NCES 2020-439 U.S. DEPARTMENT OF EDUCATION

2015–16 National Postsecondary Student Aid Study (NPSAS:16)

Supplemental Documentation of Selected Variables Related to Financial Aid

NATIONAL CENTER FOR EDUCATION STATISTICS
Institute of Education Sciences

2015–16 National Postsecondary Student Aid Study (NPSAS:16)

Supplemental Documentation of Selected Variables Related to Financial Aid

APRIL 2020

Annaliza Nunnery
Rachel Burns
T. Austin Lacy
Johnathan Conzelmann
Robin R. Henke
David Radwin
RTI International

Tracy Hunt-White
Project Officer
National Center for Education Statistics

NCES 2020-439
U.S. DEPARTMENT OF EDUCATION

NPSAS

ies NATIONAL CENTER FOR EDUCATION STATISTICS
Institute of Education Sciences

U.S. Department of Education
Betsy DeVos
Secretary

Institute of Education Sciences
Mark Schneider
Director

National Center for Education Statistics
James L. Woodworth
Commissioner

The National Center for Education Statistics (NCES) is the primary federal entity for collecting, analyzing, and reporting data related to education in the United States and other nations. It fulfills a congressional mandate to collect, collate, analyze, and report full and complete statistics on the condition of education in the United States; conduct and publish reports and specialized analyses of the meaning and significance of such statistics; assist state and local education agencies in improving their statistical systems; and review and report on education activities in foreign countries.

NCES activities are designed to address high-priority education data needs; provide consistent, reliable, complete, and accurate indicators of education status and trends; and report timely, useful, and high-quality data to the U.S. Department of Education, the Congress, the states, other education policymakers, practitioners, data users, and the general public. Unless specifically noted, all information contained herein is in the public domain.

We strive to make our products available in a variety of formats and in language that is appropriate to a variety of audiences. You, as our customer, are the best judge of our success in communicating information effectively. If you have any comments or suggestions about this or any other NCES product or report, we would like to hear from you. Please direct your comments to

NCES, IES, U.S. Department of Education
Potomac Center Plaza (PCP)
550 12th Street, SW
Washington, DC 20202

April 2020

The NCES Home Page address is https://nces.ed.gov.
The NCES Publications and Products address is https://nces.ed.gov/pubsearch.

This publication is only available online. To download, view, and print the report as a PDF file, go to the NCES Publications and Products address shown above.

This report was prepared for the National Center for Education Statistics under Contract No. ED-IES-12-C-0095 with RTI International. Mention of trade names, commercial products, or organizations does not imply endorsement by the U.S. Government.

Suggested Citation
Nunnery, A., Burns, R., Lacy, T.A., Conzelmann, J.G., Henke, R.R., and Radwin, D. (2020). *Financial Aid Measures* (NCES 2020-439). U.S. Department of Education. Washington, DC: National Center for Education Statistics. Retrieved [date] from https://nces.ed.gov/pubsearch/.

Content Contact
National Center for Education Statistics
NCES.info@ed.gov

Contents

PAGE

List of Figures in Appendix B vi

Introduction 1

Variables 4

- AGE 4
- AGEGROUP 4
- AIDSECT 5
- AIDSECTG 6
- ATTNSTAT 7
- BORAMT1 8
- BORAMT2 9
- BORAMT3 10
- BUDGETAJ 10
- CAMPAMT 13
- CINCOME 14
- DEPANY 15
- DEPEND 16
- DEPEND2 17
- DEPINC 18
- DODAMT 19
- EFC 21
- EMPLWAIV 30
- EMPLYAM3 32
- ENLEN 35
- FEDCUM1 36
- FEDCUM2 37
- FEDCUM3 38
- FEDDUE1 39
- FEDDUE2 40
- FEDDUE3 42
- FEDOWE1 43
- FEDOWE2 44
- FEDOWE3 46
- GPLUSAMT 47
- GRADDEG 47
- GRASTAMT 50
- GRGRDAMT 51
- GRINFEL 54

PAGE

GRRESAMT 57
GRTEAAMT 60
INATHAMT 62
INCOME 64
INCOMEG 66
INDEPINC 67
INGRTAMT 69
INLNAMT 70
INSMERIT 72
INSMILAMT 72
INSTAMT 74
INSTNEED 75
INSTNOND 77
INSTNOND1 77
INSTWRK 78
INSVETAMT 80
INSWAIV 82
MFT 82
MILTYPE 83
MILTYPE2 84
NETCST1 86
NETCST3 87
NETCST9 87
NFEDCUM1 88
NFEDCUM2 92
OTHFDGRT 96
OTHGTAMT 99
OTHRSCR 99
OTHTYPE 100
PELLAMT 101
PERKAMT 102
PLUSAMT 103
PRIVAID 104
PRIVLOAN 107
SEOGAMT 108
SMARITAL 109
STAFFAMT 110
STAFSUB 111
STAFUNSB 112
STATEAMT 112
STATNEED 113
STATNOND 114
STFCUM1 115

PAGE

STFCUM2 116
STFCUM3 117
STGTAMT 118
STLNAMT 119
STMERIT 120
STMILAMT 122
STNDMRT 123
STNDONLY 125
STNOND1 126
STUDMULT 128
STVETAMT 129
STWKAMT 131
STYPELST 132
T4LNAMT1 134
T4LNAMT2 135
TEACHGRT 136
TFEDAID 136
TFEDGRT 138
TFEDLN 139
TFEDLN2 140
TFEDWRK 141
TITIVAMT 142
TOTAID 143
TOTGRT 144
TOTLOAN 144
TOTWKST 145
TUITION2 146
UGDEG 155
VADODAMT 158
VETBEN 159
VOCHELP 160

References 163

Appendix A – Index of Variables by Variable Label A-1

Appendix B – Flowcharts Showing Relationships Among Components of the Total Student Aid Variable (TOTAID) B-1

List of Figures in Appendix B

Figure 1. Components of TOTAID for undergraduates in NPSAS:16, by aid type B-3

Figure 2. Components of TOTAID for graduate students in NPSAS:16, by aid type B-4

Figure 3. Components of TOTGRT for undergraduates in NPSAS:16, by aid source B-5

Figure 4. Components of TOTGRT for graduate students in NPSAS:16, by aid source B-6

Figure 5. Components of TOTLOAN in NPSAS:16, by source.. B-7

Figure 6. Components of TOTWKST in NPSAS:16, by source B-8

Figure 7. Components of OTHTYPE in NPSAS:16, by type and source B-9

Figure 8. Components of TOTAID for undergraduates in NPSAS:16, by source............. B-10

Figure 9. Components of TOTAID for graduate students in NPSAS:16, by source.......... B-11

Figure 10. Components of TFEDAID in NPSAS:16, by type.. B-12

Figure 11. Components of STATEAMT in NPSAS:16, by type B-13

Figure 12. Components of INSTAMT for undergraduates in NPSAS:16, by type B-14

Figure 13. Components of INSTAMT for graduate students in NPSAS:16, by type B-15

Figure 14. Components of OTHRSCR in NPSAS:16, by type source B-16

Introduction

This report provides detailed information on key variables related to students' financial aid in the 2015–16 National Postsecondary Student Aid Study (NPSAS:16), supplementing information provided in the *Data File Documentation* (Wine, Siegel, and Stollberg 2018). Key variables include aid variables such as total student aid (TOTAID) and its components, as well as variables that are closely related to financial aid eligibility or financial need, e.g., dependency status (DEPEND), student budget (BUDGETAJ), and measures of student and parent income. This set of key variables also encompasses aid-related variables that NCES DataLab (PowerStats, QuickStats, and TrendStats) users have often included in their analyses. Examples of these variables include cumulative amount borrowed for undergraduate education (BORAMT1) and cumulative federal loan amount for undergraduate education (FEDCUM1).

Historically, most of the information contained in this report has been available in documents that accompany the Restricted Use File for each administration of NPSAS, i.e., codebooks, a README (a text file describing the data files), and the SAS programs used to create derived variables. This report consolidates variable-specific information in a single document and provides usage statistics, which indicate the number of times each variable has been used to generate NPSAS:16 estimates in DataLab through November 25, 2019; a programming narrative, which informs non-SAS users about decisions made in variable creation; and SAS code used to generate the variable. The exhibit below displays the fields of information provided for each variable and describes each field.

Variable Name	**Name of the variable**
Variable Label	Label for the variable
Description	Short description of the variable
Assigned Values	For categorical variables, the value labels define the categories. For continuous variables, value labels define 0 (labeled as "zero") and positive values (labeled as "continuous"). Where applicable, values for reserve codes, i.e., values assigned to indicate that a value is missing and why, are also noted.

Data Source(s)	NPSAS variables are derived from any of the following sources: - NPSAS Student Records (SR)–data collected from institutions - NPSAS Student Interviews - National Student Loan Data System (NSLDS) - Free Application for Federal Student Aid (FAFSA), also referred to as the Central Processing System (CPS) (which stores information provided on the FAFSA) - National Student Clearinghouse (NSC), a nonprofit organization that collects enrollment and completion records from participating institutions - Veterans Benefits Administration (VBA), an agency of the U.S. Department of Veterans Affairs, responsible for administering the Department's programs that provide financial and other forms of assistance to veterans, their dependents, and survivors - Integrated Postsecondary Education Data System (IPEDS), a system of interrelated surveys conducted annually by the National Center for Education Statistics (NCES), a part of the Institute for Education Sciences within the U.S. Department of Education - College Board (publisher of the SAT test) - ACT (publisher of the ACT test) When values for a derived variable (e.g., student's age) are available from multiple raw data sources, this field documents those data sources in the order in which they were used to create the variable. For example, when AGE was constructed, age values were first taken from the student's FAFSA. If the FAFSA file did not include an age value for the student, a value was taken from the NPSAS interview. If no age value was available from the interview, a value was taken from the NPSAS institution's administrative records. This field does not apply for variables made from derived variables.
Component Derived Variables	Lists derived variables used to create the variable. For example, derived variables for month-level enrollment are used to calculate the number of months enrolled. The words "Not applicable" in this field indicate that no derived variables were used in the variable's construction. Not every component derived variable is available in every NPSAS study. For example, BAYEARM (year and month received bachelor's degree) was first introduced in NPSAS:16. Previous NPSAS studies used a different component variable such as BAYEAR (year received bachelor's degree). In some entries, this section mentions component derived variables that are not included in this report. Information on these variables may be found in the NPSAS:16 undergraduate codebook, https://nces.ed.gov/datalab/powerstats/codebook.aspx?dataset=121&type=subject, and the NPSAS:16 graduate student codebook, https://nces.ed.gov/datalab/powerstats/codebook.aspx?dataset=122&type=subject.
Applies to	Indicates the population of undergraduate/graduate students to whom the variable applies (e.g., undergraduates only).
Availability	Indicates the previous NPSAS studies (starting with NPSAS:96) in which the variable exists.
Change History	Indicates a change in the name of the variable or decisions regarding its derivation. For variables where the derivation has changed, this change is noted. There are three general changes not included in the entries for individual aid variables. First, beginning with NPSAS:16, the minimum amount for aid variables was reduced from $100 to $1. Second, the maximum amount for some aid variables has varied across NPSAS studies. These changes are shown in the codebooks for the respective studies. Third, in NPSAS studies prior to NPSAS:16, certain state grants that were administered by institutions were classified as institutional grants. This practice was changed in NPSAS:16 to treat institution-administered state grants as state aid, similar to how federal campus-based aid programs are classified by the source of funds. This change mainly affects the aid of undergraduate students attending public institutions in California. It directly affects the derived variables STATNEED (state need-based grants) and INSTNEED (institutional need-based grants). It indirectly affects derived variables that are based on STATNEED and INSTNEED.

N:16 Usage Statistics	Indicates the number of times the variable has been used to create data or other output on the NCES PowerStats, Quickstats, and Trendstats web tools for the NPSAS:16 data as of November 25, 2019. UG refers to use of the undergraduate version of the variable, and GR refers to use of the graduate version.
Programming Narrative	Short narrative of instructions used to create the variable. This section and the following section may refer to source variables or groups of source variables (e.g., C16DOB, N16ADOBMY, ASTHDOB) that were used to create derived variables but are not available in DataLab or the restricted-use data set. Institutions report most financial aid values within standardized and prespecified categories such as federal work-study awards, tuition waivers for staff and their families, and graduate teaching assistantships. “Custom” financial aid values, which do not fit into these categories, are recorded as separate source variables with names that start with “CFA.” Some entries describe the sequence and method of imputation used to replace missing values with valid values. The most common method, stochastic imputation, starts by matching each individual with a missing value for a particular variable to a group of individuals (“donor cases”) with similar characteristics but who have valid values for the same variable. It then replaces the missing value with the value from a randomly selected donor case. If the individual is known to have received a particular type of aid but the amount is missing, the program sets a minimum imputation flag to ensure that a donor case with a positive value is selected.
SAS Code	SAS code that was used to calculate this variable

In addition to variables indicating whether and how much of various types of aid students received from federal, state, institution, and other sources, the directory includes variables related to students’ eligibility for specific forms of aid (e.g., age, military status, marital status, whether they have children). The directory orders the variables in alphabetical order by variable name.

Appendix A provides an index of the variables discussed in this report in alphabetical order by variable label. For example, the first entry is “Age as of 12/31/2015.” Each entry includes the page number on which the corresponding variable is described.

Appendix B provides graphic representations of the relationships among the components of the total student aid variable (TOTAID), both by the source of aid (federal or state government, institutions, or other sources) and the type of aid (loans, grants, work-study, and other types of aid).

Variables

Variable Name	**AGE**
Variable Label	Age as of 12/31/2015
Description	Student's age as of 12/31/2015.

Assigned Values	**Value** | **Value label**
	c | {Continuous}
	0 | {Zero}

Data Source(s)	FAFSA:16, NPSAS:16 Interview, NPSAS:16 Student Records
Component Derived Variables	Not applicable
Applies to	All study members
Availability	NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16
Change History	Unchanged from NPSAS:96 through NPSAS:16
N:16 Usage Statistics	Number of uses: 4027 (UG); 108 (GR)
Programming Narrative	1. Take values from the data sources in the order listed in Data Source(s). 2. Take the pre-loaded values (collected from institution lists in the sampling phase) from the interview as a last resort (Y_DOB). Take values from age 15 through age 90.

SAS Code	/*-------------------- AGE: Age as of 12/31/2015 --------------------*/
	/*------------------- AGEGROUP: Age groups as of 12/31/2011 -----------------*/
	/* variable used: tage N16ADOBMY Y_DOB ASTHDOB C16DOB (N16ALT30 TAGE) */
	
	/* derive AGE from YYYYMM */
	IF C16DOB>100000 THEN AGECPS=2015-INT(C16DOB/100); /* CPS */
	IF N16ADOBMY>100000 THEN AGESI=2015-INT(N16ADOBMY/100); /* student interview */
	IF ASTHDOB>100000 THEN AGESR=2015-INT(ASTHDOB/100); /* student records */
	IF Y_DOB>100000 THEN AGEPL=2015-INT(Y_DOB/100); /* Preloaded data */
	
	IF 15<=AGECPS<=90 THEN AGE=AGECPS; /* first take CPS */
	ELSE IF 15<=AGESI<=90 THEN AGE=AGESI; /* then student interview */
	ELSE IF 15<=AGESR<=90 THEN AGE=AGESR; /* then student records */
	ELSE IF 15<=AGEPL<=90 THEN AGE=AGEPL; /* last preloaded data */

Variable Name	**AGEGROUP**
Variable Label	Age group as of 12/31/2015
Description	Student's age as of 12/31/2015 (by group)
Assigned Values	**Value** | **Value label** **1** | 15-23 **2** | 24-29 **3** | 30 or above
Data Source(s)	FAFSA:16, NPSAS:16 Interview, NPSAS:16 Student Records

Component Derived Variables	AGE
Applies to	All study members
Availability	NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16
Change History	Unchanged from NPSAS:04 through NPSAS:16
N:16 Usage Statistics	Number of uses: 311 (UG); 112 (GR)

Programming Narrative	Create a categorical variable from AGE, with categories
	a. 15-23;
	b. 24-29;
	c. 30 or above

SAS Code	/* AGEGROUP */
	if 15<=age<=23 then agegroup=1;
	else if 24<=age<=29 then agegroup=2;
	else if age>=30 then agegroup=3;

<table>
<tr><td><b>Variable Name</b></td><td><b>AIDSECT</b></td></tr>
<tr><td><b>Variable Label</b></td><td>NPSAS institution type: Undergraduate (with multiple)</td></tr>
<tr><td><b>Description</b></td><td>Type of NPSAS sample institution attended during the 2015–16 academic year for students who attended only one institution (STUDMULT=1).</td></tr>
<tr><td><b>Assigned Values</b></td><td>
<table>
<tr><th>Value</th><th>Value label</th></tr>
<tr><td>1</td><td>Public less-than-2-year</td></tr>
<tr><td>2</td><td>Public 2-year</td></tr>
<tr><td>3</td><td>Public 4-year non-doc-granting, primarily subbacca</td></tr>
<tr><td>4</td><td>Public 4-year non-doc-granting, primarily bacc</td></tr>
<tr><td>5</td><td>Public 4-year doctorate-granting</td></tr>
<tr><td>6</td><td>Private nonprofit less than-4-year</td></tr>
<tr><td>7</td><td>Private nonprofit 4-year non-doctorate-granting</td></tr>
<tr><td>8</td><td>Private nonprofit 4-year doctorate-granting</td></tr>
<tr><td>9</td><td>Private for-profit less-than-2-year</td></tr>
<tr><td>10</td><td>Private for-profit 2-year</td></tr>
<tr><td>11</td><td>Private for-profit 4-year</td></tr>
<tr><td>12</td><td>Attended more than one institution</td></tr>
</table>
</td></tr>
<tr><td><b>Data Source(s)</b></td><td>IPEDS:16, NPSAS:16 Interview, NSLDS:16, NPSAS:16 Student Records, NSC:16</td></tr>
<tr><td><b>Component Derived Variables</b></td><td>SECTOR11, STUDMULT
Note: SECTOR11 is not included in this report. Information on this variable may be found in the NPSAS:16 undergraduate codebook, https://nces.ed.gov/datalab/powerstats/codebook.aspx?dataset=121&type=subject#SECTOR11.</td></tr>
<tr><td><b>Applies to</b></td><td>All undergraduate study members</td></tr>
<tr><td><b>Availability</b></td><td>NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16</td></tr>
<tr><td><b>Change History</b></td><td>Changed in NPSAS:08, NPSAS:12, NPSAS:16 – changes occured to reflect changes in sector-level variable; NSC has only been a data source since NPSAS:12</td></tr>
</table>

N:16 Usage Statistics	Number of uses: 884 (UG); 0 (GR)
Programming Narrative	1) For students attending one institution, make AIDSECT equal to SECTOR11 2) For students attending multiple institutions, make AIDSECT equal to 12

SAS Code	AIDSECT=SECTOR11;
	
	IF STUDMULT>1 THEN DO;
	  AIDSECT=12; /*Change from 11 (NPSAS12) to 12: More than one institution. */
	END;

<table>
<tr><td><b>Variable Name</b></td><td><b>AIDSECTG</b></td></tr>
<tr><td><b>Variable Label</b></td><td>NPSAS institution type: Graduate (with multiple)</td></tr>
<tr><td><b>Description</b></td><td>Type of NPSAS sample institution attended during the 2015–16 academic year, for students who attended only one institution (STUDMULT=1).</td></tr>
<tr><td><b>Assigned Values</b></td><td>
<table>
<tr><th>Value</th><th>Value label</th></tr>
<tr><td><b>1</b></td><td>Public 4-year non-doctorate-granting</td></tr>
<tr><td><b>2</b></td><td>Public 4-year doctorate-granting</td></tr>
<tr><td><b>3</b></td><td>Private nonprofit 4-year non-doctorate-granting</td></tr>
<tr><td><b>4</b></td><td>Private nonprofit 4-year doctorate-granting</td></tr>
<tr><td><b>5</b></td><td>Private for-profit 4-year</td></tr>
<tr><td><b>6</b></td><td>Attended more than one institution</td></tr>
</table>
</td></tr>
<tr><td><b>Data Source(s)</b></td><td>IPEDS:16, NPSAS:16 Interview, NSLDS:16, NPSAS:16 Student Records; FAFSA:16</td></tr>
<tr><td><b>Component Derived Variables</b></td><td>SECTOR11, STUDMULT, STYPELST
Note: SECTOR11 is not included in this report. Information on this variable may be found in the NPSAS:16 graduate student codebook, https://nces.ed.gov/datalab/powerstats/codebook.aspx?dataset=122&type=subject#SECTOR11.</td></tr>
<tr><td><b>Applies to</b></td><td>All graduate study members</td></tr>
<tr><td><b>Availability</b></td><td>NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16</td></tr>
<tr><td><b>Change History</b></td><td>Unchanged from NPSAS:04 through NPSAS:16</td></tr>
<tr><td><b>N:16 Usage Statistics</b></td><td>Number of uses: 0 (UG); 171 (GR)</td></tr>
<tr><td><b>Programming Narrative</b></td><td>Use AIDSECT; keep only graduate institution types. Since this is a graduate level variable, set to missing for undergraduates.</td></tr>
<tr><td><b>SAS Code</b></td><td>/*AIDSECTG: Institution type - graduate (with multiple) */
/* variable used: SECTOR11 */
IF STYPELST=1 THEN AIDSECTG=-3;   /* undergraduate student */
ELSE DO;
  IF SECTOR11 IN (3,4) THEN AIDSECTG=1; /* public 4-year non-doc */
  ELSE IF SECTOR11=5 THEN AIDSECTG=2; /* public 4-year doc */
  ELSE IF SECTOR11=7 THEN AIDSECTG=3; /* private non-profit 4-year non-doc */
  ELSE IF SECTOR11=8 THEN AIDSECTG=4; /* private non-profit 4-year doc */
  ELSE IF SECTOR11=11 THEN AIDSECTG=5; /* private for profit 4-year */
  IF STUDMULT>1 THEN AIDSECTG=6;   /* more than one institution */
END;</td></tr>
</table>

