRG-31000_R2.02~M, ~Y |
| P21405 | YPRG-21405_R2 | PS31601M, PS31601Y | YPRG-31600_R2.01~M, ~Y |
| P21420 | YPRG-21420_R2 | PS31602M, PS31602Y | YPRG-31600_R2.02~M, ~Y |
| P21430 | YPRG-21430_R2 | PS325011-PS325015 | YPRG-32500_R2.01~000001 – ~000005 |
| P21900A1-P21900A3 | YPRG-21900A_R2.01-.03 | PS325021-PS325025 | YPRG-32500_R2.02~000001 – ~000005 |
| PS228001, PS228002 | YPRG-22800_R2.01, .02 | P40001M, P40001Y | YPRG-4000.01~M, ~Y |
| PS233001-PS233003 | YPRG-23300_R2.01-.03 | P40002M, P40002Y | YPRG-4000.02~M, ~Y |
| PS234001, PS234002 | YPRG-23400_R2.01, .02 | P40003M, P40003Y | YPRG-4000.03~M, ~Y |
| P23700A1-P23700A3 | YPRG-23700A_R2.01-.03 | P47001M, P47001Y | YPRG-4700.01~M, ~Y |
| P35715 | YPRG-35715_R2 | P47002M, P47002Y | YPRG-4700.02~M, ~Y |
| P35730 | YPRG-35730_R2 | P47003M, P47003Y | YPRG-4700.03~M, ~Y |
| P35740 | YPRG-35740_R2 | P102001M, P102001Y | YPRG-10200.01~M, ~Y |
| P357901-P357903 | YPRG-35790_R2.01-.03 | P109001M, P109001Y | YPRG-10900.01~M, ~Y |
| P359001-P359003 | YPRG-35900_R2.01-.03 | P194001M, P194001Y | YPRG-19400.01~M, ~Y |
| P359101 | YPRG-35910_R2.01 | P194002M, P194002Y | YPRG-19400.02~M, ~Y |
| P359501-P359503 | YPRG-35950_R2.01-.03 | P194003M, P194003Y | YPRG-19400.03~M, ~Y |
| P36087 | YPRG-36087_R2 | P200001M, P200001Y | YPRG-20000.01~M, ~Y |
| P36095 | YPRG-36095_R2 | P200002M, P200002Y | YPRG-20000.02~M, ~Y |
| P36110 | YPRG-36110_R2 | P2090011-P2090015 | YPRG-20900.01~000001 – ~000005 |
| P36120 | YPRG-36120_R2 | P2090021-P2090025 | YPRG-20900.02~000001 – ~000005 |

Appendix 5: Income and Assets Variable Creation

| | | | |
|--------------------|------------------------|------------------------|--------------------------------|
| P30900A1, P30900A2 | YPRG-30900A_R2.01, .02 | P2090031-P2090035 | YPRG-20900.03~000001 - ~000005 |
| PS318001 | YPRG-31800_R2.01 | P220001M, P220001Y | YPRG-22000.01~M, ~Y |
| PS323001, PS323002 | YPRG-32300_R2.01, .02 | P220002M, P220002Y | YPRG-22000.02~M, ~Y |
| PS324001 | YPRG-32400_R2.01 | P226001M, P226001Y | YPRG-22600.01~M, ~Y |
| P32700A1 P32700A2 | YPRG-32700A_R2.01, .02 | P2350011-P2350015 | YPRG-23500.01~000001 - ~000005 |
| P1900 | YPRG-1900 | P2350021 P2350025 | YPRG-23500.02~000001 - ~000005 |
| P2000 | YPRG-2000 | P167001M, P167001Y | YPRG-16700.01~M, ~Y |
| P2500 | YPRG-2500 | P172001M, P172001Y | YPRG-17200.01~M, ~Y |
| P2600 | YPRG-2600 | P1830011-P1830015 | YPRG-18300.01~000001 - ~000005 |
| P2400 | YPRG-2400 | P310001M, P310001Y | YPRG-31000.01~M, ~Y |
| P3300 | YPRG-3300 | P310002M, P310002Y | YPRG-31000.02~M, ~Y |
| P3400 | YPRG-3400 | P316001M, P316001Y | YPRG-31600.01~M, ~Y |
| P3500 | YPRG-3500 | P3250011-P3250015 | YPRG-32500.01~000001 - ~000005 |
| P42001-P42003 | YPRG-4200.01-.03 | P3250021-P3250025 | YPRG-32500.02~000001 - ~000005 |
| P44001-P44003 | YPRG-4400.01-.03 | YINT_D, YINT_M, YINT_Y | YINTDATE~D, ~M, ~Y |
| P45001, P45002 | YPRG-4500.01, .02 | P18900 | YPRG-18900 |
| P48001-P48003 | YPRG-4800.01-.03 | P196001-P196003 | YPRG-19600.01-.03 |
| P49001, P49002 | YPRG-4900.01, .02 | P201001-P201003 | YPRG-20100.01-.03 |
| P52001-P52003 | YPRG-5200.01-.03 | P202001, P202002 | YPRG-20200.01, .02 |
| P56001-P56003 | YPRG-5600.01-.03 | P207001-P207003 | YPRG-20700.01-.03 |
| P59001-P59003 | YPRG-5900.01-.03 | P208001 | YPRG-20800.01 |
| P61001-P61003 | YPRG-6100.01-.03 | P21500 | YPRG-21500 |
| P63001, P63002 | YPRG-6300.01, .02 | P222001, P222002 | YPRG-22200.01, .02 |
| P9550EC | YPRG-9550-ELIG-CHECK | P228001 | YPRG-22800.01 |
| P9600 | YPRG-9600 | P233001, P233002 | YPRG-23300.01, .02 |
| P9700 | YPRG-9700 | P234001 | YPRG-23400.01 |
| P103001 | YPRG-10300.01 | P23900EC | YPRG-23900-ELIG-CHECK |
| P104001 | YPRG-10400.01 | P16200 | YPRG-16200 |
| P106001 | YPRG-10600.01 | P169001 | YPRG-16900.01 |
| P110001 | YPRG-11000.01 | P175001 | YPRG-17500.01 |
| P111001 | YPRG-11100.01 | P181001 | YPRG-18100.01 |
| P114001 | YPRG-11400.01 | P182001 | YPRG-18200.01 |
| P116001 | YPRG-11600.01 | P30500 | YPRG-30500 |
| P117001 | YPRG-11700.01 | P312001, P312002 | YPRG-31200.01, .02 |
| P118001 | YPRG-11800.01 | P318001 | YPRG-31800.01 |
| P121001 | YPRG-12100.01 | P323001, P323002 | YPRG-32300.01, .02 |
| P122001 | YPRG-12200.01 | P324001 | YPRG-32400.01 |
| P126001 | YPRG-12600.01 | | |

This program creates several variables describing the respondent's participation in government programs for the economically disadvantaged. During the interview, respondents report amounts received and months of participation in Aid to Families with Dependent Children (AFDC); food stamps; and Women, Infants, and Children (WIC). There is also an "other assistance" question to capture information about any other government program from which respondents may have received assistance. The interview also records amounts received from unemployment compensation and worker's compensation in separate question series; this program also creates variables describing respondents' participation in unemployment or worker's compensation.

The program to create these variables first creates a month-by-month participation array for each of the six categories (AFDC, food stamps, WIC, other programs, unemployment compensation, and worker's compensation). These month-by-month variables constitute part of the event history array for program participation; see appendix 7 for more information. After all six arrays are created, the program merges data from the six categories to create the summary variables.

```

cm14=((bdate_y+14)-1980)*12+bdate_m;
cmb=((bdate_y)-1980)*12+bdate_m;
ym14=(BDATE_Y*100+BDATE_M)+1400;
iym=YINT_Y*100+YINT_M;

if p1210=1 then dliym=R0000202*100+(R0000201+1) and dlicm=(R0000202-1980)*12+(R0000201+1);
if p1210 ne 1 then dliym=ym14 and dlicm=cm14;

```

```

doicm=(YINT_Y-1980)*12+YINT_M;
aaicm=(YINT_Y-BDATE_Y)*12+YINT_M-BDATE_M;

array wc      (l) wc001-wc232;
array ui      (l) ui001-ui232;
array uiAMT   (l) uiamt001-uiamt232;
array wcAMT   (l) wcamt001-wcamt232;
array ahhm    (l) ahhm001- ahhm232;
array aamt    (l) aamt001- aamt232;
array a       (l) a001- a232;
array whhm   (l) whhm001- whhm232;
array wamt   (l) wamt001- wamt232;
array w       (l) w001-w232;
array fhdm   (l) fhdm001- fhdm232;
array famt   (l) famt001- famt232;
array f       (l) f001- f232;
array ohhm   (l) ohhm001- ohhm232;
array oamt   (l) oamt001- oamt232;
array o       (l) o001- o232;

if P1210>-4 then do;
  if P21397>-3 then P21500=P21397;
  if P36087>-3 then P30500=P36087;
  if P9867>-3 then P9900=P9867;

  /*2. begin - Rs eligible for a program in round 2*/
  do l=1 to 232;
    if cmb le L le doicm then do;
      a=0; w=0; f=0; o=0; wc=0; ui=0; end;
    end;

  /** start YEAR information */
  array ysa1 (j) P358001Y P358002Y P358003Y;      /**afdc, sdli**/
  array ysa2 (j) P167001Y ysa22  ysa23;          /**afdc, int**/
  array ysw1  (j) PS22001Y PS22002Y ysw13;        /**wic, sdli**/
  array ysw2  (j) P220001Y P220002Y ysw23;        /**wic, int**/
  array ysf1   (j) P191001Y P191002Y ysf13;        /**food stamps, sdli**/
  array ysf2   (j) P194001Y P194002Y P194003Y;    /**food stamps, int**/
  array yso1   (j) PS31001Y PS31002Y yso13;        /**other, sdli**/
  array yso2   (j) P310001Y P310002Y yso23;        /**other, int**/
  array ysu1   (j) PS40001Y ysu12 ysu13;          /**unemployment, sdli**/
  array ysu2   (j) P40001Y P40002Y P40003Y;        /**unemployment, int**/
  array ysc1   (j) PS10201Y ysc12 ysc13;          /**workers comp, sdil**/
  array ysc2   (j) P102001Y ysc22 ysc23;          /**workers comp, int**/

  /** start MONTH information */
  array msa1  (j) P358001M P358002M P358003M;    /**afdc, sdli**/
  array msa2  (j) P167001M msa22  msa23;          /**afdc, int**/
  array msw1  (j) PS22001M PS22002M msw13;        /**wic, sdli**/
  array msw2  (j) P220001M P220002M msw23;        /**wic, int**/
  array msf1   (j) P191001M P191002M msf13;        /**food stamps, sdli**/
  array msf2   (j) P194001M P194002M P194003M;    /**food stamps, int**/
  array mso1   (j) PS31001M PS31002M mso13;        /**other, sdli**/
  array mso2   (j) P310001M P310002M mso23;        /**other, int**/
  array msu1   (j) PS40001M msu12 msu13;          /**unemployment, sdli**/
  array msu2   (j) P40001M P40002M P40003M;        /**unemployment, int**/
  array msc1   (j) PS10201M msc12 msc13;          /**workers comp, sdil**/

```

```

array msc2 (j) P102001M msc22 msc23;           /**workers comp, int**/

array ymsa1 (j) yms01-yms03;
array ymsa2 (j) yms04 ymsa22 ymsa23;
array ymsw1 (j) yms05-yms07;
array ymsw2 (j) yms08 yms09 ymsw23;
array ymsf1 (j) yms10 yms11 ymsf13;
array ymsf2 (j) yms12-yms14;
array ymso1 (j) yms15 yms16 ymso13;
array ymso2 (j) yms17 yms18 ymso23;
array ymsu1 (j) ymsu11-ymsu13;
array ymsu2 (j) ymsu21-ymsu23;
array ymsc1 (j) ymsc11-ymsc13;
array ymsc2 (j) ymsc21-ymsc23;

array csma1 (j) csm01-csm03;
array csma2 (j) csm04 csma22 csma23;
array csmw1 (j) csm05-csm07;
array csmw2 (j) csm08 csm09 csmw23;
array csmf1 (j) csm10 csm11 csmf13;
array csmf2 (j) csm12-csm14;
array csmo1 (j) csm15 csm16 csmo13;
array csmo2 (j) csm17 csm18 csmo23;
array csmu1 (j) csmu11-csmu13;
array csmu2 (j) csmu21-csmu23;
array csmc1 (j) csmc11-csmc13;
array csmc2 (j) csmc21-csmc23;

/** end YEAR information */
array yea1 (j) P358401Y P358402Y yea13;          /**afdc, sdli**/
array yea2 (j) P172001Y yea22  yea23;            /**afdc, int**/
array yew1 (j) PS22601Y P226002Y yew13;          /**wic, sdli**/
array yew2 (j) P226001Y yew22  yew23;            /**wic, int**/
array yef1 (j) PS20001Y PS20002Y yef13;          /**food stamps, sdli**/
array yef2 (j) P200001Y P200002Y yef23;          /**food stamps, int**/
array yeo1 (j) PS31601Y PS31602Y yeo13;          /**other, sdli**/
array yeo2 (j) P316001Y yeo22  yeo23;            /**other, int**/
array yeu1 (j) PS47001Y yeu12  yeu13;             /**unemployment, sdli**/
array yeu2 (j) P47001Y P47002Y P47003Y;          /**unemployment, int**/
array yec1 (j) PS10901Y yec12  yec13;             /**workers comp, sdil**/
array yec2 (j) P109001Y yec22  yec23;             /**workers comp, int**/

/** end MONTH information */
array meal1 (j) P358401M P358402M mea13;          /**afdc, sdli**/
array mea2 (j) P172001M mea22  mea23;            /**afdc, int**/
array mew1 (j) PS22601M P226002M mew13;          /**wic, sdli**/
array mew2 (j) P226001M mew22  mew23;            /**wic, int**/
array mef1 (j) PS20001M PS20002M mef13;          /**food stamps, sdli**/
array mef2 (j) P200001M P200002M mef23;          /**food stamps, int**/
array meo1 (j) PS31601M PS31602M meo13;          /**other, sdli**/
array meo2 (j) P316001M meo22  meo23;            /**other, int**/
array meu1 (j) PS47001M meu13;                     /**unemployment, sdli**/
array meu2 (j) P47001M P47002M P47003M;          /**unemployment, int**/
array mec1 (j) PS10901M mec12  mec13;            /**workers comp, sdil**/
array mec2 (j) P109001M mec22  mec23;             /**workers comp, int**/

array ymeal1 (j) yme01-yme03;

```

```

array ymea2 (j) yme04 ymea22 ymea23;
array ymew1 (j) yme05-yme07;
array ymew2 (j) yme08 yme09 ymew23;
array ymef1 (j) yme10 yme11 ymef13;
array ymef2 (j) yme12-yme14;
array ymeo1 (j) yme15 yme16 ymeo13;
array ymeo2 (j) yme17 yme18 ymeo23;
array ymeu1 (j) ymeu11-ymeu13;
array ymeu2 (j) ymeu21-ymeu23;
array ymec1 (j) ymec11-ymec13;
array ymec2 (j) ymec21-ymec23;

array cema1 (j) cem01-cem03;
array cema2 (j) cem04 cema22 cema23;
array cemw1 (j) cem05-cem07;
array cemw2 (j) cem08 cem09 cemw23;
array cemf1 (j) cem10 cem11 cemf13;
array cemf2 (j) cem12-cem14;
array cemo1 (j) cem15 cem16 cemo13;
array cemo2 (j) cem17 cem18 cemo23;
array cemu1 (j) cemu11-cemu13;
array cemu2 (j) cemu21-cemu23;
array cemc1 (j) cemc11-cemc13;
array cemc2 (j) cemc21-cemc23;

/** CURRENTLY receiving information */
array cura1 (j) P359501 P359502 P359503;      /**afdc, no gap sdli*/
array cura2 (j) P169001 cura22 cura23;          /**afdc, int*/
array curw1 (j) P23700A1 P23700A2 P23700A3;    /**wic, no gap sdli*/
array curw2 (j) P222001 P222002 curw23;         /**wic, int*/
array curf1 (j) P21100A1 P21100A2 curf13;        /**food stamps, no gap sdli*/
array curf2 (j) P196001 P196002 P196003;         /**food stamps, int*/
array curo1 (j) P32700A1 P32700A2 curo13;       /**other, no gap sdli*/
array curo2 (j) P312001 P312002 curo23;         /**other, int*/
array curu1 (j) PS44001 curu12 curu13;           /**unemployment, sdli*/
array curu2 (j) P44001 P44002 P44003;            /**unemployment, int*/
array curc1 (j) PS106001 curc12 curc13;          /**workers comp, sdil*/
array curc2 (j) P106001 curc22 curc23;           /**workers comp, int*/

/** EDIT FLAGS for dates */
array eflaga1 (j) eflag01-eflag03;                /**afdc, sdli*/
array eflaga2 (j) eflag04 eflaga22 eflaga23;       /**afdc, int*/
array eflgw1 (j) eflag05-eflag07;                  /**wic, sdli*/
array eflgw2 (j) eflag08 eflag09 eflagw23;         /**wic, int*/
array eflagf1 (j) eflag10 eflag11 eflagf13;        /**food stamps, sdli*/
array eflagf2 (j) eflag12-eflag14;                 /**food stamps, int*/
array eflago1 (j) eflag15 eflag16 eflago13;        /**other, sdli*/
array eflago2 (j) eflag17 eflag18 eflago23;        /**other, int*/
array eflagu1 (j) eflagu11-eflagu13;              /**unemployment, sdli*/
array eflagu2 (j) eflagu21-eflagu23;              /**unemployment, int*/
array eflagc1 (j) eflagc11-eflagc13;              /**workers comp, sdil*/
array eflagc2 (j) eflagc21-eflagc23;              /**workers comp, int*/

/** EDIT FLAGS for amounts */
array aflga1 (j) aflag01-aflag03;                 /**afdc, sdli*/
array aflga2 (j) aflag04 aflga22 aflaga23;       /**afdc, int*/
array aflgw1 (j) aflag05-aflag07;                 /**wic, sdli*/

```

```

array aflagw2 (j) aflag08 aflag09 aflagw23;          /**wic, int**/
array aflagf1 (j) aflag10 aflag11 aflagf13;          /**food stamps, sdli**/
array aflagf2 (j) aflag12-aflag14;                  /**food stamps, int**/
array aflago1 (j) aflag15 aflag16 aflago13;          /**other, sdli**/
array aflago2 (j) aflag17 aflag18 aflago23;          /**other, int**/
array aflagu1 (j) aflagu11-aflagu13;                /**unemployment, sdli**/
array aflagu2 (j) aflagu21-aflagu23;                /**unemployment, int**/
array aflagc1 (j) aflagc11-aflagc13;                /**workers comp, sdil**/
array aflagc2 (j) aflagc21-aflagc23;                /**workers comp, int**/

/** AMOUNT RECEIVED - actual & estimated */
array inca1 (j) P359001 P359002 P359003;          /**afdc, amount sdli**/
array einca1 (j) P359101 einca12 einca13;          /**afdc, amount int**/
array inca2 (j) P181001 inca22 inca23;            /**afdc, est amount sdli**/
array einca2 (j) P182001 einca22 einca23;          /**afdc, est amount int**/
array incw1 (j) PS233001 PS233002 PS233003;        /**wic, amount sdli**/
array eincw1 (j) PS234001 PS234002 eincw13;        /**wic, amount int**/
array incw2 (j) P233001 P233002 incw23;            /**wic, est amount sdli**/
array eincw2 (j) P234001 eincw22 eincw23;          /**wic, est amount int**/
array incf1 (j) PS207001 PS207002 incf13;          /**food stamps, amount sdli**/
array eincf1 (j) PS208001 eincf12 eincf13;          /**food stamps, amount int**/
array incf2 (j) P207001 P207002 P207003;          /**food stamps, est amount sdli**/
array eincf2 (j) P208001 eincf22 eincf23;          /**food stamps, est amount int**/
array inco1 (j) PS323001 PS323002 inco13;          /**other, amount sdli**/
array einco1 (j) PS324001 einco12 einco13;          /**other, amount int**/
array inco2 (j) P323001 P323002 inco23;            /**other, est amount sdli**/
array einco2 (j) P324001 einco22 einco23;          /**other, est amount int**/

array wincu1 (j) PS59001 incu12 incu13;
array wincu2 (j) P59001 P59002 P59003;
array wincc1 (j) PS121001 incc12 incc13;
array wincc2 (j) P121001 incc22 incc23;
array eincc2 (j) P122001 eincc22 eincc23;

array wksa1 (j) wksa11-wksa13;
array wksa2 (j) P175001 wksa22 wksa23;
array wksw1 (j) PS228001 PS228002 wksw13;
array wksw2 (j) P228001 wksw22 wksw23;
array wksf1 (j) PS202001 wksf12 wksf13;
array wksf2 (j) P202001 P202002 wksf23;
array wkso1 (j) PS318001 wkso12 wkso13;
array wkso2 (j) P318001 wkso22 wkso23;
array wksu1 (j) wksu11-wksu13;
array wksu2 (j) P52001 wksu22 P52003;
array wksc1 (j) wksc11-wksc13;
array wksc2 (j) P114001 wksc22 wksc23;

array mosa1 (j) mosa11-mosa13;
array mosa2 (j) mosa21-mosa23;
array mosw1 (j) mosw11-mosw13;
array mosw2 (j) mosw21-mosw23;
array mosf1 (j) mosf11-mosf13;
array mosf2 (j) mosf21-mosf23;
array moso1 (j) moso11-moso13;
array moso2 (j) moso21-moso23;
array mosu1 (j) mosu11-mosu13;
array mosu2 (j) mosu21-mosu23;
array mosc1 (j) mosc11-mosc13;