<table>
<tr><td><b>Variable Name</b></td><td><b>ATTNSTAT</b></td></tr>
<tr><td><b>Variable Label</b></td><td>Attendance pattern</td></tr>
<tr><td><b>Description</b></td><td>Student’s attendance pattern at all institutions attended during the 2015–16 academic year</td></tr>
<tr><td><b>Assigned Values</b></td><td>
<table>
<tr><th>Value</th><th>Value label</th></tr>
<tr><td><b>1</b></td><td>Full-time/full year, 1 institution</td></tr>
<tr><td><b>2</b></td><td>Full-time/full year, 2 or more institutions</td></tr>
<tr><td><b>3</b></td><td>Full-time/part year</td></tr>
<tr><td><b>4</b></td><td>Part-time/full year, 1 institution</td></tr>
<tr><td><b>5</b></td><td>Part-time/full year, 2 or more institutions</td></tr>
<tr><td><b>6</b></td><td>Part-time/part year</td></tr>
</table>
</td></tr>
<tr><td><b>Data Source(s)</b></td><td>NPSAS:16 Student Records, NPSAS:16 Interview, NSC:16, NSLDS:16</td></tr>
<tr><td><b>Component Derived Variables</b></td><td>MFT, ENLEN</td></tr>
<tr><td><b>Applies to</b></td><td>All study members</td></tr>
<tr><td><b>Availability</b></td><td>NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16</td></tr>
<tr><td><b>Change History</b></td><td>Unchanged from NPSAS:2000 through NPSAS:16; was named ATTNST3 in NPSAS:96; NSC has only been a data source since NPSAS:12</td></tr>
<tr><td><b>N:16 Usage Statistics</b></td><td>Number of uses: 1,829 (UG); 193 (GR)</td></tr>
<tr><td><b>Programming Narrative</b></td><td>1. If student enrolled 9 or more months full time during the 2015–16 academic year (where additional months enrolled could be part time) then set to 1 = Full-time/full year, 1 institution
2. If student enrolled less than 9 months during the 2015–16 academic year; and enrolled full time in all of these months then set to 3 = Full-time/part year.
3. If student enrolled 9 or more months during the 2015–16 academic year; but less than 9 months were full time then set to 4 = Part-time/full year, 1 institution.
4. If student enrolled less than 9 months during the 2015–16 academic year; could be enrolled full time or part time, but not all of these months were full time, then set to 6 = Part-time/part year.
5. If student was enrolled in more than one institution and ATTNSTAT = 1, then set to 2 = Full-time/full year, 2 or more institutions.
6. If student was enrolled in more than one institution and ATTNSTAT = 4, then set to 5 = Part-time/full year, 2 or more institutions.</td></tr>
<tr><td><b>SAS Code</b></td><td><pre>
* Source variable mft="Number of full-time months enrolled";
* Source variable enlen="Number of months enrolled";
* Source variable studmult="Number of institutions attended";
***;
* If student enrolled 9 or more months full time during the 2015–16
* academic year (where additional months enrolled could be part time) then
* set to 1 = Full-time/full year, 1 institution;
  if mft >= 9   then attnstat = 1;
* If student enrolled less than 9 months during the 2015–16 academic year;
* and enrolled full time in all of these months then set to
* 3 = Full-time/part year.;
else if mft > 0 & mft = enlen then attnstat = 3;
  * If student enrolled 9 or more months during the 2015–16 academic year;
  * but less than 9 months were full time then set to
</pre></td></tr>
</table>

```
* 4 =  Part-time/full year, 1 institution.;
else if enlen >= 9          then attnstat = 4;
* If student enrolled less than 9 months during the 2015–16 academic year;
* could be enrolled full time or part time, but not all of these months
* were full time, then set to 6 = Part-time/part year.;
else if enlen > 0           then attnstat = 6;

if studmult > 1 then
 do;
  * If student was enrolled in more than one institution and
  * ATTNSTAT = 1, then set to 2 = Full-time/full year, 2 or more
  * institutions.;
  if attnstat = 1 then attnstat = 2;
  * If student was enrolled in more than one institution and
  * ATTNSTAT = 4, then set to 5 = Part-time/full year, 2 or more
  * institutions.;
  if attnstat = 4 then attnstat = 5;
 end;
```

<table>
<tr><td><b>Variable Name</b></td><td><b>BORAMT1</b></td></tr>
<tr><td><b>Variable Label</b></td><td>Cumulative amount borrowed for undergraduate education</td></tr>
<tr><td><b>Description</b></td><td>Includes all loans ever borrowed for undergraduate education in 2015–16 and prior years. Excludes Parent PLUS loans (PLUSCUM), which are only available to the parents of dependent undergraduates.</td></tr>
<tr><td><b>Assigned Values</b></td><td><table><tr><th>Value</th><th>Value label</th></tr><tr><td><b>0</b></td><td>{Zero}</td></tr><tr><td><b>c</b></td><td>{Continuous}</td></tr></table></td></tr>
<tr><td><b>Data Source(s)</b></td><td>NPSAS:16 Interview, NSLDS:16, NPSAS:16 Student Records</td></tr>
<tr><td><b>Component Derived Variables</b></td><td>FEDCUM1, NFEDCUM1</td></tr>
<tr><td><b>Applies to</b></td><td>All study members</td></tr>
<tr><td><b>Availability</b></td><td>NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16</td></tr>
<tr><td><b>Change History</b></td><td>Unchanged from NPSAS:96 through NPSAS:16. The NPSAS:16 interview asked study members to report the cumulative amount of private loans borrowed for their undergraduate education, which differs from prior NPSAS studies. To minimize double-counting and inconsistencies between different data sources, responses to non-federal loan items from the interview were compared to the cumulative federal amount borrowed for undergraduate education (FEDCUM1) derived from administrative NSLDS data as well as self-reported cumulative total amount of undergraduate loans borrowed (of any type) from the interview.</td></tr>
<tr><td><b>N:16 Usage Statistics</b></td><td>Number of uses: 4,313 (UG); 170 (GR)</td></tr>
<tr><td><b>Programming Narrative</b></td><td>1. Amount equals the sum of cumulative federal loans for undergraduate (FEDCUM1) and cumulative private/non-federal loans for undergraduate (NFEDCUM1).
2. For undergraduates (STYPELST = 1), check this amount against the total amount borrowed in the NPSAS year (TOTLOAN). Flag cases where BORAMT1 is greater than TOTLOAN for review. It is possible for study members who were graduates and undergraduates in the NPSAS year to have borrowed for both and have seemingly conflicting values.</td></tr>
</table>

SAS Code	* Source variable fedcum1="Cumulative federal loan amount for undergrad"; * Source variable nfedcum1="Cumulative non-federal loan amount for undergrad"; **; * Equal to the sum of cumulative federal amount borrowed for undergraduate * education (FEDCUM1) and cumulative non-federal (private) amount * borrowed for undergraduate education (NFEDCUM1); boramt1 = fedcum1 + nfedcum1;

<table>
<tr><td><b>Variable Name</b></td><td><b>BORAMT2</b></td></tr>
<tr><td><b>Variable Label</b></td><td>Cumulative amount borrowed for graduate education</td></tr>
<tr><td><b>Description</b></td><td>Includes all loans ever borrowed for graduate education for 2015–16 and prior years, including federal loans and loans from other sources such as banks, credit unions, states, and institutions. Includes Graduate PLUS loans (GPLUSCUM), but excludes Parent PLUS Loans (PLUSCUM), which are only available to the parents of dependent undergraduates.</td></tr>
<tr><td><b>Assigned Values</b></td><td><table>
<tr><th>Value</th><th>Value label</th></tr>
<tr><td><b>0</b></td><td>{Zero}</td></tr>
<tr><td><b>c</b></td><td>{Continuous}</td></tr>
</table></td></tr>
<tr><td><b>Data Source(s)</b></td><td>NPSAS:16 Interview, NSLDS:16; NPSAS:16 Student Records</td></tr>
<tr><td><b>Component Derived Variables</b></td><td>FEDCUM2, NFEDCUM2</td></tr>
<tr><td><b>Applies to</b></td><td>All graduate study members</td></tr>
<tr><td><b>Availability</b></td><td>NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16</td></tr>
<tr><td><b>Change History</b></td><td>Unchanged from NPSAS:96 through NPSAS:16. The NPSAS:16 interview asked study members to report the cumulative amount of private loans borrowed for their graduate education, which differs from prior NPSAS studies. To minimize double-counting and inconsistencies between different data sources, responses to non-federal loan items from the interview were compared to the cumulative federal amount borrowed for undergraduate education (FEDCUM2) derived from administrative NSLDS data as well as self-reported cumulative total amount of undergraduate loans borrowed (of any type) from the interview.</td></tr>
<tr><td><b>N:16 Usage Statistics</b></td><td>Number of uses: 0 (UG); 695 (GR)</td></tr>
<tr><td><b>Programming Narrative</b></td><td>1. Since this is a graduate level variable, set to missing for undergraduates.
2. Amount equals the sum of cumulative federal loans for graduate (FEDCUM2) and cumulative private/ non-federal loans for graduate (NFEDCUM2).
3. Check this amount against the total amount borrowed in the NPSAS year (TOTLOAN). Flag cases where BORAMT2 is greater than TOTLOAN for review. It is possible for study members who were graduates and undergraduates in the NPSAS year to have borrowed for both and have seemingly conflicting values.</td></tr>
<tr><td><b>SAS Code</b></td><td>* Source variable stypelst="Student type indicator";
* Source variable fedcum2="Cumulative federal loan amount for grad";
* Source variable nfedcum2="Cumulative non federal loan amount for grad";
    **;
* Applies to: All graduate study members;
    if stypelst = 1   then boramt2 = .y;
* Amount equals the sum of cumulative federal loans for graduate (FEDCUM2)
* and cumulative private/ non-federal loans for graduate;
    else if stypelst > 1 then boramt2 = fedcum2 + nfedcum2;</td></tr>
</table>

Variable Name	**BORAMT3**
Variable Label	Cumulative amount borrowed for undergraduate and graduate education
Description	Includes all loans ever borrowed for both graduate and undergraduate education for 2015–16 and prior years, including federal loans and loans from other sources such as banks, credit unions, states, and institutions. Includes Graduate PLUS loans (GPLUSCUM) but excludes Direct PLUS Loans (PLUSCUM), which are only available to the parents of dependent undergraduates.

Assigned Values	**Value** | **Value label**
	0 | {Zero}
	c | {Continuous}

Data Source(s)	NPSAS:16 Interview, NSLDS:16; NPSAS:16 Student Records
Component Derived Variables	BORAMT1, BORAMT2
Applies to	All graduate study members
Availability	NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16
Change History	Unchanged from NPSAS:96 through NPSAS:16. The NPSAS:16 interview asked study members to report the cumulative amount of private loans borrowed for their undergraduate and graduate education, which differs from prior NPSAS studies. To minimize double-counting and inconsistencies between different data sources, responses to non-federal loan items from the interview were compared to the cumulative federal amount borrowed for undergraduate education (FEDCUM3) derived from administrative NSLDS data as well as self-reported cumulative total amount of undergraduate loans borrowed (of any type) from the interview.
N:16 Usage Statistics	Number of uses: 0 (UG); 380 (GR)
Programming Narrative	1. Since this is a graduate level variable, set to missing for undergraduates. 2. Amount equals the sum of the cumulative amount borrowed for undergraduate education (BORAMT1) and the cumulative amount borrowed for graduate education (BORAMT2). Note: BORAMT1 and BORAMT2 are variables whose value is determined by choosing the maximum of three values. For more information, see their respective entries.

SAS Code	`* Source variable boramt1="Cumulative amount borrowed for undergrad";`
	`* Source variable boramt2="Cumulative amount borrowed for grad";`
	`******************************************************************;`
	`* Applies to: All graduate study members;`
	`if stypelst = 1      then boramt3 = .y;`
	`* Add together the cumulative amount borrowed for undergraduate education`
	`* (BORAMT1) and the cumulative amount borrowed for graduate education`
	`* (BORAMT2);`
	`else if stypelst > 1 then boramt3 = boramt1 + boramt2;`

Variable Name	**BUDGETAJ**
Variable Label	Student budget (attendance adjusted)
Description	Price of attendance or total student budget (attendance adjusted) at NPSAS institution during 2015–16 academic year.
Assigned Values	**Value** | **Value label** **-3** | {Skipped}

	c {Continuous}
Data Source(s)	NPSAS:16 Student Records, IPEDS:16, NSLDS:16
Component Derived Variables	STUDMULT, TUITION2, BUDNONAJ
Applies to	Study members who attended one institution (STUDMULT=1)
Availability	NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16
Change History	Unchanged from NPSAS:04 through NPSAS:16. Name change from BUDGETAJ to BUDGETA2 (NPSAS:2000) and back to BUDGETAJ (NPSAS:04).
N:16 Usage Statistics	Number of uses: 797 (UG); 70 (GR)
Programming Narrative	1. Set to skip if student attended multiple institutions in the NPSAS year (STUDMULT>1). 2. Derive tuition and fees paid (TUITION2) and non-tuition expenses (BUDNONAJ). 3. Add tuition and fees paid to non-tuition expenses to get the total budget (BUDGETAJ).

SAS Code

```
*Source variable budnonug="non-tuition expense budget for undergraduate (attendance
adjusted)";
*Source variable budnongr="non-tuition expense budget for graduate (attendance adjusted)";
 ****************************************************************;
*Applies to: all study members;
   if studmult > 1 then budnonaj = -3
   else if stypelst = 1 then budnonaj=budnonug;
   else if stypelst > 1 then budnonaj=budnongr;
     end;

*Applies to: undergraduates;
if STYPELST=1 and STUDMULT=1;
set pellcost;
 PELLCOA=ACTD_COST_OF_ATT;
if missing(PELLCOA)=0 and ENLEN>10 then
PELLCOA=PELLCOA+round((PELLCOA/9)*(ENLEN-10));
if PELLCOA>TUITION2_UG then PELLCOA2=PELLCOA; else PELLCOA2=.;
 if missing(PELLCOA2)=0 and ATTNSTAT=1 then PELLCOA2=PELLCOA-TUITION2_UG;
 else if (PELLCOA-TUITION2>0) and missing(PELLCOA2)=0 and TUITION2>0 and
ATTNSTAT ne 1 and
   TUITION2>=TUITION2_UG and INJURIS ne 2 then PELLCOA2=PELLCOA-TUITION2;
 else if missing(PELLCOA2)=0 and (PELLCOA-TUITION3>0) and ATTNSTAT ne 1 and
   TUITION3>=TUITION2_UG and INJURIS=2 then PELLCOA2=PELLCOA-TUITION3;
 else if missing(PELLCOA2)=0 and (PELLCOA-tuit_ftfy>0) and tuit_ftfy>=TUITION2_UG
      then PELLCOA2=PELLCOA-tuit_ftfy;
 else if missing(PELLCOA2)=0 then PELLCOA2=PELLCOA-TUITION2_UG;
if CNEBOOKS>=0 and CNESROOM>=0 and CNETRANS>=0 and CNESCOMP>=0 and
   CNEOTHER>=0 then do;
   if CNPERIOD=1 then SRCOA=sum(of CNEBOOKS CNESROOM CNETRANS
CNESCOMP CNEOTHER);
   if CNPERIOD=2 and CALSYS=1
      then SRCOA=(sum(of CNEBOOKS CNESROOM CNETRANS CNESCOMP
CNEOTHER))*2;
   if CNPERIOD=2 and CALSYS=2
      then SRCOA=(sum(of CNEBOOKS CNESROOM CNETRANS CNESCOMP
CNEOTHER))*3;
```

```
 end;
 if CNEBOOKS>=0 and CNESROOM>=0 and CNETRANS>=0 and CNESCOMP>=0 and
    CNEOTHER>=0 then do;
    if CNPERIOD in (3,5,7) and ATTNSTAT=4 then do;
     SRCOA_ADJ=sum(of CNEBOOKS CNESROOM CNETRANS CNESCOMP CNEOTHER);
    end;
    if CNPERIOD in (4,6,8) and ATTNSTAT=4 and CALSYS=1 then do;
     SRCOA_ADJ=(sum(of CNEBOOKS CNESROOM CNETRANS CNESCOMP
CNEOTHER))*2;
    end;
    if CNPERIOD in (4,6,8) and ATTNSTAT=4 and CALSYS=2 then do;
     SRCOA_ADJ=(sum(of CNEBOOKS CNESROOM CNETRANS CNESCOMP
CNEOTHER))*3;
    end;
              if CNPERIOD in (4,6,8) and ATTNSTAT=6 and ENLEN<6 then do;
               SRCOA_ADJ=sum(of CNEBOOKS CNESROOM CNETRANS CNESCOMP
CNEOTHER);
              end;
 end;
 if SRCOA<=1000 then SRCOA=.;
 if SRCOA_ADJ<=250 then SRCOA_ADJ=.;
 if PELLCOA2<=1000 then PELLCOA2=.;
 if missing(SRCOA_ADJ)=0 and missing(PELLCOA2)=0 then do;
  if SRCOA_ADJ>PELLCOA2 then PELLCOA2=.;
   else if CNPERIOD in (3,5,7) then PELLCOA2=.;
   else if CNPERIOD in (4,6,8) then SRCOA_ADJ=.;
 end;
 if PELLCOA2>=0 then BUDGET1=PELLCOA2;
  else if SRCOA>=0 then BUDGET1=SRCOA;
if BUDGET1=. and SRCOA_ADJ=. then do;
 if budimp1>0 then do;BUDGET2=budimp1; end;
 else if budimp2>0 then do;BUDGET2=budimp2; end;
 else if budimp3>0 then do;BUDGET2=budimp3; end;
 else if budimp4>0 then do;BUDGET2=budimp4; end;
 else if budimp5>0 then do;BUDGET2=budimp5; end;
 else if budimp6>0 then do;BUDGET2=budimp6; end;
 else if budimp7>0 then do;BUDGET2=budimp7; end;
 else if budimp8>0 then do;BUDGET2=budimp8; end;
end;
if BUDGET1>=0 then BUDGET3=round(BUDGET1);
 else BUDGET3=BUDGET2;
if ATTEND4=4 then BUDNONUG=BUDGET3;
 if BUDNONUG=. and BUDGET3>0 then do; BUDNONUG=round(BUDGET3*(ATTEND4/4));
   end;
 if BUDNONUG=. and SRCOA_ADJ>0 then do; BUDNONUG=SRCOA_ADJ; end;

*Applies to: graduate students;
if STYPELST>1 and STUDMULT=1;
 if CNEBOOKS>=0 and CNESROOM>=0 and CNETRANS>=0 and CNESCOMP>=0 and
    CNEOTHER>=0 then do;
    if CNPERIOD=1 then SRCOA=sum(of CNEBOOKS CNESROOM CNETRANS
CNESCOMP CNEOTHER);
```

```
    if CNPERIOD=2 and CALSYS=1
      then SRCOA=(sum(of CNEBOOKS CNESROOM CNETRANS CNESCOMP
CNEOTHER))*2;
    if CNPERIOD=2 and CALSYS=2
      then SRCOA=(sum(of CNEBOOKS CNESROOM CNETRANS CNESCOMP
CNEOTHER))*3;
 end;
 if CNEBOOKS>=0 and CNESROOM>=0 and CNETRANS>=0 and CNESCOMP>=0 and
    CNEOTHER>=0 then do;
    if CNPERIOD in (3,5,7) and ATTNSTAT=4 then do;
     SRCOA_ADJ=sum(of CNEBOOKS CNESROOM CNETRANS CNESCOMP CNEOTHER);
    end;
    if CNPERIOD in (4,6,8) and ATTNSTAT=4 and CALSYS=1 then do;
     SRCOA_ADJ=(sum(of CNEBOOKS CNESROOM CNETRANS CNESCOMP
CNEOTHER))*2;
    end;
    if CNPERIOD in (4,6,8) and ATTNSTAT=4 and CALSYS=2 then do;
     SRCOA_ADJ=(sum(of CNEBOOKS CNESROOM CNETRANS CNESCOMP
CNEOTHER))*3;
    end;
             if CNPERIOD in (4,6,8) and ATTNSTAT=6 and ENLEN<6 then do;
              SRCOA_ADJ=sum(of CNEBOOKS CNESROOM CNETRANS CNESCOMP
CNEOTHER);
             end;
 end;
 if SRCOA<=1000 then SRCOA=.;
 if SRCOA_ADJ<=250 then SRCOA_ADJ=.;
 if SRCOA>=0 then BUDGET1=SRCOA;
if BUDGET1=. and SRCOA_ADJ=. then do;
 if budimp1>0 then do;BUDGET2=budimp1; end;
 else if budimp2>0 then do;BUDGET2=budimp2; end;
 else if budimp3>0 then do;BUDGET2=budimp3; end;
 else if budimp4>0 then do;BUDGET2=budimp4; end;
 else if budimp5>0 then do;BUDGET2=budimp5; end;
 else if budimp6>0 then do;BUDGET2=budimp6; end;
 else if budimp7>0 then do;BUDGET2=budimp7; end;
 else if budimp8>0 then do;BUDGET2=budimp8; end;
 else if budimp9>0 then do;BUDGET2=budimp9; end;
end;
if BUDGET1>=0 then BUDGET3=round(BUDGET1);
 else BUDGET3=BUDGET2;
if ATTEND4=4 then BUDNONGR=BUDGET3;
 if BUDNONGR=. and BUDGET3>0 then do; BUDNONGR=round(BUDGET3*(ATTEND4/4));
    end;
if BUDNONGR=. and SRCOA_ADJ>0 then do; BUDNONGR=SRCOA_ADJ; end;
```

Variable Name	**CAMPAMT**
Variable Label	Federal campus-based aid (Perkins, SEOG, FWS)
Description	Total amount of federal aid received during the 2015–16 academic year which is allocated by the institution’s financial aid office (i.e., campus-based).

<table>
<tr><td><b>Assigned Values</b></td><td><table><tr><th>Value</th><th>Value label</th></tr><tr><td><b>-3</b></td><td>{Skipped}</td></tr><tr><td><b>0</b></td><td>{Zero}</td></tr><tr><td><b>c</b></td><td>{Continuous}</td></tr></table></td></tr>
<tr><td><b>Data Source(s)</b></td><td>NPSAS:16 Student Records, NSLDS:16, NPSAS:16 Interview</td></tr>
<tr><td><b>Component Derived Variables</b></td><td>SEOGAMT, PERKAMT, TFEDWRK</td></tr>
<tr><td><b>Applies to</b></td><td>All study members</td></tr>
<tr><td><b>Availability</b></td><td>NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16</td></tr>
<tr><td><b>Change History</b></td><td>Unchanged from NPSAS:96 through NPSAS:16.</td></tr>
<tr><td><b>N:16 Usage Statistics</b></td><td>Number of uses: 16 (UG); 0 (GR)</td></tr>
<tr><td><b>Programming Narrative</b></td><td>1. For undergraduate students, sum of the Supplemental Educational Opportunity (SEOG) grant (SEOGAMT), federal work-study (TFEDWRK), and Perkins loans (PERKAMT).
2. For graduate students, sum of federal work-study (TFEDWRK) and Perkins loans (PERKAMT).</td></tr>
</table>

SAS Code

```
*Source variable seogamt="Federal Supplemental Educational Opportunity Grant (SEOG)";
*Source variable perkamt="Perkins loan";
*Source variable tfedwrk="Federal work-study";
  ****************************************************************;
*Applies to: all study members;
  if stypelst = 1 then campamt = seogamt + perkamt + tfedwrk;
           else campamt = perkamt + tfedwrk;
  if missing(campamt) = 1 then
   do;
    campamt = .x;
   end;
```

<table>
<tr><td><b>Variable Name</b></td><td><b>CINCOME</b></td></tr>
<tr><td><b>Variable Label</b></td><td>Total income (continuous)</td></tr>
<tr><td><b>Description</b></td><td>Total income in 2014 for independent students (including spouse) or parents of dependent students.</td></tr>
<tr><td><b>Assigned Values</b></td><td><table><tr><th>Value</th><th>Value label</th></tr><tr><td><b>0</b></td><td>{Zero}</td></tr><tr><td><b>c</b></td><td>{Continuous}</td></tr></table></td></tr>
<tr><td><b>Data Source(s)</b></td><td>FAFSA:16, NSLDS:16 NPSAS:16 Interview</td></tr>
<tr><td><b>Component Derived Variables</b></td><td>DEPEND, DEPINC, INDEPINC</td></tr>
<tr><td><b>Applies to</b></td><td>All study members</td></tr>
<tr><td><b>Availability</b></td><td>NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16</td></tr>
<tr><td><b>Change History</b></td><td>Unchanged from NPSAS:96 through NPSAS:12; NPSAS:16 changed to allow values over $999,999, largely due to a change in the FAFSA.</td></tr>
<tr><td><b>N:16 Usage Statistics</b></td><td>Number of uses: 1,415 (UG); 143 (GR)</td></tr>
</table>

Programming Narrative	1. If dependent (DEPEND=1), then total income = dependent parents’ income (DEPINC). 2. If independent (DEPEND=2), then total income = independent student and spouse’s income (INDEPINC).

SAS Code	/*----- CINCOME: Total income-parents and independent (continuous) -----*/
	/* variable used: DEPINC INDEPINC */
	
	IF DEPINC>=0 THEN CINCOME=DEPINC;
	IF INDEPINC>=0 THEN CINCOME=INDEPINC;

Variable Name	**DEPANY**
Variable Label	Dependents: Has any dependents
Description	Indicates whether the student had any dependents during the 2015–16 academic year.
Assigned Values	**Value** **Value Label** **0** No dependents **1** Has dependents
Data Source(s)	FAFSA:16, NPSAS:16 Interview
Component Derived Variables	Not applicable
Applies to	All study members
Availability	NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16
Change History	Unchanged from NPSAS:04 through NPSAS:16. Named ANYDEP in NPSAS:2000.
N:16 Usage Statistics	Number of uses: 98 (UG); 19 (GR)
Programming Narrative	1. Use the dependent classification used to calculate expected family contribution (EFC) from the FAFSA first: a. If C16EFCPT in (1,2,4,5) then study member has no dependents (DEPANY = 0) b. Else if C16EFCPT in (3,6) then study member has dependents. 2. Else if that information is missing, use responses to the student interview and the FAFSA questions pertaining to dependents. a. If study member indicated having children or other dependents in the student interview (N16EDEPS=1 or N16EOTDEPS=1) or on the FAFSA (C16DEPS=1 or C16DEPO=1), then study member has dependents (DEPANY = 1). b. Else if study member indicated they had neither children nor other dependents on the FAFSA or student interview, then DEPANY = 0. 3. Consistency edit: If student is later classified as a dependent (DEPEND = 1), then ensure they also have no dependents (DEPANY = 0).
SAS Code	/*---------------- DEPANY: Dependents - has dependents ----------------*/ /* variable used: N16EDEPS N16EOTDEPS C16DEPS C16DEPO C16EFCPT */ /* CPS has precedence over dependency set to "0" if CPS EFC indicated student has no dependent (C16EFCPT=1,2,4,5) else set to "1" if CPS EFC indicated student has dependent (C16EFCPT=3,6) else if no EFC: -set to "1" if either student int or CPS indicates student has dependent - else set to "0" if either student interview or CPS indicates student has no dependent children AND other dependent */ IF C16EFCPT IN (1,2,4,5) THEN DO; DEPANY=0; DP=1; END; ELSE IF C16EFCPT IN (3,6) THEN DO; DEPANY=1; DP=1; END;

	ELSE IF (N16EDEPS=1 AND N16EDEP2 ne 0) OR (N16EOTDEPS=1 AND
	    N16EOTDEPS2 ne 0) OR C16DEPS=1 OR C16DEPO=1 THEN DO; DEPANY=1; END;
	ELSE IF (N16EDEPS=0 OR C16DEPS=2) AND (N16EOTDEPS=0 OR C16DEPO=2)
	    THEN DO; DEPANY=0; END;

Variable Name	**DEPEND**
Variable Label	Dependency status
Description	Student's dependency status during the 2015–16 academic year
Assigned Values	**Value** **Value label** **1** Dependent student **2** Independent student
Data Source(s)	FAFSA:16, NPSAS:16 Interview, NPSAS:16 Student Records, VBA:16
Component Derived Variables	AGEGROUP, MILTYPE2, STYPELST, SMARITAL, ORPHAN, DEPANY, HOMELESS Note: ORPHAN and HOMELESS are restricted-use file variables only. They are not described in this document.
Applies to	All undergraduate study members
Availability	NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16
Change History	Unchanged from NPSAS:96 through NPSAS:08; Homelessness was added as a criterion for determining independent status between NPSAS:08 and NPSAS:12. Added VBA as data source in NPSAS:16.
N:16 Usage Statistics	Number of uses: 3,218 (UG); 0 (GR)
Programming Narrative	1. If student meets any of the 8 criteria, set to INDEPENDENT a. Age 24 or older on December 31, 2015 (AGEGROUP>1) b. A veteran of the U.S. Armed Forces or on active duty (MILTYPE2 in [1, 3]) c. Enrolled in a graduate or professional program beyond a bachelor's degree (STYPELST>1) d. Married (SMARITAL>1) e. Orphan, ward of court, emancipated minor, or in legal (ORPHAN=1) f. Have legal dependents other than a spouse (DEPANY=1) g. Homeless or at risk of homelessness (HOMELESS=1) 2. Else use variable from CPS used to calculate EFC (C16EFCPT: Primary EFC Type) a. If C16EFCPT in (1, 4) then DEPENDENT b. If C16EFCPT in (2, 3, 5, 6) then INDEPENDENT 3. Else if study member is younger than 24 (AGEGROUP=1), not in the military (MILTYPE2 in [0,2,4]), single (SMARITAL=1), not an orphan (ORPHAN in [-3,0]), has no dependents (DEPANY=0), and is not homeless (HOMELESS in [-3,0]) then set to DEPENDENT 4. Consistency edit: If DEPENDENT, set study member to have no dependents of his/her own (DEPANY=0).
SAS Code	/*------------------------ DEPEND: Dependency status ----------------------*/ /* variable used: C16EFCPT AGEGROUP MILTYPE2 STYPELST SMARITAL ORPHAN DEPANY */ /* 1. set to "independent" if student met any of the 8 criterias: a. Age 24 or older on December 31, 2015 (AGE>23) b. Married (SMARITAL>1) c. Orphan, ward of court, emancipated minor, or in legal (ORPHAN=1) d. Have legal dependents other than a spouse (DEPANY=1) e. A veteran of the U.S. Armed Forces (MILTYPE2=3)

	f. U.S. Armed Forces active duty (MILTYPE2=1) g. Enrolled in a graduate or professional program beyond a bachelor's degree (STYPELST>1) h. Homeless or at risk of homelessness (HOMELESS=1) 2. else use CPS EFC (C16EFCPT) 3. else set to "dependent" if student didn't meet all (available) criterias */ IF AGEGROUP>1 OR MILTYPE2 IN (1,3) OR STYPELST>1 OR SMARITAL>1 OR ORPHAN=1 OR DEPANY=1 OR HOMELESS=1 THEN DO; DEPEND=2; DS=1; END; /* CPS */ ELSE IF C16EFCPT IN (1,4) THEN DO; DEPEND=1; DS=2; END; /* dependent */ ELSE IF C16EFCPT IN (2,3,5,6) THEN DO; DEPEND=2; DS=2; END; /* independent */ ELSE IF AGEGROUP=1 AND MILTYPE2 IN (0,2,4) AND STYPELST=1 AND SMARITAL=1 AND ORPHAN=0 AND DEPANY=0 AND HOMELESS=0 THEN DO; DEPEND=1; DS=3; END;

<table>
<tr><td><b>Variable Name</b></td><td><b>DEPEND2</b></td></tr>
<tr><td><b>Variable Label</b></td><td>Dependency status (3 categories)</td></tr>
<tr><td><b>Description</b></td><td>Student’s dependency status (3 categories) for federal financial aid need analysis purposes during the 2015–16 academic year. This variable divides independent students into two separate categories based on whether the students have dependents.</td></tr>
<tr><td><b>Assigned Values</b></td><td>
<table>
<tr><th>Value</th><th>Value label</th></tr>
<tr><td><b>1</b></td><td>Dependent</td></tr>
<tr><td><b>2</b></td><td>Independent without dependents</td></tr>
<tr><td><b>3</b></td><td>Independent with dependents</td></tr>
</table>
</td></tr>
<tr><td><b>Data Source(s)</b></td><td>FAFSA:16, NPSAS:16 Interview, NPSAS:16 Student Records, VBA:16</td></tr>
<tr><td><b>Component Derived Variables</b></td><td>DEPEND, DEPANY</td></tr>
<tr><td><b>Applies to</b></td><td>All undergraduate study members</td></tr>
<tr><td><b>Availability</b></td><td>NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16</td></tr>
<tr><td><b>Change History</b></td><td>Unchanged from NPSAS:96 through NPSAS:08; Homelessness was added as a criterion for determining independent status between NPSAS:08 and NPSAS:12. Added VBA as data source in NPSAS:16.</td></tr>
<tr><td><b>N:16 Usage Statistics</b></td><td>Number of uses: 419 (UG); 0 (GR)</td></tr>
<tr><td><b>Programming Narrative</b></td><td>Programming occurs after imputation of component variables
1. If respondent is a dependent (DEPEND = 1), then set variable to one (DEPEND2 = 1)
2. If respondent is an independent (DEPEND = 2) and does not have any dependents (DEPANY = 0), then set to two (DEPEND2 = 2).
3. If respondent is an independent (DEPEND = 2) and has dependents (DEPANY = 1), then set to three (DEPEND2 = 3)</td></tr>
<tr><td><b>SAS Code</b></td><td>/*----------------- DEPEND2: Dependency status (3 categories) --------------*/
/* variable used: DEPEND DEPANY */

IF DEPEND=1 THEN DEPEND2=1; /* dependent */</td></tr>
</table>

```
IF DEPEND=2 AND DEPANY=0 THEN DEPEND2=2;    /* indep no dependent */
IF DEPEND=2 AND DEPANY=1 THEN DEPEND2=3;    /* indep with dependent */
```

Variable Name	**DEPINC**
Variable Label	Dependent students: Parents' income
Description	Total income in 2014 for parents of dependent student.

Assigned Values	**Value** **Value label**
	-3 {Skipped}
	0 {Zero}
	c {Continuous}

Data Source(s)	FAFSA:16, NPSAS:16 Interview
Component Derived Variables	DEPEND, DEPEND2
Applies to	Undergraduate dependent study members (DEPEND=1)
Availability	NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16
Change History	Unchanged from NPSAS:96 through NPSAS:16.
N:16 Usage Statistics	Number of uses: 2,499 (UG); 0 (GR)
Programming Narrative	1. If dependent student (DEPEND=1): a. Use "total income" (C16TI) from CPS if it is greater than or equal to $0. b. Else set DEPINC to zero if c16ti<0 and c16ti != -9 (indicates a negative total income). c. Else use the parents' adjusted gross income (C16PTXAGI) from CPS. d. Else use the sum of both parents' earnings from work (C16PAR1INC & C16PAR2INC), as long as both are non-missing. e. Else use each of the parents' income separately (C16PAR1INC first) as long as one is >=0. 2. Use historical FAFSA data a. See the instructions on parent adjusted gross income for data management. b. Replace DEPINC = nfagi14_2015 if DEPINC is missing, DEPEND2 = 1, and nfefcpt2_2015 = 1, and the agi14 from 2015 is greater than or equal to zero. c. Replace DEPINC = nfagi14_2014 if DEPINC is missing, DEPEND2 = 1, and nfefcpt2_2014 = 1, and the agi14 from 2014 is greater than or equal to zero. d. Replace DEPINC = nfagi14_2013 if DEPINC is missing, DEPEND2 = 1, and nfefcpt2_2013 = 1, and the agi14 from 2013 is greater than or equal to zero. 3. If independent student (DEPEND=2), set DEPINC to legitimate skip (-3). 4. If still missing, use the student interview categorical responses from parents' income (N16EPARNC) to help with stochastic imputation.

SAS Code

```
/*---------------------------- INCOME --------------------------------*/
/*------------- DEPINC: Dependent parent income (cont) ---------------------*/
/* variable used: DEPEND C16TI C16PTXAGI C16FINC C16MINC N16EPARNC

/*------------------ from CPS (continuous) ------------------*/

/* dependent student */
IF DEPEND=1 THEN DO;
   /* parents income */
   IF C16TI>=0 THEN DEPINC=C16TI;
```

```
  ELSE IF C16PTXAGI>=0 THEN DEPINC=C16PTXAGI;
  ELSE IF C16PAR1INC>=0 AND C16PAR2INC>=0 THEN DEPINC=SUM(OF C16PAR1INC C16PAR2INC);
  ELSE IF C16PAR1INC>=0 THEN DEPINC=C16PAR1INC;  /* prevent neg values */
  ELSE IF C16PAR2INC>=0 THEN DEPINC=C16PAR2INC;

  /* Fill in missing using the historical FAFSA data */
  IF DEPINC=. AND DEPEND2=1 THEN DO;
  IF nfefcpt2_2015=1 AND nfagi14_2015>=0 THEN DO; DEPINC=nfagi14_2015; END;
   ELSE IF nfefcpt2_2014=1 AND nfagi14_2014>=0 THEN DO; DEPINC=nfagi14_2014; END;
   ELSE IF nfefcpt2_2013=1 AND nfagi14_2013>=0 THEN DO; DEPINC=nfagi14_2013; END;
  END;

  /* student income */
  IF C16STI>=0 THEN DSTUINC=C16STI ;
  ELSE IF C16TXAGI>=0 THEN DSTUINC=C16TXAGI;
  ELSE IF C16INC>=0 THEN DSTUINC=C16INC;
  ELSE IF N16EINCOM=1 THEN DO; DSTUINC=0; ZDSTUINC=2; END;

  INDEPINC=-3; ISTUINC=-3; /* set skips */
END;
```

Variable Name	**DODAMT**
Variable Label	Department of Defense (military) aid
Description	Total amount of federal military tuition grants and fee waivers (including ROTC) received during the 2015–16 academic year.

Assigned Values	**Value** **Value label**
	0 {Zero}
	c {Continuous}

Data Source(s)	NPSAS:16 Student Records
Component Derived Variables	Not applicable
Applies to	All study members
Availability	NPSAS:16
Change History	New variable in NPSAS:16
N:16 Usage Statistics	Number of uses: 10 (UG); 3 (GR)

Programming Narrative

Sum federal military and ROTC benefits from student records (type=14).

Other aid section:

If CFAOTHSRC = 3 and CFAOTHTYP = 14 and CFAOTHAMT>0

Else if CFAOTHSRC = 3 and CFAOTHTYP = 14 and CFAOTHAMT = missing, then set minimum imputation flag to 1, to impute positive value.

Private/other government aid section (type=4):

If CFAGOVTYP = 4 and CFAGOVAMT>0

Else if CFAGOVTYP = 4 and CFAGOVAMT = missing, then set minimum imputation flag to 1, to impute a positive value.

SAS Code

```
* Source variables cfa1othamt-cfa3othamt="Other aid amount";
* Source variables cfa1othsrc-cfa3othsrc="Other aid source";
* Source variables cfa1othtyp-cfa3othtyp="Other aid type";
* Source variables cfa1govamt–cfa3govamt="Private aid program amount";
* Source variables cfa1govtyp-cfa3govtyp="Private aid program type";
  **************************************************************;
  array othamtd(3)  othamtd1-othamtd3;
  array srothamt(3) cfa1othamt cfa2othamt cfa3othamt;
  array srothsrc(3) cfa1othsrc cfa2othsrc cfa3othsrc;
  array srothtyp(3) cfa1othtyp cfa2othtyp cfa3othtyp;
  array govamtd(3) govamtd1-govamtd3;
  array srgovamt(3) cfa1govamt cfa2govamt cfa3govamt;
  array srgovtyp(3) cfa1govtyp cfa2govtyp cfa3govtyp;
  do i = 1 to 3;
    othamtd(i) = 0;
    if cfaothaid ~= 0 then
     do;
      if srothsrc(i) = 3 & srothtyp(i) = 14 then
       do;
        othamtd(i) = srothamt(i);
        if missing(othamtd(i)) = 1 then othamtd(i) = .m;
       end;
      else if cfaothaid = .x then othamtd(i) = .x;
     end;

    govamtd(i) = 0;
    if cfaothgov ~= 0 then
     do;
      if srgovtyp(i) = 4 then
       do;
        govamtd(i) = srgovamt(i);
        if missing(govamtd(i)) = 1 then govamtd(i) = .m;
       end;
      else if cfaothgov = .x then govamtd(i) = .x;
     end;
  end;
  array forDodamt(6) othamtd1 othamtd2 othamtd3 govamtd1 govamtd2 govamtd3;
  do i = 1 to 6;
    if forDodamt(i) > 0 then
     do j = 1 to 6;
      if forDodamt(j) = .x then forDodamt(j) = 0;
     end;
  end;
  dodamt = othamtd1 + othamtd2 + othamtd3 + govamtd1 + govamtd2 + govamtd3;
  dodamt_minimp = sum(of othamtd1 othamtd2 othamtd3 govamtd1 govamtd2 govamtd3);
  if dodamt_minimp = . then dodamt_minimp = 0;
  do i = 1 to 6;
    if forDodamt(i) = .m then Dodamt_minimp = Dodamt_minimp + 1;
  end;
  if missing(dodamt) = 1 then
   do;
    dodamt = .x;
   end;
```

<table>
<tr><td><b>Variable Name</b></td><td><b>EFC</b></td></tr>
<tr><td><b>Variable Label</b></td><td>Expected Family Contribution</td></tr>
<tr><td><b>Description</b></td><td>Federal Expected Family Contribution (EFC) used in need analysis.</td></tr>
<tr><td><b>Assigned Values</b></td><td><table><tr><th>Value</th><th>Value label</th></tr><tr><td>0</td><td>{Zero}</td></tr><tr><td>c</td><td>{Continuous}</td></tr></table></td></tr>
<tr><td><b>Data Source(s)</b></td><td>NSLDS:16, FAFSA:16, NPSAS:16 Interview, NPSAS:16 Student Records</td></tr>
<tr><td><b>Component Derived Variables</b></td><td>DEPEND, DEPEND2, DEPEND5B, PTAXFILE, PFEDTAX, PMARITAL, PINCOL, PFAMNUM, CINCOME, STUSTATE, DSTUINC, STAXFILE, SFEDTAX, SMARITAL, ISTUINC, SPSINC, SINCOL, SPINCOL, SFAMNUM
Note: DEPEND5B, PTAXFILE, PFEDTAX, PMARITAL, PINCOL, PFAMNUM, STUSTATE, DSTUINC, STAXFILE, SFEDTAX, ISTUINC, SPSINC, SINCOL, SPINCOL, and SFAMNUM are not included in this report. Information on these variables may be found in the NPSAS:16 undergraduate codebook, https://nces.ed.gov/datalab/powerstats/codebook.aspx?dataset=121&type=subject and (except for PTAXFILE, PFEDTAX, PMARITAL, PINCOL, and PFAMNUM) the NPSAS:16 graduate student codebook, https://nces.ed.gov/datalab/powerstats/codebook.aspx?dataset=122&type=subject.</td></tr>
<tr><td><b>Applies to</b></td><td>All study members</td></tr>
<tr><td><b>Availability</b></td><td>NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16</td></tr>
<tr><td><b>Change History</b></td><td>Name change from EFC4 to EFC in NPSAS:2000. Variable name unchanged from NPSAS:04 through NPSAS:16.
NPSAS:2000 used logistic regression to predict if EFC was equal to 0 and used Ordinarly Least Squares (OLS) regression to impute non-zero EFC. NPSAS:04, NPSAS:08, and NPSAS:12 used regression to impute missing EFC using dependency, family size, income, and number of family members in college as predictors. NPSAS:16 used a simplified EFC formula based on dependency and number of dependents to calculate reasonable estimates of missing EFC.
Maximum values of EFC appear larger in NPSAS:16 due to changes in the 2014-15 FAFSA EFC field from five digits to six.</td></tr>
<tr><td><b>N:16 Usage Statistics</b></td><td>Number of uses: 1,499 (UG); 37 (GR)</td></tr>
<tr><td><b>Programming Narrative</b></td><td>1. Use EFC from the 2015–16 CPS for those who filed a FAFSA
2. For Pell grant recipients, use the EFC reported with the grant in NSLDS if the EFC differs from that in CPS. If there are 2 Pell grant EFCs (e.g., two different institutions, or a re-calculation), use the EFC from the NPSAS institution, or the most recent (based on enrl_dt) if there are multiple.</td></tr>
</table>

3. If still missing, use EFC from the 2016-17 CPS, as long as the dependency status used to calculate it is the same as the derived variable for NPSAS (DEPEND2 and c17efcpt).
4. Logically impute all remaining cases using the simplified EFC formula. This requires dependency, family, income, and residency variables be imputed PRIOR to the EFC derivation
   a. GROUP 1: Dependent students (DEPEND2=1)
      i. If Parents' income is $24,000 or less (CINCOME<=$24,000) and they received federal benefits of any type (FEDBEN = 1), automatically set EFC to zero and skip the rest of the steps in this portion.
      ii. PARENT PORTION:
         1. Estimate allowances against parents' income
            a. Federal income tax = PFEDTAX if parents filed taxes (PTAXFILE = 1), else = 0.
            b. State tax = use state of residence (STUSTATE) and table A1 in the EFC worksheet to calculate an amount
            c. Parents' social security tax = use parents' income (CINCOME) and table A2 in the EFC worksheet
            d. Income protection allowance = see table A3 in the EFC worksheet
            e. Employment expense allowance: If single parent family (PMARITAL>1) minimum of ($4000, 35% parent's income); else = 0.
            f. Total parent allowances = sum of (a – e)
         2. Estimate of adjusted available parents' income
            a. Total income (CINCOME) – total parent allowances. This interim variable CAN be a negative number and will be used again in the student portion.
            b. Set to zero here if negative, then adjust the value using table A6 in the EFC worksheet. This takes into account number of family members (PFAMNUM) and number in college (PINCOL) to determine an amount
         3. Estimate of parents' contribution to EFC
            a. Adjusted available income (from above) divided by number in college in 2015–16 (PINCOL).
      iii. STUDENT PORTION
         1. Estimate allowances against student income
            a. State tax = use state of residence (STUSTATE) and percentage from table A7 in the EFC worksheet
            b. Social security tax = use student income (DSTUINC) and table A2 in the EFC worksheet
            c. Income protection allowance = $6,310
            d. Parents' negative adjusted available income: If value from 2a in Parents' portion is negative, add it as a positive number here; else = 0.
            e. Total student allowances = Sum of (a – d)
         2. Estimate student's contribution from income
            a. Total income (DSTUINC) – total allowances = available income
            b. Take half of available income from 2a. (Student contrib = Available inc*0.5)

      iv. CALCULATE EXPECTED FAMILY CONTRIBUTION
         1. EFC = Parent contribution + Student contribution
         2. If negative, set to zero

   b. GROUP 2: Independent students with no dependents (besides spouse) (DEPEND2 = 2)

i. There are no automatic zero EFC conditions for this group.

ii. Estimate allowances

1. Federal income tax = SFEDTAX if student filed taxes (STAXFILE = 1), else = 0.

2. State tax = use state of residence and percentage from table B1 in the EFC worksheet

3. Social security tax = find student and spouse's separately (ISTUINC & SPSINC) if applicable, and use table B2 in the EFC worksheet

4. Income protection allowance:

a. If single, separated, or divorced (SMARITAL in [1,3]), then = $9,810

b. Else if married and spouse is enrolled ½ time (SPINCOL = 1), then = $9,810

c. Else if married and just the student is enrolled ½ time (SPINCOL = 0), then = $15,720

5. Employment allowance:

a. If student is not married, separated, or divorced, then = 0

b. Else if student is married but only one person is working (ISTUINC = 0 or SPSINC = 0), then = 0

c. Else if student is married and both are working, then take minimum of (35% of the lesser of the 2 incomes or $4,000)

6. Total allowances = Sum of (1 – 5)

iii. Estimated contribution from available income

1. Total Income (CINCOME) – allowances = Available income

2. Take half of available income from 1 (Available inc * 0.5)

iv. CALCULATE EXPECTED FAMILY CONTRIBUTION

1. EFC = contribution from available income divided by the number in college in 2015–16 (SINCOL)

2. If negative, set to zero

c. GROUP 3: Independent students with other dependents (besides spouse) (DEPEND2 = 3)

i. If student (and the student's spouse if any) income is $24,000 or less (ISTUINC + SPSINC <= 24,000) and they received any type of federal benefits (FEDBEN = 1), automatically set EFC to zero and skip the rest of the steps in this portion.

ii. Estimate allowances against income

1. Federal income tax = SFEDTAX if student filed taxes (STAXFILE = 1), else = 0.

2. State tax = use state of residence (STUSTATE) and table C1 in the EFC worksheet to calculate an amount

3. Social security tax = find student and spouse's separately (ISTUINC & SPSINC) if applicable, and use table C2 in the EFC worksheet

4. Income protection allowance = use number of people in student's family (SFAMNUM), the number of them in college (SINCOL), and table C3 in the EFC worksheet

5. Employment expense allowance:

a. If both student and spouse are working (ISTUINC > 0 & SPSINC > 0) then take minimum of (35% of the lesser the 2 incomes or $4,000).

b. Else if one parent family (SMARITAL in [1,3]) then take minimum of (35% income and $4,000)

c. Else if either student or spouse is working (not both), set to zero.

d. Total allowances = sum of (a – c)

iii. Estimate of available income

1. Total income (CINCOME) – total allowances. This interim variable CAN be a negative number

iv. Estimate Total contribution from available income using the number from 2a., and table C6 from the EFC worksheet.