```

```

array mosc2 (j) mosc21-mosc23;
array rcvra1 (j) P3592011 P3592021 P3592031;      /**afdc, sdli**/
array rcvra2 (j) P1830011 rcvra22 rcvra23;        /**afdc, int**/
array rcrvw1 (j) PS235011 PS235021 PS235031;      /**wic, sdli**/
array rcrvw2 (j) P2350011 P2350021 rcrvw23;        /**wic, int**/
array rcvrf1 (j) rcvrf11-rcvrf13;                  /**food stamps, sdli**/
array rcvrf2 (j) P2090011 P2090021 P2090021;      /**food stamps, int**/
array rcvro1 (j) PS325011 PS325021 rcvro13;       /**other, sdli**/
array rcvro2 (j) P3250011 P3250021 rcvro23;       /**other, int**/

/** PERSON receiving amount, spouse **/
array rcvsal (j) P3592012 P3592022 P3592032;      /**afdc, sdli**/
array rcvsal (j) P1830012 rcvsal22 rcvsal23;       /**afdc, int**/
array rcvsw1 (j) PS235012 PS235022 PS235032;      /**wic, sdli**/
array rcvsw2 (j) P2350012 P2350022 rcvsw23;       /**wic, int**/
array rcvsf1 (j) rcvsf11-rcvsf13;                  /**food stamps, sdli**/
array rcvsf2 (j) P2090012 P2090022 P2090022;      /**food stamps, int**/
array rcvso1 (j) PS325012 PS325022 rcvso13;       /**other, sdli**/
array rcvso2 (j) P3250012 P3250022 rcvso23;       /**other, int**/

/** PERSON receiving amount, child **/
array rcvcal (j) P3592013 P3592023 P3592033;      /**afdc, sdli**/
array rcvcal (j) P1830013 rcvcal22 rcvcal23;       /**afdc, int**/
array rcvcw1 (j) PS235013 PS235023 PS235033;      /**wic, sdli**/
array rcvcw2 (j) P2350013 P2350023 rcvcw23;       /**wic, int**/
array rcvcf1 (j) rcvcf11-rcvcf13;                  /**food stamps, sdli**/
array rcvcf2 (j) P2090013 P2090023 P2090023;      /**food stamps, int**/
array rcvc01 (j) PS325013 PS325023 rcvc013;       /**other, sdli**/
array rcvc02 (j) P3250013 P3250023 rcvc023;       /**other, int**/

/** PERSON receiving amount family member **/
array rcvfa1 (j) P3592014 P3592024 P3592034;      /**afdc, sdli**/
array rcvfa2 (j) P1830014 rcvfa22 rcvfa23;        /**afdc, int**/
array rcvfw1 (j) PS235014 PS235024 PS235034;      /**wic, sdli**/
array rcvfw2 (j) P2350014 P2350024 rcvfw23;       /**wic, int**/
array rcvff1 (j) rcvff11-rcvff13;                  /**food stamps, sdli**/
array rcvff2 (j) P2090014 P2090024 P2090024;      /**food stamps, int**/
array rcvfo1 (j) PS325014 PS325024 rcvfo13;       /**other, sdli**/
array rcvfo2 (j) P3250014 P3250024 rcvfo23;       /**other, int**/

/** PERSON receiving amount, other person **/
array rcvoa1 (j) P3592015 P3592025 P3592035;      /**afdc, sdli**/
array rcvoa2 (j) P1830015 rcvoa22 rcvoa23;        /**afdc, int**/
array rcvw1 (j) PS235015 PS235025 PS235035;      /**wic, sdli**/
array rcvw2 (j) P2350015 P2350025 rcvw23;        /**wic, int**/
array rcvof1 (j) rcvof11-rcvof13;                  /**food stamps, sdli**/
array rcvof2 (j) P2090015 P2090025 P2090025;      /**food stamps, int**/
array rcvo01 (j) PS325015 PS325025 rcvo013;       /**other, sdli**/
array rcvo02 (j) P3250015 P3250025 rcvo023;       /**other, int**/

array whoa1 (j) who01-who03;
array whoa2 (j) who04 whoa22 whoa23;
array whow1 (j) who05-who07;
array whow2 (j) who08 who09 whow23;
array whof1 (j) who10 who11 whof13;

```

```

array whof2 (j) who12-who14;
array whoo1 (j) who15 who16 whoo13;
array whoo2 (j) who17 who18 whoo23;

array dlia1 (j) P357901 dlia12 dlia13;
array dliw1 (j) P21900A1 dliw12 dliw13;
array dlif1 (j) P19063A1 dlif12 dlif13;
array dlio1 (j) P30900A1 dlio12 dlio13;

/** these arrays encompass all programs for the arrays */
array ys (k) ysa1 ysa2 ysw1 ysw2 ysf1 yso1 yso2 ysu1 ysu2 ysc1 ysc2;
array ms (k) msa1 msa2 msw1 msw2 msf1 msf2 mso1 mso2 msu1 msu2 msc1 msc2;
array ye (k) yea1 yea2 yew1 yew2 yef1 yef2 yeo1 yeo2 yeu1 yeu2 yec1 yec2;
array me (k) meal mea2 mew1 mew2 mef1 mef2 meo1 meo2 meu1 meu2 mec1 mec2;
array cur (k) cura1 cura2 curw1 curw2 curf1 curf2 curo1 curo2 curu1 curu2 curc1 curc2;
array eflag (k) eflaga1 eflaga2 eflagw1 eflagw2 eflagf1 eflagf2 eflago1 eflago2 eflagu1 eflagu2 eflagc1 eflagc2;
array aflag (k) aflaga1 aflaga2 aflagw1 aflagw2 aflagf1 aflagf2 aflago1 aflago2 aflagu1 aflagu2 aflagc1 aflagc2;
array yms (k) ymsa1 ymsa2 ymsw1 ymsw2 ymsf1 ymsf2 yms01 yms02 ymsu1 ymsu2 ymsc1 ymsc2;
array wks (k) wksa1 wksa2 wksw1 wksw2 wksf1 wksf2 wkso1 wkso2 wksu1 wksu2 wksc1 wksc2;
array cwks (k) P35740 P16200 P21430 P21500 P16410 P18900 P36120 P30500 P1930 P3500 P9900 P9700;
array yme (k) ymeal ymea2 ymew1 ymew2 ymef1 ymef2 ymeo1 ymeo2 ymeu1 ymeu2 ymec1 ymec2;
array mos (k) mos1 mosa2 mosw1 mosw2 mosf1 mosf2 moso1 moso2 mosu1 mosu2 mosc1 mosc2;
array csm (k) csma1 csma2 csmw1 csmw2 csmf1 csmf2 csmo1 csmo2 csmu1 csmu2 csmc1 csmc2;
array cem (k) cema1 cema2 cemw1 cemw2 cemf1 cemf2 cemo1 cemo2 cemu1 cemu2 cemc1 cemc2;
array rcvr (k) rcvra1 rcvra2 rcvrw1 rcvrw2 rcvrf1 rcvrf2 rcvro1 rcvro2 rcvr09-rcvr12;
array rcvs (k) rcvs1 rcvs2 rcvsw1 rcvsw2 rcvsf1 rcvsf2 rcvso1 rcvso2 rcvs09-rcvs12;
array rcvc (k) rcvca1 rcvca2 rcvcw1 rcvcw2 rcvcf1 rcvcf2 rcvc01 rcvc02 rcvc09-rcvc12;
array rcvf (k) rcvfa1 rcvfa2 rcfw1 rcfw2 rcvff1 rcvff2 rcvfo1 rcvfo2 rcvf09-rcvf12;
array rcvo (k) rcvoa1 rcvoa2 rcvow1 rcvow2 rcvo1 rcvo2 rcvo3 rcvo4 rcvo5 rcvo6 rcvo7 rcvo8 rcvo9 rcvo10 rcvo11 rcvo12;
array who (k) whoa1 whoa2 whow1 whow2 whof1 whof2 whoo1 whoo2 who9-who12;
array dli (k) dlia1 dlia2 dliw1 dliw2 dlif1 dlif2 dlio1 dlio2 dli09-dli12;
array inc (k) inca1 inca2 incw1 incw2 incf1 incf2 inco1 inco2 incu1 incu2 inccl inc2;

array agefla1 (j) agefla11 agefla12 agefla13;
array agefla2 (j) agefla21 agefla22 agefla23;
array ageflw1 (j) ageflw11 ageflw12 ageflw13;
array ageflw2 (j) ageflw21 ageflw22 ageflw23;
array ageflf1 (j) ageflf11 ageflf12 ageflf13;
array ageflf2 (j) ageflf21 ageflf22 ageflf23;
array ageflo1 (j) ageflo11 ageflo12 ageflo13;
array ageflo2 (j) ageflo21 ageflo22 ageflo23;
array ageflu1 (j) ageflu11 ageflu12 ageflu13;
array ageflu2 (j) ageflu21 ageflu22 ageflu23;
array ageflc1 (j) ageflc11 ageflc12 ageflc13;
array ageflc2 (j) ageflc21 ageflc22 ageflc23;

array agefl (k) agefla1 agefla2 ageflw1 ageflw2 ageflf1 ageflf2 ageflo1 ageflo2 ageflu1 ageflu2 ageflc1 ageflc2;

NU80=0;      NU81=0;      NU82=0;      NU83=0;
NU84=0;      NU85=0;      NU86=0;      NU87=0;
NU88=0;      NU89=0;      NU90=0;      NU91=0;
NU92=0;      NU93=0;      NU94=0;      NU95=0;
NU96=0;      NU97=0;      NU98=0;      NU99=0;

```

***** This portion of the SAS program defines the start and end dates. If the respondent reports still receiving, the interview date is used as the temporary end date for the last loop reported. In the next survey round, the respondent will be asked if he or she is still receiving and, if not, a permanent end date equivalent to the interview date of the

previous round will be assigned. Users will be able to tell which method was used by looking at the following participation flag variable created during the program. The categories are the following:

| | |
|--|--------------------------------|
| 1=respondent reported participation dates | 2=start month imputed |
| 3=start month and year imputed | 4=stop month imputed |
| 5=stop month and year imputed | 6=start and stop dates imputed |
| 7=error in data due to round 2 programming error ******/ | |

```

do k=1 to 12;
  do j=1 to 3;
    if ys>0 and ms>0 then yms=(ys*100)+ms;
    if dli=1 then yms=dliym;
    if cur=1 then yme=iym;
    if ye>0 and me>0 then yme=(ye*100)+me;
    if yme>iym then yme=iym;
    if yms>0 and yme ge dliym then eflag=1;
    if wks ge 0 then mos=round (wks/4.3,1);
    if mos=0 then mos=1;

agefl=0;
if ys>0 and ys<1990 then agefl=1;

*****1. if start year is known and month is unknown ****/

$$\begin{aligned} &\text{** 1.a' If weeks are known and currently receiving, then count backwards by the number of weeks from the} \\ &\text{interview date. If the number of weeks falls short of the start year, the start month is December of that year. 0If the} \\ &\text{number of weeks is past the start year, then the start month is January of that year.****/} \end{aligned}$$

if ys>0 and -3 le ms le -1 then do;
  if (wks>0 and cwks=1) then do;
    yme=iym;
    if mos le YINT_M then yms=iym-mos;
    if YINT_M le mos le (YINT_M+12) then yms=iym-100+(mos-12);
    if (YINT_M+12) le mos le (YINT_M+24) then yms=iym-200+(mos-24);
    if (YINT_M+24) le mos le (YINT_M+36) then yms=iym-300+(mos-36);
    if (YINT_M+36) le mos le (YINT_M+48) then yms=iym-400+(mos-48);
    if (YINT_M+48) le mos le (YINT_M+60) then yms=iym-500+(mos-60);
    if (YINT_M+60) le mos le (YINT_M+72) then yms=iym-600+(mos-72);
    if (YINT_M+72) le mos le (YINT_M+84) then yms=iym-700+(mos-84);
    if (YINT_M+84) le mos le (YINT_M+96) then yms=iym-800+(mos-96);
    if (YINT_M+96) le mos le (YINT_M+108) then yms=iym-900+(mos-108);
    if (YINT_M+108) le mos le (YINT_M+120) then yms=iym-1000+(mos-120);
    if yms<((ys*100)+01) then yms=((ys*100)+01);
    if yms<dliym then yms=dliym;
    if yms>((ys*100)+12) then yms=((ys*100)+12);
  end;


$$\begin{aligned} &\text{/**weeks missing and continuously receiving - set end date to current interview date and start date to December of} \\ &\text{start year**/} \end{aligned}$$

else if (-3 le wks le 0 and cwks=1) then do;
  yme=iym;
  yms=((ys*100)+12);
  if yms>iym then yms=iym;
end;


$$\begin{aligned} &\text{/**weeks missing and not continuously receiving - set end date to December and start date to January of start year**/} \end{aligned}$$

else if (-3 le wks le 0 and cwks=0) then do;
  yms=((ys*100)+01);
  yme=((ys*100)+12);

```

```

if yme>iym then yme=iym;
end;

/** 1.a" if weeks are known and not currently receiving, then count forward by the number of weeks from January
of the start year. If the count exceeds the interview date then stop counting at the interview date.****/
else if (wks>0 and cwks=0) then do;
    yms=((ys*100)+01);
    if yms<dliym then yms=dliym;
    if 0 le mos le 12 then yme=yms+mos;
    if wks>0 and cwks=0 and 13 le mos le 24 then yme=((ys+1)*100)+mos-12;
    if wks>0 and cwks=0 and 25 le mos le 36 then yme=((ys+2)*100)+mos-24;
    if wks>0 and cwks=0 and 37 le mos le 48 then yme=((ys+3)*100)+mos-36;
    if wks>0 and cwks=0 and 49 le mos le 60 then yme=((ys+4)*100)+mos-48;
    if wks>0 and cwks=0 and 61 le mos le 72 then yme=((ys+5)*100)+mos-60;
    if wks>0 and cwks=0 and 73 le mos le 84 then yme=((ys+6)*100)+mos-72;
    if wks>0 and cwks=0 and 85 le mos le 96 then yme=((ys+7)*100)+mos-84;
    if wks>0 and cwks=0 and 97 le mos le 108 then yme=((ys+8)*100)+mos-96;
    if wks>0 and cwks=0 and 109 le mos le 120 then yme=((ys+9)*100)+mos-108;
    if yme>iym then yme=iym;
end;

if yme>0 and cwks=0 then yms=((ys*100)+01);
if yms<dliym then yms=dliym;
eflag=2;
end;

*****2. if start year is unknown but weeks are known then count back from interview date if currently receiving. If
not currently receiving, then count back from interview date to find the most recent year the respondent could have
begun receiving and receive for that number of months; then count forward the number of months from January of
that year ****/
if -3 le ys le -1 then do;
    if (wks>0 and cwks=1) then do;
        if 0 le mos le YINT_M then yms=iym-mos;
        if YINT_M le mos le (YINT_M+12) then yms=iym-100-(mos-12);
        if YINT_M+12 le mos le (YINT_M+24) then yms=iym-200-(mos-24);
        if (YINT_M+24) le mos le (YINT_M+36) then yms=iym-300-(mos-36);
        if (YINT_M+36) le mos le (YINT_M+48) then yms=iym-400-(mos-48);
        if (YINT_M+48) le mos le (YINT_M+60) then yms=iym-500-(mos-60);
        if (YINT_M+60) le mos le (YINT_M+72) then yms=iym-600-(mos-72);
        if (YINT_M+72) le mos le (YINT_M+84) then yms=iym-700-(mos-84);
        if (YINT_M+84) le mos le (YINT_M+96) then yms=iym-800-(mos-96);
        if (YINT_M+96) le mos le (YINT_M+108) then yms=iym-900-(mos-108);
        if (YINT_M+108) le mos le (YINT_M+120) then yms=iym-1000-(mos-120);
        yme=iym;
    end;

    if (-3 le wks le 0 and cwks=1) then do;
        yms=dliym;
        yme=iym;
    end;

    if (wks>0 and cwks=0) then do;
        if 0 le mos le YINT_M then do;
            yms=(YINT_Y*100)+01;
            yme=(YINT_Y*100)+01+mos;
        end;
    end;

```

```

if (YINT_M+01) le mos le (YINT_M+12) then do;
  yms=((YINT_Y-1)*100)+01;
  if (YINT_M+01) le mos le 12 then yme=((YINT_Y-1)*100)+01+mos;
  if 13 le mos le (YINT_M+12) then yme=(YINT_Y*100)+01+mos;
end;
if (YINT_M+13) le mos le (YINT_M+24) then do;
  yms=((YINT_Y-2)*100)+01;
  if (YINT_M+13) le mos le 24 then yme=((YINT_Y-1)*100)+01+mos;
  if 25 le mos le (YINT_M+24) then yme=(YINT_Y*100)+01+mos;
end;
if yme>iym then yme=iym;
end;
eflag=3;
end;

if yme>0 and yms=. then do;
  yms=dliym;
  if j=1 then eflag=3;
  else if j>1 then eflag=7;
end;

```

*****3. If stop year is known and weeks are known and not currently receiving, but stop month is not known, then count forward from start year. If the number of months falls short of the stop year, then use January of the end year as the stop date; if the number of months exceeds the stop year, then end the array in the December of the stop year. If the stop year is equal to the interview year and the stop month exceeds the interview month, then stop at the interview date. If currently receiving, use interview date as the stop date. *****

```

if yms>0 and ye>0 and -2 le me le -1 then do;
  if wks>0 then do;
    if 01 le ((yms- round (yms,100))+mos) le 12 then yme=yms+mos;
    if 12 lt ((yms- round (yms,100))+mos) le 24 then yme=yms+100+mos-12;
    if 24 lt ((yms- round (yms,100))+mos) le 36 then yme=yms+200+mos-24;
    if yme>((ye*100)+12) then yme=((ye*100)+12);
    if yme<((ye*100)+01) then yme=((ye*100)+01);
    if yme>iym then yme=iym;
  end;
  if -3 le wks le 0 then yme=((ye*100)+12);
  if yme>iym then yme=iym;
  eflag=4;
end;

```

*** 4. if stop year is unknown, and weeks are known***

```

if -3 le ye le -1 then do;
  if yms>0 and wks>0 then do;
    if 01 le ((yms- round (yms,100))+mos) le 12 then yme=yms+mos;
    if 12 lt ((yms- round (yms,100))+mos) le 24 then yme=yms+100+mos-12;
    if 24 lt ((yms- round (yms,100))+mos) le 36 then yme=yms+200+mos-24;
    if yme>iym then yme=iym;
  end;

```

*** 5. if stop year is unknown, and weeks are unknown***

```

if yms>0 and -3 le wks le 0 then yme=(round (yms,100))+12;
  eflag=5;
end;

```

*** 6. if the start and stop years are unknown and the weeks are unknown, use current and last interview date***

```

if -3 le ys le -1 and cwks=0 and -3 le wks le 0 then do;
  yme=iym;

```

```

yms=dliym;
if j=1 then eflag=6;
else if j>1 then eflag=7;
end;

```

***** This portion of the program creates a variable that determines who receives afdc in the household. It collapses the answers to 8 categories. The coding is the following -

| | |
|---|--|
| 1=respondent only 2=spouse/partner only 3=child only 4=respondent and spouse/partner | 5=respondent and child 6=spouse/partner and child 7=respondent and spouse/partner and child 8=other |
|---|--|

The first 7 categories may include an 'other' person as captured by response categories 4 and 5 in the original question. If only another person is listed as receiving, then the 8th answer category is used in the created variable. **/

```

if rcvr=1 and rcvs=0 and rcvc=0 then who=01;
if rcvr=0 and rcvs=1 and rcvc=0 then who=02;
if rcvr=0 and rcvs=0 and rcvc=1 then who=03;
if rcvr=1 and rcvs=1 and rcvc=0 then who=04;
if rcvr=1 and rcvs=0 and rcvc=1 then who=05;
if rcvr=0 and rcvs=1 and rcvc=1 then who=06;
if rcvr=1 and rcvs=1 and rcvc=1 then who=07;
if -3 le rcvr lt 0 then who=rcvr;
if rcvr=0 and rcvs=0 and rcvc=0 and (rcvf=1 or rcvo=1) then who=08;

```

***** This portion of the program uses the category reported by the respondent to create an estimated amount. The estimated amount is the midpoint rounded down. Note that the 12th category lists \$1251 as the amount. This amount was chosen since the category is unbounded - the number represents one dollar above the lower bound.****/

```

if -3 le incf1 le -1 and 1 le eincf1 le 10 then incf1=(eincf1*100)-50;
if -3 le incf1 le -1 and eincf1=11 then incf1=1125;
if -3 le incf1 le -1 and eincf1=12 then incf1=1251;
if -3 le incf2 le -1 and 1 le eincf2 le 10 then incf2=(eincf2*100)-50;
if -3 le incf2 le -1 and eincf2=11 then incf2=1125;
if -3 le incf2 le -1 and eincf2=12 then incf2=1251;
if -3 le inca1 le -1 and 1 le einca1 le 10 then inca1=(einca1*100)-50;
if -3 le inca1 le -1 and einca1=11 then inca1=1125;
if -3 le inca1 le -1 and einca1=12 then inca1=1251;
if -3 le inca2 le -1 and 1 le einca2 le 10 then inca2=(einca2*100)-50;
if -3 le inca2 le -1 and einca2=11 then inca2=1125;
if -3 le inca2 le -1 and einca2=12 then inca2=1251;
if -3 le inco1 le -1 and 1 le einco1 le 10 then inco1=(einco1*100)-50;
if -3 le inco1 le -1 and einco1=11 then inco1=1125;
if -3 le inco1 le -1 and einco1=12 then inco1=1251;
if -3 le inco2 le -1 and 1 le einco2 le 10 then inco2=(einco2*100)-50;
if -3 le inco2 le -1 and einco2=11 then inco2=1125;
if -3 le inco2 le -1 and einco2=12 then inco2=1251;
if -3 le incw1 le -1 and 1 le eincw1 le 5 then incw1=(eincw1*20)-10;
if -3 le incw1 le -1 and eincw1=6 then incw1=101;
if -3 le incw2 le -1 and 1 le eincw2 le 5 then incw2=(eincw2*20)-10;
if -3 le incw2 le -1 and eincw2=6 then incw2=101;
if wincu1>0 then incu1= round ((wincu1*52)/12,1);
if wincu1<0 then incu1=wincu1;
if wincu2>0 then incu2= round ((wincu2*52)/12,1);
if wincu2<0 then incu2=wincu2;
if wincc1>0 then incc1= round ((wincc1*52)/12,1);
if wincc1<0 then incc1=wincc1;
if wincc2>0 then incc2= round ((wincc2*52)/12,1);
if wincc2<0 then incc2=wincc2;