v. CALCULATE EXPECTED FAMILY CONTRIBUTION

1. Take the number from 3, and divide it by the number in college in 2015–16 (SINCOL)
2. If negative, set to zero

SAS Code

```
***************************************************************;
* Source variable pellefc="EFC reported with the grant in NSLDS";
* Source variable efccps="EFC from CPS";
***************************************************************;
* If PELLAMT>0, take PELLEFC;
if pellamt > 0 then do; efc = pellefc; end;
* Else take EFC from CPS if it’s >=0 (C16EFCP);
else if efccps >= 0 then do; efc = efccps; end;

**********************************************************************;
    * Parent portion
    **********************************************************************;
    * 1. Estimate allowances against parents’ income;
    *   a. Federal income tax = PFEDTAX if parents filed taxes (PTAXFILE = 1), else = 0;
    if ptaxfile = 1 then allowances_para = pfedtax;
                else allowances_para = 0;
    *   b. State tax = use state of residence and table A1 in the EFC worksheet to calculate an 
amount;
    if 0 <= cincome < 15000  then allowances_parb = cincome * tabA1_pct1;
    else if cincome >= 15000 then allowances_parb = cincome * tabA1_pct2;
    *   c. Parents’ social security tax = use parents’ income (CINCOME) and table A2 in the 
EFC worksheet;
    if 0 <= cincome <= 117000 then allowances_parc = 0.0765 * cincome;
    else if cincome > 117000  then allowances_parc = 8950.5 + (0.0145 * (cincome - 117000));
    *   d. Income protection allowance = see table A3 in the EFC worksheet;
    if pfamnum = 2 then
      do;
        if pincol = 1      then allowances_pard = 17580;
        else if pincol = 2 then allowances_pard = 14570;
      end;
    else if pfamnum = 3 then
      do;
        if pincol = 1      then allowances_pard = 21890;
        else if pincol = 2 then allowances_pard = 18900;
        else if pincol = 3 then allowances_pard = 15890;
      end;
    else if pfamnum = 4 then
      do;
        if pincol = 1      then allowances_pard = 27040;
        else if pincol = 2 then allowances_pard = 24030;
        else if pincol = 3 then allowances_pard = 21040;
        else if pincol = 4 then allowances_pard = 18030;
      end;
    else if pfamnum = 5 then
```

```
     do;
      if pincol = 1      then allowances_pard = 31900;
      else if pincol = 2 then allowances_pard = 28890;
      else if pincol = 3 then allowances_pard = 25900;
      else if pincol = 4 then allowances_pard = 22890;
      else if pincol = 5 then allowances_pard = 19900;
     end;
    else if pfamnum = 6 then
     do;
      if pincol = 1      then allowances_pard = 37310;
      else if pincol = 2 then allowances_pard = 34310;
      else if pincol = 3 then allowances_pard = 31310;
      else if pincol = 4 then allowances_pard = 28310;
      else if pincol = 5 then allowances_pard = 25320;
      else if pincol = 6 then allowances_pard = 22330;
     end;
    else if pfamnum > 6 then
     do;
      if pincol = 1      then allowances_pard = 37310 + (4210 * (pfamnum - 6));
      else if pincol = 2 then allowances_pard = 34310 + (4210 * (pfamnum - 6));
      else if pincol = 3 then allowances_pard = 31310 + (4210 * (pfamnum - 6));
      else if pincol = 4 then allowances_pard = 28310 + (4210 * (pfamnum - 6));
      else if pincol = 5 then allowances_pard = 25320 + (4210 * (pfamnum - 6));
      else if pincol = 6 then allowances_pard = 22330 + (4210 * (pfamnum - 6));
      else if pincol > 6 then allowances_pard = 22330 + (4210 * (pfamnum - 6)) - (2990 * (pincol - 6));
     end;
    *    e. Employment expense allowance: If single parent family (PMARITAL>1) minimum of ($4000, 35% parent’s income); else = 0;
    if pmarital in (2,3,4) then allowances_pare = min(4000,0.35 * cincome);
                       else allowances_pare = 0;
    *    f.  Total parent allowances = sum of (a – e);
    allowances_par = allowances_para + allowances_parb + allowances_parc + allowances_pard + allowances_pare;
    * 2. Estimate of adjusted available parents’ income;
    * Total income (CINCOME) – total parent allowances;
    ai_par = round(cincome - allowances_par);
    * Adjust the value using table A6 in the EFC worksheet;
    if ai_par < -3409 & missing(ai_par) = 0 then contrib_paraai = -750;
    else if -3409 <= ai_par <= 15700       then contrib_paraai = 0.22 * ai_par;
    else if 15701 <= ai_par <= 19700       then contrib_paraai = 3454 + (0.25 * (ai_par - 15700));
    else if 19701 <= ai_par <= 23700       then contrib_paraai = 4454 + (0.29 * (ai_par - 19700));
    else if 23701 <= ai_par <= 27700       then contrib_paraai = 5614 + (0.34 * (ai_par - 23700));
    else if 27701 <= ai_par <= 31700       then contrib_paraai = 6974 + (0.4 *  (ai_par - 27700));
    else if ai_par >= 31701                then contrib_paraai = 8574 + (0.47 * (ai_par - 31700));
    if contrib_paraai < 0 & missing(contrib_paraai) = 0 then contrib_paraai = 0;
    * 3. Estimate of parents’ contribution to EFC. Adjusted available
    * income (from above) divided by number in college in 2015–16 (PINCOL);
    if pincol > 0 then contrib_par = contrib_paraai / pincol;

    ************************************************************************;
    * Student portion
    ************************************************************************;
    * 1. Estimate allowances against student income;
    *    a. State tax = use state of residence and percentage from table A7 in the EFC
worksheet;
    if missing(dstuinc) = 0 then allowances_stua = dstuinc * tabA7_pct;
    *    b. Social security tax = use student income (DSTUINC) and table A2 in the EFC
worksheet;
    if 0 <= dstuinc <= 117000 then allowances_stub = 0.0765 * dstuinc;
    else if dstuinc > 117000  then allowances_stub = 8950.5 + (0.0145 * (dstuinc - 117000));
    *    c. Income protection allowance;
    allowances_stuc = 6310;
    *    d. Parents' negative adjusted available income: If value from 2a in Parents' portion is
negative, add it as a positive number here. else = 0.;
    if missing(ai_par) = 0 & ai_par < 0 then allowances_stud = abs(ai_par);
                                   else allowances_stud = 0;
    *    e. Add SFEDTAXD (federal taxes paid for dependent students) in the student
allowances portion for dependent students;
    if staxfiled = 1 then allowances_stue = sfedtaxd;
             else allowances_stue = 0;
    *    Total student allowances;
    allowances_stu = allowances_stua + allowances_stub + allowances_stuc +
allowances_stud + allowances_stue;
    * 2. Estimate student's contribution from income;
    * Total income (DSTUINC) – total allowances = available income;
    ai_stu = dstuinc - allowances_stu;
    * Take half of available income from 2a. (Student contrib = Available inc*0.5);
    contrib_stu = .5 * ai_stu;
    if contrib_stu < 0 & missing(contrib_stu) = 0 then contrib_stu = 0;

    ************************************************************************;
    * Calculate expected family contribution
    ************************************************************************;
    if missing(efc) = 1 then
     do;
      * EFC = Parent contribution + Student contribution;
      efc = contrib_par + contrib_stu;
      * If negative, set to zero;
      if missing(efc) = 0 & efc < 0 then efc = 0;
      * Cap: If any value for EFC is 999,999, then set it to 1,000,000;
      if efc >= 999999 then efc = 1000000;
   end;

************************************************************************;
  * Independent students/no dependents (besides spouse);
  ************************************************************************;
  else if depend5b in (2,3) then
   do;
```

```
    **********************************************************************;
    * Estimate allowances
    **********************************************************************;
    * 1. Federal income tax = SFEDTAX if student filed taxes (STAXFILE = 1), else = 0;
    if staxfile = 1 then allowances_inodep1 = sfedtax;
            else allowances_inodep1 = 0;
    * 2. State tax = use state of residence and percentage from table B1 in the EFC worksheet;
    if missing(cincome) = 0 then allowances_inodep2 = cincome * tabB1_pct;
    * 3. Social security tax = find student and spouse's separately (ISTUINC & SPSINC) if applicable, and use table B2 in the EFC worksheet;
    if 0 <= istuinc <= 117000 then allowances_inodep3a = 0.0765 * istuinc;
    else if istuinc > 117000  then allowances_inodep3a = 8950.5 + (0.0145 * (istuinc - 117000));
    if 0 <= spsinc <= 117000  then allowances_inodep3b = 0.0765 * spsinc;
    else if spsinc > 117000   then allowances_inodep3b = 8950.5 + (0.0145 * (spsinc - 117000));
    allowances_inodep3 = sum(of allowances_inodep3a allowances_inodep3b);
    * 4. Income protection allowance;
    if smarital in (1,3) then allowances_inodep4 = 9810;
    else if spincol = 1  then allowances_inodep4 = 9810;
    else if spincol = 0  then allowances_inodep4 = 15720;
    * 5. Employment allowance;
    if smarital in (1,3)          then allowances_inodep5 = 0;
    else if istuinc = 0 | spsinc = 0 then allowances_inodep5 = 0;
                    else allowances_inodep5 = min(0.35 * min(istuinc,spsinc),4000);
    * 6. Total allowances = Sum of (1 – 5);
    allowances_inodep = allowances_inodep1 + allowances_inodep2 + allowances_inodep3 + allowances_inodep4 + allowances_inodep5;

    **********************************************************************;
    * Estimated contribution from available income;
    **********************************************************************;
    * Total Income (CINCOME) – allowances = Available income;
    ai_inodep = cincome - allowances_inodep;
    * Take half of available income from 1 (Available inc * 0.5);
    contrib_inodep = .5 * ai_inodep;

    **********************************************************************;
    * Calculate expected family contribution;
    **********************************************************************;
    if missing(efc) = 1 then
     do;
      * EFC = contribution from available income divided by the number in college in 2015–16 (SINCOL);
      if sincol > 0 then efc = contrib_inodep / sincol;
      * If negative, set to zero;
      if missing(efc) = 0 & efc < 0 then efc = 0;
      * Cap: If any value for EFC is 999,999, then set it to 1,000,000;
      if efc >= 999999 then efc = 1000000;
   end;
 **********************************************************************;
  * Independent students with dependents (besides spouse);
  **************************************************************************;
  else if depend5b in (4,5) then
   do;

    **********************************************************************;
    * Estimate allowances
    **********************************************************************;
    * 1. Federal income tax = SFEDTAX if student filed taxes (STAXFILE = 1), else = 0;
    if staxfile = 1 then allowances_idep1 = sfedtax;
            else allowances_idep1 = 0;
    * 2. State tax = use state of residence and table C1 in the EFC worksheet to calculate an
amount;
    if 0 <= cincome < 15000  then allowances_idep2 = cincome * tabC1_pct1;
    else if cincome >= 15000 then allowances_idep2 = cincome * tabC1_pct2;
    * 3. Social security tax = find student and spouse’s separately (ISTUINC & SPSINC) if
applicable, and use table C2 in the EFC worksheet;
    if 0 <= istuinc <= 117000 then allowances_idep3a = 0.0765 * istuinc;
    else if istuinc > 117000  then allowances_idep3a = 8950.5 + (0.0145 * (istuinc - 117000));
    if 0 <= spsinc <= 117000  then allowances_idep3b = 0.0765 * spsinc;
    else if spsinc > 117000   then allowances_idep3b = 8950.5 + (0.0145 * (spsinc - 117000));
    allowances_idep3 = sum(of allowances_idep3a allowances_idep3b);
    * 4. Income protection allowance = use number of people in student’s family (SFAMNUM),
the number of them in college (SINCOL), and table C3 in the EFC worksheet;
    if sfamnum = 2 then
     do;
      if sincol = 1      then allowances_idep4 = 24840;
      else if sincol = 2 then allowances_idep4 = 20590;
     end;
    else if sfamnum = 3 then
     do;
      if sincol = 1      then allowances_idep4 = 30920;
      else if sincol = 2 then allowances_idep4 = 26700;
      else if sincol = 3 then allowances_idep4 = 22450;
     end;
    else if sfamnum = 4 then
     do;
      if sincol = 1      then allowances_idep4 = 38180;
      else if sincol = 2 then allowances_idep4 = 33950;
      else if sincol = 3 then allowances_idep4 = 29720;
      else if sincol = 4 then allowances_idep4 = 25470;
     end;
    else if sfamnum = 5 then
     do;
      if sincol = 1      then allowances_idep4 = 45060;
      else if sincol = 2 then allowances_idep4 = 40800;
      else if sincol = 3 then allowances_idep4 = 36570;
      else if sincol = 4 then allowances_idep4 = 32340;
      else if sincol = 5 then allowances_idep4 = 28110;
     end;
    else if sfamnum = 6 then
     do;
```

```
      if sincol = 1      then allowances_idep4 = 52690;
      else if sincol = 2 then allowances_idep4 = 48450;
      else if sincol = 3 then allowances_idep4 = 44240;
      else if sincol = 4 then allowances_idep4 = 39970;
      else if sincol = 5 then allowances_idep4 = 35760;
      else if sincol = 6 then allowances_idep4 = 31530;
     end;
    else if sfamnum > 6 then
     do;
      if sincol = 1      then allowances_idep4 = 52690 + (5950 * (sfamnum - 6));
      else if sincol = 2 then allowances_idep4 = 48450 + (5950 * (sfamnum - 6));
      else if sincol = 3 then allowances_idep4 = 44240 + (5950 * (sfamnum - 6));
      else if sincol = 4 then allowances_idep4 = 39970 + (5950 * (sfamnum - 6));
      else if sincol = 5 then allowances_idep4 = 35760 + (5950 * (sfamnum - 6));
      else if sincol = 6 then allowances_idep4 = 31530 + (5950 * (sfamnum - 6));
      else if sincol > 6 then allowances_idep4 = 31530 + (5950 * (sfamnum - 6)) - (4230 *
(sincol - 6));
     end;
    * 5. Employment expense allowance;
    if istuinc > 0 & spsinc > 0                    then allowances_idep5 = min(0.35 *
min(istuinc,spsinc),4000);
    else if smarital in (1,3)                    then allowances_idep5 = min(0.35 * istuinc,4000);
    else if smarital = 2 & (istuinc > 0 | spsinc > 0) then allowances_idep5 = 0;
                                            else allowances_idep5 = 0;
    * 6. Total allowances;
    allowances_idep = allowances_idep1 + allowances_idep2 + allowances_idep3 +
allowances_idep4 + allowances_idep5;

    ************************************************************************;
    * Estimate of available income;
    ************************************************************************;
    * Total income (CINCOME) – total allowances;
    ai_idep = round(cincome - allowances_idep);

    ************************************************************************;
    * Estimate Total contribution
    ************************************************************************;
    if ai_idep < -3409 & missing(ai_idep) = 0 then contrib_idepaai = -750;
    else if -3409 <= ai_idep <= 15700       then contrib_idepaai = 0.22 * ai_idep;
    else if 15701 <= ai_idep <= 19700       then contrib_idepaai = 3454 + (0.25 * (ai_idep -
15700));
    else if 19701 <= ai_idep <= 23700       then contrib_idepaai = 4454 + (0.29 * (ai_idep -
19700));
    else if 23701 <= ai_idep <= 27700       then contrib_idepaai = 5614 + (0.34 * (ai_idep -
23700));
    else if 27701 <= ai_idep <= 31700       then contrib_idepaai = 6974 + (0.4 *  (ai_idep -
27700));
    else if ai_idep >= 31701              then contrib_idepaai = 8574 + (0.47 * (ai_idep -
31700));
    if contrib_idepaai < 0 & missing(contrib_idepaai) = 0 then contrib_idepaai = 0;

    ************************************************************************;
```

```
      * Calculate expected family contribution;
      ***********************************************************************;
      if missing(efc) = 1 then
       do;
        if sincol > 0 then efc = contrib_idepaai / sincol;
        * If negative, set to zero;
        if missing(efc) = 0 & efc < 0 then efc = 0;
        * Cap: If any value for EFC is 999,999, then set it to 1,000,000;
        if efc >= 999999 then efc = 1000000;
     end;

  ***************************************************************************;
    * Final edits to EFC;
    ***************************************************************************;
    * Cap: If any value for EFC is 999,999, then set it to 1,000,000;
    if efc >= 999999 then efc = 1000000;
    efc = round(efc);
```

Variable Name	**EMPLWAIV**
Variable Label	Institutional tuition waivers for staff
Description	Total amount of tuition waivers for staff and families of staff at the institution attended during the 2015–16 academic year.

Assigned Values	**Value** – **Value label**
	0 – {Zero}
	c – {Continuous}

Data Source(s)	NPSAS:16 Student Records
Component Derived Variables	Not applicable
Applies to	All study members
Availability	NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16
Change History	Unchanged from NPSAS:2000 through NPSAS:16.
N:16 Usage Statistics	Number of uses: 7 (UG); 3 (GR)
Programming Narrative	Sum of “Tuition waivers for faculty/staff” from student records. 1. Graduate aid section: a. If CFAGRTYP = 9 and CFAGRAMT>0 b. Else if CFAGRTYP = 9 and CFAGRAMT = missing, then set minimum imputation flag to 1, to impute positive value. 2. Institutional aid section: a. If CFAITYP = 9 and CFAIAMT>0 b. Else if CFAITYP = 9 and CFAIAMT = missing, then set minimum imputation flag to 1, to impute positive value. 3. Other aid section: a. If CFAOTHSRC = 1 and CFAOTHTYP = 9 and CFAOTHAMT>0 b. Else if CFAOTHSRC = 1 and CFAOTHTYP = 9 and CFAOTHAMT = missing, then set minimum imputation flag to 1, to impute positive value.

SAS Code

```
* Source variable cfagrtyp01-cfagrtyp03="Graduate aid program type";
* Source variable cfagramt01-cfagramt03="Graduate aid program amount";  *** 9 =
Faculty/staff tuition waivers ;
* Source variable cfaityp01-cfaityp03="Institution aid program type";
* Source variable cfaiamt01-cfaiamt03="Institution aid program amount";
* Source variables cfa1othsrc-cfa3othsrc="Other aid source";
* Source variables cfa1othtyp-cfa3othtyp="Other aid type";
* Source variables cfa1othamt-cfa3othamt="Other aid amount";
* Source variable cfacvans="Received any financial aid";
  ****************************************************************;
  array gtyp    (3) cfagrtyp01 cfagrtyp02 cfagrtyp03;
  array gtemp   (3) emplg1     emplg2     emplg3;
  array gamt    (3) cfagramt01 cfagramt02 cfagramt03;
  array ityp    (3) cfaityp01  cfaityp02  cfaityp03;
  array itemp   (3) empli1     empli2     empli3;
  array iamt    (3) cfaiamt01  cfaiamt02  cfaiamt03;
  array otemp   (3) emplo1     emplo2     emplo3;
  array othsrc  (3) cfa1othsrc cfa2othsrc cfa3othsrc;
  array othtype (3) cfa1othtyp cfa2othtyp cfa3othtyp;
  array oamt    (3) cfa1othamt cfa2othamt cfa3othamt;

  do i = 1 to 3;
    gtemp(i) = 0;
    if cfagraid ~= 0 then
     do;
      if gtyp(i) = 9 then
       do;
        gtemp(i) = gamt(i);
        if missing(gtemp(i)) = 1 then gtemp(i) = .m;
       end;
      else if cfagraid = .x then gtemp(i) = .x;
     end;

    itemp(i) = 0;
    if cfainstaid ~= 0 then
     do;
      if ityp(i) = 9 then
       do;
        itemp(i) = iamt(i);
        if missing(itemp(i)) = 1 then itemp(i) = .m;
       end;
      else if cfainstaid = .x then itemp(i) = .x;
     end;

    otemp(i) = 0;
    if cfaothaid ~= 0 then
     do;
      if othsrc(i) = 1 & othtype(i) = 9 then
       do;
        otemp(i) = oamt(i);
        if missing(otemp(i)) = 1 then otemp(i) = .m;
```

```
              end;
            else if cfaothaid = .x then otemp(i) = .x;
          end;
    end;
    array forEmplwaiv(9) emplg1 emplg2 emplg3 empli1 empli2 empli3 emplo1 emplo2 emplo3;
    do i = 1 to 9;
        if forEmplwaiv(i) > 0 then
          do j = 1 to 9;
            if forEmplwaiv(j) = .x then forEmplwaiv(j) = 0;
          end;
    end;
    emplwaiv = emplg1 + emplg2 + emplg3 + empli1 + empli2 + empli3 + emplo1 + emplo2 +
emplo3;
    if missing(emplwaiv) = 1 then
      do;
        emplwaiv = .x;
       end;
```

<table>
<tr><td><b>Variable Name</b></td><td><b>EMPLYAM3</b></td></tr>
<tr><td><b>Variable Label</b></td><td>Employer aid (student & parents)</td></tr>
<tr><td><b>Description</b></td><td>Total amount of aid received from the student’s or the parents’ employers during the 2015–16 academic year.</td></tr>
<tr><td><b>Assigned Values</b></td><td><table><tr><th>Value</th><th>Value label</th></tr><tr><td>0</td><td>{Zero}</td></tr><tr><td>c</td><td>{Continuous}</td></tr></table></td></tr>
<tr><td><b>Data Source(s)</b></td><td>NPSAS:16 Student Records, NPSAS:16 Interview</td></tr>
<tr><td><b>Component Derived Variables:</b></td><td>Not applicable</td></tr>
<tr><td><b>Applies to</b></td><td>All study members</td></tr>
<tr><td><b>Availability</b></td><td>NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16</td></tr>
<tr><td><b>Change History</b></td><td>Unchanged from NPSAS:04 through NPSAS:08. In NPSAS:04 and NPSAS:08, calculated as the sum of EMPLYAM1 (aid from student’s employer, capped at $20,000) and EMPLYAM2 (aid from parents’ employers, capped at $10,000). In NPSAS:12, collected as a single item in the interview and student records and capped at $50,000. In NPSAS:16, collected as a single item in the interview and student records but not capped.</td></tr>
<tr><td><b>N:16 Usage Statistics</b></td><td>Number of uses: 380 (UG); 140 (GR)</td></tr>
<tr><td><b>Programming Narrative</b></td><td>1. Use student records:
a. Other source aid:
i. Source = 4 (private) and type = 19 (employer aid).
b. Government/Private aid:
i. Type = 3.
c. Else, take student interview response (N16CEMPGTAMT).
2. Post-imputation edits:
a. Remove double counting:
i. If abs(INGRTAMT - EMPLYAM3)/INGRTAMT<=.05 and (INGRTAMT + EMPLYAM3>BUDGETAJ and STUDMULT = 1 or INGRTAMT + EMPLYAM3 > $40,000 and STUDMULT>1), then set EMPLYAM3 = 0.</td></tr>
</table>

ii. If abs(GRASTAMT - EMPLYAM3)/GRASTAMT<=.05 and (GRASAMT + EMPLYAM3>BUDGETAJ and STUDMULT = 1 or GRASTAMT + EMPLYAM3 > $40,000 and STUDMULT>1), then set EMPLYAM3 = 0.

b. Cap edits:

i. Edit values greater than $200,000 to zero (Bad values, like $999,999).

ii. Cap at $40,000 if CONTROL = 1, and the EMPLYAM3 amount is greater than the BUDGETAJ (and STUDMULT = 1). Else if STUDMULT>1 and CONTROL = 1, cap at $40,000.

iii. Cap at $60,000 if CONTROL>1 and is greater than the BUDGETAJ (and STUDMULT = 1). Else if STUDMULT>1 and CONTROL > 1, cap at $60,000.

c. TOTAID and COA edit to EMPLYAM3 and PRIVAID.

i. Flag cases where OTHGTAMT (PRIVAID + EMPLYAM3) plus all other aid except private loans and VADODAMT is greater than the cost of attendance by 125%. This is prior to the hard edits to BUDGETAJ, where the non-tuition expenses (BUDNONAJ) are raised to the total aid amount.

ii. Reduce both PRIVAID and EMPLYAM3 by the ratio of the difference between total aid (minus PRIVLOAN & VADODAMT) and the BUDGETAJ divided by OTHGTAMT, if the case was flagged in Part 3a.

SAS Code

```
* Source variables cfa1othamt-cfa3othamt="Other aid amount";
* Source variables cfa1othsrc-cfa3othsrc="Other aid source";
* Source variables cfa1othtyp-cfa3othtyp="Other aid type";
* Source variables cfa1govamt–cfa3govamt="Private aid program amount";
* Source variables cfa1govtyp-cfa3govtyp="Private aid program type";
* Source variables n16caidemp1="Undergraduate aid 2015–16: employer financial
assistance";
* Source variables n16caidemp2="Graduate aid 2015–16: employer financial assistance";
* Source variables n16cempgtamt="Aid amount: employer financial assistance";
* Source variables grinfel="Graduate fellowship amount";
* Source variables grastamt="Total assistantships amount";
  *****************************************************************;
  array othamte(3)  othamte1-othamte3;
  * array srothamt(3) cfa1othamt cfa2othamt cfa3othamt;
  * array srothsrc(3) cfa1othsrc cfa2othsrc cfa3othsrc;
  * array srothtyp(3) cfa1othtyp cfa2othtyp cfa3othtyp;
  array govamte(3) govamte1-govamte3;
  * array srgovamt(3) cfa1govamt cfa2govamt cfa3govamt;
  * array srgovtyp(3) cfa1govtyp cfa2govtyp cfa3govtyp;
  do i = 1 to 3;
    othamte(i) = 0;
    if cfaothaid ~= 0 then
     do;
      if srothsrc(i) = 4 & srothtyp(i) = 19 then
       do;
        othamte(i) = srothamt(i);
        if missing(othamte(i)) = 1 then othamte(i) = .m;
       end;
      else if cfaothaid = .x then othamte(i) = .x;
     end;

    govamte(i) = 0;
    if cfaothgov ~= 0 then
     do;
```

```
        if srgovtyp(i) = 3 then
          do;
            govamte(i) = srgovamt(i);
            if missing(govamte(i)) = 1 then govamte(i) = .m;
          end;
        else if cfaothgov = .x then govamte(i) = .x;
      end;
end;

array forEmplyam3(6) othamte1 othamte2 othamte3 govamte1 govamte2 govamte3;
do i = 1 to 6;
    if forEmplyam3(i) > 0 then
      do j = 1 to 6;
        if forEmplyam3(j) = .x then forEmplyam3(j) = 0;
      end;
end;

if othamte1 + othamte2 + othamte3 + govamte1 + govamte2 + govamte3 > 0 then
  do;
    emplyam3 = othamte1 + othamte2 + othamte3 + govamte1 + govamte2 + govamte3;
  end;
else if n16cempgtamt > 0 then
  do;
    emplyam3 = n16cempgtamt;
  end;
else if sum(of othamte1 othamte2 othamte3 govamte1 govamte2 govamte3) > 0 |
    othamte1 = .m | othamte2 = .m | othamte3 = .m |
    govamte1 = .m | govamte2 = .m | govamte3 = .m then
  do;
    emplyam3 = othamte1 + othamte2 + othamte3 + govamte1 + govamte2 + govamte3;
    do i = 1 to 6;
        if forEmplyam3(i) = .m then emplyam3_minimp = emplyam3_minimp + 1;
    end;
    if missing(emplyam3) = 1 then do; emplyam3 = .x; end;
  end;
* Only taking positive values from student records because employer aid is underreported in that data;
else if n16cempgtamt = 0 | n16caidemp1 = 0 | n16caidemp2 = 0 then
  do;
    emplyam3 = 0;
  end;
else
  do;
    emplyam3 = .x;
  end;

* Consistency edit: If amount for graduate students (STYPELST>1) is
* identical to graduate assistantship (GRASTAMT) or graduate fellowship
* (GRINFEL) amount, set the value to zero (EMPLYAM3 = 0) to avoid duplicating aid values;
```

```
    if stypelst > 1 & emplyam3 > 0 & ((emplyam3 = grastamt & zgrastam ~= 17) | emplyam3 =
grinfel) then
      do;
        emplyam3 = 0;
      end;
```

Variable Name	ENLEN
Variable Label	Number of total months enrolled
Description	Number of months enrolled between July 2015 and June 2016.

Assigned Values	**Value** | **Value Label**
	0 | {Zero}
	c | {Continuous}

Data Source(s)	NPSAS:16 Student Records, NPSAS:16 Interview, NSC:16, NSLDS:16
Component Derived Variables	ENR01-ENR12 Note: ENR01-ENR12 are not included in this report. Information on these variables may be found in the NPSAS:16 undergraduate codebook, https://nces.ed.gov/datalab/powerstats/codebook.aspx?dataset=121&type=subject#ENR01, and the NPSAS:16 graduate student codebook, https://nces.ed.gov/datalab/powerstats/codebook.aspx?dataset=122&type=subject#ENR01.
Applies to	All study members
Availability	NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16
Change History	Unchanged from NPSAS:96 through NPSAS:16; National Student Clearinghouse has only been a data source since NPSAS:12
N:16 Usage Statistics	Number of uses: 20 (UG); 8 (GR)
Programming Narrative	Count the number of months between ENR01-ENR12 that have enrollment (values 1 or 2) to calculate the number of months enrolled.
SAS Code	

```
* Source variable enr1-enr12="Monthly enrollment status";
  ****************************************************************************;
  array enr{12} enr01-enr12;
  array em{0:2} enlen mft mpt;

  do i = 0 to 2;
    em{i} = 0;
  end;

  do i = 1 to 12;
    * Count total months enrolled;
    if enr{i} > 0 then enlen + 1;
    do j = 1 to 2;
    * Count months enrolled full-time and part-time;
      if enr{i} = j then em{j} + 1;
    end;
  end;
```

Variable Name	**FEDCUM1**
Variable Label	Cumulative federal loan amount for undergraduate education
Description	Cumulative federal loan amounts borrowed for undergraduate education through June 30, 2016.

Assigned Values	**Value** | **Value label**
	0 | {Zero}
	c | {Continuous}

Data Source(s)	NSLDS:16, NPSAS:16 Student Records
Component Derived Variables	STFCUM1, PERKCUM1, STYPELST, BAYEARM, STAFFAMT, PERKAMT Note: PERKCUM1 and BAYEARM are not included in this report. Information on these variables may be found in the NPSAS:16 undergraduate codebook, https://nces.ed.gov/datalab/powerstats/codebook.aspx?dataset=121&type=subject#PERKCUM1, and the NPSAS:16 graduate student codebook, https://nces.ed.gov/datalab/powerstats/codebook.aspx?dataset=122&type=subject#PERKCUM1 and https://nces.ed.gov/datalab/powerstats/codebook.aspx?dataset=122&type=subject#BAYEARM.
Applies to	All study members
Availability	NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16
Change History	Unchanged from NPSAS:04 through NPSAS:16.
N:16 Usage Statistics	Number of uses: 698 (UG); 47 (GR)
Programming Narrative	1. Assign all non-consolidated loans with missing academic levels to undergraduate or graduate based on the following criteria: a. If undergraduate in NPSAS (STYPELST=1), set loans to undergraduate b. For graduate students (STYPELST>1), set to undergraduate if beginning enrollment date of loan is before their bachelor's degree award date (per_beg_dt<BAYEARM); else set loan to graduate-level. 2. Calculate cumulative amount disbursed according to loan types using *tot_dis*, a variable sourced from NSLDS data that represents the total amount disbursed for a loan: a. pastStsbcum1, the cumulative subsidized Stafford loan amount for undergraduate education before the NPSAS year (loan_type in "D1" "SF") if academic level of loan is undergraduate (1<=acad_lvl<=5). b. pastUnsbcum1, the cumulative unsubsidized Stafford loan amount for undergraduate education before the NPSAS year (loan_type in "D2" "SU" "SL") if academic level of loan is undergraduate (1<=acad_lvl<=5). c. pastPerkcum1, the cumulative Perkins amount for undergraduate education before the NPSAS year (loan_type in "PU" "NU" "EU") if academic level of loan is undergraduate (1<=acad_lvl<=5). 3. Create individual derived variables for Stafford and Perkins loans. a. STFCUM1, the cumulative amount for undergraduate-level Stafford loans: SUM of pastStsbcum1, pastUnsbcum1 and STAFFAMT (derived variable created in federalaid1.sas that is the cumulative loan amount for both subsidized and unsubsidized Stafford loans in the NPSAS year). i. For graduate students (STYPELST>1), do not include STAFFAMT. b. Perkcum1, the cumulative amount for undergraduate-level Perkins loans: SUM of pastPerkcum1 and PERKAMT (derived variable created in federalaid1.sas that is the cumulative loan amount for Perkins loans in the NPSAS year) i. For graduate students (STYPELST>1), do not include PERKAMT.

	4. Add the derived variables together: FEDCUM1 = STFCUM1 + PERKCUM1 5. Consistency edit: If FEDCUM1 = 0, check for whether or not the cumulative federal loan amount owed, principal and interest, for undergraduate education (FEDDUE1) is positive. If it is (i.e. FEDDUE1 > 0), set FEDCUM1 = FEDDUE1. For more details on FEDDUE1, see associated entry.

SAS Code	* Source variable stfcum1="Cumulative Direct Subsidized and Unsubsidized Loans for undergrad";
	* Source variable perkcum1="Cumulative Perkins amount for undergrad";
	**;
	fedcum1 = stfcum1 + perkcum1;

<table>
<tr><td><b>Variable Name</b></td><td><b>FEDCUM2</b></td></tr>
<tr><td><b>Variable Label</b></td><td>Cumulative federal loan amount for graduate education</td></tr>
<tr><td><b>Description</b></td><td>Cumulative federal loan amount borrowed for graduate education through June 30, 2016.</td></tr>
<tr><td><b>Assigned Values</b></td><td><table><tr><th>Value</th><th>Value label</th></tr><tr><td><b>0</b></td><td>{Zero}</td></tr><tr><td><b>c</b></td><td>{Continuous}</td></tr></table></td></tr>
<tr><td><b>Data Source(s)</b></td><td>NSLDS:16, NPSAS:16 Student Records</td></tr>
<tr><td><b>Component Derived Variables</b></td><td>STFCUM2, PERKCUM2, GPLUSCUM, STYPELST, BAYEARM
Note: PERKCUM2, GPLUSCUM, and BAYEARM are not included in this report. Information on these variables may be found in the NPSAS:16 graduate student codebook, https://nces.ed.gov/datalab/powerstats/codebook.aspx?dataset=122&type=subject#PERKCUM2, https://nces.ed.gov/datalab/powerstats/codebook.aspx?dataset=122&type=subject#GPLUSCUM, and https://nces.ed.gov/datalab/powerstats/codebook.aspx?dataset=122&type=subject#BAYEARM.</td></tr>
<tr><td><b>Applies to</b></td><td>All graduate study members</td></tr>
<tr><td><b>Availability</b></td><td>NPSAS:04, NPSAS: 08, NPSAS:12, NPSAS:16</td></tr>
<tr><td><b>Change History</b></td><td>Unchanged from NPSAS:96 through NPSAS:16.</td></tr>
<tr><td><b>N:16 Usage Statistics</b></td><td>Number of uses: 0 (UG); 543 (GR)</td></tr>
<tr><td><b>Programming Narrative</b></td><td>1. Since this is a graduate level variable, set to missing for undergraduates.
2. Assign loans with missing academic levels to undergraduate or graduate based on the following criteria:
a. If undergraduate in NPSAS (STYPELST=1), set loans to undergraduate
b. For graduate students (STYPELST>1), set to undergraduate if beginning enrollment date of loan is before their bachelor’s degree award date (per_beg_dt<BAYEARM); else set loan to graduate-level.
3. Calculate cumulative amount disbursed according to loan types using <i>tot_dis</i>, a variable sourced from NSLDS data that represents the total amount disbursed for a loan:
a. pastStsbcum2, the cumulative subsidized Stafford loan amount for graduate education before the NPSAS year (loan_type in “D1” “SF”) if academic level of loan is graduate (6<=acad_lvl<=10).
b. pastUnsbcum2, the cumulative unsubsidized Stafford loan amount for graduate education before the NPSAS year (loan_type in “D2” “SU” “SL”) if academic level of loan is graduate (6<=acad_lvl<=10).</td></tr>
</table>

   c. pastPerkcum2, the cumulative Perkins amount for graduate education before the NPSAS year (loan_type in “PU” “NU” “EU”) if academic level of loan is graduate (6<=acad_lvl<=10).
   d. pastGpluscum, the cumulative graduate plus amount before the NPSAS year (loan_type in “D3” “GB”) if academic level of loan is graduate (6<=acad_lvl<=10).

4. Create individual derived variables for Stafford and Perkins loans.
   a. STFCUM2, the cumulative amount for graduate-level Stafford loans: SUM of pastStsbcum2, pastUnsbcum2 and STAFFAMT (derived variable created in federalaid1.sas that is the cumulative loan amount for both subsidized and unsubsidized Stafford loans in the NPSAS year).
   b. PERKCUM2, the cumulative amount for graduate-level Perkins loans: SUM of pastPerkcum2 and PERKAMT (derived variable created in federalaid1.sas that is the cumulative loan amount for Perkins loans in the NPSAS year)
   c. GPLUSCUM, the cumulative amount for graduate plus loans: Sum of pastGpluscum and GPLUSAMT (derived variable created in federalaid1.sas that is the cumulative amount of graduate plus loans in the NPSAS year).
5. Add the derived variables together: FEDCUM2 = STFCUM2 + PERKCUM2 + GPLUSCUM
6. Consistency edit: If FEDCUM2 = 0, check for whether the cumulative federal loan amount owed, principal and interest, for undergraduate education (FEDDUE2) is positive. If it is (i.e. FEDDUE2 > 0), set FEDCUM2 = FEDDUE2. For more details on FEDDUE2, see associated entry.

SAS Code

```
* Source variable stfcum2="Cumulative Direct Subsidized and Unsubsidized Loans for grad";
* Source variable perkcum2="Cumulative Perkins amount for grad";
* Source variable gpluscum="Cumulative Graduate PLUS Loan amount";
**************************************************************;
* Applies to: All graduate study members;
  if stypelst = 1      then fedcum2 = .y;
  else if stypelst > 1 then fedcum2 = stfcum2 + perkcum2 + gpluscum;
```

<table>
<tr><td><b>Variable Name</b></td><td><b>FEDCUM3</b></td></tr>
<tr><td><b>Variable Label</b></td><td>Cumulative federal loan amount for undergraduate and graduate education</td></tr>
<tr><td><b>Description</b></td><td>Cumulative federal loan amount borrowed for undergraduate and graduate education through June 30, 2016.</td></tr>
<tr><td><b>Assigned Values</b></td><td><table><tr><th>Value</th><th>Value label</th></tr><tr><td><b>0</b></td><td>{Zero}</td></tr><tr><td><b>c</b></td><td>{Continuous}</td></tr></table></td></tr>
<tr><td><b>Data Source(s)</b></td><td>NSLDS:16, NPSAS:16 Student Records</td></tr>
<tr><td><b>Component Derived Variables</b></td><td>FEDCUM1, FEDCUM2, STYPELST</td></tr>
<tr><td><b>Applies to</b></td><td>All graduate study members</td></tr>
<tr><td><b>Availability</b></td><td>NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16</td></tr>
<tr><td><b>Change History</b></td><td>Unchanged from NPSAS:96 through NPSAS:16.</td></tr>
<tr><td><b>N:16 Usage Statistics</b></td><td>Number of uses: 0 (UG); 106 (GR)</td></tr>
</table>

Programming Narrative	Add together cumulative federal loan amount for undergraduate education (FEDCUM1) and cumulative federal loan amount for graduate education (FEDCUM2). Since this is a graduate level variable, set to missing for undergraduates. Note: FEDCUM1 and FEDCUM2 are determined by a large variety of interim variables. For more information, see their respective entries.

SAS Code	* Source variable fedcum1="Cumulative federal loan amount for undergrad";
	* Source variable fedcum2="Cumulative federal loan amount for grad";
	**;
	* Applies to: All graduate study members;
	   if stypelst = 1 then fedcum3 = .y;
	* Add together cumulative federal loan amount for undergraduate education
	* (FEDCUM1) and cumulative federal loan amount for graduate education
	* (FEDCUM2);
	   else if stypelst > 1 then fedcum3 = fedcum1 + fedcum2;

<table>
<tr><td><b>Variable Name</b></td><td><b>FEDDUE1</b></td></tr>
<tr><td><b>Variable Label</b></td><td>Cumulative federal loan amount owed, principal & interest, undergraduate education</td></tr>
<tr><td><b>Description</b></td><td>Indicates total amount owed, both principal and interest, on all federal loans for undergraduate education as of late 2016. Excludes Direct PLUS Loans to parents of undergraduate students.</td></tr>
<tr><td><b>Assigned Values</b></td><td><table><tr><th>Value</th><th>Value label</th></tr><tr><td><b>0</b></td><td>{Zero}</td></tr><tr><td><b>c</b></td><td>{Continuous}</td></tr></table></td></tr>
<tr><td><b>Data Source(s)</b></td><td>NSLDS:16</td></tr>
<tr><td><b>Component Derived Variables</b></td><td>FEDCUM1, STYPELST, BAYEARM
Note: BAYEARM is not included in this report. Information on this variable may be found in the NPSAS:16 graduate student codebook, https://nces.ed.gov/datalab/powerstats/codebook.aspx?dataset=122&type=subject#BAYEARM.</td></tr>
<tr><td><b>Applies to</b></td><td>All study members</td></tr>
<tr><td><b>Availability</b></td><td>NPSAS:12, NPSAS:16</td></tr>
<tr><td><b>Change History</b></td><td>Unchanged from NPSAS:12 through NPSAS:16.</td></tr>
<tr><td><b>N:16 Usage Statistics</b></td><td>Number of uses: 96 (UG); 15 (GR)</td></tr>
<tr><td><b>Programming Narrative</b></td><td>1. Assign all non-consolidated loans with missing academic levels to undergraduate or graduate based on the following criteria:
a. If undergraduate in NPSAS (STYPELST=1), set loans to undergraduate
b. For graduate students (STYPELST>1), set to undergraduate if beginning enrollment date of loan is before their bachelor’s degree award date (per_beg_dt<BAYEARM); else set loan to graduate-level.
2. Derive the necessary interim variables
a. Use the same interim variables derived for FEDOWE1, for the outstanding principal balance.
b. Create outstanding interest variables using <i>outint</i>, a variable sourced from NSLDS data that represents the outstanding interest balance:
i. pastPerkint1: outstanding interest balance for undergraduate Perkins loans before the NPSAS year</td></tr>
</table>

	ii. npsasPerkint1: outstanding interest balance for undergraduate Perkins loans in the NPSAS year iii. pastStsbint1: outstanding interest balance for undergraduate subsidized Stafford loans before the NPSAS year iv. npsasStsbint1: outstanding interest balance for undergraduate subsidized Stafford loans in the NPSAS year v. pastUnsbint1: outstanding interest balance for undergraduate unsubsidized Stafford loans before the NPSAS year vi. npsasUnsbint1: outstanding interest balance for undergraduate unsubsidized Stafford loans in the NPSAS year c. For graduate students only who have consolidated loans, derive interim variables uconoutx and uconintx. These are variables that allocate the unknown consolidated principal or interest (listed as conoutx and conintx in the data set respectively) based on the ratio of undergraduate loans to total loans. Note: this is necessary because consolidated loans do not get assigned academic levels in NSLDS and could contain both undergraduate and/or graduate loans. 3. Sum the interim variables a. Perkdue1 (outstanding principal and interest undergraduate Perkins loans amounts combined) = pastPerkout1 + npsasPerkout1 + pastPerkint1 + npsasPerkint1 b. Stfdue1 (outstanding principal and interest undergraduate Stafford loans amounts combined; includes both subsidized and unsubsidized Stafford loans) = pastStsbout1 + npsasStsbout1 + pastUnsbout1 + npsasUnsbout1 + pastStsbint1 + npsasStsbint1 + pastUnsbint1 + npsasUnsbint1 c. If the loan item falls under the correct loan type and academic level, conout1 = outstand and conint1 = outint i. For graduate students, conout1 = outstand + uconoutx and conint1 = outint + uconintx 4. Create the derived variable a. FEDDUE1 = perkdue1 + stfdue1 + conout1 + conint1
SAS Code	* Source variable perkdue1="Outstanding principal and interest * undergraduate Perkins loans amounts combined"; * Source variable stfdue1="Outstanding principal and interest * undergraduate Stafford loans amounts combined"; * Source variable conout1="Outstanding principal balance on consolidated * undergraduate loans"; * Source variable conint1="Outstanding interest balance on consolidated * undergraduate loans"; **; feddue1 = perkdue1 + stfdue1 + conout1 + conint1;

<table>
<tr><td><b>Variable Name</b></td><td><b>FEDDUE2</b></td></tr>
<tr><td><b>Variable Label</b></td><td>Cumulative federal loan amount owed, principal & interest, graduate education</td></tr>
<tr><td><b>Description</b></td><td>Indicates total amount owed, both principal and interest, on all federal loans for graduate education as of late 2016.</td></tr>
<tr><td><b>Assigned Values</b></td><td><table><tr><th>Value</th><th>Value label</th></tr><tr><td><b>0</b></td><td>{Zero}</td></tr><tr><td><b>c</b></td><td>{Continuous}</td></tr></table></td></tr>
<tr><td><b>Data Source(s)</b></td><td>NSLDS:16</td></tr>
</table>

Component Derived Variables	FEDCUM2, BAYEARM, STYPELST Note: BAYEARM is not included in this report. Information on this variable may be found in the NPSAS:16 graduate student codebook, https://nces.ed.gov/datalab/powerstats/codebook.aspx?dataset=122&type=subject#BAYEARM.
Applies to	All graduate study members
Availability	NPSAS:12, NPSAS:16
Change History	Unchanged from NPSAS:12 through NPSAS:16.
N:16 Usage Statistics	Number of uses: 0 (UG); 113 (GR)
Programming Narrative	1. Assign all non-consolidated loans with missing academic levels to undergraduate or graduate based on the following criteria: a. If undergraduate in NPSAS (STYPELST=1), set loans to undergraduate. Since this is a graduate level variable, set to missing for undergraduates. b. For graduate students (STYPELST>1), set to undergraduate if beginning enrollment date of loan is before their bachelor's degree award date (per_beg_dt<BAYEARM); else set loan to graduate-level. 2. Derive the necessary interim variables a. Use the same interim variables derived for FEDOWE2, for the outstanding principal balance. b. Create outstanding interest variables using *outint*, a variable sourced from NSLDS data that represents the outstanding interest balance: i. pastPerkint2: outstanding interest balance for graduate Perkins loans before the NPSAS year ii. npsasPerkint2: outstanding interest balance for graduate Perkins loans in the NPSAS year iii. pastStsbint2: outstanding interest balance for graduate subsidized Stafford loans before the NPSAS year iv. npsasStsbint2: outstanding interest balance for graduate subsidized Stafford loans in the NPSAS year v. pastUnsbint2: outstanding interest balance for graduate unsubsidized Stafford loans before the NPSAS year vi. npsasUnsbint2: outstanding interest balance for graduate unsubsidized Stafford loans in the NPSAS year vii. pastGplusint: outstanding interest balance for graduate plus loans before the NPSAS year viii. npsasGplusint: outstanding interest balance for graduate plus loans in the NPSAS year. c. For graduate students only, who may have consolidated loans of "unknown" status (i.e. the grade level to which the loan corresponds to is unknown), derive interim variables uconoutx and uconintx. These are variables that allocate the unknown consolidated principal or interest (listed as conoutx and conintx in the data set respectively) based on the ratio of graduate loans to total loans. 3. Sum the interim variables a. Perkdue2 (outstanding principal and interest graduate Perkins loans amounts combined) = pastPerkout2 + npsasPerkout2 + pastPerkint2 + npsasPerkint2 b. Stfdue2 (outstanding principal and interest graduate Stafford loans amounts combined; includes both subsidized and unsubsidized Stafford loans) = pastStsbout2 + npsasStsbout2 + pastUnsbout2 + npsasUnsbout2 + pastStsbint2 + npsasStsbint2 + pastUnsbint2 + npsasUnsbint2 c. Gplusdue (outstanding principal and interest graduate plus amount) = pastGplusout + npsasGplusout + pastGplusint + npsasGplusint d. If the loan item falls under the correct loan type and academic level, conout2 = outstand and conint2 = outint

	i. For graduate students, conout2 = outstand + (conoutx-uconoutx) and conint2 = outint + (conintx-uconintx) 4. Add the cumulative variables together a. FEDDUE2 = perkdue2 + stfdue2 + gplusdue + conout2 + conint2

SAS Code	* Source variable stypelst="Student type indicator";
	* Source variable perkdue2="Outstanding principal and interest graduate
	* Perkins loans amounts combined";
	* Source variable stfdue2="Outstanding principal and interest graduate
	* Stafford loans amounts combined";
	* Source variable gplusdue="Outstanding principal and interest graduate
	* plus amount";
	* Source variable conout2="Outstanding principal balance on consolidated
	* graduate loans";
	* Source variable conint2="Outstanding interest balance on consolidated
	* graduate loans";
	**;
	if stypelst = 1 then feddue2 = .y;
	else if stypelst > 1 then feddue2 = perkdue2 + stfdue2 + gplusdue + conout2 + conint2;

Variable Name	**FEDDUE3**
Variable Label	Cumulative federal loan amount owed, principal & interest, undergraduate and graduate education
Description	Indicates total amount owed, both principal and interest, on all federal loans as of late 2016.