```

```

if inc>-4 then aflag=0;

csm=(round(yms,100)-198000)*.12+(yms-round(yms,100));
cem=(round(yme,100)-198000)*.12+(yme-round(yme,100));

*****to compute the spells of ui*****
if 9 le K le 10 then DO;
  if 1 le CSM le 12 then NU80=NU80+1;
  if 13 le CSM le 24 then NU81=NU81+1;
  if 25 le CSM le 36 then NU82=NU82+1;
  if 37 le CSM le 48 then NU83=NU83+1;
  if 49 le CSM le 60 then NU84=NU84+1;
  if 61 le CSM le 72 then NU85=NU85+1;
  if 73 le CSM le 84 then NU86=NU86+1;
  if 85 le CSM le 96 then NU87=NU87+1;
  if 97 le CSM le 108 then NU88=NU88+1;
  if 109 le CSM le 120 then NU89=NU89+1;
  if 121 le CSM le 132 then NU90=NU90+1;
  if 133 le CSM le 144 then NU91=NU91+1;
  if 145 le CSM le 156 then NU92=NU92+1;
  if 157 le CSM le 168 then NU93=NU93+1;
  if 169 le CSM le 180 then NU94=NU94+1;
  if 181 le CSM le 192 then NU95=NU95+1;
  if 193 le CSM le 204 then NU96=NU96+1;
  if 205 le CSM le 216 then NU97=NU97+1;
  if 217 le CSM le 228 then NU98=NU98+1;
  if 229 le CSM le 232 then NU99=NU99+1;
end;

C=0;
do L=1 to 232;
  C=C+1;
  if 0 le L le doicm then do;
    if csm LE C LE cem then do;
      if 1 le k le 2 then do;
        a=eflag; ahhm=who; aamt=inc;
        if aamt>1229 then do;
          amtodd=1;
        end;
      end;
      if 3 le k le 4 then do;
        w=eflag; whhm=who; wamt=inc;
      end;
      if 5 le k le 6 then do;
        f=eflag; fhlm=who; famt=inc;
        if famt>1000 then do;
          amtodd=2;
        end;
      end;
      if 7 le k le 8 then do;
        o=eflag; ohhm=who; oamt=inc;
      end;
      if 9 le k le 10 then do;
        ui=eflag; uiamt=inc;
      end;
      if 11 le k le 12 then do;
        wc=eflag; wcamt=inc;
      end;
    end;
  end;
end;

```



```

wa4=0; wn4=0; wa5=0; wn5=0; wa6=0; wn6=0;
ua7=0; un7=0; ua8=0; un8=0; ua9=0; un9=0;
wa7=0; wn7=0; wa8=0; wn8=0; wa9=0; wn9=0;
incprg=0; prgamt=0; nnp=0; nap=0; out=0;

/*total months - part 1*/
do L=1 to 232;
  if 1 le L le 232 then do;
    if TTLM ge 0 and A>0 or W>0 or F>0 or O>0
      then do;
        TTLM=TTLM+1;
      end;
    if a>1 then TTLM=-3;
    if w>1 then TTLM=-3;
    if f>1 then TTLM=-3;
    if o>1 then TTLM=-3;
    if TTLU ge 0 and UI>0 then do;
      TTLU=TTLU+1;
    end;
    if UI>1 then TTLU=-3;
    if TTLW ge 0 and WC>0 then do;
      TTLW=TTLW+1;
    end;
    if WC>1 then TTLW=-3;
  end;

  if 1 le L le 12 then do;
    if M80 ge 0 and a>0 or w>0 or f>0 or o>0 then do;
      M80=M80+1;
    end;
    if a>1 then M80=-3;
    if w>1 then M80=-3;
    if f>1 then M80=-3;
    if o>1 then M80=-3;
    array allamt80 aamt wamt famt oamt;
    do over allamt80;
      if -3 le allamt80 le -1 then do;
        nn80=nn80+1;
        amt80=allamt80;
      end;
    if allamt80 ge 0 then do;
      na80=na80+1;
      if na80=0 and nn80=0 then amt80=allamt80;
      if na80>1 and nn80=0 then
        amt80=amt80+allamt80;
      if M80=-3 then amt80=-3;
    end;
  end;
  if UM80 ge 0 and UI>0 then do;
    UM80=UM80+1;
  end;
  if UI>1 then UM80=-3;
  if -3 le uiamt le -1 then do;
    un80=un80+1;
    uamt80=uiamt;
  end;
  if uiamt ge 0 then do;
    ua80=ua80+1;
    if ua80=1 and un80=0 then UAMT80=UIAMT;
    if ua80>1 and un80=0 then
      uamt80=uamt80+uiamt;
    if UM80=-3 then uamt80=-3;
  end;
  if WM80 ge 0 and WC>0 then do;
    WM80=WM80+1;
  end;
  if WC>1 then WM80=-3;
  if -3 le wcamt le -1 then do;
    wn80=wn80+1;
    camt80=wcamt;
  end;
  if wcamt ge 0 then do;
    wa80=wa80+1;
    if wa80=1 and wn80=0 then
      cAMT80=WCAMT;
    if wa80>1 and wn80=0 then
      camt80=camt80+wcamt;
    if WM80=-3 then camt80=-3;
  end;
  if 13 le L le 24 then do;
    if M81 ge 0 and a>0 or w>0 or f>0 or o>0 then do;
      M81=M81+1;
    end;
    if a>1 then M81=-3;
    if w>1 then M81=-3;
    if f>1 then M81=-3;
    if o>1 then M81=-3;
    array allamt81 aamt wamt famt oamt;
    do over allamt81;
      if -3 le allamt81 le -1 then do;
        nn81=nn81+1;
        amt81=allamt81;
      end;
    if allamt81 ge 0 then do;
      na81=na81+1;
      if na81=0 and nn81=0 then amt81=allamt81;
      if na81>1 and nn81=0 then
        amt81=amt81+allamt81;
      if M81=-3 then amt81=-3;
    end;
  end;
  if UM81 ge 0 and UI>0 then do;
    UM81=UM81+1;
  end;
  if UI>1 then UM81=-3;
  if -3 le uiamt le -1 then do;
    un81=un81+1;
  end;

```

| | |
|--|---|
| <pre> uamt81=uiamt; end; if uiamt ge 0 then do; ua81=ua81+1; if ua81=1 and un81=0 then UAMT81=UIAMT; if ua81>1 and un81=0 then uamt81=uamt81+uiamt; if UM81=-3 then uamt81=-3; end; if WM81 ge 0 and WC>0 then do; WM81=WM81+1; end; if WC>1 then WM81=-3; if -3 le wcamt le -1 then do; wn81=wn81+1; camt81=wcamt; end; if wcamt ge 0 then do; wa81=wa81+1; if wa81=1 and wn81=0 then cAMT81=WCAMT; if wa81>1 and wn81=0 then camt81=camt81+wcamt; if WM81=-3 then camt81=-3; end; end; if 25 le L le 36 then do; if M82 ge 0 and a>0 or w>0 or f>0 or o>0 then do; M82=M82+1; end; if a>1 then M82=-3; if w>1 then M82=-3; if f>1 then M82=-3; if o>1 then M82=-3; array allamt82 aamt wamt famt oamt; do over allamt82; if -3 le allamt82 le -1 then do; nn82=nn82+1; amt82=allamt82; end; if allamt82 ge 0 then do; na82=na82+1; if na82=0 and nn82=0 then amt82=allamt82; if na82>1 and nn82=0 then amt82=amt82+allamt82; if M82=-3 then amt82=-3; end; end; if UM82 ge 0 and UI>0 then do; UM82=UM82+1; end; if UI>1 then UM82=-3; if -3 le uiamt le -1 then do; un82=un82+1; uamt82=uiamt; end; if uiamt ge 0 then do; </pre> | <pre> ua82=ua82+1; if ua82=1 and un82=0 then UAMT82=UIAMT; if ua82>1 and un82=0 then uamt82=uamt82+uiamt; if UM82=-3 then uamt82=-3; end; if WM82 ge 0 and WC>0 then do; WM82=WM82+1; end; if WC>1 then WM82=-3; if -3 le wcamt le -1 then do; wn82=wn82+1; camt82=wcamt; end; if wcamt ge 0 then do; wa82=wa82+1; if wa82=1 and wn82=0 then cAMT82=WCAMT; if wa82>1 and wn82=0 then camt82=camt82+wcamt; if WM82=-3 then camt82=-3; end; end; if 37 le L le 48 then do; if M83 ge 0 and a>0 or w>0 or f>0 or o>0 then do; M83=M83+1; end; if a>1 then M83=-3; if w>1 then M83=-3; if f>1 then M83=-3; if o>1 then M83=-3; array allamt83 aamt wamt famt oamt; do over allamt83; if -3 le allamt83 le -1 then do; nn83=nn83+1; amt83=allamt83; end; if allamt83 ge 0 then do; na83=na83+1; if na83=0 and nn83=0 then amt83=allamt83; if na83>1 and nn83=0 then amt83=amt83+allamt83; if M83=-3 then amt83=-3; end; end; if UM83 ge 0 and UI>0 then do; UM83=UM83+1; end; if UI>1 then UM83=-3; if -3 le uiamt le -1 then do; un83=un83+1; uamt83=uiamt; end; if uiamt ge 0 then do; ua83=ua83+1; if ua83=1 and un83=0 then UAMT83=UIAMT; </pre> |
|--|---|

```

if ua83>1 and un83=0 then
    uamt83=uamt83+uiamt;
    if UM83=-3 then uamt83=-3;
end;
if WM83 ge 0 and WC>0 then do;
    WM83=WM83+1;
end;
if WC>1 then WM83=-3;
if -3 le wcamt le -1 then do;
    wn83=wn83+1;
    camt83=wcamt;
end;
if wcamt ge 0 then do;
    wa83=wa83+1;
    if wa83=1 and wn83=0 then cAMT83=WCAMT;
    if wa83>1 and wn83=0 then
        camt83=camt83+wcamt;
    if WM83=-3 then camt83=-3;
end;
end;

if 49 le L le 60 then do;
    if M84 ge 0 and a>0 or w>0 or f>0 or o>0 then do;
        M84=M84+1;
    end;
    if a>1 then M84=-3;
    if w>1 then M84=-3;
    if f>1 then M84=-3;
    if o>1 then M84=-3;
    array allamt84 aamt wamt famt oamt;
    do over allamt84;
        if -3 le allamt84 le -1 then do;
            nn84=nn84+1;
            amt84=allamt84;
        end;
        if allamt84 ge 0 then do;
            na84=na84+1;
            if na84=0 and nn84=0 then amt84=allamt84;
            if na84>1 and nn84=0 then
                amt84=amt84+allamt84;
            if M84=-3 then amt84=-3;
        end;
    end;
    if UM84 ge 0 and UI>0 then do;
        UM84=UM84+1;
    end;
    if UI>1 then UM84=-3;
    if -3 le uiamt le -1 then do;
        un84=un84+1;
        uamt84=uiamt;
    end;
    if uiamt ge 0 then do;
        ua84=ua84+1;
        if ua84=1 and un84=0 then UAMT84=UIAMT;
        if ua84>1 and un84=0 then
            uamt84=uamt84+uiamt;
        if UM84=-3 then uamt84=-3;
    end;
end;

```

```

end;
if WM84 ge 0 and WC>0 then do;
    WM84=WM84+1;
end;
if WC>1 then WM84=-3;
if -3 le wcamt le -1 then do;
    wn84=wn84+1;
    camt84=wcamt;
end;
if wcamt ge 0 then do;
    wa84=wa84+1;
    if wa84=1 and wn84=0 then cAMT84=WCAMT;
    if wa84>1 and wn84=0 then
        camt84=camt84+wcamt;
    if WM84=-3 then camt84=-3;
end;
end;

if 61 le L le 72 then do;
    if M85 ge 0 and a>0 or w>0 or f>0 or o>0 then do;
        M85=M85+1;
    end;
    if a>1 then M85=-3;
    if w>1 then M85=-3;
    if f>1 then M85=-3;
    if o>1 then M85=-3;
    array allamt85 aamt wamt famt oamt;
    do over allamt85;
        if -3 le allamt85 le -1 then do;
            nn85=nn85+1;
            amt85=allamt85;
        end;
        if allamt85 ge 0 then do;
            na85=na85+1;
            if na85=0 and nn85=0 then amt85=allamt85;
            if na85>1 and nn85=0 then
                amt85=amt85+allamt85;
            if M85=-3 then amt85=-3;
        end;
    end;
    if UM85 ge 0 and UI>0 then do;
        UM85=UM85+1;
    end;
    if UI>1 then UM85=-3;
    if -3 le uiamt le -1 then do;
        un85=un85+1;
        uamt85=uiamt;
    end;
    if uiamt ge 0 then do;
        ua85=ua85+1;
        if ua85=1 and un85=0 then UAMT85=UIAMT;
        if ua85>1 and un85=0 then
            uamt85=uamt85+uiamt;
        if UM85=-3 then uamt85=-3;
    end;
    if WM85 ge 0 and WC>0 then do;
        WM85=WM85+1;
    end;
end;

```

| | |
|--|--|
| <pre> end; if WC>1 then WM85=-3; if -3 le wcamt le -1 then do; wn85=wn85+1; camt85=wcamt; end; if wcamt ge 0 then do; wa85=wa85+1; if wa85=1 and wn85=0 then cAMT85=WCAMT; if wa85>1 and wn85=0 then camt85=camt85+wcamt; if WM85=-3 then camt85=-3; end; end; if 73 le L le 84 then do; if M86 ge 0 and a>0 or w>0 or f>0 or o>0 then do; M86=M86+1; end; if a>1 then M86=-3; if w>1 then M86=-3; if f>1 then M86=-3; if o>1 then M86=-3; array allamt86 aamt wamt famt oamt; do over allamt86; if -3 le allamt86 le -1 then do; nn86=nn86+1; amt86=allamt86; end; if allamt86 ge 0 then do; na86=na86+1; if na86=0 and nn86=0 then amt86=allamt86; if na86>1 and nn86=0 then amt86=amt86+allamt86; if M86=-3 then amt86=-3; end; end; if UM86 ge 0 and UI>0 then do; UM86=UM86+1; end; if UI>1 then UM86=-3; if -3 le uiamt le -1 then do; un86=un86+1; uamt86=uiamt; end; if uiamt ge 0 then do; ua86=ua86+1; if ua86=1 and un86=0 then UAMT86=UIAMT; if ua86>1 and un86=0 then uamt86=uamt86+uiamt; if UM86=-3 then uamt86=-3; end; if WM86 ge 0 and WC>0 then do; WM86=WM86+1; end; if WC>1 then WM86=-3; if -3 le wcamt le -1 then do; </pre> | <pre> wn86=wn86+1; camt86=wcamt; end; if wcamt ge 0 then do; wa86=wa86+1; if wa86=1 and wn86=0 then cAMT86=WCAMT; if wa86>1 and wn86=0 then camt86=camt86+wcamt; if WM86=-3 then camt86=-3; end; end; if 85 le L le 96 then do; if M87 ge 0 and a>0 or w>0 or f>0 or o>0 then do; M87=M87+1; end; if a>1 then M87=-3; if w>1 then M87=-3; if f>1 then M87=-3; if o>1 then M87=-3; array allamt87 aamt wamt famt oamt; do over allamt87; if -3 le allamt87 le -1 then do; nn87=nn87+1; amt87=allamt87; end; if allamt87 ge 0 then do; na87=na87+1; if na87=0 and nn87=0 then amt87=allamt87; if na87>1 and nn87=0 then amt87=amt87+allamt87; if M87=-3 then amt87=-3; end; end; if UM87 ge 0 and UI>0 then do; UM87=UM87+1; end; if UI>1 then UM87=-3; if -3 le uiamt le -1 then do; un87=un87+1; uamt87=uiamt; end; if uiamt ge 0 then do; ua87=ua87+1; if ua87=1 and un87=0 then UAMT87=UIAMT; if ua87>1 and un87=0 then uamt87=uamt87+uiamt; if UM87=-3 then uamt87=-3; end; if WM87 ge 0 and WC>0 then do; WM87=WM87+1; end; if WC>1 then WM87=-3; if -3 le wcamt le -1 then do; wn87=wn87+1; camt87=wcamt; end; </pre> |
|--|--|

| | |
|---|--|
| <pre> if wcamt ge 0 then do; wa87=wa87+1; if wa87=1 and wn87=0 then cAMT87=WCAMT; if wa87>1 and wn87=0 then camt87=camt87+wcamt; if WM87=-3 then camt87=-3; end; end; if 97 le L le 108 then do; if M88 ge 0 and a>0 or w>0 or f>0 or o>0 then do; M88=M88+1; end; if a>1 then M88=-3; if w>1 then M88=-3; if f>1 then M88=-3; if o>1 then M88=-3; array allamt88 aamt wamt famt oamt; do over allamt88; if -3 le allamt88 le -1 then do; nn88=nn88+1; amt88=allamt88; end; if allamt88 ge 0 then do; na88=na88+1; if na88=0 and nn88=0 then amt88=allamt88; if na88>1 and nn88=0 then amt88=amt88+allamt88; if M88=-3 then amt88=-3; end; end; if UM88 ge 0 and UI>0 then do; UM88=UM88+1; end; if UI>1 then UM88=-3; if -3 le uiamt le -1 then do; un88=un88+1; uamt88=uiamt; end; if uiamt ge 0 then do; ua88=ua88+1; if ua88=1 and un88=0 then UAMT88=UIAMT; if ua88>1 and un88=0 then uamt88=uamt88+uiamt; if UM88=-3 then uamt88=-3; end; if WM88 ge 0 and WC>0 then do; WM88=WM88+1; end; if WC>1 then WM88=-3; if -3 le wcamt le -1 then do; wn88=wn88+1; camt88=wcamt; end; if wcamt ge 0 then do; wa88=wa88+1; if wa88=1 and wn88=0 then cAMT88=WCAMT; </pre> | <pre> if wa88>1 and wn88=0 then camt88=camt88+wcamt; if WM88=-3 then camt88=-3; end; end; if 109 le L le 120 then do; if M89 ge 0 and a>0 or w>0 or f>0 or o>0 then do; M89=M89+1; end; if a>1 then M89=-3; if w>1 then M89=-3; if f>1 then M89=-3; if o>1 then M89=-3; array allamt89 aamt wamt famt oamt; do over allamt89; if -3 le allamt89 le -1 then do; nn89=nn89+1; amt89=allamt89; end; if allamt89 ge 0 then do; na89=na89+1; if na89=0 and nn89=0 then amt89=allamt89; if na89>1 and nn89=0 then amt89=amt89+allamt89; if M89=-3 then amt89=-3; end; end; if UM89 ge 0 and UI>0 then do; UM89=UM89+1; end; if UI>1 then UM89=-3; if -3 le uiamt le -1 then do; un89=un89+1; uamt89=uiamt; end; if uiamt ge 0 then do; ua89=ua89+1; if ua89=1 and un89=0 then UAMT89=UIAMT; if ua89>1 and un89=0 then uamt89=uamt89+uiamt; if UM89=-3 then uamt89=-3; end; if WM89 ge 0 and WC>0 then do; WM89=WM89+1; end; if WC>1 then WM89=-3; if -3 le wcamt le -1 then do; wn89=wn89+1; camt89=wcamt; end; if wcamt ge 0 then do; wa89=wa89+1; if wa89=1 and wn89=0 then cAMT89=WCAMT; if wa89>1 and wn89=0 then camt89=camt89+wcamt; if WM89=-3 then camt89=-3; end; </pre> |
|---|--|

| | |
|--|---|
| <pre> end; end; if 121 le L le 132 then do; if M90 ge 0 and a>0 or w>0 or f>0 or o>0 then do; M90=M90+1; end; if a>1 then M90=-3; if w>1 then M90=-3; if f>1 then M90=-3; if o>1 then M90=-3; array allamt0 aamt wamt famt oamt; do over allamt0; if -3 le allamt0 le -1 then do; nn0=nn0+1; amt90=allamt0; end; if allamt0 ge 0 then do; na0=na0+1; if na0=0 and nn0=0 then amt90=allamt0; if na0>1 and nn0=0 then amt90=amt90+allamt0; if M90=-3 then amt90=-3; end; end; if UM90 ge 0 and UI>0 then do; UM90=UM90+1; end; if UI>1 then UM90=-3; if -3 le uiamt le -1 then do; un0=un0+1; uamt90=uiamt; end; if uiamt ge 0 then do; ua0=ua0+1; if ua0=1 and un0=0 then UAMT90=UIAMT; if ua0>1 and un0=0 then uamt90=uamt90+uiamt; if UM90=-3 then uamt90=-3; end; if WM90 ge 0 and WC>0 then do; WM90=WM90+1; end; if WC>1 then WM90=-3; if -3 le wcamt le -1 then do; wn0=wn0+1; camt90=wcamt; end; if wcamt ge 0 then do; wa0=wa0+1; if wa0=1 and wn0=0 then cAMT90=WCAMT; if wa0>1 and wn0=0 then camt90=camt90+wcamt; if WM90=-3 then camt90=-3; end; end; if 133 le L le 144 then do; if M91 ge 0 and a>0 or w>0 or f>0 or o>0 then do; </pre> | <pre> M91=M91+1; end; if a>1 then M91=-3; if w>1 then M91=-3; if f>1 then M91=-3; if o>1 then M91=-3; array allamt1 aamt wamt famt oamt; do over allamt1; if -3 le allamt1 le -1 then do; nn1=nn1+1; amt91=allamt1; end; if allamt1 ge 0 then do; na1=na1+1; if na1=1 and nn1=0 then amt91=allamt1; if na1>1 and nn1=0 then amt91=amt91+allamt1; if M91=-3 then amt91=-3; end; if UM91 ge 0 and UI>0 then do; UM91=UM91+1; end; if UI>1 then UM91=-3; if -3 le uiamt le -1 then do; un1=un1+1; uamt91=uiamt; end; if uiamt ge 0 then do; ua1=ua1+1; if ua1=1 and un1=0 then UAMT91=UIAMT; if ua1>1 and un1=0 then uamt91=uamt91+uiamt; if UM91=-3 then uamt91=-3; end; if WM91 ge 0 and WC>0 then do; WM91=WM91+1; end; if WC>1 then WM91=-3; if -3 le wcamt le -1 then do; wn1=wn1+1; camt91=wcamt; end; if wcamt ge 0 then do; wa1=wa1+1; if wa1=1 and wn1=0 then cAMT91=WCAMT; if wa1>1 and wn1=0 then camt91=camt91+wcamt; if WM91=-3 then camt91=-3; end; end; if 145 le L le 156 then do; if M92 ge 0 and a>0 or w>0 or f>0 or o>0 then do; M92=M92+1; end; if a>1 then M92=-3; if w>1 then M92=-3; if f>1 then M92=-3; </pre> |
|--|---|

```

if o>1 then M92=-3;
array allamt2 aamt wamt famt oamt;
do over allamt2;
  if -3 le allamt2 le -1 then do;
    nn2=nn2+1;
    amt92=allamt2;
  end;
  if allamt2 ge 0 then do;
    na2=na2+1;
    if na2=1 and nn2=0 then amt92=allamt2;
    if na2>1 and nn2=0 then amt92=amt92+allamt2;
    if M92=-3 then amt92=-3;
  end;
end;
if UM92 ge 0 and UI>0 then do;
  UM92=UM92+1;
end;
if UI>1 then UM92=-3;
if -3 le uiamt le -1 then do;
  un2=un2+1;
  uamt92=uiamt;
end;
if uiamt ge 0 then do;
  ua2=ua2+1;
  if ua2=1 and un2=0 then UAMT92=UIAMT;
  if ua2>1 and un2=0 then uamt92=uamt92+uiamt;
  if UM92=-3 then uamt92=-3;
end;
if WM92 ge 0 and WC>0 then do;
  WM92=WM92+1;
end;
if WC>1 then WM92=-3;
if -3 le wcamt le -1 then do;
  wn2=wn2+1;
  camt92=wcamt;
end;
if wcamt ge 0 then do;
  wa2=wa2+1;
  if wa2=1 and wn2=0 then cAMT92=WCAMT;
  if wa2>1 and wn2=0 then
    camt92=camt92+wcamt;
  if WM92=-3 then camt92=-3;
end;
end;

if 157 le L le 168 then do;
  if M93 ge 0 and a>0 or w>0 or f>0 or o>0 then do;
    M93=M93+1;
  end;
  if a>1 then M93=-3;
  if w>1 then M93=-3;
  if f>1 then M93=-3;
  if o>1 then M93=-3;
  array allamt3 aamt wamt famt oamt;
  do over allamt3;
    if -3 le allamt3 le -1 then do;
      nn3=nn3+1;
      amt93=allamt3;
    end;
    if allamt3 ge 0 then do;
      na3=na3+1;
      if na3=1 and nn3=0 then amt93=allamt3;
      if na3>1 and nn3=0 then amt93=amt93+allamt3;
      if M93=-3 then amt93=-3;
    end;
    if UM93 ge 0 and UI>0 then do;
      UM93=UM93+1;
    end;
    if UI>1 then UM93=-3;
    if -3 le uiamt le -1 then do;
      un3=un3+1;
      uamt93=uiamt;
    end;
    if uiamt ge 0 then do;
      ua3=ua3+1;
      if ua3=1 and un3=0 then UAMT93=UIAMT;
      if ua3>1 and un3=0 then uamt93=uamt93+uiamt;
      if UM93=-3 then uamt93=-3;
    end;
    if WM93 ge 0 and WC>0 then do;
      WM93=WM93+1;
    end;
    if WC>1 then WM93=-3;
    if -3 le wcamt le -1 then do;
      wn3=wn3+1;
      camt93=wcamt;
    end;
    if wcamt ge 0 then do;
      wa3=wa3+1;
      if wa3=1 and wn3=0 then cAMT93=WCAMT;
      if wa3>1 and wn3=0 then
        camt93=camt93+wcamt;
      if WM93=-3 then camt93=-3;
    end;
    end;

    if 169 le L le 180 then do;
      if M94 ge 0 and a>0 or w>0 or f>0 or o>0 then do;
        M94=M94+1;
      end;
      if a>1 then M94=-3;
      if w>1 then M94=-3;
      if f>1 then M94=-3;
      if o>1 then M94=-3;
      array allamt4 aamt wamt famt oamt;
      do over allamt4;
        if -3 le allamt4 le -1 then do;
          nn4=nn4+1;
          amt94=allamt4;
        end;
        if allamt4 ge 0 then do;
          na4=na4+1;
          if na4=1 and nn4=0 then amt94=allamt4;
        end;
      end;
    end;
  end;
end;

```

| | |
|---|---|
| <pre> if na4>1 and nn4=0 then amt94=amt94+allamt4; if M94=-3 then amt94=-3; end; if UM94 ge 0 and UI>0 then do; UM94=UM94+1; end; if UI>1 then UM94=-3; if -3 le uiamt le -1 then do; un4=un4+1; uamt94=uamt; end; if uiamt ge 0 then do; ua4=ua4+1; if ua4=1 and un4=0 then UAMT94=UIAMT; if ua4>1 and un4=0 then uamt94=uamt94+uamt; if UM94=-3 then uamt94=-3; end; if WM94 ge 0 and WC>0 then do; WM94=WM94+1; end; if WC>1 then WM94=-3; if -3 le wcamt le -1 then do; wn4=wn4+1; camt94=wcamt; end; if wcamt ge 0 then do; wa4=wa4+1; if wa4=1 and wn4=0 then cAMT94=WCAMT; if wa4>1 and wn4=0 then camt94=camt94+wcamt; if WM94=-3 then camt94=-3; end; end; if 181 le L le 192 then do; if M95 ge 0 and a>0 or w>0 or f>0 or o>0 then do; M95=M95+1; end; if a>1 then M95=-3; if w>1 then M95=-3; if f>1 then M95=-3; if o>1 then M95=-3; array allamt5 aamt wamt famt oamt; do over allamt5; if -3 le allamt5 le -1 then do; nn5=nn5+1; amt95=allamt5; end; if allamt5 ge 0 then do; na5=na5+1; if na5=1 and nn5=0 then amt95=allamt5; if na5>1 and nn5=0 then amt95=amt95+allamt5; if M95=-3 then amt95=-3; end; end; if UM95 ge 0 and UI>0 then do; </pre> | <pre> UM95=UM95+1; end; if UI>1 then UM95=-3; if -3 le uiamt le -1 then do; un5=un5+1; uamt95=uamt; end; if uiamt ge 0 then do; ua5=ua5+1; if ua5=1 and un5=0 then UAMT95=UIAMT; if ua5>1 and un5=0 then uamt95=uamt95+uamt; if UM95=-3 then uamt95=-3; end; if WM95 ge 0 and WC>0 then do; WM95=WM95+1; end; if WC>1 then WM95=-3; if -3 le wcamt le -1 then do; wn5=wn7+1; camt95=wcamt; end; if wcamt ge 0 then do; wa5=wa5+1; if wa5=1 and wn5=0 then cAMT95=WCAMT; if wa5>1 and wn5=0 then camt95=camt95+wcamt; if WM95=-3 then camt95=-3; end; end; if 193 le L le 204 then do; if M96 ge 0 and a>0 or w>0 or f>0 or o>0 then do; M96=M96+1; end; if a>1 then M96=-3; if w>1 then M96=-3; if f>1 then M96=-3; if o>1 then M96=-3; array allamt6 aamt wamt famt oamt; do over allamt6; if -3 le allamt6 le -1 then do; nn6=nn6+1; amt96=allamt6; end; if allamt6 ge 0 then do; na6=na6+1; if na6=1 and nn6=0 then amt96=allamt6; if na6>1 and nn6=0 then amt96=amt96+allamt6; if M96=-3 then amt96=-3; end; end; if UM96 ge 0 and UI>0 then do; UM96=UM96+1; end; if UI>1 then UM96=-3; if -3 le uiamt le -1 then do; un6=un6+1; </pre> |
|---|---|

```

uamt96=uiamt;
end;
if uiamt ge 0 then do;
ua6=ua6+1;
if ua6=1 and un6=0 then UAMT96=UIAMT;
if ua6>1 and un6=0 then uamt96=uamt96+uiamt;
if UM96=-3 then uamt96=-3;
end;
if WM96 ge 0 and WC>0 then do;
WM96=WM96+1;
end;
if WC>1 then WM96=-3;
if -3 le wcamt le -1 then do;
wn6=wn6+1;
camt96=wcamt;
end;
if wcamt ge 0 then do;
wa6=wa6+1;
if wa6=1 and wn6=0 then cAMT96=WCAMT;
if wa6>1 and wn6=0 then
camt96=camt96+wcamt;
if WM96=-3 then camt96=-3;
end;
end;

if 205 le L le 216 then do;
if M97 ge 0 and a>0 or w>0 or f>0 or o>0 then do;
M97=M97+1;
end;
if a>1 then M97=-3;
if w>1 then M97=-3;
if f>1 then M97=-3;
if o>1 then M97=-3;
array allamt7 aamt wamt famt oamt;
do over allamt7;
if -3 le allamt7 le -1 then do;
nn7=nn7+1;
amt97=allamt7;
end;
if allamt7 ge 0 then do;
na7=na7+1;
if na7=1 and nn7=0 then amt97=allamt7;
if na7>1 and nn7=0 then amt97=amt97+allamt7;
if M97=-3 then amt97=-3;
end;
end;
if UM97 ge 0 and UI>0 then do;
UM97=UM97+1;
end;
if UI>1 then UM97=-3;
if -3 le uiamt le -1 then do;
un7=un7+1;
uamt97=uiamt;
end;
if uiamt ge 0 then do;
ua7=ua7+1;
if ua7=1 and un7=0 then UAMT97=UIAMT;
if ua7>1 and un7=0 then uamt97=uamt97+uiamt;
if UM97=-3 then uamt97=-3;
end;
if WM97 ge 0 and WC>0 then do;
WM97=WM97+1;
end;
if WC>1 then WM97=-3;
if -3 le wcamt le -1 then do;
wn7=wn7+1;
camt97=wcamt;
end;
if wcamt ge 0 then do;
wa7=wa7+1;
if wa7=1 and wn7=0 then cAMT97=WCAMT;
if wa7>1 and wn7=0 then
camt97=camt97+wcamt;
if WM97=-3 then camt97=-3;
end;
end;

if 217 le L le 228 then do;
if M98 ge 0 and a>0 or w>0 or f>0 or o>0 then do;
M98=M98+1;
end;
if a>1 then M98=-3;
if w>1 then M98=-3;
if f>1 then M98=-3;
if o>1 then M98=-3;
array allamt8 aamt wamt famt oamt;
do over allamt8;
if -3 le allamt8 le -1 then do;
nn8=nn8+1;
amt98=allamt8;
end;
if allamt8 ge 0 then do;
na8=na8+1;
if na8=1 and nn8=0 then amt98=allamt8;
if na8>1 and nn8=0 then amt98=amt98+allamt8;
if M98=-3 then amt98=-3;
end;
end;
if UM98 ge 0 and UI>0 then do;
UM98=UM98+1;
end;
if UI>1 then UM98=-3;
if -3 le uiamt le -1 then do;
un8=un8+1;
uamt98=uiamt;
end;
if uiamt ge 0 then do;
ua8=ua8+1;
if ua8=1 and un8=0 then UAMT98=UIAMT;
if ua8>1 and un8=0 then uamt98=uamt98+uiamt;
if UM98=-3 then uamt98=-3;
end;
if WM98 ge 0 and WC>0 then do;
WM98=WM98+1;
end;

```

```

end;
if WC>1 then WM98=-3;
if -3 le wcamt le -1 then do;
  wn8=wn8+1;
  camt98=wcamt;
end;
if wcamt ge 0 then do;
  wa8=wa8+1;
  if wa8=1 and wn8=0 then cAMT98=WCAMT;
  if wa8>1 and wn8=0 then
    camt98=camt98+wcamt;
  if WM98=-3 then camt98=-3;
end;
end;
if 229 le L le 232 then do;
  if M99 ge 0 and a>0 or w>0 or f>0 or o>0 then do;
    M99=M99+1;
  end;
  if a>1 then M99=-3;
  if w>1 then M99=-3;
  if f>1 then M99=-3;
  if o>1 then M99=-3;

array allamt9 aamt wamt famt oamt;
do over allamt9;
  if -3 le allamt9 le -1 then do;
    nn9=nn9+1;
    amt99=allamt9;
  end;
  if allamt9 ge 0 then do;
    na9=na9+1;
    if na9=1 and nn9=0 then amt99=allamt9;
    if na9>1 and nn9=0 then amt99=amt99+allamt9;
    if M99=-3 then amt99=-3;
  end;
end;

if UM99 ge 0 and UI>0 then do;
  UM99=UM99+1;
end;
if UI>1 then UM99=-3;
if -3 le uiamt le -1 then do;
  un9=un9+1;
  uamt99=uiamt;
end;
if uiamt ge 0 then do;
  ua9=ua9+1;
  if ua9=1 and un9=0 then UAMT99=UIAMT;
  if ua9>1 and un9=0 then uamt99=uamt99+uiamt;
  if UM99=-3 then uamt99=-3;
end;
if WM99 ge 0 and WC>0 then do;
  WM99=WM99+1;
end;
if WC>1 then WM99=-3;
if -3 le wcamt le -1 then do;
  wn9=wn9+1;
  camt99=wcamt;
end;
if wcamt ge 0 then do;
  wa9=wa9+1;
  if wa9=1 and wn9=0 then cAMT99=WCAMT;
  if wa9>1 and wn9=0 then
    camt99=camt99+wcamt;
  if WM99=-3 then camt99=-3;
end;
end;

**for income program amounts;
if 205 le L le 216 then do;
  if w>0 or f>0 then do;
    out=1;
  end;
  if incprg ge 0 and (a>0 or o>0 or ui>0 or wc>0)
    then do;
    incprg=incprg+1;
  end;
  if a>1 or o>1 or ui>1 or wc>1 then incprg=-3;
  array prgamts aamt oamt uiamt wcamt;
  do over prgamts;
    if -3 le prgamts le -1 then do;
      nnp=nnp+1;
      prgamt=prgamts;
    end;
    if prgamts ge 0 then do;
      nap=nap+1;
      if nap=1 and nnp=0 then prgamt=prgamts;
      if nap>1 and nnp=0 then
        prgamt=prgamt+prgamts;
      if incprg=-3 then prgamt=-3;
    end;
    end;
  end;
end;
/*3. end*/
if total80=. then total80=0;
if total81=. then total81=0;
if total82=. then total82=0;
if total83=. then total83=0;
if total84=. then total84=0;
if total85=. then total85=0;
if total86=. then total86=0;
if total87=. then total87=0;
if total88=. then total88=0;
if total89=. then total89=0;
if total90=. then total90=0;
if total91=. then total91=0;
if total92=. then total92=0;
if total93=. then total93=0;
if total94=. then total94=0;
if total95=. then total95=0;
if total96=. then total96=0;

```

| | |
|--|--|
| <pre> if total97=. then total97=0; if total98=. then total98=0; if totalall=. then totalall=0; ckM80=total80; ckM81=total81; ckM82=total82; ckM83=total83; ckM84=total84; ckM85=total85; ckM86=total86; ckM87=total87; ckM88=total88; ckM89=total89; ckM90=total90; ckM91=total91; ckM92=total92; ckM93=total93; ckM94=total94; ckM95=total95; ckM96=total96; ckM97=total97; ckM98=total98; if M80 ge 0 and total80 ge 0 then do; total80=M80+total80; end; if M81 ge 0 and total81 ge 0 then do; total81=M81+total81; end; if M82 ge 0 and total82 ge 0 then do; total82=M82+total82; end; if M83 ge 0 and total83 ge 0 then do; total83=M83+total83; end; if M84 ge 0 and total84 ge 0 then do; total84=M84+total84; end; if M85 ge 0 and total85 ge 0 then do; total85=M85+total85; end; if M86 ge 0 and total86 ge 0 then do; total86=M86+total86; end; if M87 ge 0 and total87 ge 0 then do; total87=M87+total87; end; if M88 ge 0 and total88 ge 0 then do; total88=M88+total88; end; if M89 ge 0 and total89 ge 0 then do; total89=M89+total89; end; if M90 ge 0 and total90 ge 0 then do; total90=M90+total90; end; if M91 ge 0 and total91 ge 0 then do; total91=M91+total91; end; if M92 ge 0 and total92 ge 0 then do; total92=M92+total92; end; if M93 ge 0 and total93 ge 0 then do; total93=M93+total93; end; if M94 ge 0 and total94 ge 0 then do; total94=M94+total94; end; if M95 ge 0 and total95 ge 0 then do; total95=M95+total95; end; </pre> | <pre> if M96 ge 0 and total96 ge 0 then do; total96=M96+total96; end; if M97 ge 0 and total97 ge 0 then do; total97=M97+total97; end; if M98 ge 0 and total98 ge 0 then do; total98=M98+total98; end; total99=m99; if M80<0 or total80<0 then do; total80=-3; end; if M81<0 or total81<0 then do; total81=-3; end; if M82<0 or total82<0 then do; total82=-3; end; if M83<0 or total83<0 then do; total83=-3; end; if M84<0 or total84<0 then do; total84=-3; end; if M85<0 or total85<0 then do; total85=-3; end; if M86<0 or total86<0 then do; total86=-3; end; if M87<0 or total87<0 then do; total87=-3; end; if M88<0 or total88<0 then do; total88=-3; end; if M89<0 or total89<0 then do; total89=-3; end; if M90<0 or total90<0 then do; total90=-3; end; if M91<0 or total91<0 then do; total91=-3; end; if M92<0 or total92<0 then do; total92=-3; end; if M93<0 or total93<0 then do; total93=-3; end; if M94<0 or total94<0 then do; total94=-3; end; if M95<0 or total95<0 then do; total95=-3; end; if M96<0 or total96<0 then do; total96=-3; end; if M97<0 or total97<0 then do; total97=-3; end; if M98<0 or total98<0 then do; total98=-3; end; if m99<0 then do; total99=-3; end; *total months, part 2; if ttlm ge 0 and totalall ge 0 then do; ttlm=ttlm+totalall; end; if ttlm<0 or totalall<0 then do; ttlm=-3; end; ckamt80=atotal80; ckamt81=atotal81; ckamt82=atotal82; ckamt83=atotal83; ckamt84=atotal84; ckamt85=atotal85; ckamt86=atotal86; ckamt87=atotal87; ckamt88=atotal88; ckamt89=atotal89; ckamt90=atotal90; ckamt91=atotal91; ckamt92=atotal92; ckamt93=atotal93; ckamt94=atotal94; ckamt95=atotal95; ckamt96=atotal96; ckamt97=atotal97; ckamt98=atotal98; if atotal80=-4 then atotal80=0; if atotal81=-4 then atotal81=0; </pre> |
|--|--|

```

if atotal82=-4 then atotal82=0;
if atotal83=-4 then atotal83=0;
if atotal84=-4 then atotal84=0;
if atotal85=-4 then atotal85=0;
if atotal86=-4 then atotal86=0;
if atotal87=-4 then atotal87=0;
if atotal88=-4 then atotal88=0;
if atotal89=-4 then atotal89=0;
if atotal90=-4 then atotal90=0;
if atotal91=-4 then atotal91=0;
if atotal92=-4 then atotal92=0;
if atotal93=-4 then atotal93=0;
if atotal94=-4 then atotal94=0;
if atotal95=-4 then atotal95=0;
if atotal96=-4 then atotal96=0;
if atotal97=-4 then atotal97=0;
if atotal98=-4 then atotal98=0;

if amt80 ge 0 and atotal80 ge 0 then do;
  atotal80=amt80+atotal80; end;
if amt81 ge 0 and atotal81 ge 0 then do;
  atotal81=amt81+atotal81; end;
if amt82 ge 0 and atotal82 ge 0 then do;
  atotal82=amt82+atotal82; end;
if amt83 ge 0 and atotal83 ge 0 then do;
  atotal83=amt83+atotal83; end;
if amt84 ge 0 and atotal84 ge 0 then do;
  atotal84=amt84+atotal84; end;
if amt85 ge 0 and atotal85 ge 0 then do;
  atotal85=amt85+atotal85; end;
if amt86 ge 0 and atotal86 ge 0 then do;
  atotal86=amt86+atotal86; end;
if amt87 ge 0 and atotal87 ge 0 then do;
  atotal87=amt87+atotal87; end;
if amt88 ge 0 and atotal88 ge 0 then do;
  atotal88=amt88+atotal88; end;
if amt89 ge 0 and atotal89 ge 0 then do;
  atotal89=amt89+atotal89; end;
if amt90 ge 0 and atotal90 ge 0 then do;
  atotal90=amt90+atotal90; end;
if amt91 ge 0 and atotal91 ge 0 then do;
  atotal91=amt91+atotal91; end;
if amt92 ge 0 and atotal92 ge 0 then do;
  atotal92=amt92+atotal92; end;
if amt93 ge 0 and atotal93 ge 0 then do;
  atotal93=amt93+atotal93; end;
if amt94 ge 0 and atotal94 ge 0 then do;
  atotal94=amt94+atotal94; end;
if amt95 ge 0 and atotal95 ge 0 then do;
  atotal95=amt95+atotal95; end;
if amt96 ge 0 and atotal96 ge 0 then do;
  atotal96=amt96+atotal96; end;
if amt97 ge 0 and atotal97 ge 0 then do;
  atotal97=amt97+atotal97; end;
if amt98 ge 0 and atotal98 ge 0 then do;
  atotal98=amt98+atotal98; end;
atotal99=amt99;