Assigned Values	**Value** | **Value label**
	0 | {Zero}
	c | {Continuous}

Data Source(s)	NSLDS:16
Component Derived Variables	FEDDUE1, FEDDUE2
Applies to	All graduate study members
Availability	NPSAS:12, NPSAS:16.
Change History	Unchanged from NPSAS:12 through NPSAS:16.
N:16 Usage Statistics	Number of uses: 0 (UG); 59 (GR)
Programming Narrative	Add together cumulative federal loan amount owed, principal and interest, for undergraduate education (FEDDUE1) and cumulative federal loan amount owed, principal and interest, for graduate education (FEDDUE2). Since this is a graduate level variable, set to missing for undergraduates. Note: FEDDUE1 and FEDDUE2 are determined by a large variety of interim variables. For more information, see their respective entries.

<table>
<tr><td><b>SAS Code</b></td><td>**;
* Source variable stypelst="Student type indicator";
* Source variable feddue1="Cum. federal loan amount owed, principal &
* interest, undergraduate";
* Source variable feddue2="Cum. federal loan amount owed, principal &
* interest, graduate";
**;
if stypelst = 1 then feddue3 = .y;
* Add together cumulative federal loan amount owed, principal and
* interest, for undergraduate education (FEDDUE1) and cumulative federal
* loan amount owed, principal and interest, for graduate education
* (FEDDUE2);
else if stypelst > 1 then feddue3 = feddue1 + feddue2;</td></tr>
</table>

<table>
<tr><td><b>Variable Name</b></td><td><b>FEDOWE1</b></td></tr>
<tr><td><b>Variable Label</b></td><td>Cumulative federal loan amount owed for undergraduate education</td></tr>
<tr><td><b>Description</b></td><td>Indicates total amount owed (principal only) on all federal loans for undergraduate education as of late 2016. Excludes Direct PLUS Loans to parents of undergraduate students.</td></tr>
<tr><td><b>Assigned Values</b></td><td><table><tr><th>Value</th><th>Value label</th></tr><tr><td><b>0</b></td><td>{Zero}</td></tr><tr><td><b>c</b></td><td>{Continuous}</td></tr></table></td></tr>
<tr><td><b>Data Source(s)</b></td><td>NSLDS:16</td></tr>
<tr><td><b>Component Derived Variables</b></td><td>STYPELST, BAYEARM
Note: BAYEARM is not included in this report. Information on this variable may be found in the NPSAS:16 graduate student codebook, https://nces.ed.gov/datalab/powerstats/codebook.aspx?dataset=122&type=subject#BAYEARM.</td></tr>
<tr><td><b>Applies to</b></td><td>All study members</td></tr>
<tr><td><b>Availability</b></td><td>NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16</td></tr>
<tr><td><b>Change History</b></td><td>Unchanged from NPSAS:04 through NPSAS:16. Variable was labeled OWEFED in NPSAS:1996. Split into graduate versus undergraduate loans in NPSAS:2000 and named OWEFED1. Changed name to FEDOWE1 in NPSAS:04-NPSAS:16.</td></tr>
<tr><td><b>N:16 Usage Statistics</b></td><td>Number of uses: 96 (UG); 21 (GR)</td></tr>
<tr><td><b>Programming Narrative</b></td><td>1. Assign all non-consolidated loans with missing academic levels to undergraduate or graduate based on the following criteria:
a. If undergraduate in NPSAS (STYPELST=1), set loans to undergraduate
b. For graduate students (STYPELST>1), set to undergraduate if beginning enrollment date of loan is before their bachelor’s degree award date (per_beg_dt<BAYEARM); else set loan to graduate-level.
2. Calculate the necessary interim variables.
a. For each variable, sum the total for <i>outstand</i>, a variable sourced from NSLDS data that represents the outstanding principal balance:
i. pastPerkout1: outstanding principal balance for undergraduate Perkins loans before the NPSAS year (loan_type in “PU” “NU” “EU” and academic level of loan is undergraduate [1<=acad_lvl<=5] and academic year < 2016).</td></tr>
</table>

	ii. npsasPerkout1: outstanding principal balance for undergraduate Perkins loans in the NPSAS year (if loan_type in “PU” “NU” “EU” and academic level of loan is undergraduate [1<=acad_lvl<=5] and academic year = 2016). iii. pastStsbout1: outstanding principal balance for undergraduate subsidized Stafford loans before the NPSAS year (loan_type in “D1” “SF” and academic level is undergraduate [1<=acad_lvl<=5] and academic year < 2016). iv. npsasStsbout1: outstanding principal balance for undergraduate subsidized Stafford loans in the NPSAS year (loan_type in “D1” “SF” and academic level is undergraduate [1<=acad_lvl<=5] and academic year = 2016). v. pastUnsbout1: outstanding principal balance for undergraduate unsubsidized Stafford loans before the NPSAS year (loan_type in “D2” “SU” “SL” and academic level is undergraduate [1<=acad_lvl<=5] and academic year < 2016). vi. npsasUnsbout1: outstanding principal balance for undergraduate unsubsidized Stafford loans in the NPSAS year (loan_type in “D2” “SU” “SL” and academic level is undergraduate [1<=acad_lvl<=5] and academic year = 2016). vii. conout1: outstanding balance on consolidated undergraduate loans (loan_type in “CL” “D5” “D6” “D7”) b. For graduate students only, who have consolidated loans, derive interim variable uconoutx, to allocate the consolidated loan principal balance based on the ratio of undergraduate loans to total loans. Note: this is necessary because consolidated loans do not get assigned academic levels in NSLDS and could contain both undergraduate and/or graduate loans. 3. Create the composite variables. a. Perkout1 (outstanding principal amount for undergraduate Perkins loans) = pastPerkout1 + npsasPerkout1 b. Stfout1 (outstanding principal amount for undergraduate Stafford loans) = pastStsbout1 + pastUnsbout1 + npsasStsbout1 + npsasUnsbout1 c. For graduate students only, conout1 = conout1 + uconoutx 4. Derive the variable. a. FEDOWE1=perkout1 + stfout1 + conout1

SAS Code	* Source variable perkout1="Outstanding principal amount for
	* undergraduate Perkins loans";
	* Source variable stfout1="Outstanding principal amount for undergraduate
	* Stafford loans";
	* Source variable conout1="Outstanding balance on consolidated
	* undergraduate loans";
	**;
	fedowe1 = perkout1 + stfout1 + conout1;

<table>
<tr><td><b>Variable Name</b></td><td><b>FEDOWE2</b></td></tr>
<tr><td><b>Variable Label</b></td><td>Cumulative federal loan amount owed for graduate education</td></tr>
<tr><td><b>Description</b></td><td>Indicates total amount owed (principal only) on all federal loans for graduate education as of late 2016.</td></tr>
<tr><td><b>Assigned Values</b></td><td><table><tr><th>Value</th><th>Value label</th></tr><tr><td><b>0</b></td><td>{Zero}</td></tr><tr><td><b>c</b></td><td>{Continuous}</td></tr></table></td></tr>
<tr><td><b>Data Source(s)</b></td><td>NSLDS:16</td></tr>
</table>

Component Derived Variables	STYPELST, BAYEARM Note: BAYEARM is not included in this report. Information on this variable may be found in the NPSAS:16 graduate student codebook, https://nces.ed.gov/datalab/powerstats/codebook.aspx?dataset=122&type=subject#BAYEARM.
Applies to	All graduate study members
Availability	NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16
Change History	Variable was labeled OWEFED in NPSAS:1996. Split into graduate versus undergraduate loans in NPSAS:2000 and named OWEFED2. Changed name to FEDOWE2 in NPSAS:04-NPSAS:16.
N:16 Usage Statistics	Number of uses: 0 (UG); 56(GR)
Programming Narrative	1. Since this is a graduate-level variable, set to missing for undergraduates. 2. Assign all non-consolidated loans with missing academic levels to undergraduate or graduate based on the following criteria: a. If undergraduate in NPSAS (STYPELST=1), set loans to undergraduate b. For graduate students (STYPELST>1), set to undergraduate if beginning enrollment date of loan is before their bachelor's degree award date (per_beg_dt<BAYEARM); else set loan to graduate-level. 3. Calculate the necessary interim variables. a. For each variable, sum the total for *outstand*, a variable sourced from NSLDS data that represents the outstanding principal balance: i. pastPerkout2: outstanding principal balance for graduate Perkins loans before the NPSAS year (loan_type in "PU" "NU" "EU" and academic level of loan is graduate [6<=acad_lvl<=10] and academic year < 2016). ii. npsasPerkout2: outstanding principal balance for graduate Perkins loans in the NPSAS year (if loan_type in "PU" "NU" "EU" and academic level of loan is graduate [6<=acad_lvl<=10] and academic year = 2016). iii. pastStsbout2: outstanding principal balance for graduate subsidized Stafford loans before the NPSAS year (loan_type in "D1" "SF" and academic level is graduate [6<=acad_lvl<=10] and academic year < 2016). iv. npsasStsbout2: outstanding principal balance for graduate subsidized Stafford loans in the NPSAS year (loan_type in "D1" "SF" and academic level is graduate [6<=acad_lvl<=10] and academic year = 2016). v. pastUnsbout2: outstanding principal balance for graduate unsubsidized Stafford loans before the NPSAS year (loan_type in "D2" "SU" "SL" and academic level is graduate [6<=acad_lvl<=10] and academic year < 2016). vi. npsasUnsbout2: outstanding principal balance for graduate unsubsidized Stafford loans in the NPSAS year (loan_type in "D2" "SU" "SL" and academic level is graduate [6<=acad_lvl<=10] and academic year = 2016). vii. pastGplusout: outstanding principal balance for graduate plus loans before the NPSAS year (loan_type in "D3" "GB" and academic level is graduate [6<=acad_lvl<=10] and academic year < 2016). viii. npsasGplusout: outstanding principal balance for graduate plus loans for the NPSAS year (loan_type in "D3" "GB" and academic level is graduate [6<=acad_lvl<=10] and academic year = 2016). ix. Conout2: outstanding balance on consolidated graduate loans (loan_type in "CL" "D5" "D6" "D7") 4. Create the composite variables. a. Perkout2 (outstanding principal amount for graduate Perkins loans) = pastPerkout2 + npsasPerkout2 b. Stfout2 (outstanding principal amount for graduate Stafford loans) = pastStsbout2 + pastUnsbout2 + npsasStsbout2 + npsasUnsbout2 c. Gplusout (outstanding principal amount for graduate plus loans) = pastGplusout + npsasGplusout d. Conout2 = conout2 + uconoutx 5. Derive the variable.

	a. FEDOWE2=perkout2 + stfout2 + gplusout + conout2 b. This variable is set to missing for undergraduates, as this is a graduate level variable.

SAS Code	* Source variable stypelst="Student type indicator";
	* Source variable perkout2="Outstanding principal amount for graduate
	* Perkins loans";
	* Source variable stfout2="Outstanding principal amount for graduate
	* Stafford loans";
	* Source variable gplusout="Outstanding principal amount for graduate
	* plus loans";
	* Source variable conout2="Outstanding principal balance on consolidated
	* graduate loans";
	**;
	if stypelst = 1 then fedowe2 = .y;
	else if stypelst > 1 then fedowe2 = perkout2 + stfout2 + gplusout + conout2;

Variable Name	**FEDOWE3**
Variable Label	Cumulative federal loan amount owed for undergraduate or graduate education
Description	Indicates total amount owed (principal only) on all federal loans for undergraduate or graduate education as of late 2016. Excludes Direct PLUS Loans to parents of undergraduate students, which are only available to the parents of dependent undergraduates.

Assigned Values	**Value** **Value label**
	0 {Zero}
	c {Continuous}

Data Source(s)	NSLDS:16
Component Derived Variables	FEDOWE1, FEDOWE2
Applies to	All graduate study members
Availability	NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16
Change History	Variable was labled OWEFED in NPSAS:96. Split into graduate versus undergraduate loans in NPSAS:2000 and named OWEFED3. Changed name to FEDOWE3 in NPSAS:04-NPSAS:16.
N:16 Usage Statistics	Number of uses: 0 (UG); 32 (GR)
Programming Narrative	1. Since this is a graduate-level variable, set to missing for undergraduates. 2. Set to skip if study member did not borrow any undergrad or grad federal loans (FEDCUM1=0 and FEDCUM2 = 0). 3. Add together cumulative federal loan principal amount owed for undergraduate education (FEDOWE1) and cumulative federal loan principal amount owed for graduate education (FEDOWE2). Note: FEDOWE1 and FEDOWE2 are determined by a large variety of interim variables. For more information, see their respective entries.
SAS Code	* Source variable stypelst="Student type indicator"; * Source variable fedowe1="Cumulative federal loan amount owed for undergrad"; * Source variable fedowe2="Cumulative federal loan amount owed for grad"; **; * Applies to: All graduate study members; if stypelst = 1 then fedowe3 = .y;

	* Add together cumulative federal loan principal amount owed for * undergraduate education (FEDOWE1) and cumulative federal loan principal * amount owed for graduate education (FEDOWE2); else if stypelst > 1 then fedowe3 = fedowe1 + fedowe2;

Variable Name	**GPLUSAMT**
Variable Label	Direct PLUS Loans to graduate students
Description	Total amount of Direct PLUS loans to graduate students received at all institutions attended during the 2015–16 academic year.

Assigned Values	**Value** — **Value label**
	0 — {Zero}
	c — {Continuous}

Data Source(s)	NSLDS:16
Component Derived Variables	Not applicable
Applies to	All graduate study members
Availability	NPSAS:08, NPSAS:12, NPSAS:16
Change History	Unchanged from NPSAS:08 through NPSAS:16
N:16 Usage Statistics	Number of uses: 0 (UG); 350 (GR)
Programming Narrative	1. Data management: a. Subset to loans with a total disbursed amount greater than or equal to $1. b. Subset to loans whose enrollment period coincided with the 2016 federal award year (July 1, 2015 - June 30, 2016). 2. Set all undergraduate students (STYPELST = 1) to a skip (GPLUSAMT = -3). Since this is a graduate level variable, set to missing for undergraduates. 3. Sum the total amount disbursed of Federal PLUS Loans to graduate students (loan_type = "D3"). Note, no FFELP loans exist past 2010, so they are not included in this variable specification (loan_type = "GB").

SAS Code	* Source variable ndgplus="Direct PLUS Loans to graduate students (NSLDS)";
	* Source variable stypelst="Student type indicator";
	**;
	* Set all undergraduate students (STYPELST = 1) to a skip (GPLUSAMT = -3);
	if stypelst = 1 then do; gplusamt = .y; end;
	else if stypelst > 1 then
	do;
	gplusamt = ndgplus;
	end;

Variable Name	**GRADDEG**
Variable Label	Graduate degree program
Description	Indicates the general type of graduate degree program in which the student was enrolled during the 2015–16 academic year.

Assigned Values

Value	Value label
1	Master's degree
2	Post-baccalaureate or post-master's certificate
3	Doctor's degree - research/scholarship
4	Doctor's degree - professional practice
5	Doctor's degree - other
6	Not in a degree program

Data Source(s)

NPSAS:16 Interview, NPSAS:16 Student Records, FAFSA:16, NSC:16

Component Derived Variables

STYPELST

Applies to

All graduate study members

Availability

NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16

Change History

Unchanged from NPSAS:96 through NPSAS:16. NSC has only been a data source since NPSAS:12

N:16 Usage Statistics

Number of uses: 0 (UG); 1,670 (GR)

Programming Narrative

1. Since this is a graduate level variable, set to missing for undergraduates.
2. Create individual degree variables for all the sources (interview, student records, and CPS)
3. Set the source degree variables to missing when they conflict with the level of the institution
   a. If level4 is 1 (less-than 2 yr) but the degree isn't either a certificate or undergrad classes, set to missing
   b. If level4 is not 4 (4-year doctorate granting), but the degree indicates it is (7,8, or 9), set to missing
   c. If level4 is 2 (2-year institution), but degree is not (1,2,4,6), set to missing (NEW STEP)
4. First, take the student interview as the degree if it is not just classes (4 or 10) AND the source for STYPELST was the interview (ZSTYPE in (1,3,5,7))
5. Second, take student records as the degree if it is not just classes (4 or 10)
6. Third, take CPS as the degree if it is not just classes (4 or 10)
7. Fourth, take NSC as the degree.
8. If STYPELST = 3, set the degree to 4 (professional practice).
9. If degree is still missing and If STYPELST > 1 and any of the source degrees indicate the student just took graduate classes (degsi = 10 or degsr = 10 or degcps = 10) set degree to 6
10. Fix cases when 2/3 sources say one degree type, but we took another (like STYPELST)
    a. If degree from student records matches degree from cps, and both are greater than zero, set degree to the student records degree
    b. If degree from the student interview AND student records is just undergraduate classes (4) or just graduate classes (10), then use that value, rather than the valid degree from CPS.

SAS Code

```
/*--------------------- GRADDEG: Graduate degree type --------------------*/
/* variable used: STYPELST LEVEL4 LEVEL TDEGREE BENLADEG C12DEG */

/*------- (1) create variables with same categories in all 3 sources -------
DEGSI: Degree programs from student interview
DEGSR: Degree programs from student records
DEGCPS: Degree programs from CPS
```

```
5 (1)=Master's degree
6 (2)=Post-BA or post-master's certificate
7 (3)=Doctoral degree - research/scholarship
8 (4)=Doctoral degree - professional practice
9 (5)=Doctoral degree - other
10(6)=Not in a graduate degree program

contradicting cases with STYPELST:
-1=UG student who has grad degree
-2=grad student who has UG degree
*/
/*---------------- Graduate student ----------------*/

/* DEGSI */
ELSE IF STYPELST>1 THEN DO;
  IF TDEGREE=7 THEN DEGSI=5;
  ELSE IF TDEGREE IN (6,8) THEN DEGSI=6;
  ELSE IF TDEGREE=9 THEN DEGSI=7;
  ELSE IF TDEGREE=10 THEN DEGSI=8;
  ELSE IF TDEGREE=11 THEN DEGSI=9;
  ELSE IF TDEGREE=5 THEN DEGSI=10;
  ELSE IF TDEGREE>0 THEN DEGSI=-2;   /* Grad status with UG deg */

/* DEGSR */
  IF BENLADEG=7 THEN DEGSR=5;
  ELSE IF BENLADEG IN (6,8) THEN DEGSR=6;
  ELSE IF BENLADEG=9 THEN DEGSR=7;
  ELSE IF BENLADEG=10 THEN DEGSR=8;
  ELSE IF BENLADEG=11 THEN DEGSR=9;
  ELSE IF BENLADEG=5 THEN DEGSR=10;
  ELSE IF BENLADEG>0 THEN DEGSR=-2;   /* Grad status with UG deg */

/* DEGCPS: CPS do not have specific graduate degree */
END;

/* more contradicting cases with level:
-3=student enrolled at a lt-2-yr college but
  has degree type other than "certificate" and "no degree"
-4=student who did not enrolled at a doc-granting college but has doctoral degree
*/

ARRAY D DEGSI DEGSR DEGCPS;

DO OVER D;
  IF LEVEL4=1 AND D NOT IN (1,4) AND D>0 THEN D=-3;
  IF LEVEL4~=4 AND D IN (7,8,9) THEN D=-4;
END;

/*--- (2) use the 3 sources to create one variable that has all degree programs ---*/
```

	/* if missing, take student interview first, then student records, then CPS */
	IF DEGREE=. THEN DO;
	  IF DEGSI>0 AND DEGSI NOT IN (4,10) THEN DEGREE=DEGSI;
	  ELSE IF DEGSR>0 AND DEGSR NOT IN (4,10) THEN DEGREE=DEGSR;
	  ELSE IF DEGCPS>0 AND DEGCPS NOT IN (4,10) THEN DEGREE=DEGCPS;
	END;
	
	/* set professional-practice student to "8" */
	IF STYPELST=3 THEN DEGREE=8;
	
	/* if DEGREE still missing, set to UG/Grad others (only few hundreds cases) */
	IF DEGREE=. THEN DO;
	  IF STYPELST=1 THEN DEGREE=4; ELSE DEGREE=10;
	END;
	
	/*--------- (3) create UGDEG (UG) and GRADDEG (GRAD) from DEGREE ---------*/
	
	IF STYPELST=1 THEN DO; UGDEG=DEGREE; GRADDEG=-3; END;
	ELSE IF STYPELST>1 THEN DO; GRADDEG=DEGREE-4; UGDEG=-3; END;

Variable Name	**GRASTAMT**
Variable Label	Total assistantships amount
Description	Total amount of all graduate student assistantships received during the 2015–16 academic year.

Assigned Values	**Value** **Value label**
	0 {Zero}
	c {Continuous}

Data Source(s)	NPSAS:16 Interview, NPSAS:16 Student Records
Component Derived Variables	GRRESAMT, GRTEAAMT, GRGRDAMT
Applies to	All graduate study members
Availability	NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16
Change History	Unchanged from NPSAS:04 through NPSAS:16. ASTAMT from NPSAS:96 through NPSAS:2000.
N:16 Usage Statistics	Number of uses: 0 (UG); 195 (GR)
Programming Narrative	Sum of graduate research assistantship amount (GRRESAMT), graduate teaching assistantship amount (GRTEAAMT), and other graduate assistantship amount (GRGRDAMT). Since this is a graduate level variable, set to missing for undergraduates.