```

```

if amt80<0 or atotal80<0 or total80<0 then do;
  atotal80=-3; end;
if amt81<0 or atotal81<0 or total81<0 then do;
  atotal81=-3; end;
if amt82<0 or atotal82<0 or total82<0 then do;
  atotal82=-3; end;
if amt83<0 or atotal83<0 or total83<0 then do;
  atotal83=-3; end;
if amt84<0 or atotal84<0 or total84<0 then do;
  atotal84=-3; end;
if amt85<0 or atotal85<0 or total85<0 then do;
  atotal85=-3; end;
if amt86<0 or atotal86<0 or total86<0 then do;
  atotal86=-3; end;
if amt87<0 or atotal87<0 or total87<0 then do;
  atotal87=-3; end;
if amt88<0 or atotal88<0 or total88<0 then do;
  atotal88=-3; end;
if amt89<0 or atotal89<0 or total89<0 then do;
  atotal89=-3; end;
if amt90<0 or atotal90<0 or total90<0 then do;
  atotal90=-3; end;
if amt91<0 or atotal91<0 or total91<0 then do;
  atotal91=-3; end;
if amt92<0 or atotal92<0 or total92<0 then do;
  atotal92=-3; end;
if amt93<0 or atotal93<0 or total93<0 then do;
  atotal93=-3; end;
if amt94<0 or atotal94<0 or total94<0 then do;
  atotal94=-3; end;
if amt95<0 or atotal95<0 or total95<0 then do;
  atotal95=-3; end;
if amt96<0 or atotal96<0 or total96<0 then do;
  atotal96=-3; end;
if amt97<0 or atotal97<0 or total97<0 then do;
  atotal97=-3; end;
if amt98<0 or atotal98<0 or total98<0 then do;
  atotal98=-3; end;

if amt80=0 and ckamt80=-4 and total80=0 then do;
  atotal80=-4; end;
if amt81=0 and ckamt81=-4 and total81=0 then do;
  atotal81=-4; end;
if amt82=0 and ckamt82=-4 and total82=0 then do;
  atotal82=-4; end;
if amt83=0 and ckamt83=-4 and total83=0 then do;
  atotal83=-4; end;
if amt84=0 and ckamt84=-4 and total84=0 then do;
  atotal84=-4; end;
if amt85=0 and ckamt85=-4 and total85=0 then do;
  atotal85=-4; end;
if amt86=0 and ckamt86=-4 and total86=0 then do;
  atotal86=-4; end;
if amt87=0 and ckamt87=-4 and total87=0 then do;
  atotal87=-4; end;
if amt88=0 and ckamt88=-4 and total88=0 then do;

```

```

        atotal88=-4; end;
if amt89=0 and ckamt89=-4 and total89=0 then do;
  atotal89=-4; end;
if amt90=0 and ckamt90=-4 and total90=0 then do;
  atotal90=-4; end;
if amt91=0 and ckamt91=-4 and total91=0 then do;
  atotal91=-4; end;
if amt92=0 and ckamt92=-4 and total92=0 then do;
  atotal92=-4; end;
if amt93=0 and ckamt93=-4 and total93=0 then do;
  atotal93=-4; end;
if amt94=0 and ckamt94=-4 and total94=0 then do;
  atotal94=-4; end;
if amt95=0 and ckamt95=-4 and total95=0 then do;
  atotal95=-4; end;
if amt96=0 and ckamt96=-4 and total96=0 then do;
  atotal96=-4; end;
if amt97=0 and ckamt97=-4 and total97=0 then do;
  atotal97=-4; end;
if amt98=0 and ckamt98=-4 and total98=0 then do;
  atotal98=-4; end;
if amt99=0 and total99=0 then do;
  atotal99=-4; end;

if uamt80=0 and um80=0 then do; uamt80=-4; end;
if uamt81=0 and um81=0 then do; uamt81=-4; end;
if uamt82=0 and um82=0 then do; uamt82=-4; end;
if uamt83=0 and um83=0 then do; uamt83=-4; end;
if uamt84=0 and um84=0 then do; uamt84=-4; end;
if uamt85=0 and um85=0 then do; uamt85=-4; end;
if uamt86=0 and um86=0 then do; uamt86=-4; end;
if uamt87=0 and um87=0 then do; uamt87=-4; end;
if uamt88=0 and um88=0 then do; uamt88=-4; end;
if p1210=-4 then do;

total80=-4; total81=-4; total82=-4; total83=-4; total84=-4; total85=-4; total86=-4; total87=-4;
total88=-4; total89=-4; total90=-4; total91=-4; total92=-4; total93=-4; total94=-4; total95=-4;
total96=-4; total97=-4; total98=-4; total99=-4;
atotal80=-4; atotal81=-4; atotal82=-4; atotal83=-4; atotal84=-4; atotal85=-4; atotal86=-4; atotal87=-4;
atotal88=-4; atotal89=-4; atotal90=-4; atotal91=-4; atotal92=-4; atotal93=-4; atotal94=-4; atotal95=-4;
atotal96=-4; atotal97=-4; atotal98=-4; atotal99=-4;
m80=-4; m81=-4; M82=-4; M83=-4; M84=-4; M85=-4; M86=-4; M87=-4; M88=-4; M89=-4;
m90=-4; m91=-4; M92=-4; M93=-4; M94=-4; M95=-4; M96=-4; M97=-4; M98=-4; M99=-4; ttlM=-4;
amt80=-4; amt81=-4; amt82=-4; amt83=-4; amt84=-4; amt85=-4; amt86=-4; amt87=-4; amt88=-4; amt89=-4;
amt90=-4; amt91=-4; amt92=-4; amt93=-4; amt94=-4; amt95=-4; amt96=-4; amt97=-4; amt98=-4; amt99=-4;
UM80=-4; UM81=-4; UM82=-4; UM83=-4; UM84=-4; UM85=-4; UM86=-4; UM88=-4; UM87=-4;
UM88=-4; UM89=-4; UM90=-4; UM91=-4; UM92=-4; UM93=-4; UM94=-4; UM95=-4; UM96=-4;
UM97=-4; UM98=-4; UM99=-4;
WM80=-4; WM81=-4; WM82=-4; WM83=-4; WM84=-4; WM85=-4; WM86=-4; WM88=-4; WM87=-4;
WM88=-4; WM89=-4; WM90=-4; WM91=-4; WM92=-4; WM93=-4; WM94=-4; WM95=-4; WM96=-4;
WM97=-4; WM98=-4; WM99=-4; TTLU=-4; TTLW=-4;
camt80=-4; camt81=-4; camt82=-4; camt83=-4; camt84=-4; camt85=-4; camt86=-4; camt87=-4; camt88=-4;
camt89=-4; camt90=-4; camt91=-4; camt92=-4; camt93=-4; camt94=-4; camt95=-4; camt96=-4; camt97=-4;
camt98=-4; camt99=-4;
uamt80=-4; uamt81=-4; uamt82=-4; uamt83=-4; uamt84=-4; uamt85=-4; uamt86=-4; uamt87=-4; uamt88=-4;
uamt89=-4; uamt90=-4; uamt91=-4; uamt92=-4; uamt93=-4; uamt94=-4; uamt95=-4; uamt96=-4; uamt97=-4;
uamt98=-4; uamt99=-4;

```

```

na80=-4; nn80=-4; na81=-4; nn81=-4; na82=-4; nn82=-4; na83=-4; nn83=-4; na84=-4; nn84=-4; na85=-4;
nn85=-4; na86=-4; nn86=-4; na87=-4; nn87=-4; na88=-4; nn88=-4; na89=-4; nn89=-4;
na0=-4; nn0=-4; na1=-4; nn1=-4; na2=-4; nn2=-4; na3=-4; nn3=-4;
na4=-4; nn4=-4; na5=-4; nn5=-4; na6=-4; nn6=-4; na7=-4; nn7=-4;
na8=-4; nn8=-4; na9=-4; nn9=-4;
ua80=-4; un80=-4; ua81=-4; un81=-4;
wa80=-4; wn80=-4; wa81=-4; wn81=-4;
ua82=-4; un82=-4; ua83=-4; un83=-4; ua84=-4; un84=-4;
wa82=-4; wn82=-4; wa83=-4; wn83=-4; wa84=-4; wn84=-4;
ua85=-4; un85=-4; ua86=-4; un86=-4; ua87=-4; un87=-4;
wa85=-4; wn85=-4; wa86=-4; wn86=-4; wa87=-4; wn87=-4;
ua88=-4; un88=-4; ua89=-4; un89=-4; ua0=-4; un0=-4;
wa88=-4; wn88=-4; wa89=-4; wn89=-4; wa0=-4; wn0=-4;
ua1=-4; un1=-4; ua2=-4; un2=-4; ua3=-4; un3=-4;
wa1=-4; wn1=-4; wa2=-4; wn2=-4; wa3=-4; wn3=-4;
ua4=-4; un4=-4; ua5=-4; un5=-4; ua6=-4; un6=-4;
wa4=-4; wn4=-4; wa5=-4; wn5=-4; wa6=-4; wn6=-4;
ua7=-4; un7=-4; ua8=-4; un8=-4; ua9=-4; un9=-4;
wa7=-4; wn7=-4; wa8=-4; wn8=-4; wa9=-4; wn9=-4;
incprg=-4; prgamt=-4; nnp=-4; nap=-4; out=-4;
NU80=-4; NU81=-4; NU82=-4; NU83=-4; NU84=-4; NU85=-4; NU86=-4; NU87=-4; NU88=-4; NU89=-4;
NU90=-4; NU91=-4; NU92=-4; NU93=-4; NU94=-4; NU95=-4; NU96=-4; NU97=-4; NU98=-4; NU99=-4;

end;

if p1210=-5 then do;
total80=-5; total81=-5; total82=-5; total83=-5; total84=-5; total85=-5; total86=-5; total87=-5;
total88=-5; total89=-5; total90=-5; total91=-5; total92=-5; total93=-5; total94=-5; total95=-5;
total96=-5; total97=-5; total98=-5; total99=-5;
atotal80=-5; atotal81=-5; atotal82=-5; atotal83=-5; atotal84=-5; atotal85=-5; atotal86=-5; atotal87=-5;
atotal88=-5; atotal89=-5; atotal90=-5; atotal91=-5; atotal92=-5; atotal93=-5; atotal94=-5; atotal95=-5;
atotal96=-5; atotal97=-5; atotal98=-5; atotal99=-5;
m80=-5; m81=-5; M82=-5; M83=-5; M84=-5; M85=-5; M86=-5; M87=-5; M88=-5; M89=-5;
m90=-5; m91=-5; M92=-5; M93=-5; M94=-5; M95=-5; M96=-5; M97=-5; M98=-5; M99=-5; ttlM=-5;
amt80=-5; amt81=-5; amt82=-5; amt83=-5; amt84=-5; amt85=-5; amt86=-5; amt87=-5; amt88=-5; amt89=-5;
amt90=-5; amt91=-5; amt92=-5; amt93=-5; amt94=-5; amt95=-5; amt96=-5; amt97=-5; amt98=-5; amt99=-5;
UM80=-5; UM81=-5; UM82=-5; UM83=-5; UM84=-5; UM85=-5; UM86=-5; UM88=-5; UM87=-5;
UM88=-5; UM89=-5; UM90=-5; UM91=-5; UM92=-5; UM93=-5; UM94=-5; UM95=-5; UM96=-5;
UM97=-5; UM98=-5; UM99=-5;
WM80=-5; WM81=-5; WM82=-5; WM83=-5; WM84=-5; WM85=-5; WM86=-5; WM88=-5; WM87=-5;
WM88=-5; WM89=-5; WM90=-5; WM91=-5; WM92=-5; WM93=-5; WM94=-5; WM95=-5; WM96=-5;
WM97=-5; WM98=-5; WM99=-5; TTLU=-5; TTLW=-5;
camt80=-5; camt81=-5; camt82=-5; camt83=-5; camt84=-5; camt85=-5; camt86=-5; camt87=-5; camt88=-5;
camt89=-5; camt90=-5; camt91=-5; camt92=-5; camt93=-5; camt94=-5; camt95=-5; camt96=-5; camt97=-5;
camt98=-5; camt99=-5;
uamt80=-5; uamt81=-5; uamt82=-5; uamt83=-5; uamt84=-5; uamt85=-5; uamt86=-5; uamt87=-5; uamt88=-5;
uamt89=-5; uamt90=-5; uamt91=-5; uamt92=-5; uamt93=-5; uamt94=-5; uamt95=-5; uamt96=-5; uamt97=-5;
uamt98=-5; uamt99=-5;
na80=-5; nn80=-5; na81=-5; nn81=-5; na82=-5; nn82=-5; na83=-5; nn83=-5; na84=-5; nn84=-5; na85=-5;
nn85=-5; na86=-5; nn86=-5; na87=-5; nn87=-5; na88=-5; nn88=-5; na89=-5; nn89=-5;
na0=-5; nn0=-5; na1=-5; nn1=-5; na2=-5; nn2=-5; na3=-5; nn3=-5;
na4=-5; nn4=-5; na5=-5; nn5=-5; na6=-5; nn6=-5; na7=-5; nn7=-5;
na8=-5; nn8=-5; na9=-5; nn9=-5;
ua80=-5; un80=-5; ua81=-5; un81=-5;
wa80=-5; wn80=-5; wa81=-5; wn81=-5;
ua82=-5; un82=-5; ua83=-5; un83=-5; ua84=-5; un84=-5;
wa82=-5; wn82=-5; wa83=-5; wn83=-5; wa84=-5; wn84=-5;

```

```
ua85=-5; un85=-5; ua86=-5; un86=-5; ua87=-5; un87=-5;
wa85=-5; wn85=-5; wa86=-5; wn86=-5; wa87=-5; wn87=-5;
ua88=-5; un88=-5; ua89=-5; un89=-5; ua0=-5; un0=-5;
wa88=-5; wn88=-5; wa89=-5; wn89=-5; wa0=-5; wn0=-5;
ua1=-5; un1=-5; ua2=-5; un2=-5; ua3=-5; un3=-5;
wa1=-5; wn1=-5; wa2=-5; wn2=-5; wa3=-5; wn3=-5;
ua4=-5; un4=-5; ua5=-5; un5=-5; ua6=-5; un6=-5;
wa4=-5; wn4=-5; wa5=-5; wn5=-5; wa6=-5; wn6=-5;
ua7=-5; un7=-5; ua8=-5; un8=-5; ua9=-5; un9=-5;
wa7=-5; wn7=-5; wa8=-5; wn8=-5; wa9=-5; wn9=-5;
incprg=-5; prgamt=-5; nnp=-5; nap=-5; out=-5;
NU80=-5; NU81=-5; NU82=-5; NU83=-5; NU84=-5; NU85=-5; NU86=-5; NU87=-5; NU88=-5; NU89=-5;
NU90=-5; NU91=-5; NU92=-5; NU93=-5; NU94=-5; NU95=-5; NU96=-5; NU97=-5; NU98=-5; NU99=-5;

end;
endsas;
```

NLSY97 Appendix 6:

Event History Creation and Documentation

The NLSY97 survey records significant life-course transitions experienced by young people, such as education, employment, program participation, and marital history, in a longitudinal format. The event history arrays document these events in a chronological format that records the significant transitions in a meaningful manner while maintaining data quality. Using these arrays, researchers can extract the status of a respondent at a point in time or over time. Event history arrays are generated for four distinct areas: employment, marital/cohabitation status, program participation, and schooling. This section presents information on each type of event history array; for details on the chronological format of the arrays and the naming conventions used to identify the variables, users should refer to appendix 7.

Employment Event History Arrays

Three employment arrays provide information on the respondent's employment on a weekly basis. These arrays include information about employer jobs only; freelance jobs were not included in the arrays. All employment arrays provide information starting in the week that the respondent turned 14 and ending in the week that he or she was last interviewed.

1. EMP_STATUS

This main array presents the employment status of a respondent in a particular week. The codes and their explanations follow:

| Code | Definition |
|---|---|
| Status=0: No information reported to account for week | Not assigned during round 1. |
| Status=1: Not associated with an employer, not actively searching for an employer job | Refers to weeks during a between-jobs gap in which the respondent is not actively searching and reports working at a freelance job. Since the actual weeks working at a freelance job cannot be determined, all weeks in which the respondent is not actively searching are coded in this manner. Similar information was not collected for the within-job gaps during round 1. |
| Status=2: Not working (unemployment vs. out of labor force cannot be determined) | Assigned when the respondent is not asked follow-up questions about his or her search activity during a within-job gap or a between-jobs gap. |
| Status=3: Associated with an employer, periods not working for employer are missing | Used when a respondent reports an indeterminate start or stop date for a within-job gap. |
| Status=4: Unemployed | Indicates that the respondent reports actively searching for work during a within-job gap or a between-jobs gap. When the number of weeks unemployed do not account for the entire gap period, weeks unemployed are assumed to occur in the middle of that period. |
| Status=5: Out of the labor force | Assigned during a between-jobs gap or a within-job gap when the respondent is either not actively searching for work or on layoff from a job. |
| Status=6: Active military service | Indicates that the respondent is tied to the military. |
| Status=9701 to 9809: Employer on roster | Refers to the employer number on the employer roster (YEMP_UID.01 to YEMP_UID.09). Presence of an employer number indicates that the respondent was working during a given week. Civilian work takes precedence over other activities, such as job search. Respondents who report working at an employer job for one day in a given week are listed as having worked at that job for the entire week, regardless of other activities. |

2. EMP_DUAL_JOB#

If a respondent holds more than one employer job during a week, the second employer job is presented in a dual job array. These arrays contain only the job number of the overlapping job; labor force status information is only included in the main array. For example, if a respondent held two employer jobs (e.g., the first and third jobs listed on the employer roster), during the 52nd week of 1997, the employer number for the first job would be recorded in the EMP_STATUS array and the employer number for the third job would be recorded in the EMP_DUAL_2 array. If a respondent held three jobs (e.g., jobs #01, #04, and #05 on the roster) in one week, the first job would be recorded in the EMP_STATUS array, the employer ID for job #04 would be recorded in the EMP_DUAL_2 array, and the employer ID for job #05 would be recorded in the EMP_DUAL_3 array.

3. EMP_HOURS

This final array calculates the total number of hours worked by a respondent at any employer job during a given week. Hours per week worked at each job are assumed constant except during a reported gap, when the hours for that job are assumed to be zero. Each week is assigned a code of ‘-3 (invalid skip)’ when any of the jobs has an indeterminate gap date.

A secondary set of variables translates the reported beginning and ending dates (day, month, and year) of employer jobs and the gaps within those jobs to the week and year naming scheme (e.g., EMP_GAP_START_YEAR.01.01 and EMP_GAP_END_YEAR.01.01 provide the start and end dates of the respondent’s first gap in the continuous week and year format). More information about the week and year naming scheme is provided in appendix 7 in this document.

The created event history variables can be used in conjunction with the main file information about the respondent’s employment. The 9701–9809 codes that appear in the status and dual job arrays are unique employer ID numbers that are also used in the main file data. In the main data, these codes are listed in the YEMP_UID.xx variables (e.g., R24761.). Using these unique ID codes, researchers can identify the comparable job information (e.g., start and stop dates, fringe benefits, job satisfaction, industry and occupation, etc.) from the main file. Employers first reported in the round 1 interview have codes beginning with “97”; new employers in round 2 were assigned codes beginning with “98.” This system permits users to link employers across survey rounds and across data files and to identify the round in which an employer was first reported.

Marital Status Event History Arrays

The NLSY97 marital and cohabitation arrays record changes in the respondent’s marital status and cohabitation changes on a monthly basis. The marital/cohabitation history program converts dates reported in the marriage section (beginning and ending dates of cohabitations, marriages, separations, divorces, and widowhoods) to an actual month number, using January 1980 as month #1. Used jointly, these arrays allow the researcher to obtain a detailed history of the respondent’s partners and changes in his/her marital and cohabitation status on a monthly basis. All marital/cohabitation arrays provide information beginning in the month that the respondent turned 14 and ending in the month that he or she was last interviewed. Additionally, the beginning dates of the youth’s first marriage and first cohabitation are provided in two variables: CV_FIRST_MARRY_MONTH and CV_FIRST_COHAB_MONTH.

Four types of arrays record transitions between living without a partner of the opposite sex to cohabiting or to marriage.

1. MAR_STATUS

The main array presents the status (e.g., never married/not cohabiting, cohabiting, married, divorced) of a respondent during a particular month. Marital status takes precedence over cohabiting; for example, if a

respondent is divorced and living with another partner, the status listed in this array will be ‘divorced.’ Respondents who are married but not living with their spouse are coded as married. When a respondent reports an annulment, the marriage is maintained and the marital status code after the annulment is ‘divorced.’

2. MAR_COHABITATION

This second array details the partner that the respondent is living with in a particular month. For example, if the respondent is cohabiting, the variable for each month identifies whether the respondent lives with partner 1, partner 2, spouse 1, spouse 2, etc. Users should note that “1” and “2” in this case refer to the respondent’s partners/spouses in chronological order. The numbers do not necessarily refer to the same person as the spouse/partner questions asked directly of the respondent during the survey.

If a respondent reports living with two partners in the same month, the first partner is listed in this array, subsequent partners are listed in the MAR_DUAL array.

3. MAR_DUAL

If there is an overlap of partners (e.g., partner 1 leaves at the beginning of the month and partner 2 moves in at the end of the month), this array records the presence of the new partner. The format of these variables is the same as that of the MAR_COHABITATION variables.

4. MAR_PARTNER_LINK

The fourth array links the cohabiting partner or spouse to the partner order in the main survey questions. For example, a number of codebook variables report the status of “Partner #01” or “Partner #02.” If the person the respondent reports cohabiting with in a given month is the first person asked about in the 1997 survey, the MAR_PARTNER_LINK variable for that month is 9701. If the partner in a given month is the second partner asked about in 1997, this variable is coded 9702, and so on. This array allows the researcher to identify characteristics of the respondent’s partner and to link them with spells of marriage or cohabitation. In round 2, if a partner was reported in round 1, he or she retains the same identification number. Otherwise, new partners are coded as 9801, 9802, etc., in the same manner that was used in round 1.

Program Participation Event History Arrays

Program participation arrays are constructed individually for five programs—Worker’s Compensation, Unemployment Compensation, AFDC, Food Stamps, and WIC. The AFDC array includes all federal and state programs created under Temporary Assistance to Needy Families (TANF) or any government program for needy families that replaces AFDC. All other programs (e.g., LIHEAP, SSI, other) are combined into a sixth array entitled ‘Other.’ For each program type, except Worker’s and Unemployment Compensation, three arrays are created. All program participation arrays provide information starting in the month that the respondent turned 14 and ending in the month that he or she was last interviewed.

A secondary set of variables translates the reported beginning and ending dates (month and year) of a spell within the program into the continuous month scheme (e.g., AFDC_START_MONTH and AFDC_STOP_MONTH). More information about the continuous month scheme is provided in appendix 7 in this document.

1. STATUS

The main array, (e.g., AFDC_STATUS), presents the status—receiving or not—of a respondent during each month. When asked for the start or stop date of a spell, the respondent could respond ‘don’t know’ or ‘refuse’ to any component. In this case, the respondent was then asked how many weeks the spell lasted. The number of reported weeks was then divided by 4.3 to determine the equivalent number of months. If a fraction of a month was reported, then the entire month was counted as a month receiving

benefits. Using a combination of start date, stop date, and week information, each spell was defined and a value of '1' inserted into the status array to indicate months of receipt. The months that a respondent did not receive that benefit, but was eligible to receive it, have a value of '0.' An edit variable, (e.g., AFDC_EDIT_DATE) flags respondent-reported and imputed dates. The process by which imputed dates and the corresponding edit flag were assigned is described below:

| Flag | Definition |
|---|--|
| Edit Flag=1: Respondent reported participation dates | Respondent reported a complete start and stop date and is not currently receiving. If the respondent reports still receiving at the time of the interview, the interview date is assigned as the temporary stop date. In the next survey round, the respondent will be asked if he or she is still receiving; if not, a permanent stop date equivalent to the previous round's interview date will be assigned. If the respondent reports receiving, participation will continue in filling the array. |
| Edit Flag=2: Start month imputed | Total weeks known: If the respondent reports not currently receiving, then set the month equal to January and count forward by the number of weeks to imply a stop date. If currently receiving, then count back by the number of weeks from the interview date to impute a start month. If the month indicated by the count falls short of the start year, the start month is December of the start year. If the month occurs in the year before the reported start year, then the start month is January of the start year. |
| | Total weeks unknown: If the respondent reports not currently receiving, then the start month is set to January. Use December as the stop month and the start year as the stop year. If the respondent reports currently receiving, use December as the start month. |
| Edit Flag=3: Start month and year imputed | Total weeks known: Count back by the number of weeks from the interview date if currently receiving. If not currently receiving, then count back from interview date to find the most recent year the respondent could have begun receiving and call the start date January of that year; then count forward the number of weeks from that date to imply a stop date. |
| | Total weeks unknown: If currently receiving, begin the spell at the respondent's 14 th birthday. |
| Edit Flag=4: Stop month imputed | Total weeks known: If not currently receiving, then count forward from start date. If the month indicated falls short of the stop year, then use January of the stop year as the stop month; if the number of months exceeds the stop year, then set the stop month to December of the stop year. If the stop year is equal to the interview year and the stop month exceeds the interview month, then stop at the interview date. |
| | Total weeks unknown: If not currently receiving, then use December of stop year for the stop month. |
| Edit Flag=5: Stop month and year imputed. | Total weeks known: If not still receiving, count forward from the start date. |
| | Total weeks unknown: If not currently receiving, then use December of the start year as the stop month and the start year as the stop year. |
| Edit Flag=6: Start and stop dates imputed. | Total weeks unknown: The imputed dates are based on the previous interview's date (start date) to the current interview date (stop date); in round 1, the last interview date is the respondent's 14 th birth month and year. |
| Edit Flag=7: Start and stop dates complete but gap information missing. | In these cases, the respondent began a spell of receipt in round 1 and ended it in round 2. The start and stop dates are accurate, but no information was collected about gaps in receipt. |

2. AMOUNT RECEIVED

If a respondent reports receiving in a particular month, a second array presents the amount received in each month (e.g., AFDC_AMT). The dollar values asked about during the interview were meant to be monthly values. However, some responses were higher than the federal or state limits on the amount

received from a particular benefit. A likely reason is that the respondent mistakenly reported a total value rather than a monthly value. Values determined to be too high were divided by the number of months the respondent reported receiving the benefit. These values were used in the AMT arrays instead. A second set of edit variables (e.g., AFDC_EDIT_AMT) flags these values for a particular spell. In round 1, the edit values were set for the latest year available. For AFDC, the highest benefit offered was \$1229; reported amounts falling above this maximum were edited. The maximum amount accepted for food stamps was \$936; the edit flag indicates reported amounts that fall above this maximum.

3. HOUSEHOLD MEMBERS RECEIVING

If a respondent reports receiving in a particular month, the persons in the household who benefit from the program in each month (e.g., respondent only, child only, respondent and child) are recorded in a third array (e.g., AFDC_HH). This program condenses the set of answers from the question in the survey that collects this information; for example, see YPRG-18300.01_001 to YPRG-18300.01_005 for AFDC. Users should note that Worker's Compensation and Unemployment Compensation are not included in this array because these programs are collected for the respondent only.

IV. Schooling Event History Arrays

Yearly Schooling Variables: A set of schooling variables provides information for each year beginning in 1980, the year when the first information is available in the survey, through 1998. These education arrays are somewhat different than the other event history arrays. Information on a respondent's education is reported on a yearly basis, rather than monthly or weekly. This approach is used to combine information from the youth questionnaire, which collects more detailed data, and from the round 1 parent questionnaire, which presented information only for each year. In general, these variables refer to the school year rather than the calendar year. That is, 1991 in a variable title or in the data for a variable generally indicates the school year starting in fall 1991 and ending in spring 1992.

Users should be aware that, because questions were not identical in the round 1 parent questionnaire and the round 2 youth questionnaire, the transition between the two data sources was not seamless and some information for the yearly variables had to be imputed. If they feel that a given value is questionable, researchers may wish to compare created variables to the raw data and to the monthly schooling arrays described below.

1. SCH_YEAR_TO_GRADE

This array presents the grade the respondent attended during the school year. The last two digits of the question name indicate the school year. For example, SCH_YEAR_TO_GRADE.90 refers to the grade attended by the respondent during the school year that starts in fall 1990 and ends in spring 1991.

2. SCH_GRADE_TO_YEAR

This array refers to the year the respondent attended a certain grade. For example, if the respondent attended second grade in 1992–93, then SCH_GRADE_TO_YEAR.2 would have the value 1992.

3. SCH_CHANGES

This array counts the number of times the respondent changed the school attended during the school year. For example, SCH_CHANGES.90 shows how many different schools the respondent attended during the school year that started in fall 1990 and ended in spring 1991.

4. SCH_MNTHS_MISSED

This array presents the number of months during the school year that the respondent did not attend school. For example, if SCH_MNTHS_MISSED.90 has a value of 3 for a respondent, then that respondent had a gap in attendance of three months during the school year that started in the fall of 1990

and ended in the spring of 1991. A gap is defined as missing school for one or more months (not including summer vacation); gaps do not have to be consecutive.

5. SCH_SUMMER_SCHOOL

This array refers to extra school classes during an educational break in a given school year, such as summer school. For example, SCH_SUMMER_SCHOOL.90 shows whether the respondent attended school during a break in the 1990–91 school year.

6. SCH_GRADE_PROGRESS

This array has positive values if there are any special events that occurred during the school grade. For example, a positive value in SCH_GRADE_PROGRESS.2 indicates that the respondent was skipped or demoted during second grade. Researchers should note that parents might have been confused as to how to answer the skip grade questions asked during the interview. For example, there are parents who say their child skipped from 5th to 6th grade, while others say from 4th to 6th grades. Both of these cases are probably stating that the child missed most or all of the 5th grade. To resolve this ambiguity, the code states that if a child is skipped consecutive years then the first year (i.e. 5th grade) was missed. If a parent reports non-consecutive years (i.e. 4th to 6th) then the program assumes the year(s) in the middle are the ones not attended.

7. SCH_YEAR_PROGRESS

This array refers to any special events that occurred during the school year. The question name's last two digits indicate the school year this variable refers to. For example, SCH_YEAR_PROGRESS.90 shows special events that occurred during the school year that starts in fall 1990 and ends in spring 1991. The special events, such as grades skipped or demoted to, are defined in the same way as in the previous array.

8. SCH_SUSPENSIONS

This array counts the number of days during the school year the respondent was suspended from school. For example, if SCH_SUSPENSIONS.90 has a value of 3 then the respondent was suspended from school 3 days during the school year that started in fall 1990 and ended in spring 1991.

Monthly Schooling Variables. In round 2, 3 types of monthly arrays and one flag variable (SCH_DUAL_1998) were created. Each array captures information for each month from the respondent's interview date in round 1 to the interview date in round 2.

1. SCH_STATUS

This array reports the respondent's enrollment status during each month from the round 1 interview date through the current interview date. Coding categories include unknown, not enrolled, in grades K to 12, in college, on vacation, expelled, and other.

2. SCH_TERM

These variables report the respondent's school type and grade for each month in the time period. The first two digits represent the type of school (public = 10, private = 20, and religious = 30). The last two digits provide the respondent's grade in school (1–12) or year in college (1–8). To determine whether the school is a K–12 school or a college, researchers are advised to combine this variable with the SCH_STATUS variable described above.

3. SCH_ID

This variable permits users to link array information to the school roster in the main data file and access other information about the school. The first two digits represent the survey year the respondent first reported attending the school. The last two digits provide the number of the school on the current survey year's roster. For example, a value of 9701 indicates that the school was first reported in round 1 and is school #01 on the round 2 school roster.

4. SCH_DUAL_1998

A small number of NLSY97 respondents went to two different schools in the same month. Because only the first school can be reported in the other arrays, this variable flags these special cases. There is only one variable for each school for the period between the round 1 and 2 interviews; the exact month when the overlap occurred is not indicated, and overlap may have occurred in more than one month.

| Flag | Definition |
|--------|--|
| Flag=1 | More than one K–12 school attended in the same month |
| Flag=2 | More than one college attended in the same month |
| Flag=3 | More than one K–12 school attended in the same month AND More than one college attended in the same month |
| Flag=4 | K–12 school and college attended in the same month |
| Flag=5 | More than one K–12 school attended in the same month AND K–12 school and college attended in the same month |
| Flag=6 | More than one college attended in the same month AND K–12 school and college attended in the same month |
| Flag=7 | More than one K–12 school attended in the same month AND more than one college attended in the same month AND K–12 school and college attended in the same month |

NLSY97 Appendix 7:

Continuous Month Scheme and Crosswalk

Continuous Month Timeline

The non-employment event history arrays are presented using two different timelines. The first, a continuous month approach, labels January 1980 as month 1, February 1980 as month 2, and so on. Thus, a respondent born in month 4 might start receiving public assistance in month 193 and leave the program in month 198. Key events occurring during the life of each cohort member can be indexed within this month-by-month structure. To aid users, a number of variables other than the event history arrays have already been created using the event history format. For example, the month of the respondent's birth and the month of the respondent's interview are both presented in continuous month format. Second, the event history arrays can be constructed using actual month and year dates. Using this approach, the same respondent, born in April 1980, started receiving public assistance in January 1997 and left the program in June 1997.

The data set includes variables for the event history arrays for every month number under the continuous month scheme. However, actual month and year dates are only included for a given respondent in months in which transitions occurred. That is, the respondent in the example above would have continuous month variables relating to program participation status for every month beginning with 172 (the month of the respondent's 14th birthday) but would only have actual date variables for January and June 1997. To aid users in moving between the two dating schemes, table 1 in this appendix contains a crosswalk between actual dates and continuous month numbers.

The continuous month system allows researchers to easily compare the times at which various events occurred in the youth's life. The following example illustrates a case in which a youth turns 17 in February 1997 and, in the same month (month 194 in the continuous timeline), begins receiving AFDC. She remains on AFDC for 2 months (month 194 and month 195). The following table illustrates this information using a month-by-month timeline.

Non-Employment Event History Array Incorporating the Month-by-Month Timeline

| Question Name | Participation Status | Month Number (created by the user) | Birth date: CV_CHILD_BIRTH_MONTH |
|-------------------|----------------------|---------------------------------------|-------------------------------------|
| AFDC_STATUS.97.01 | 0 | 193 | |
| AFDC_STATUS.97.02 | 1 | 194 | 194 |
| AFDC_STATUS.97.03 | 1 | 195 | |
| AFDC_STATUS.97.04 | 0 | 196 | |

In addition linking the dates through the crosswalk, any user who selects an event array will be able to extract from the CD additional characteristics that relate to that event. The following table presents the additional information that a user may chose to link to the non-employment event history arrays. A user is able to create these and other variable combinations for any status array.

| Status | Variable 1 | Variable 2 | Variable 3 |
|----------------------|------------------------|-------------------------|-----------------------|
| Enrollment | Highest grade attended | Highest grade completed | Highest degree earned |
| Cohabitation/Marital | Partner's race | Partner's religion | Partner's grade level |
| AFDC | Received during month | Benefit amount | People benefiting |

Continuous Week Timeline

In the event history section of the CD-ROM, an employment history of the number of weeks worked is presented in a continuous week-by-week status array. This array is very similar to the month-by-month scheme used for the rest of the event history variables. In this format, the first week of January 1980 is number week 1, the second week of January 1980 is numbered week 2, and so on; weeks are listed by

exact date as well. This week-by-week array lists each respondent's employment status for each week since age 14.

As with the month-by-month arrays, respondents have data available for every week in the continuous week arrays but only for transition weeks in the actual date arrays. However, users should note that, due to technical considerations, the variable titles in the data number weeks by calendar year rather than from week 1 through week 992. For example, a variable might read "Employment: Employment Status in Week 07 Year 94" instead of "Employment: Employment Status in Week 737." Table 2 in this appendix provides a crosswalk between the actual beginning date of each week, the continuous week number, and the week number for each week in a calendar year. This table begins with January 1994 because this is the first week in which any NLSY97 respondent could have reached age 14, the youngest age for which the employemnt event history variables were created. Users who need to see earlier continuous week numbers should contact NLS User Services.

Naming Conventions

The question names of the NLSY97 event history variables incorporate the dates to which the variables apply. The marital status and program participation array titles include the month and year (e.g., AFDC_AMT.97.12 corresponds to December 1997). Likewise, employment array variables are listed for each week and year (e.g., EMP_STATUS.52.97 corresponds to the 52nd week in 1997). The schooling variables are yearly and employ a slightly different system. In general, these variables refer to the school year rather than the calendar year. That is, 1991 in a variable title or in the data for a variable generally indicates the school year starting in fall 1991 and ending in spring 1992.

Table 1. Continuous Month Crosswalk

| Month and year | Continuous month number | Month and year | Continuous month number |
|-----------------------|--------------------------------|-----------------------|--------------------------------|
| January 1980 | 1 | July 1983 | 43 |
| February 1980 | 2 | August 1983 | 44 |
| March 1980 | 3 | September 1983 | 45 |
| April 1980 | 4 | October 1983 | 46 |
| May 1980 | 5 | November 1983 | 47 |
| June 1980 | 6 | December 1983 | 48 |
| July 1980 | 7 | January 1984 | 49 |
| August 1980 | 8 | February 1984 | 50 |
| September 1980 | 9 | March 1984 | 51 |
| October 1980 | 10 | April 1984 | 52 |
| November 1980 | 11 | May 1984 | 53 |
| December 1980 | 12 | June 1984 | 54 |
| January 1981 | 13 | July 1984 | 55 |
| February 1981 | 14 | August 1984 | 56 |
| March 1981 | 15 | September 1984 | 57 |
| April 1981 | 16 | October 1984 | 58 |
| May 1981 | 17 | November 1984 | 59 |
| June 1981 | 18 | December 1984 | 60 |
| July 1981 | 19 | January 1985 | 61 |
| August 1981 | 20 | February 1985 | 62 |
| September 1981 | 21 | March 1985 | 63 |
| October 1981 | 22 | April 1985 | 64 |
| November 1981 | 23 | May 1985 | 65 |
| December 1981 | 24 | June 1985 | 66 |
| January 1982 | 25 | July 1985 | 67 |
| February 1982 | 26 | August 1985 | 68 |
| March 1982 | 27 | September 1985 | 69 |
| April 1982 | 28 | October 1985 | 70 |
| May 1982 | 29 | November 1985 | 71 |
| June 1982 | 30 | December 1985 | 72 |
| July 1982 | 31 | January 1986 | 73 |
| August 1982 | 32 | February 1986 | 74 |
| September 1982 | 33 | March 1986 | 75 |
| October 1982 | 34 | April 1986 | 76 |
| November 1982 | 35 | May 1986 | 77 |
| December 1982 | 36 | June 1986 | 78 |
| January 1983 | 37 | July 1986 | 79 |
| February 1983 | 38 | August 1986 | 80 |
| March 1983 | 39 | September 1986 | 81 |
| April 1983 | 40 | October 1986 | 82 |
| May 1983 | 41 | November 1986 | 83 |
| June 1983 | 42 | December 1986 | 84 |

Appendix 7: Continuous Month Scheme and Crosswalk

Table 1. Continuous Month Crosswalk (Continued)

| Month and year | Continuous month number | Month and year | Continuous month number |
|----------------|-------------------------|----------------|-------------------------|
| January 1987 | 85 | July 1990 | 127 |
| February 1987 | 86 | August 1990 | 128 |
| March 1987 | 87 | September 1990 | 129 |
| April 1987 | 88 | October 1990 | 130 |
| May 1987 | 89 | November 1990 | 131 |
| June 1987 | 90 | December 1990 | 132 |
| July 1987 | 91 | January 1991 | 133 |
| August 1987 | 92 | February 1991 | 134 |
| September 1987 | 93 | March 1991 | 135 |
| October 1987 | 94 | April 1991 | 136 |
| November 1987 | 95 | May 1991 | 137 |
| December 1987 | 96 | June 1991 | 138 |
| January 1988 | 97 | July 1991 | 139 |
| February 1988 | 98 | August 1991 | 140 |
| March 1988 | 99 | September 1991 | 141 |
| April 1988 | 100 | October 1991 | 142 |
| May 1988 | 101 | November 1991 | 143 |
| June 1988 | 102 | December 1991 | 144 |
| July 1988 | 103 | January 1992 | 145 |
| August 1988 | 104 | February 1992 | 146 |
| September 1988 | 105 | March 1992 | 147 |
| October 1988 | 106 | April 1992 | 148 |
| November 1988 | 107 | May 1992 | 149 |
| December 1988 | 108 | June 1992 | 150 |
| January 1989 | 109 | July 1992 | 151 |
| February 1989 | 110 | August 1992 | 152 |
| March 1989 | 111 | September 1992 | 153 |
| April 1989 | 112 | October 1992 | 154 |
| May 1989 | 113 | November 1992 | 155 |
| June 1989 | 114 | December 1992 | 156 |
| July 1989 | 115 | January 1993 | 157 |
| August 1989 | 116 | February 1993 | 158 |
| September 1989 | 117 | March 1993 | 159 |
| October 1989 | 118 | April 1993 | 160 |
| November 1989 | 119 | May 1993 | 161 |
| December 1989 | 120 | June 1993 | 162 |
| January 1990 | 121 | July 1993 | 163 |
| February 1990 | 122 | August 1993 | 164 |
| March 1990 | 123 | September 1993 | 165 |
| April 1990 | 124 | October 1993 | 166 |
| May 1990 | 125 | November 1993 | 167 |
| June 1990 | 126 | December 1993 | 168 |

Table 1. Continuous Month Crosswalk (Continued)

| Month and year | Continuous month number | Month and year | Continuous month number |
|-----------------------|--------------------------------|-----------------------|--------------------------------|
| January 1994 | 169 | January 1997 | 205 |
| February 1994 | 170 | February 1997 | 206 |
| March 1994 | 171 | March 1997 | 207 |
| April 1994 | 172 | April 1997 | 208 |
| May 1994 | 173 | May 1997 | 209 |
| June 1994 | 174 | June 1997 | 210 |
| July 1994 | 175 | July 1997 | 211 |
| August 1994 | 176 | August 1997 | 212 |
| September 1994 | 177 | September 1997 | 213 |
| October 1994 | 178 | October 1997 | 214 |
| November 1994 | 179 | November 1997 | 215 |
| December 1994 | 180 | December 1997 | 216 |
| January 1995 | 181 | January 1998 | 217 |
| February 1995 | 182 | February 1998 | 218 |
| March 1995 | 183 | March 1998 | 219 |
| April 1995 | 184 | April 1998 | 220 |
| May 1995 | 185 | May 1998 | 221 |
| June 1995 | 186 | June 1998 | 222 |
| July 1995 | 187 | July 1998 | 223 |
| August 1995 | 188 | August 1998 | 224 |
| September 1995 | 189 | September 1998 | 225 |
| October 1995 | 190 | October 1998 | 226 |
| November 1995 | 191 | November 1998 | 227 |
| December 1995 | 192 | December 1998 | 228 |
| January 1996 | 193 | January 1999 | 229 |
| February 1996 | 194 | February 1999 | 230 |
| March 1996 | 195 | March 1999 | 231 |
| April 1996 | 196 | April 1999 | 232 |
| May 1996 | 197 | May 1999 | 233 |
| June 1996 | 198 | June 1999 | 234 |
| July 1996 | 199 | July 1999 | 235 |
| August 1996 | 200 | August 1999 | 236 |
| September 1996 | 201 | September 1999 | 237 |
| October 1996 | 202 | October 1999 | 238 |
| November 1996 | 203 | November 1999 | 239 |
| December 1996 | 204 | December 1999 | 240 |