SAS Code	* Source variable grteaamt="Graduate teaching assistantship amount";
	* Source variable grresamt="Graduate research assistantship amount";
	* Source variable grgrdamt="Other graduate assistantship amount";
	  **;
	Applies to: All graduate study members;
	if stypelst = 1 then do; grastamt = .y; end;
	  else if stypelst > 1 then grastamt = grteaamt + grresamt + grgrdamt;

	if missing(grastamt) = 1 then grastamt = .x;
	    end;

Variable Name	**GRGRDAMT**
Variable Label	Other graduate assistantship amount
Description	Total amount of graduate assistantships other than research or teaching assistantships received during the 2015–16 academic year

Assigned Values	**Value** | **Value label**
	0 | {Zero}
	c | {Continuous}

Data Source(s)	NPSAS:16 Student Records, NPSAS:16 Interview
Component Derived Variables	Not applicable
Applies to	All graduate study members
Availability	NPSAS:04, NPSAS:08; NPSAS:12; NPSAS:16
Change History	Unchanged from NPSAS:04 through NPSAS:16.
N:16 Usage Statistics	Number of uses: 0 (UG); 38 (GR)
Programming Narrative	Since this is a graduate level variable, set to missing for undergraduates. Raw student interview data were first passed through a double-counting program to identify self-reported amounts that appeared to be identical (or nearly). All assistantship types were part of this and were sometimes set to missing if there was evidence of double-counting. Raw student records data were also passed through a similar double-counting program that eliminated some aid amounts that appeared to be double counted, largely institutional aid amounts and graduate aid amounts. Create before imputation: 1. Derive individual assistantship amounts from student records variables only: a. Grteaamt_sr = sum of (graduate aid teaching assistantships, institutional aid teaching assistantships, and other aid institutional teaching assistantships) b. Grresamt_sr = sum of (graduate aid research assistantships, institutional aid research assistantships, and other aid institutional research assistantships) c. Grgrdamt_sr = sum of (graduate aid other assistantships, institutional aid other assistantships, and other aid institutional other assistantships) d. Flag the cases where the amount is missing, but the aid type was specified as the respective assistantship type (using the .m system missing designation). 2. Derive individual assistantship amounts from the student interview only: a. Grteaamt_si = n16cgrtaamt, where the amount is greater than or equal to zero (and stypelst>1). b. Grresamt_si = n16cgrraamt, where the amount is greater than or equal to zero (and stypelst>1). c. Grgrdamt_si = n16cgrothamt, where the amount is greater than or equal to zero (and stypelst>1). 3. Use student records for all three assistantship types when: a. None of the three student records assistantship type variables are non-missing (grteaamt_sr, grresamt_sr, and grgrdamt_sr), AND

   b. All three of the student interview versions of the variable are either zero or missing (grteaamt_si, grresamt_si, and grgrdamt_si)

4. Use the student interview for all three assistantship types when
   a. All three student record assistantships are missing, AND
   b. At least one of the three student interview types are non-missing.

5. Set each assistantship type to zero if both the student records and student interview version of the variable is zero.

6. Use student records for all three assistantship types if:
   a. The case wasn’t part of either step 3 or 4, where wholesale student records or interview assistantships were taken, AND
   b. At least one assistantship value from student records for an assistantship type is greater than or equal to $5,000. This arbitrary value takes care of a handful of cases where an amount in student records is very small, but other types of assistantships were reported in the interview, for much larger amounts. These will be used in the coming steps.

7. Replace missing values for the cases from step 6 with zero. This is a logical adjustment, and, in most cases, the interview corroborates the zero value. Only a few would need to be stochastically imputed if we left it, but it's more likely this student received just one type of assistantship, and zero for the other types (better than stochastically imputing). Note: In NPSAS:16 this step amounted to 19 replacements for teaching assistantships; 82 for research; and 85 for other.

8. For those remaining that weren’t part of step 3, 4, or 6:
   a. Use positive values from student records for each type. Then use positive values from the student interview by type.
   b. If the value is still missing and student records says zero, use it.
   c. If the value is still missing and the interview says zero, use it.

9. Set a minimum imputation flag: If the student records value for a particular assistantship type was missing, but the institution reported they had this type of assistantship, but did not give us an amount, set the minimum imputation flag to 1, ONLY IF the value of the derived variable is still currently missing.

SAS Code

```
* Source variable grteaamt_sr="Graduate teaching assistantship amount (student records)";
* Source variable grteaamt_si="Graduate teaching assistantship amount (student interview)";
* Source variable grresamt_sr="Graduate research assistantship amount (student records)";
* Source variable grresamt_si="Graduate research assistantship amount (student interview)";
* Source variable grgrdamt_sr="Other graduate assistantship amount (student records)";
* Source variable grgrdamt_si="Other graduate assistantship amount (student interview)";
**************************************************************************;
if stypelst = 1 then
      do;
        grgrdamt = .y;
      end;
    else if stypelst > 1 then
      do;

* Step 3: Use student records for all three assistantship types when:
* 1.    None of the three student records assistantship type variables
*       are non-missing (grteaamt_sr, grresamt_sr, and grgrdamt_sr) AND
* 2.    All three of the student interview versions of the variable
*       are either zero or missing (grteaamt_si, grresamt_si, and grgrdamt_si);
      if missing(grteaamt_sr) = 0 & missing(grresamt_sr) = 0 & missing(grgrdamt_sr) = 0 &
         (missing(grteaamt_si) = 1 | grteaamt_si = 0) &
         (missing(grresamt_si) = 1 | grresamt_si = 0) &
         (missing(grgrdamt_si) = 1 | grgrdamt_si = 0) then
       do;
         grgrdamt = grgrdamt_sr;
         step3flag = 1;
       end;
* Step 4: Use the student interview for all three assistantship types when:
* 1.    All three student record assistantships are missing AND
* 2.    At least one of the three student interview types are non-missing;
      else if missing(grteaamt_sr) = 1 & missing(grresamt_sr) = 1 & missing(grgrdamt_sr) = 1 &
           (missing(grteaamt_si) = 0 | missing(grresamt_si) = 0 | missing(grgrdamt_si) = 0) then
       do;
         grgrdamt = grgrdamt_si;
         step4flag = 1;
       end;
* Step 5: Set each assistantship type to zero if:
* Both the student records and student interview version of the variable is zero.;
      if grgrdamt = . & grgrdamt_sr = 0 & grgrdamt_si = 0 then do; grgrdamt = 0; end;
* Step 6: Use student records for all three assistantship types if:
* 1.    The case wasn't part of either step 3 or 4, where wholesale
*       student records or interview assistantships were taken, AND
* 2.    At least one assistantship value from student records for an
*       assistantship type is greater than or equal to $5,000.;
      if step3flag ~= 1 & step4flag ~= 1 & (grteaamt_sr >= 5000 | grresamt_sr >= 5000 |
grgrdamt_sr >= 5000) then
       do;
         grgrdamt = grgrdamt_sr; if missing(grgrdamt_sr) = 0 then zgrgrdam = 2;
         step6flag = 1;

 * Step 7: Replace missing values for the cases from step 6 with zero.;
         if grgrdamt = . then do; grgrdamt = 0; end;
       end;
 * Step 8: For those remaining that weren't part of step 3, 4, or 6:
 * 1.    Use positive values from student records for each type. Then
 *       use positive values from the student interview by type.
 * 2.    If the value is still missing and student records says zero,
 *       use it.
 * 3.    If the value is still missing and the interview says zero,
 *       use it.;
      if step3flag ~= 1 & step4flag ~= 1 & step6flag ~= 1 then
       do;
         array grder grteaamt    grresamt    grgrdamt;
         array grsr  grteaamt_sr grresamt_sr grgrdamt_sr;
         array grsi  grteaamt_si grresamt_si grgrdamt_si;
         do over grder;
           if grsr > 0      then do; grder = grsr; end;
           else if grsi > 0 then do; grder = grsi; end;
```

	else if grsr = 0 then do; grder = grsr; end;
	else if grsi = 0 then do; grder = grsi; end;

<table>
<tr><td><b>Variable Name</b></td><td><b>GRINFEL</b></td></tr>
<tr><td><b>Variable Label</b></td><td>Graduate fellowship amount</td></tr>
<tr><td><b>Description</b></td><td>Total amount of graduate fellowships or grants from institutional sources received during the 2015–16 academic year.</td></tr>
<tr><td><b>Assigned Values</b></td><td><table><tr><th>Value</th><th>Value label</th></tr><tr><td>0</td><td>{Zero}</td></tr><tr><td>c</td><td>{Continuous}</td></tr></table></td></tr>
<tr><td><b>Data Source(s)</b></td><td>NPSAS:16 Student Records, NPSAS:16 Interview</td></tr>
<tr><td><b>Component Derived Variables</b></td><td>Not applicable</td></tr>
<tr><td><b>Applies to</b></td><td>All graduate study members</td></tr>
<tr><td><b>Availability</b></td><td>NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16</td></tr>
<tr><td><b>Change History</b></td><td>Unchanged from NPSAS:96 through NPSAS:16.</td></tr>
<tr><td><b>N:16 Usage Statistics</b></td><td>Number of uses: 0 (UG); 67 (GR)</td></tr>
<tr><td><b>Programming Narrative</b></td><td>Since this is a graduate level variable, set to missing for undergraduates.
Raw student interview data were first passed through a double-counting program to identify self-reported amounts that appeared to be identical (or nearly). N16CFELLAMT was one of many variables that were set to missing if there was evidence it was double-counted.
Raw student records data were also passed through a similar double-counting program that eliminated aid amounts that appeared to be double counted, largely institutional aid amounts and graduate aid amounts.
1. Sum together all institutional grants.
2. Graduate aid section:
a. If CFAGRTYP = 1 and CFAGRAMT>0
b. Else if CFAGRTYP = 1 and CFAGRAMT = missing, then set minimum imputation flag to 1, to impute positive value.
3. Institutional aid section
a. If CFAITYP in (1, 2, 3, 4, 8, 13, 17) and CFAIAMT>0
b. Else if CFAITYP in (1, 2, 3, 4, 8, 13, 17) and CFAIAMT = missing, then set minimum imputation flag to 1, to impute positive value.
4. Other aid section
a. If CFAOTHSRC = 1 and CFAOTHTYP in (1, 2, 3, 4, 8, 13, 17) and CFAOTHAMT>0
b. Else if CFAOTHSRC = 1 and CFAOTHTYP in (1, 2, 3, 4, 8, 13, 17) and CFAOTHAMT = missing, then set minimum imputation flag to 1, to impute positive value.
5. If the sums from above are not greater than zero, and the student said they received fellowships in 2015–16 in the interview (N16CAIDFELL2 = 1), then set GRINFEL to N16CFELLAMT, the self-reported amount from the interview.
6. Else if the sum of all the student records items are zero, then set GRINFEL to zero.
7. Else if the student did not say they received a fellowship in the interview, set GRINFEL to zero (N16CAIDFELL1 = 0 or N16CAIDFELL2 = 0).</td></tr>
<tr><td><b>SAS Code</b></td><td>* Source variable cfagrtyp01-cfagrtyp03="Graduate aid program type";
* Source variable cfagramt01-cfagramt03="Graduate aid program amount"; *** 1 = Fellowship/scholarship ;</td></tr>
</table>

```
* Source variable cfaityp01-cfaityp03="Institution aid program type";
* Source variable cfaiamt01-cfaiamt03="Institution aid program amount";
* Source variables cfa1othsrc-cfa3othsrc="Other aid source";
* Source variables cfa1othtyp-cfa3othtyp="Other aid type";
* Source variables cfa1othamt-cfa3othamt="Other aid amount";
* Source variable n16cfellamt="Aid amount: fellowship";
* Source variable n16caidfell2="Graduate aid 2015–16: fellowship";
* Source variable cfacvans="Received any financial aid";
*************************************************************************;
  array gtyp    (3) cfagrtyp01 cfagrtyp02 cfagrtyp03;
  array gtemp   (3) felg1     felg2     felg3;
  array gamt    (3) cfagramt01 cfagramt02 cfagramt03;
  array ityp    (3) cfaityp01  cfaityp02  cfaityp03;
  array itemp   (3) feli1     feli2     feli3;
  array iamt    (3) cfaiamt01  cfaiamt02  cfaiamt03;
  array otemp   (3) felo1     felo2     felo3;
  array othsrc  (3) cfa1othsrc cfa2othsrc cfa3othsrc;
  array othtype (3) cfa1othtyp cfa2othtyp cfa3othtyp;
  array oamt    (3) cfa1othamt cfa2othamt cfa3othamt;

  do i = 1 to 3;
    gtemp(i) = 0;
    if cfagraid ~= 0 then
     do;
      if gtyp(i) = 1 then
       do;
        gtemp(i) = gamt(i);
        if missing(gtemp(i)) = 1 then gtemp(i) = .m;
       end;
      else if cfagraid = .x then gtemp(i) = .x;
     end;

    itemp(i) = 0;
    if cfainstaid ~= 0 then
     do;
      if ityp(i) in (1, 2, 3, 4, 8, 13, 17) then
       do;
        itemp(i) = iamt(i);
        if missing(itemp(i)) = 1 then itemp(i) = .m;
       end;
      else if cfainstaid = .x then itemp(i) = .x;
     end;

    otemp(i) = 0;
    if cfaothaid ~= 0 then
     do;
      if othsrc(i) = 1 & othtype(i) in (1, 2, 3, 4, 8, 13, 17) then
       do;
        otemp(i) = oamt(i);
        if missing(otemp(i)) = 1 then otemp(i) = .m;
       end;
```

```
        else if cfaothaid = .x then otemp(i) = .x;
      end;
end;

array forGrinfel(9) felg1 felg2 felg3 feli1 feli2 feli3 felo1 felo2 felo3;
do i = 1 to 9;
    if forGrinfel(i) > 0 then
      do j = 1 to 9;
        if forGrinfel(j) = .x then forGrinfel(j) = 0;
      end;
end;

* Applies to: All graduate study members;
if stypelst = 1 then do; grinfel = .y; grinfel_minimp = .y; end;
else if stypelst > 1 then
  do;
    if felg1 + felg2 + felg3 + feli1 + feli2 + feli3 + felo1 + felo2 + felo3 > 0 then
      do;
        grinfel = felg1 + felg2 + felg3 + feli1 + feli2 + feli3 + felo1 + felo2 + felo3;
        grinfel_minimp = sum(of felg1 felg2 felg3 feli1 feli2 feli3 felo1 felo2 felo3);
       end;
    else if n16cfellamt > 0 & n16caidfell2 = 1 then
      do;
        grinfel = n16cfellamt;
        grinfel_minimp = n16cfellamt;
       end;
    else if sum(of felg1 felg2 felg3 feli1 feli2 feli3 felo1 felo2 felo3) > 0 |
            felg1 = .m | felg2 = .m | felg3 = .m |
            feli1 = .m | feli2 = .m | feli3 = .m |
            felo1 = .m | felo2 = .m | felo3 = .m then
      do;
        grinfel = felg1 + felg2 + felg3 + feli1 + feli2 + feli3 + felo1 + felo2 + felo3;
        grinfel_minimp = sum(of felg1 felg2 felg3 feli1 feli2 feli3 felo1 felo2 felo3);
        do i = 1 to 9;
            if forGrinfel(i) = .m then Grinfel_minimp = Grinfel_minimp + 1;
        end;
        if missing(grinfel) = 1 then do; grinfel = .x; end;
      end;
    else if felg1 + felg2 + felg3 + feli1 + feli2 + feli3 + felo1 + felo2 + felo3 = 0 then
      do;
        grinfel = 0;
        grinfel_minimp = 0;
      end;
    else if n16cfellamt = 0 | n16caidfell2 = 0 then
      do;
        grinfel = 0;
        grinfel_minimp = 0;
      end;
    else
      do;
        grinfel = felg1 + felg2 + felg3 + feli1 + feli2 + feli3 + felo1 + felo2 + felo3;
      grinfel_minimp = sum(of felg1 felg2 felg3 feli1 feli2 feli3 felo1 felo2 felo3);
      if grinfel_minimp = . then grinfel_minimp = 0;
      if missing(grinfel) = 1 then do; grinfel = .x; end;
     end;
   end;
  if grinfel = .x & grinfel_minimp = 0 & n16caidfell2 = 1 & n16cfellamt ~= .n then
grinfel_minimp = 1;
```

<table>
<tr><td><b>Variable Name</b></td><td><b>GRRESAMT</b></td></tr>
<tr><td><b>Variable Label</b></td><td>Graduate research assistantship amount</td></tr>
<tr><td><b>Description</b></td><td>Total amount of graduate research assistantships received during the 2015–16 academic year.</td></tr>
<tr><td><b>Assigned Values</b></td><td>
<table>
<tr><th>Value</th><th>Value label</th></tr>
<tr><td><b>0</b></td><td>{Zero}</td></tr>
<tr><td><b>c</b></td><td>{Continuous}</td></tr>
</table>
</td></tr>
<tr><td><b>Data Source(s)</b></td><td>NPSAS:16 Student Records, NPSAS:16 Interview</td></tr>
<tr><td><b>Component Derived Variables</b></td><td>Not applicable</td></tr>
<tr><td><b>Applies to</b></td><td>All graduate study members</td></tr>
<tr><td><b>Availability</b></td><td>NPSAS:04, NPSAS:08; NPSAS:12; NPSAS:16</td></tr>
<tr><td><b>Change History</b></td><td>Unchanged from NPSAS:04 through NPSAS:16.</td></tr>
<tr><td><b>N:16 Usage Statistics</b></td><td>Number of uses: 0 (UG); 87 (GR)</td></tr>
<tr><td><b>Programming Narrative</b></td><td>Since this is a graduate level variable, set to missing for undergraduates.
Raw student interview data were first passed through a double-counting program to identify self-reported amounts that appeared to be identical (or nearly). All assistantship types were part of this and were sometimes set to missing if there was evidence of double-counting.
Raw student records data were also passed through a similar double-counting program that eliminated some aid amounts that appeared to be double counted, largely institutional aid amounts and graduate aid amounts.

Create before imputation:
1. Derive individual assistantship amounts from student records variables only:
a. Grteaamt_sr = sum of (graduate aid teaching assistantships, institutional aid teaching assistantships, and other aid institutional teaching assistantships)
b. Grresamt_sr = sum of (graduate aid research assistantships, institutional aid research assistantships, and other aid institutional research assistantships)
c. Grgrdamt_sr = sum of (graduate aid other assistantships, institutional aid other assistantships, and other aid institutional other assistantships)
d. Flag the cases where the amount is missing, but the aid type was specified as the respective assistantship type (using the .m system missing designation).

2. Derive individual assistantship amounts from the student interview only:
a. Grteaamt_si = n16cgrtaamt, where the amount is greater than or equal to zero (and stypelst>1).
b. Grresamt_si = n16cgrraamt, where the amount is greater than or equal to zero (and stypelst>1).
c. Grgrdamt_si = n16cgrothamt, where the amount is greater than or equal to zero (and stypelst>1).</td></tr>
</table>

3. Use student records for all three assistantship types when:
    a. None of the three student records assistantship type variables are non-missing (grteaamt_sr, grresamt_sr, and grgrdamt_sr), AND
    b. All three of the student interview versions of the variable are either zero or missing (grteaamt_si, grresamt_si, and grgrdamt_si)

4. Use the student interview for all three assistantship types when
    a. All three student record assistantships are missing, AND
    b. At least one of the three student interview types are non-missing.

5. Set each assistantship type to zero if both the student records and student interview version of the variable is zero.

6. Use student records for all three assistantship types if:
    a. The case wasn’t part of either step 3 or 4, where wholesale student records or interview assistantships were taken, AND
    b. At least one assistantship value from student records for an assistantship type is greater than or equal to $5,000. This arbitrary value takes care of a handful of cases where an amount in student records is very small, but other types of assistantships were reported in the interview, for much larger amounts. These will be used in the coming steps.

7. Replace missing values for the cases from step 6 with zero. This is a logical adjustment and in most cases, the interview corroborates the zero value. Only a few would need to be stochastically imputed if we left it, but it's more likely this student received just one type of assistantship, and zero for the other types (better than stochastically imputing). Note: In NPSAS:16 this step amounted to 19 replacements for teaching assistantships; 82 for research; and 85 for other.

8. For those remaining that weren’t part of step 3, 4, or 6:
    a. Use positive values from student records for each type. Then use positive values from the student interview by type.
    b. If the value is still missing and student records says zero, use it.
    c. If the value is still missing and the interview says zero, use it.

9. Set a minimum imputation flag: If the student records value for a particular assistantship type was missing, but the institution reported they had this type of assistantship, but did not give us an amount, set the minimum imputation flag to 1, ONLY IF the value of the derived variable is still currently missing.

SAS Code

```
* Source variable grteaamt_sr="Graduate teaching assistantship amount (student records)";
* Source variable grteaamt_si="Graduate teaching assistantship amount (student interview)";
* Source variable grresamt_sr="Graduate research assistantship amount (student records)";
* Source variable grresamt_si="Graduate research assistantship amount (student interview)";
* Source variable grgrdamt_sr="Other graduate assistantship amount (student records)";
* Source variable grgrdamt_si="Other graduate assistantship amount (student interview)";
*************************************************************************;
if stypelst = 1 then
    do;
     grresamt = .y;
    end;
  else if stypelst > 1 then
    do;
```

```
* Step 3: Use student records for all three assistantship types when:
* 1.   None of the three student records assistantship type variables are non-missing
(grteaamt_sr, grresamt_sr, and grgrdamt_sr) AND
* 2.   All three of the student interview versions of the variable are either zero or missing
(grteaamt_si, grresamt_si, and grgrdamt_si);
      if missing(grteaamt_sr) = 0 & missing(grresamt_sr) = 0 & missing(grgrdamt_sr) = 0 &
        (missing(grteaamt_si) = 1 | grteaamt_si = 0) &
        (missing(grresamt_si) = 1 | grresamt_si = 0) &
        (missing(grgrdamt_si) = 1 | grgrdamt_si = 0) then
       do;
        grresamt = grresamt_sr;
        step3flag = 1;
       end;
* Step 4: Use the student interview for all three assistantship types when:
* 1.   All three student record assistantships are missing AND
* 2.   At least one of the three student interview types are non-missing;
      else if missing(grteaamt_sr) = 1 & missing(grresamt_sr) = 1 & missing(grgrdamt_sr) = 1 &
(missing(grteaamt_si) = 0 | missing(grresamt_si) = 0 | missing(grgrdamt_si) = 0) then
       do;
         grresamt = grresamt_si;
         step4flag = 1;
       end;
* Step 5: Set each assistantship type to zero if:
* Both the student records and student interview version of the variable is zero.;
      if grresamt = . & grresamt_sr = 0 & grresamt_si = 0 then do; grresamt = 0; end;
* Step 6: Use student records for all three assistantship types if:
* 1.   The case wasn't part of either step 3 or 4, where wholesale student records or
* interview assistantships were taken, AND
* 2.   At least one assistantship value from student records for an assistantship type is
* greater than or equal to $5,000.;
      if step3flag ~= 1 & step4flag ~= 1 & (grteaamt_sr >= 5000 | grresamt_sr >= 5000 |
grgrdamt_sr >= 5000) then
       do;
         grresamt = grresamt_sr;
         step6flag = 1;
* Step 7: Replace missing values for the cases from step 6 with zero.;
         if grresamt = . then do; grresamt = 0;
       end;
* Step 8: For those remaining that weren't part of step 3, 4, or 6:
* 1.   Use positive values from student records for each type. Then
*      use positive values from the student interview by type.
* 2.   If the value is still missing and student records says zero,
*      use it.
* 3.   If the value is still missing and the interview says zero,
*      use it.;
      if step3flag ~= 1 & step4flag ~= 1 & step6flag ~= 1 then
       do;
         array grder grteaamt   grresamt   grgrdamt;
         array grsr  grteaamt_sr grresamt_sr grgrdamt_sr;
         array grsi  grteaamt_si grresamt_si grgrdamt_si;
         do over grder;
           if grsr > 0     then do; grder = grsr; end;
```

	else if grsi > 0 then do; grder = grsi; end;
	else if grsr = 0 then do; grder = grsr; end;
	else if grsi = 0 then do; grder = grsi; end;

<table>
<tr><td><b>Variable Name</b></td><td><b>GRTEAAMT</b></td></tr>
<tr><td><b>Variable Label</b></td><td>Graduate teaching assistantship amount</td></tr>
<tr><td><b>Description</b></td><td>Total amount of graduate teaching assistantships received during the 2015–16 academic year.</td></tr>
<tr><td><b>Assigned Values</b></td><td><table><tr><th>Value</th><th>Value label</th></tr><tr><td><b>0</b></td><td>{Zero}</td></tr><tr><td><b>c</b></td><td>{Continuous}</td></tr></table></td></tr>
<tr><td><b>Data Source(s)</b></td><td>NPSAS:16 Student Records, NPSAS:16 Interview</td></tr>
<tr><td><b>Component Derived Variables</b></td><td>Not applicable</td></tr>
<tr><td><b>Applies to</b></td><td>All graduate study members</td></tr>
<tr><td><b>Availability</b></td><td>NPSAS:04, NPSAS:08; NPSAS:12; NPSAS:16</td></tr>
<tr><td><b>Change History</b></td><td>Unchanged from NPSAS:04 through NPSAS:16.</td></tr>
<tr><td><b>N:16 Usage Statistics</b></td><td>Number of uses: 0 (UG); 78 (GR)</td></tr>
<tr><td><b>Programming Narrative</b></td><td>Since this is a graduate level variable, set to missing for undergraduates.
Raw student interview data were first passed through a double-counting program to identify self-reported amounts that appeared to be identical (or nearly). All assistantship types were part of this and were sometimes set to missing if there was evidence of double-counting.
Raw student records data were also passed through a similar double-counting program that eliminated some aid amounts that appeared to be double counted, largely institutional aid amounts and graduate aid amounts.