Appendix 7: Continuous Month Scheme and Crosswalk

Table 2. Continuous Week Crosswalk

| Week start date | Continuous week number | Calendar year week number | Week start date | Continuous week number | Calendar year week number |
|-----------------|------------------------|---------------------------|-----------------|------------------------|---------------------------|
| 12/26/93 | 731 | 1 | 10/30/94 | 775 | 45 |
| 01/02/94 | 732 | 2 | 11/06/94 | 776 | 46 |
| 01/09/94 | 733 | 3 | 11/13/94 | 777 | 47 |
| 01/16/94 | 734 | 4 | 11/20/94 | 778 | 48 |
| 01/23/94 | 735 | 5 | 11/27/94 | 779 | 49 |
| 01/30/94 | 736 | 6 | 12/04/94 | 780 | 50 |
| 02/06/94 | 737 | 7 | 12/11/94 | 781 | 51 |
| 02/13/94 | 738 | 8 | 12/18/94 | 782 | 52 |
| 02/20/94 | 739 | 9 | 12/25/94 | 783 | 53 |
| 02/27/94 | 740 | 10 | 01/01/95 | 784 | 1 |
| 03/06/94 | 741 | 11 | 01/08/95 | 785 | 2 |
| 03/13/94 | 742 | 12 | 01/15/95 | 786 | 3 |
| 03/20/94 | 743 | 13 | 01/22/95 | 787 | 4 |
| 03/27/94 | 744 | 14 | 01/29/95 | 788 | 5 |
| 04/03/94 | 745 | 15 | 02/05/95 | 789 | 6 |
| 04/10/94 | 746 | 16 | 02/12/95 | 790 | 7 |
| 04/17/94 | 747 | 17 | 02/19/95 | 791 | 8 |
| 04/24/94 | 748 | 18 | 02/26/95 | 792 | 9 |
| 05/01/94 | 749 | 19 | 03/05/95 | 793 | 10 |
| 05/08/94 | 750 | 20 | 03/12/95 | 794 | 11 |
| 05/15/94 | 751 | 21 | 03/19/95 | 795 | 12 |
| 05/22/94 | 752 | 22 | 03/26/95 | 796 | 13 |
| 05/29/94 | 753 | 23 | 04/02/95 | 797 | 14 |
| 06/05/94 | 754 | 24 | 04/09/95 | 798 | 15 |
| 06/12/94 | 755 | 25 | 04/16/95 | 799 | 16 |
| 06/19/94 | 756 | 26 | 04/23/95 | 800 | 17 |
| 06/26/94 | 757 | 27 | 04/30/95 | 801 | 18 |
| 07/03/94 | 758 | 28 | 05/07/95 | 802 | 19 |
| 07/10/94 | 759 | 29 | 05/14/95 | 803 | 20 |
| 07/17/94 | 760 | 30 | 05/21/95 | 804 | 21 |
| 07/24/94 | 761 | 31 | 05/28/95 | 805 | 22 |
| 07/31/94 | 762 | 32 | 06/04/95 | 806 | 23 |
| 08/07/94 | 763 | 33 | 06/11/95 | 807 | 24 |
| 08/14/94 | 764 | 34 | 06/18/95 | 808 | 25 |
| 08/21/94 | 765 | 35 | 06/25/95 | 809 | 26 |
| 08/28/94 | 766 | 36 | 07/02/95 | 810 | 27 |
| 09/04/94 | 767 | 37 | 07/09/95 | 811 | 28 |
| 09/11/94 | 768 | 38 | 07/16/95 | 812 | 29 |
| 09/18/94 | 769 | 39 | 07/23/95 | 813 | 30 |
| 09/25/94 | 770 | 40 | 07/30/95 | 814 | 31 |
| 10/02/94 | 771 | 41 | 08/06/95 | 815 | 32 |
| 10/09/94 | 772 | 42 | 08/13/95 | 816 | 33 |
| 10/16/94 | 773 | 43 | 08/20/95 | 817 | 34 |
| 10/23/94 | 774 | 44 | 08/27/95 | 818 | 35 |

Table 2. Continuous Week Crosswalk (Continued)

| Week start date | Continuous week number | Calendar year week number | Week start date | Continuous week number | Calendar year week number |
|-----------------|------------------------|---------------------------|-----------------|------------------------|---------------------------|
| 09/03/95 | 819 | 36 | 07/07/96 | 863 | 28 |
| 09/10/95 | 820 | 37 | 07/14/96 | 864 | 29 |
| 09/17/95 | 821 | 38 | 07/21/96 | 865 | 30 |
| 09/24/95 | 822 | 39 | 07/28/96 | 866 | 31 |
| 10/01/95 | 823 | 40 | 08/04/96 | 867 | 32 |
| 10/08/95 | 824 | 41 | 08/11/96 | 868 | 33 |
| 10/15/95 | 825 | 42 | 08/18/96 | 869 | 34 |
| 10/22/95 | 826 | 43 | 08/25/96 | 870 | 35 |
| 10/29/95 | 827 | 44 | 09/01/96 | 871 | 36 |
| 11/05/95 | 828 | 45 | 09/08/96 | 872 | 37 |
| 11/12/95 | 829 | 46 | 09/15/96 | 873 | 38 |
| 11/19/95 | 830 | 47 | 09/22/96 | 874 | 39 |
| 11/26/95 | 831 | 48 | 09/29/96 | 875 | 40 |
| 12/03/95 | 832 | 49 | 10/06/96 | 876 | 41 |
| 12/10/95 | 833 | 50 | 10/13/96 | 877 | 42 |
| 12/17/95 | 834 | 51 | 10/20/96 | 878 | 43 |
| 12/24/95 | 835 | 52 | 10/27/96 | 879 | 44 |
| 12/31/95 | 836 | 1 | 11/03/96 | 880 | 45 |
| 01/07/96 | 837 | 2 | 11/10/96 | 881 | 46 |
| 01/14/96 | 838 | 3 | 11/17/96 | 882 | 47 |
| 01/21/96 | 839 | 4 | 11/24/96 | 883 | 48 |
| 01/28/96 | 840 | 5 | 12/01/96 | 884 | 49 |
| 02/04/96 | 841 | 6 | 12/08/96 | 885 | 50 |
| 02/11/96 | 842 | 7 | 12/15/96 | 886 | 51 |
| 02/18/96 | 843 | 8 | 12/22/96 | 887 | 52 |
| 02/25/96 | 844 | 9 | 12/29/96 | 888 | 1 |
| 03/03/96 | 845 | 10 | 01/05/97 | 889 | 2 |
| 03/10/96 | 846 | 11 | 01/12/97 | 890 | 3 |
| 03/17/96 | 847 | 12 | 01/19/97 | 891 | 4 |
| 03/24/96 | 848 | 13 | 01/26/97 | 892 | 5 |
| 03/31/96 | 849 | 14 | 02/02/97 | 893 | 6 |
| 04/07/96 | 850 | 15 | 02/09/97 | 894 | 7 |
| 04/14/96 | 851 | 16 | 02/16/97 | 895 | 8 |
| 04/21/96 | 852 | 17 | 02/23/97 | 896 | 9 |
| 04/28/96 | 853 | 18 | 03/02/97 | 897 | 10 |
| 05/05/96 | 854 | 19 | 03/09/97 | 898 | 11 |
| 05/12/96 | 855 | 20 | 03/16/97 | 899 | 12 |
| 05/19/96 | 856 | 21 | 03/23/97 | 900 | 13 |
| 05/26/96 | 857 | 22 | 03/30/97 | 901 | 14 |
| 06/02/96 | 858 | 23 | 04/06/97 | 902 | 15 |
| 06/09/96 | 859 | 24 | 04/13/97 | 903 | 16 |
| 06/16/96 | 860 | 25 | 04/20/97 | 904 | 17 |
| 06/23/96 | 861 | 26 | 04/27/97 | 905 | 18 |
| 06/30/96 | 862 | 27 | 05/04/97 | 906 | 19 |

Appendix 7: Continuous Month Scheme and Crosswalk

Table 2. Continuous Week Crosswalk (Continued)

| Week start date | Continuous week number | Calendar year week number | Week start date | Continuous week number | Calendar year week number |
|-----------------|------------------------|---------------------------|-----------------|------------------------|---------------------------|
| 05/11/97 | 907 | 20 | 03/08/98 | 950 | 11 |
| 05/18/97 | 908 | 21 | 03/15/98 | 951 | 12 |
| 05/25/97 | 909 | 22 | 03/22/98 | 952 | 13 |
| 06/01/97 | 910 | 23 | 03/29/98 | 953 | 14 |
| 06/08/97 | 911 | 24 | 04/05/98 | 954 | 15 |
| 06/15/97 | 912 | 25 | 04/12/98 | 955 | 16 |
| 06/22/97 | 913 | 26 | 04/19/98 | 956 | 17 |
| 06/29/97 | 914 | 27 | 04/26/98 | 957 | 18 |
| 07/06/97 | 915 | 28 | 05/03/98 | 958 | 19 |
| 07/13/97 | 916 | 29 | 05/10/98 | 959 | 20 |
| 07/20/97 | 917 | 30 | 05/17/98 | 960 | 21 |
| 07/27/97 | 918 | 31 | 05/24/98 | 961 | 22 |
| 08/03/97 | 919 | 32 | 05/31/98 | 962 | 23 |
| 08/10/97 | 920 | 33 | 06/07/98 | 963 | 24 |
| 08/17/97 | 921 | 34 | 06/14/98 | 964 | 25 |
| 08/24/97 | 922 | 35 | 06/21/98 | 965 | 26 |
| 08/31/97 | 923 | 36 | 06/28/98 | 966 | 27 |
| 09/07/97 | 924 | 37 | 07/05/98 | 967 | 28 |
| 09/14/97 | 925 | 38 | 07/12/98 | 968 | 29 |
| 09/21/97 | 926 | 39 | 07/19/98 | 969 | 30 |
| 09/28/97 | 927 | 40 | 07/26/98 | 970 | 31 |
| 10/05/97 | 928 | 41 | 08/02/98 | 971 | 32 |
| 10/12/97 | 929 | 42 | 08/09/98 | 972 | 33 |
| 10/19/97 | 930 | 43 | 08/16/98 | 973 | 34 |
| 10/26/97 | 931 | 44 | 08/23/98 | 974 | 35 |
| 11/02/97 | 932 | 45 | 08/30/98 | 975 | 36 |
| 11/09/97 | 933 | 46 | 09/06/98 | 976 | 37 |
| 11/16/97 | 934 | 47 | 09/13/98 | 977 | 38 |
| 11/23/97 | 935 | 48 | 09/20/98 | 978 | 39 |
| 11/30/97 | 936 | 49 | 09/27/98 | 979 | 40 |
| 12/07/97 | 937 | 50 | 10/04/98 | 980 | 41 |
| 12/14/97 | 938 | 51 | 10/11/98 | 981 | 42 |
| 12/21/97 | 939 | 52 | 10/18/98 | 982 | 43 |
| 12/28/97 | 940 | 1 | 10/25/98 | 983 | 44 |
| 01/04/98 | 941 | 2 | 11/01/98 | 984 | 45 |
| 01/11/98 | 942 | 3 | 11/08/98 | 985 | 46 |
| 01/18/98 | 943 | 4 | 11/15/98 | 986 | 47 |
| 01/25/98 | 944 | 5 | 11/22/98 | 987 | 48 |
| 02/01/98 | 945 | 6 | 11/29/98 | 988 | 49 |
| 02/08/98 | 946 | 7 | 12/06/98 | 989 | 50 |
| 02/15/98 | 947 | 8 | 12/13/98 | 990 | 51 |
| 02/22/98 | 948 | 9 | 12/20/98 | 991 | 52 |
| 03/01/98 | 949 | 10 | 12/27/98 | 992 | 1 |

Table 2. Continuous Week Crosswalk (Continued)

| Week start date | Continuous week number | Calendar year week number | Week start date | Continuous week number | Calendar year week number |
|-----------------|------------------------|---------------------------|-----------------|------------------------|---------------------------|
| 1/3/99 | 993 | 2 | 7/4/99 | 1019 | 28 |
| 1/10/99 | 994 | 3 | 7/11/99 | 1020 | 29 |
| 1/17/99 | 995 | 4 | 7/18/99 | 1021 | 30 |
| 1/24/99 | 996 | 5 | 7/25/99 | 1022 | 31 |
| 1/31/99 | 997 | 6 | 8/1/99 | 1023 | 32 |
| 2/7/99 | 998 | 7 | 8/8/99 | 1024 | 33 |
| 2/14/99 | 999 | 8 | 8/15/99 | 1025 | 34 |
| 2/21/99 | 1000 | 9 | 8/22/99 | 1026 | 35 |
| 2/28/99 | 1001 | 10 | 8/29/99 | 1027 | 36 |
| 3/7/99 | 1002 | 11 | 9/5/99 | 1028 | 37 |
| 3/14/99 | 1003 | 12 | 9/12/99 | 1029 | 38 |
| 3/21/99 | 1004 | 13 | 9/19/99 | 1030 | 39 |
| 3/28/99 | 1005 | 14 | 9/26/99 | 1031 | 40 |
| 4/4/99 | 1006 | 15 | 10/3/99 | 1032 | 41 |
| 4/11/99 | 1007 | 16 | 10/10/99 | 1033 | 42 |
| 4/18/99 | 1008 | 17 | 10/17/99 | 1034 | 43 |
| 4/25/99 | 1009 | 18 | 10/24/99 | 1035 | 44 |
| 5/2/99 | 1010 | 19 | 10/31/99 | 1036 | 45 |
| 5/9/99 | 1011 | 20 | 11/7/99 | 1037 | 46 |
| 5/16/99 | 1012 | 21 | 11/14/99 | 1038 | 47 |
| 5/23/99 | 1013 | 22 | 11/21/99 | 1039 | 48 |
| 5/30/99 | 1014 | 23 | 11/28/99 | 1040 | 49 |
| 6/6/99 | 1015 | 24 | 12/5/99 | 1041 | 50 |
| 6/13/99 | 1016 | 25 | 12/12/99 | 1042 | 51 |
| 6/20/99 | 1017 | 26 | 12/19/99 | 1043 | 52 |
| 6/27/99 | 1018 | 27 | 12/26/99 | 1044 | 53 |

NLSY97 Appendix 8:
Instrument Rosters

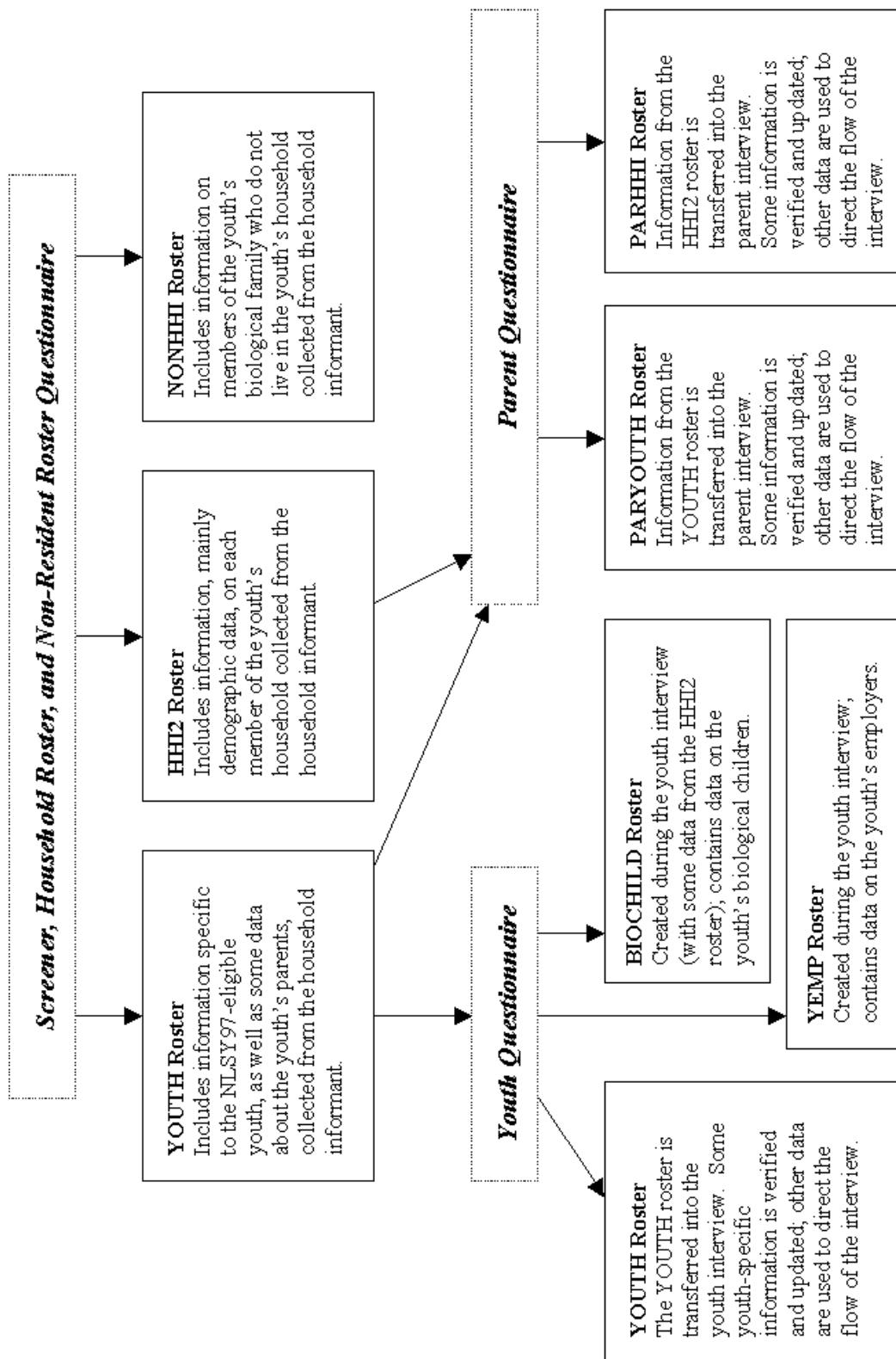
Appendix 8: Instrument Rosters

During the course of the CAPI survey, a number of rosters, or matrices of data, are constructed. These rosters contain one or more pieces of information on a given subject. Rosters are often presented to the interviewers as lists of information that are used to verify information, or from which one of the subjects on the roster is chosen as the answer to a survey question. For example, the EMPLOYER roster (a list of employers for whom the respondent has worked since his or her 14th birthday) is used to verify that the list of employers is accurate. Many of the rosters used during the administration of the survey are presented as consolidated blocks of data on the public release CD-ROM.

In round 1, a number of rosters were used to organize information from the Screener, Household Roster, and Non-Resident Roster Questionnaire. Some of this information was then transferred into the parent and youth interviews for verification and for use in determining question paths. Figure 1 below identifies the key rosters in the round 1 survey and shows how they were used in different parts of the survey.

Because the round 2 interview involved only one main questionnaire (and the paper *Household Income Update*), the construction of rosters is somewhat simpler to understand. The rosters were created during the interview using information provided by the youth and, in some cases, information pertaining to the roster from the previous interview (for example, the HHI item indicating whether a household member was on the round 1 roster or was added in round 2; the FREELANCE item indicating whether a given job was current at the previous interview date).

The following pages list the variable names, titles, and reference numbers for the various instrument rosters used during the round 1 and round 2 interviews. These lists are intended to aid researchers in identifying the types of information organized in each roster and to better follow the flow of information through the interview.



ROUND 1

HHI2 Roster (Household enumeration)

PREFIX = "HHI2" (e.g., HHI2_AGE.01)

| Variable Name | Title: All end in (Scr Ros Item) | Reference # |
|----------------|--|-------------------|
| _AGE.xx | Age of HH Member xx as of Intdate | R10803.–R10818. |
| _ASVAB.xx | Was HH Member xx Flagged for ASVAB | R10819.–R10834. |
| _DOB.xx | Date of Birth of HH Member xx | R10851.–R10865.02 |
| _DADID.xx | ID Number of HH Member xx Bio Dad | R10835.–R10850. |
| _EMPLOYED.xx | Employment Status of HH Member xx | R10898.–R10913. |
| _ENROLLNEXT.xx | Will HH Member xx Be Enrolled Next Fall | R10914.–R10929. |
| _ENROLLSTAT.xx | Is HH Member xx Currently Enrolled | R10930.–R10945. |
| _ETHNICITY.xx | Is HH Member xx Hispanic? | R10946.–R10961. |
| _HHIID.xx | ID of HH Member xx from HHI1 Roster | R10978.–R10993. |
| _HIGHGRADE.xx | HH Member xx Highest Grade Completed | R10994.–R11009. |
| _HHID.xx | HH Member xx ID | R11010.–R11025. |
| _INFORMANT.xx | Is HH Member xx the Informant | R11026.–R11041. |
| _MARSTAT.xx | HH Member xx Marital Status | R11042.–R11057. |
| _MOMID.xx | HH Member xx Bio Moms ID | R11058.–R11073. |
| _PARTNER.xx | HH Member xx Have a Partner | R11122.–R11137. |
| _PARTNERID.xx | ID of HH Member xx Partner | R11138.–R11153. |
| _RACE.xx | Race of HH Member xx | R11154.–R11169. |
| _REL1.xx | Relationship of Person 1 to HH Member xx | R11170.–R11185. |
| _REL2.xx | Relationship of Person 2 to HH Member xx | R11319.–R11334. |
| _REL3.xx | Relationship of Person 3 to HH Member xx | R11342.–R11357. |
| _REL4.xx | Relationship of Person 4 to HH Member xx | R11358.–R11373. |
| _REL5.xx | Relationship of Person 5 to HH Member xx | R11374.–R11389. |
| _REL6.xx | Relationship of Person 6 to HH Member xx | R11390.–R11405. |
| _REL7.xx | Relationship of Person 7 to HH Member xx | R11406.–R11421. |
| _REL8.xx | Relationship of Person 8 to HH Member xx | R11422.–R11437. |
| _REL9.xx | Relationship of Person 9 to HH Member xx | R11438.–R11453. |
| _REL10.xx | Relationship of Person 10 to HH Member xx | R11186.–R11201. |
| _REL11.xx | Relationship of Person 11 to HH Member xx | R11202.–R11217. |
| _REL12.xx | Relationship of Person 12 to HH Member xx | R11218.–R11233. |
| _REL13.xx | Relationship of Person 13 to HH Member xx | R11234.–R11249. |
| _REL14.xx | Relationship of Person 14 to HH Member xx | R11250.–R11265. |
| _REL15.xx | Relationship of Person 15 to HH Member xx | R11266.–R11281. |
| _REL16.xx | Relationship of Person 16 to HH Member xx | R11282.–R11297. |
| _REVDOLEL.xx | Is HH Member xx DOL Eligible (Revised) | R11454.–R11469. |
| _REVETPEL.xx | Is HH Member xx ETP Eligible (Revised) | R11470.–R11485. |
| _REVSTPEL.xx | Is HH Member xx STP Eligible (Revised) | R11486.–R11501. |
| _SEX.xx | Gender of HH Member xx | R11502.–R11517. |
| _SPECIAL.xx | HH Member xx Approved Special Accommodations | R11518.–R11533. |
| _SOPARID.xx | ID of HH Member xx Spouse or Partner | R11541.–R11556. |
| _SPOUSEID.xx | ID of HH Member xx Spouse in HH | R11557.–R11572. |
| _UID.xx | HH Member xx Unique ID | R11621.–R11636. |

Appendix 8: Instrument Rosters

NONHHI Roster (Non-resident relative enumeration)

PREFIX = "NONHHI" (e.g., NONHHI_AGE.01)

| Variable Name | Title: All end in (Scr Ros Item) | Reference # |
|---------------|---|-----------------|
| _AGE.xx | Age Non-Res Member xx at Interview Date, Youth #1 | R11637.-R11659. |
| _DECEASED.xx | Is Non-Res Member xx Deceased, Youth #1 | R11660.-R11682. |
| _DEGREE.xx | Non-Res Member xx Have a Degree, Youth #1 | R11683.-R11703. |
| _EMPLOYED.xx | Employment Status of Non-Res Member xx, Youth #1 | R11704.-R11724. |
| _ETHNICITY.xx | Is Non-Res Member xx Hispanic, Youth #1 | R11725.-R11745. |
| _HIGHGRADE.xx | HGC by Non-Res Member xx, Youth #1 | R11769.-R11789. |
| _MARSTAT.xx | Marital Status of Non-Res Member xx, Youth #1 | R11799.-R11821. |
| _RACE.xx | Race of Non-Res Member xx, Youth #1 | R11845.-R11865. |
| _RELATION.xx | Relationship of Non-Res Member xx to Youth #1 | R11866.-R11883. |
| _SEX.xx | Gender of Non-Res Member xx, Youth #1 | R11884.-R11906. |
| _UID.xx | Unique ID of Non-Res Member xx, Youth #1 | R11907.-R11929. |

BIOCHILD Roster (Youth's children)

PREFIX = "BIOCHILD" (e.g., BIOCHILD_BDATE.01)

| Variable Name | Title: All end in (Ros Item) | Reference # |
|---------------|--|-------------------|
| _BDATE.xx | Birthdate of R Bio Child xx, Final | R05202.-R05203.02 |
| _DEAD.xx | Is R Bio Child xx Deceased, Final | R05204.-R05205. |
| _RESIDE.xx | Does R Bio Child xx Reside in Household, Final | R05212.-R05213. |
| _SEX.xx | Gender of R Bio Child xx, Final | R05214.-R05215. |
| _ID.xx | ID Number of Biochild xx | R05216.-R05217. |

YEMP Roster (Youth's employers)

PREFIX = "YEMP" (e.g., YEMP_CURFLAG.01)

| Variable Name | Title: All end in (Ros Item) | Reference # |
|---------------|---|-------------------|
| _CURFLAG.xx | Was R Currently Employed at Interview Date Job xx | R05252.-R05258. |
| _INTERN.xx | Is This an Internship Employer Job xx | R05265.-R05271. |
| _STARTDATE.xx | Employer Start Month/Day/Year Job xx | R05297.-R05303.02 |
| _STOPDATE.xx | Employer Stop Month/Day/Year Job xx | R05304.-R05310.02 |
| _UID.xx | Employer Unique ID Number Job xx | R05311.-R05317. |

Appendix 8: Instrument Rosters

YOUTH Roster (Youth respondent information)
 PREFIX = "YOUTH" (e.g., YOUTH_ADOPDADID.01)

| Variable Name | Title: All end in (Ros Item) | Reference # |
|----------------|--|-------------|
| _ADOPDADID.01 | R 01 Adoptive Dads ID | R05318. |
| _ADOPMOMID.01 | R 01 Adoptive Moms ID | R05319. |
| _BOTHBIO.01 | Does R 01 Live with Both Bio Parents? | R05322. |
| _DADID.10 | R 01 Bio Dads ID | R05323. |
| _EMANCIPAT.01 | Is R 01 Emancipated? | R05326. |
| _FOSTDADID.01 | R 01 Foster Dads ID | R05327. |
| _FOSTMOMID.01 | R 01 Foster Moms ID | R05328. |
| _GRADE.01 | R 01 Current Grade | R05329. |
| _HHADOPTKID.01 | Does R 01 Have Any Adopted Kids in HH? | R05330. |
| _HHBIOKID.01 | Does R 01 Have Any Bio Kids in HH? | R05331. |
| _HHID.01 | R 01 HH Id Number | R05332. |
| _HHSTEPKID.01 | Does R 01 Have Any Step Kids? | R05333. |
| _ID.01 | R 01 ID Number | R05334. |
| _MOMID.01 | R 01 Bio Moms ID | R05336. |
| _NONR1DEAD.01 | Is R 01 1 st Non-Resp Bio Parent Deceased? | R05338. |
| _NONR1ID.01 | ID of R 01 1 st Non-Resp Bio Parent | R05339. |
| _NONR1INHH.01 | Does 1 st Non-Resp Bio Parent of R 01 Live in HH? | R05340. |
| _NONR1SEX.01 | Gender of R 01 1 st Non-Resp Bio Parent | R05342. |
| _NONR2DEAD.01 | Is R 01 2 nd Non-Resp Bio Parent Deceased? | R05343. |
| _NONR2ID.01 | R 01 2 nd Non-Resp Bio Parents ID | R05344. |
| _NONR2INHH.01 | Is R 01 2 nd Non-Resp Parent in HH | R05345. |
| _NONR2SEX.01 | Gender of R 01 2 nd Non-Resp Bio Parent | R05347. |
| _NRADOPTKID.01 | Does R 01 Have Any Non-Resident Adopted Kids | R05348. |
| _NRBIOKID.01 | Does R 01 Have Any Non-Resident Bio Kids? | R05349. |
| _NRDADID.01 | ID of R 01 Non-Resident Bio Dad | R05350. |
| _NRMOMID.01 | ID of R 01 Non-Resident Bio Mom | R05351. |
| _NRSTEPKID.01 | Does R 01 Have Any Non-Resident Step Kids | R05352. |
| _PARENT.01 | Relationship of Resp Parent to R 01 | R05353. |
| _PARENTGUAR.01 | Does R 01 Have a Resp Parent or Guardian in HH | R05354. |
| _PARENTID.01 | ID of R 01 Resp Parent | R05355. |
| _PARENTSEX.01 | Gender of R 01 Resp Parent | R05356. |
| _SOPARID.01 | ID of R 01 Spouse or Partner | R05358. |
| _STEPDADID.01 | ID of R 01 Step Dad | R05359. |
| _STEPSMOMID.01 | ID of R 01 Step Mom | R05360. |

Appendix 8: Instrument Rosters

PARHHI Roster (Household information in parent interview)

PREFIX = "PARHHI" (e.g., PARHHI_AGE.01)

| Variable Name | Title: All end in (Par Ros Item) | Reference # |
|----------------|--|-------------------|
| _AGE.xx | Age of HH Member xx as of Interview Date | R06945.-R06953. |
| _AGEDOL.xx | Age of HH Member xx as of 12/31/1996 | R06954.-R06962. |
| _DADID.xx | Member xx Bio Dads ID | R06963.-R06971. |
| _DOB.xx | Member xx Date of Birth | R06972.-R06980.02 |
| _DOLEL.xx | Is HH Member xx DOL Eligible (Preliminary) | R06981.-R06989 |
| _ELIGIBLE.xx | Member xx DOL, ETP or STP Eligible | R06990.-R06998. |
| _EMPLOYED.xx | Employment Status of HH Member xx | R06999.-R07001. |
| _ENROLLSTAT.xx | Is HH Member xx Currently Enrolled | R07002.-R07009. |
| _ETPEL.xx | Is HH Member xx ETP Eligible (Preliminary) | R07010.-R07018. |
| _GRADE.xx | Member xx Current Grade | R07019.-R07027. |
| _HIGHGRADE.xx | Member xx Highest Grade Completed | R07028.-R07036. |
| _ID.xx | ID of HH Member xx | R07037.-R07045. |
| _MARSTAT.xx | Member xx Marital Status | R07046.-R07054. |
| _MOB.xx | Member xx Month of Birth | R07055.-R07063. |
| _MOMID.xx | Member xx Bio Moms ID | R07064.-R07072. |
| _PARTNER.xx | Member xx Have a Partner? | R07082.-R07090. |
| _RACE.xx | Race of HH Member xx | R07091.-R07099. |
| _REL1.xx | Relationship of Person 1 to HH Member xx | R07100.-R07108. |
| _REL10.xx | Relationship of Person 10 to HH Member xx | R07109.-R07117. |
| _REL11.xx | Relationship of Person 11 to HH Member xx | R07118.-R07126. |
| _REL12.xx | Relationship of Person 12 to HH Member xx | R07127.-R07135. |
| _REL13.xx | Relationship of Person 13 to HH Member xx | R07136.-R07144. |
| _REL14.xx | Relationship of Person 14 to HH Member xx | R07145.-R07153. |
| _REL15.xx | Relationship of Person 15 to HH Member xx | R07154.-R07162. |
| _REL16.xx | Relationship of Person 16 to HH Member xx | R07163.-R07167. |
| _REL17.xx | Relationship of Person 17 to HH Member xx | R07168.-R07172. |
| _REL18.xx | Relationship of Person 18 to HH Member xx | R07173.-R07177. |
| _REL19.xx | Relationship of Person 19 to HH Member xx | R07178.-R07182. |
| _REL2.xx | Relationship of Person 2 to HH Member xx | R07183.-R07191. |
| _REL20.xx | Relationship of Person 20 to HH Member xx | R07192.-R07196. |
| _REL3.xx | Relationship of Person 3 to HH Member xx | R07197.-R07205. |
| _REL4.xx | Relationship of Person 4 to HH Member xx | R07206.-R07214. |
| _REL5.xx | Relationship of Person 5 to HH Member xx | R07215.-R07223. |
| _REL6.xx | Relationship of Person 6 to HH Member xx | R07224.-R07232. |
| _REL7.xx | Relationship of Person 7 to HH Member xx | R07233.-R07241. |
| _REL8.xx | Relationship of Person 8 to HH Member xx | R07242.-R07250. |
| _REL9.xx | Relationship of Person 9 to HH Member xx | R07251.-R07259. |
| _REVDOLEL.xx | Is HH Member xx DOL Eligible (Revised) | R07260.-R07268. |
| _REVETPEL.xx | Is HH Member xx ETP Eligible (Revised) | R07269.-R07277. |
| _REVSTPEL.xx | Is HH Member xx STP Eligible (Revised) | R07278.-R07286. |
| _SEX.xx | Member xx Sex | R07287.-R07295. |
| _SPOPARID.xx | ID of HH Member xx Spouse or Partner | R07296.-R07304. |
| _STPEL.xx | Is Member xx STP Eligible (Preliminary) | R07305.-R07313. |

Appendix 8: Instrument Rosters

PARYOUTH Roster (Youth information for parent interview)

PREFIX = "PARYOUTH" (e.g., PARYOUTH_AGE.01)

| Variable Name | Title: All end in (Par Ros Item) | Reference # |
|---------------|--|-------------------|
| _ADOPDADID | Rs Adoptive Dads ID | R07314. |
| _ADOPMOMID | Rs Adoptive Moms ID | R07315. |
| _AGE | Age of R as of Interview Date | R07316. |
| _AGEDOL | Age of R as of 12/31/96 | R07317. |
| _BOTHBIO | Does R Live with Both Bio Parents? | R07318. |
| _DADID | Rs Bio Dads ID | R07319. |
| _DOB | Date of Rs Birth | R07320.-R07320.02 |
| _ELIGIBLE | Is R Eligible for DOL, ETP or STP? | R07321. |
| _EMANCIPAT | Is R Emancipated? | R07322. |
| _FOSTDADID | Rs Foster Dads ID | R07323. |
| _FOSTMOMID | Rs Foster Moms ID | R07324. |
| _GRADE | Rs Current Grade | R07325. |
| _HHADOPTKID | Does R Have Any Adopted Kids in HH? | R07326. |
| _HHBIOKID | Does R Have Any Bio Kids in HH? | R07327. |
| _HHID | Rs HH ID Number | R07328. |
| _HHSTEPKID | Does R Have Any Step Kids? | R07329. |
| _ID | Rs ID Number | R07330. |
| _MARSTAT | Rs Marital Status | R07331. |
| _MOMID | Rs Bio Moms ID | R07332. |
| _NONR1DEAD | Is Rs 1 st Non-Resp Bio Parent Deceased? | R07333. |
| _NONR1ID | ID of Rs 1 st Non-Resp Bio Parent | R07334. |
| _NONR1INHH | 1 st Non-Resp Bio Parent of R Live in HH? | R07335. |
| _NONR1SEX | Gender of Rs 1 st Non-Resp Bio Parent | R07337. |
| _NONR2DEAD | Is Rs 2 nd Non-Resp Bio Parent Deceased? | R07338. |
| _NONR2ID | Rs 2 nd Non-Resp Bio Parents ID | R07339. |
| _NONR2INHH | Is Rs 2 nd Non-Resp Bio Parent In HH | R07340. |
| _NONR2SEX | Gender of Rs 2 nd Non-Resp Bio Parent | R07342. |
| _NRADOPTKID | Does R Have Any Non-Resident Adopted Kids | R07343. |
| _NRBIOKID | Does R Have Any Non-Resident Bio Kids? | R07344. |
| _NRDADID | ID of Rs Non-Resident Bio Dad | R07345. |
| _NRMOMID | ID of Rs Non-Resident Bio Mom | R07346. |
| _NRSTEPKID | Does R Have Any Non-Resident Step Kids | R07347. |
| _PARENT | Relationship of Resp Parent to R | R07348. |
| _PARENTGUAR | R Have a Resp Parent or Guardian in HH | R07349. |
| _PARENTID | ID of Rs Resp Parent | R07350. |
| _PARENTSEX | Gender of Rs Resp Parent | R07351. |
| _SEX | Gender of R | R07352. |
| _SOPARID | ID of Rs Spouse or Partner | R07353. |
| _STEPDADID | ID of Rs Stepdad | R07354. |
| _STEPSMOMID | ID of Rs Stepmom | R07355. |

ROUND 2

HHI Roster (Household enumeration)

PREFIX = "HHI" (e.g., HHI_AGE.01)

| Variable Name | Title: All begin with HHI and end in (Ros Item) | Reference # |
|----------------|---|-----------------|
| _AGE.xx | Age of Household Member xx as of Intdate | R23999.-R24012. |
| _DEGREE.xx | HH Member xx Highest Degree Earned | R24013.-R24026. |
| _EMPLOYED.xx | HH Member xx Employment Status | R24032.-R24045. |
| _ENROLLSTAT.xx | Is HH Member xx Currently Enrolled | R24046.-R24050. |
| _ETHNICITY.xx | HH Member xx Ethnicity | R24051.-R24064. |
| _HHFLAG.xx | HH Member xx Status | R24065.-R24078. |
| _HIGHGRADE.xx | HH Member xx Highest Grade Completed | R24079.-R24092. |
| _UID.xx | HH Member xx Unique ID | R24093.-R24106. |
| _MARSTAT.xx | HH Member xx Marital Status | R24121.-R24134. |
| _RACE.xx | HH Member xx Race | R24149.-R24162. |
| _RELY.xx | HH Member xx Relationship to R (Numeric) | R24163.-R24176. |
| _SEX.xx | HH Member xx Gender | R24191.-R24204. |

NONHHI Roster (Non-resident relative enumeration)

PREFIX = "NONHHI" (e.g., NONHHI_AGE.01)

| Variable Name | Title: All begin with NONHHI and end in (Ros Item) | Reference # |
|---------------|--|-----------------|
| _AGE.xx | Age of Nonres Member xx as of Intdate | R24205.-R24230. |
| _ETHNICITY.xx | Is Nonres Member xx of Hispanic Origin? | R24257.-R24277. |
| _UID.xx | Nonres Member xx Unique ID | R24304.-R24329. |
| _HHFLAG.xx | Nonres Member xx Status | R24330.-R24355. |
| _MARSTAT.xx | Nonres Member xx Marital Status | R24357.-R24382. |
| _RACE.xx | Nonres Member xx Race | R24409.-R24429. |
| _RELY.xx | Nonres Member xx Relationship to R (Numeric) | R24430.-R24446. |
| _SEX.xx | Nonres Member xx Gender | R24473.-R24494. |

NEWSCHOOL Roster (Respondent's schools)

PREFIX = "NEWSCHOOL" (e.g., NEWSCHOOL_PERIODS.01)

| Variable Name | Title: All begin with NEWSCHOOL and end in (Ros Item) | Reference # |
|---------------|---|---------------------|
| _PERIODS.xx | Number of Times R Enrolled in School xx | R24605.-R24610. |
| _START1.xx | Month/Year R Start 1st Enrollment in School xx | R24611.00-R24616.01 |
| _START2.xx | Month/Year R Start 2nd Enrollment in School xx | R24617.00-R24620.01 |
| _START3.xx | Month/Year R Start 3rd Enrollment in School xx | R24621.00-R24621.01 |
| _STOP1.xx | Month/Year R End 1st Enrollment in School xx | R24622.00-R24627.01 |
| _STOP2.xx | Month/Year R End 2nd Enrollment in School xx | R24628.00-R24631.01 |
| _STOP3.xx | Month/Year R End 3rd Enrollment in School xx | R24632.00-R24632.01 |
| _SCHCODE.xx | School Code Elementary, Middle, High, College | R24633.-R24638. |
| _INTERVIEW.xx | Which Survey Round School xx Reported in | R24639.-R24644. |
| _TYPE.xx | Type of School xx R has Attended | R24645.-R24650. |
| _PUBID.xx | PUBID of School xx R has Attended | R24651.-R24656. |

Appendix 8: Instrument Rosters

YEMP Roster (Respondent's employers)

PREFIX = "YEMP" (e.g., YEMP_CURFLAG.01)

| Variable Name | Title: All begin with YEMP and end in (Ros Item) | Reference # |
|---------------|--|---------------------|
| _CURFLAG.xx | Is R Currently Working for Employer xx | R24681.–R24689. |
| _MILFLAG.xx | Is Employer xx a Military Employer | R24690.–R24698. |
| _MILCODE.xx | Employer xx Military Code | R24699.–R24706. |
| _STARTDATE.xx | Month/Year Start Working for Employer xx | R24725.00–R24733.02 |
| _STOPDATE.xx | Month/Year Stop Working for Employer xx | R24734.00–R24742.02 |
| _INTERN.xx | Employer xx is an Internship | R24752.–R24760. |
| _UID.xx | Employer xx Unique ID | R24761.–R24769. |

FREELANCE Roster (Respondent's freelance jobs)

PREFIX = "FREELANCE" (e.g., FREELANCE_JOBS-COD.01)

| Variable Name | Title: All begin with FREELANCE and end in (Ros Item) | Reference # |
|----------------|---|---------------------|
| _JOBS-COD.xx | Job Code Frame xx | R24776.–R24781. |
| _STARTDATEL.xx | Start Date Reported at DLI Job xx | R24782.00–R24782.01 |
| _STARTDATEC.xx | Start Date Current Freelance Job xx | R24783.00–R24787.01 |
| _STOPDATECU.xx | Stop Date Current Freelance Job xx | R24788.00–R24793.01 |
| _CURATLI.xx | Job was Current at DLI Job xx | R24794.–R24798. |
| _CURRNOW.xx | Job xx is Current Freelance Job | R24799.–R24804. |
| _STARTDATEI.xx | Start Date Info Job xx | R24805.–R24809. |

BIOCHILD Roster (Respondent's children)

PREFIX = "BIOCHILD" (e.g., BIOCHILD_SEX.01)

| Variable Name | Title: All begin with BIOCHILD and end in (Ros Item) | Reference # |
|---------------|--|---------------------|
| _SEX.xx | Bio Child xx Sex | R24906.–R24908. |
| _BDATE.xx | Bio Child xx Bdate | R24909.00–R24911.02 |
| _DEAD.xx | Bio Child is Deceased | R24912.–R24914. |
| _RESIDE.xx | Bio Child xx Resides in Rs Household | R24915.–R24917. |
| _ID.xx | Bio Child xx ID | R24918.–R24920. |
| _UID.xx | Bio Child Unique Household ID Number xx ID | R24921.–R24923. |