Create before imputation:
1. Derive individual assistantship amounts from student records variables only:
a. Grteaamt_sr = sum of (graduate aid teaching assistantships, institutional aid teaching assistantships, and other aid institutional teaching assistantships)
b. Grresamt_sr = sum of (graduate aid research assistantships, institutional aid research assistantships, and other aid institutional research assistantships)
c. Grgrdamt_sr = sum of (graduate aid other assistantships, institutional aid other assistantships, and other aid institutional other assistantships)
d. Flag the cases where the amount is missing, but the aid type was specified as the respective assistantship type (using the .m system missing designation).
2. Derive individual assistantship amounts from the student interview only:
a. Grteaamt_si = n16cgrtaamt, where the amount is greater than or equal to zero (and stypelst>1).
b. Grresamt_si = n16cgrraamt, where the amount is greater than or equal to zero (and stypelst>1).
c. Grgrdamt_si = n16cgrothamt, where the amount is greater than or equal to zero (and stypelst>1).
3. Use student records for all three assistantship types when
a. None of the three student records assistantship type variables are non-missing (grteaamt_sr, grresamt_sr, and grgrdamt_sr) AND
b. All three of the student interview versions of the variable are either zero or missing (grteaamt_si, grresamt_si, and grgrdamt_si)</td></tr>
</table>

4. Use the student interview for all three assistantship types when
   a. All three student record assistantships are missing AND
   b. At least one of the three student interview types are non-missing.

5. Set each assistantship type to zero if both the student records and student interview versions of the variable are zero.

6. Use student records for all three assistantship types if
   a. The case wasn't part of either step 3 or 4, where wholesale student records or interview assistantships were taken, AND
   b. At least one assistantship value from student records for an assistantship type is greater than or equal to $5,000. This arbitrary value takes care of a handful of cases where an amount in student records is very small, but other types of assistantships were reported in the interview, for much larger amounts. These will be used in the coming steps.

7. Replace missing values for the cases from step 6 with zero. This is a logical adjustment, and in most cases, the interview corroborates the zero value. Only a few would need to be stochastically imputed if we left it, but it's more likely this student received just one type of assistantship, and zero for the other types (better than stochastically imputing).
   a. Note: In NPSAS:16 this step amounted to 19 replacements for teaching assistantships; 82 for research; and 85 for other.

8. For those remaining that weren't part of step 3, 4, or 6:
   a. Use positive values from student records for each type. Then use positive values from the student interview by type.
   b. If the value is still missing and student records says zero, use it.
   c. If the value is still missing and the interview says zero, use it.

9. Set a minimum imputation flag: If the student records value for a particular assistantship type was missing, but the institution reported they had this type of assistantship, but did not give us an amount, set the minimum imputation flag to 1, ONLY IF the value of the derived variable is still currently missing.

SAS Code

```
*******************************************************************************;
* Source variable grteaamt_sr="Graduate teaching assistantship amount (student records)";
* Source variable grteaamt_si="Graduate teaching assistantship amount (student interview)";
* Source variable grresamt_sr="Graduate research assistantship amount (student records)";
* Source variable grresamt_si="Graduate research assistantship amount (student interview)";
* Source variable grgrdamt_sr="Other graduate assistantship amount (student records)";
* Source variable grgrdamt_si="Other graduate assistantship amount (student interview)";
  *****************************************************************************;
if stypelst = 1 then
     do;
       grteaamt = .y;
     end;
   else if stypelst > 1 then
     do;
* Step 3: Use student records for all three assistantship types when:
* 1.    None of the three student records assistantship type variables are non-missing
(grteaamt_sr, grresamt_sr, and grgrdamt_sr) AND
* 2.    All three of the student interview versions of the variable are either zero or missing
(grteaamt_si, grresamt_si, and grgrdamt_si);
if missing(grteaamt_sr) = 0 & missing(grresamt_sr) = 0 & missing(grgrdamt_sr) = 0 &
          (missing(grteaamt_si) = 1 | grteaamt_si = 0) &
          (missing(grresamt_si) = 1 | grresamt_si = 0) &
          (missing(grgrdamt_si) = 1 | grgrdamt_si = 0) then
          do;
            grteaamt = grteaamt_sr;
          end;
* Step 4: Use the student interview for all three assistantship types when:
* 1.   All three student record assistantships are missing AND
* 2.   At least one of the three student interview types are non-missing;
        else if missing(grteaamt_sr) = 1 & missing(grresamt_sr) = 1 & missing(grgrdamt_sr) = 1 &
(missing(grteaamt_si) = 0 | missing(grresamt_si) = 0 | missing(grgrdamt_si) = 0) then
          do;
            grteaamt = grteaamt_si;
          end;
* Step 5: Set each assistantship type to zero if:
* Both the student records and student interview version of the variable is zero.;
        if grteaamt = . & grteaamt_sr = 0 & grteaamt_si = 0 then do; grteaamt = 0; end;
* Step 6: Use student records for all three assistantship types if:
* 1.   The case wasn't part of either step 3 or 4, where wholesale
*      student records or interview assistantships were taken, AND
* 2.   At least one assistantship value from student records for an
*      assistantship type is greater than or equal to $5,000.;
        if step3flag ~= 1 & step4flag ~= 1 & (grteaamt_sr >= 5000 | grresamt_sr >= 5000 |
grgrdamt_sr >= 5000) then
          do;
            grteaamt = grteaamt_sr;
* Step 7: Replace missing values for the cases from step 6 with zero.;
            if grteaamt = . then do; grteaamt = 0; end;
* Step 8: For those remaining that weren't part of step 3, 4, or 6:
* 1.   Use positive values from student records for each type. Then
*      use positive values from the student interview by type.
* 2.   If the value is still missing and student records says zero,
*      use it.
* 3.   If the value is still missing and the interview says zero,
*      use it.;
if step3flag ~= 1 & step4flag ~= 1 & step6flag ~= 1 then
          do;
            array grder grteaamt    grresamt    grgrdamt;
            array grsr  grteaamt_sr grresamt_sr grgrdamt_sr;
            array grsi  grteaamt_si grresamt_si grgrdamt_si;
            do over grder;
              if grsr > 0  then do; grder = grsr; end;
              else if grsi > 0 then do; grder = grsi; end;
              else if grsr = 0 then do; grder = grsr; end;
              else if grsi = 0 then do; grder = grsi; end;
            end;
          end;
         end;
```

Variable Name	**INATHAMT**
Variable Label	Athletic scholarships amount
Description	Total amount of institutional athletic scholarships received during the 2015–16 academic year.

Assigned Values

Value	Value label
0	{Zero}
c	{Continuous}

Data Source(s)	NPSAS:16 Student Records
Component Derived Variables	Not applicable
Applies to	All undergraduate study members
Availability	NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16
Change History	Unchanged from NPSAS:96 through NPSAS:16.
N:16 Usage Statistics	Number of uses: 25 (UG); 0 (GR)
Programming Narrative	Sum athletic scholarships from student records (type=8). 1. Institutional aid section: a. if CFAITYP = 8 and CFAIAMT >0 b. Else if CFAITYP = 8 and CFAIAMT = missing, then set minimum imputation flag to 1 (to impute positive value). 2. Other aid section: a. If CFAOTHSRC = 1 and CFAOTHTYP = 8 and CFAOTHAMT>0 b. Else if CFAOTHSRC = 1 and CFAOTHTYP = 8 and CFAOTHAMT = missing, then set minimum imputation flag to 1 (to impute positive value). 3. Cap edit: If amount is greater than $66,500, then set the amount to $66,500.

SAS Code

```
* Source variable cfaityp01-cfaityp03="Institution aid program type";
* Source variable cfaiamt01-cfaiamt03="Institution aid program amount";
* Source variables cfa1othsrc-cfa3othsrc="Other aid source";
* Source variables cfa1othtyp-cfa3othtyp="Other aid type";
* Source variables cfa1othamt-cfa3othamt="Other aid amount";
* Source variable cfacvans="Received any financial aid";
  ***************************************************************;
  array ityp    (3) cfaityp01  cfaityp02  cfaityp03;
  array itemp   (3) inathi1    inathi2    inathi3;
  array iamt    (3) cfaiamt01  cfaiamt02  cfaiamt03;
  array otemp   (3) inatho1    inatho2    inatho3;
  array othsrc  (3) cfa1othsrc cfa2othsrc cfa3othsrc;
  array othtype (3) cfa1othtyp cfa2othtyp cfa3othtyp;
  array oamt    (3) cfa1othamt cfa2othamt cfa3othamt;

  do i = 1 to 3;
    itemp(i) = 0;
    if cfainstaid ~= 0 then
     do;
      if ityp(i) = 8 then
       do;
        itemp(i) = iamt(i);
        if missing(itemp(i)) = 1 then itemp(i) = .m;
       end;
      else if cfainstaid = .x then itemp(i) = .x;
     end;
    otemp(i) = 0;
    if cfaothaid ~= 0 then
     do;
      if othsrc(i) = 1 & othtype(i) = 8 then
       do;
        otemp(i) = oamt(i);
        if missing(otemp(i)) = 1 then otemp(i) = .m;
       end;
      else if cfaothaid = .x then otemp(i) = .x;
     end;
  end;

  array forInathamt(6) inathi1 inathi2 inathi3 inatho1 inatho2 inatho3;
  do i = 1 to 6;
    if forInathamt(i) > 0 then
     do j = 1 to 6;
      if forInathamt(j) = .x then forInathamt(j) = 0;
     end;
  end;

  * Applies to: All undergraduate study members;
  if stypelst > 1 then do; inathamt = .y; inathamt_minimp = .y; end;
  else if stypelst = 1 then
   do;
    inathamt = inathi1 + inathi2 + inathi3 + inatho1 + inatho2 + inatho3;
    if missing(inathamt) = 1 then
     do;
      inathamt = .x;
     end;
```

Variable Name	**INCOME**
Variable Label	Total income by dependency (categorical)
Description	Total income (categorical) in 2014 for independent students or parents of dependent students.

<table>
<tr><td><b>Assigned Values</b></td><td>
<table>
<tr><th>Value</th><th>Value label</th></tr>
<tr><td>1</td><td>Dependent: Less than $10,000</td></tr>
<tr><td>2</td><td>Dependent: $10,000-$19,999</td></tr>
<tr><td>3</td><td>Dependent: $20,000-$29,999</td></tr>
<tr><td>4</td><td>Dependent: $30,000-$39,999</td></tr>
<tr><td>5</td><td>Dependent: $40,000-$49,999</td></tr>
<tr><td>6</td><td>Dependent: $50,000-$59,999</td></tr>
<tr><td>7</td><td>Dependent: $60,000-$69,999</td></tr>
<tr><td>8</td><td>Dependent: $70,000-$79,999</td></tr>
<tr><td>9</td><td>Dependent: $80,000-$99,999</td></tr>
<tr><td>10</td><td>Dependent: $100,000-$119,999</td></tr>
<tr><td>11</td><td>Dependent: $120,000-$149,999</td></tr>
<tr><td>12</td><td>Dependent: $150,000 or more</td></tr>
<tr><td>13</td><td>Independent: Less than $5,000</td></tr>
<tr><td>14</td><td>Independent: $5,000-$9,999</td></tr>
<tr><td>15</td><td>Independent: $10,000-$19,999</td></tr>
<tr><td>16</td><td>Independent: $20,000-$29,999</td></tr>
<tr><td>17</td><td>Independent: $30,000-$49,999</td></tr>
<tr><td>18</td><td>Independent: $50,000 or more</td></tr>
</table>
</td></tr>
<tr><td><b>Data Source(s)</b></td><td>FAFSA:16, NPSAS:16 Interview</td></tr>
<tr><td><b>Component Derived Variables</b></td><td>DEPEND, CINCOME, STYPELST</td></tr>
<tr><td><b>Applies to</b></td><td>All undergraduate study members</td></tr>
<tr><td><b>Availability</b></td><td>NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16</td></tr>
<tr><td><b>Change History</b></td><td>Unchanged from NPSAS:96 through NPSAS:16</td></tr>
<tr><td><b>N:16 Usage Statistics</b></td><td>Number of uses: 818 (UG); 0 (GR)</td></tr>
<tr><td><b>Programming Narrative</b></td><td>1. If dependent (DEPEND=1), create categorical variable according to the continuous income variable (CINCOME).
2. If independent (DEPEND=2) add onto same categorical variable according to continuous income variable (CINCOME).
3. If graduate student, set INCOME to skip (-3).</td></tr>
</table>

SAS Code

```
IF DEPEND=1 THEN DO;
  IF CINCOME<10000 THEN INCOME=1;
  ELSE IF CINCOME<20000 THEN INCOME=2;
  ELSE IF CINCOME<30000 THEN INCOME=3;
  ELSE IF CINCOME<40000 THEN INCOME=4;
  ELSE IF CINCOME<50000 THEN INCOME=5;
  ELSE IF CINCOME<60000 THEN INCOME=6;
  ELSE IF CINCOME<70000 THEN INCOME=7;
  ELSE IF CINCOME<80000 THEN INCOME=8;
  ELSE IF CINCOME<100000 THEN INCOME=9;
  ELSE IF CINCOME<120000 THEN INCOME=10;
  ELSE IF CINCOME<150000 THEN INCOME=11;
  ELSE INCOME=12;
END;
ELSE IF DEPEND=2 THEN DO;
  IF CINCOME<5000 THEN INCOME=13;
  ELSE IF CINCOME<10000 THEN INCOME=14;
  ELSE IF CINCOME<20000 THEN INCOME=15;
  ELSE IF CINCOME<30000 THEN INCOME=16;
  ELSE IF CINCOME<50000 THEN INCOME=17;
  ELSE INCOME=18;
END;

/* skip grad students */
IF STYPELST>1 THEN INCOME=-3;
```

<table>
<tr><td><b>Variable Name</b></td><td><b>INCOMEG</b></td></tr>
<tr><td><b>Variable Label</b></td><td>Total income (categorical)</td></tr>
<tr><td><b>Description</b></td><td>Total income in 2016 (categorical).</td></tr>
<tr><td><b>Assigned Values</b></td><td>
<table>
<tr><th>Value</th><th>Value label</th></tr>
<tr><td><b>1</b></td><td>Less than $5,000</td></tr>
<tr><td><b>2</b></td><td>$5,000 - $9,999</td></tr>
<tr><td><b>3</b></td><td>$10,000 - $19,999</td></tr>
<tr><td><b>4</b></td><td>$20,000 - $29,000</td></tr>
<tr><td><b>5</b></td><td>$30,000 - $49,999</td></tr>
<tr><td><b>6</b></td><td>$50,000 - $99,999</td></tr>
<tr><td><b>7</b></td><td>$100,000 or more</td></tr>
</table>
</td></tr>
<tr><td><b>Data Source(s)</b></td><td>FAFSA:16, NPSAS:16 Interview</td></tr>
<tr><td><b>Component Derived Variables</b></td><td>STYPELST, CINCOME</td></tr>
<tr><td><b>Applies to</b></td><td>All graduate study members</td></tr>
<tr><td><b>Availability</b></td><td>NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16</td></tr>
<tr><td><b>Change History</b></td><td>Unchanged from NPSAS:96 through NPSAS:16</td></tr>
<tr><td><b>N:16 Usage Statistics</b></td><td>Number of uses: 0 (UG); 41 (GR)</td></tr>
</table>

Programming Narrative	1. If graduate student, create categorical variable according to the continuous income variable (CINCOME). 2. If undergraduate (STYPELST = 1), set to skip (-3).

SAS Code	/*--- INCOMEG: Total income for graduate students (categorical) (GR) ---*/
	/* variable used: STYPELST INCOME */
	/* values:
	1=Independent: Less than $5,000
	2=Independent: $5,000-$9,999
	3=Independent: $10,000-$19,999
	4=Independent: $20,000-$29,999
	5=Independent: $30,000-$49,999
	6=Independent: $50,000-$99,999
	7=Independent: $100,000 or more
	*/
	
	if CINCOME<5000 then INCOMEG=1;
	else if CINCOME<10000 then INCOMEG=2;
	else if CINCOME<20000 then INCOMEG=3;
	else if CINCOME<30000 then INCOMEG=4;
	else if CINCOME<50000 then INCOMEG=5;
	else if CINCOME<100000 then INCOMEG=6;
	else INCOMEG=7;
	
	/* skip ug students */
	if STYPELST=1 then INCOMEG=-3;

Variable Name	**INDEPINC**
Variable Label	Independent students: student and spouse’s income
Description	Independent student’s total income for 2014, including income of the spouse (SPSINC), if any.

Assigned Values	**Value** — **Value label**
	-3 — {Skipped}
	0 — {Zero}
	c — {Continuous}

Data Source(s)	FAFSA:16, NSLDS:16, NPSAS:16 Interview
Component Derived Variables	DEPEND, ISTUINC, SPSINC, DEPEND2, SMARITAL Note: ISTUINC and SPSINC are not included in this report. Information on these variables may be found in the NPSAS:16 undergraduate codebook, https://nces.ed.gov/datalab/powerstats/codebook.aspx?dataset=121&type=subject#ISTUINC and https://nces.ed.gov/datalab/powerstats/codebook.aspx?dataset=121&type=subject#SPSINC, and the NPSAS:16 graduate student codebook, https://nces.ed.gov/datalab/powerstats/codebook.aspx?dataset=122&type=subject#ISTUINC and https://nces.ed.gov/datalab/powerstats/codebook.aspx?dataset=122&type=subject#SPSINC.
Applies to	Independent study members (DEPEND=2) and all graduate study members
Availability	NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16

Change History	Unchanged from NPSAS:96 through NPSAS:16.
N:16 Usage Statistics	Number of uses: 634 (UG); 7 (GR)
Programming Narrative	1. If study member is a dependent (DEPEND=1) set to a legitimate skip (-3) 2. If study member is an independent (DEPEND=2) a. Use "Total income" (C16TI) from the FAFSA if it is greater than or equal to $0 b. If total income is less than zero, but not missing (-9) then set INDEPINC to zero. c. Else use adjusted gross income (C16TXAGI) from the FAFSA d. Else if the student is married (SMARITAL=2) and both the student's and spouse's income are greater than or equal to zero, add the student and spouse's income from work (C16INC + C16INCSP) e. Else if the student is not married, and the student's income from work is greater than or equal to zero, use it (C16INC). 3. Use historical FAFSA data a. See instructions on SAGI (student adjusted gross income). b. Replace INDEPINC = nfagi14_2015 if INDEPINC is missing, DEPEND2 = nfefcpt2_2015, nfmarr2015=smarr, STAXFILE is not equal to zero, and the agi14 from 2015 is greater than or equal to zero. c. Replace INDEPINC = nfagi14_2014 if INDEPINC is missing, DEPEND2 = nfefcpt2_2014, nfmarr2014=smarr, STAXFILE is not equal to zero, and the agi14 from 2014 is greater than or equal to zero. d. Replace INDEPINC = nfagi14_2013 if INDEPINC is missing, DEPEND2 = nfefcpt2_2013, nfmarr2013=smarr, STAXFILE is not equal to zero, and the agi14 from 2013 is greater than or equal to zero. 4. Use this variable to create spouse's income (SPSINC) and independent student's income (ISTUINC). a. After creating SPSINC and ISTUINC, there are a few cases with the addition of student interview responses, where we can now calculate INDEPINC using SPSINC and ISTUINC. Add edits for: i. replace INDEPINC = SPSINC + ISTUINC if both SPSINC>=0 and ISTUINC>=0, and SMARITAL=2 and INDEPINC< 0. ii. replace INDEPINC = ISTUINC if ISTUINC>=0 and SMARITAL ne 2 and INDEPINC < 0. 5. Stochastically impute spouse's income and independent student's income. 6. Create the final derived variable, which is equal to the sum of spouse's income (SPSINC) and independent student's income (ISTUINC)

SAS Code	/* independent student */
	IF DEPEND=2 THEN DO;
	/* total income (student & spouse) */
	IF C16TI>=0 THEN INDEPINC=C16TI;
	ELSE IF C16TXAGI>=0 THEN INDEPINC=C16TXAGI;
	ELSE IF SMARITAL=2 AND C16INC>=0 AND C16INCSP>=0 THEN INDEPINC=SUM(OF C16INC C16INCSP);
	ELSE IF SMARITAL NE 2 AND C16INC>=0 THEN INDEPINC=C16INC;
	
	/* Fill in missing using the historical FAFSA data */
	IF INDEPINC=. THEN DO;
	IF DEPEND2=nfefcpt2_2015 AND nfmarr_2015=SMARR AND nfagi14_2015>=0 THEN DO;
	INDEPINC=nfagi14_2015; END;
	ELSE IF DEPEND2=nfefcpt2_2014 AND nfmarr_2014=SMARR AND nfagi14_2014>=0 THEN DO;
	INDEPINC=nfagi14_2014; END;

```
  ELSE IF DEPEND2=nfefcpt2_2013 AND nfmarr_2013=SMARR AND nfagi14_2013>=0
THEN DO;
          INDEPINC=nfagi14_2013; END;
  END;
```

Variable Name	**INGRTAMT**
Variable Label	Institutional grants amount
Description	Total amount of all institutional grants received during the 2015–16 academic year.

Assigned Values	**Value** | **Value label**
	0 | {Zero}
	c | {Continuous}

Data Source(s)	NPSAS:16 Student Records, NPSAS:16 Interview
Component Derived Variables	INSTNEED, INSTNOND, GRINFEL, INSWAIV, INSMILAMT, INSVETAMT
Applies to	All study members
Availability	NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16
Change History	Unchanged from NPSAS:96 through NPSAS:16.
N:16 Usage Statistics	Number of uses: 1,369 (UG); 118 (GR)
Programming Narrative	1. Undergraduate: sum institutional need-based grants and institutional non-need and merit grants. 2. Graduate: sum GR fellowship amount and institutional tuition, fee waivers, institutional military amount, and institutional veterans amount

SAS Code

```
* Source variable instneed="Institutional need-based grants";
* Source variable instnond="Institutional non-need and merit grants";
* Source variable grinfel="Graduate fellowship amount";
* Source variable inswaiv="Institutional tuition & fee waivers";
  ****************************************************************;
  if stypelst = 1 then
    do;
      ingrtamt = instneed + instnond;
      if missing(ingrtamt) = 1 then
        do;
          ingrtamt = .x;
         end;
    end;
  else if stypelst > 1 then
    do;
      ingrtamt = grinfel + inswaiv + insmilamt + insvetamt;
      if missing(ingrtamt) = 1 then
        do;
          ingrtamt = .x;
        end;
```

<table>
<tr><td><b>Variable Name</b></td><td><b>INLNAMT</b></td></tr>
<tr><td><b>Variable Label</b></td><td>Institutional loans amount</td></tr>
<tr><td><b>Description</b></td><td>Total amount of all institutional loans (funded by the educational institution) received during the 2015–16 academic year</td></tr>
<tr><td><b>Assigned Values</b></td><td>
<table>
<tr><th>Value</th><th>Value label</th></tr>
<tr><td><b>0</b></td><td>{Zero}</td></tr>
<tr><td><b>c</b></td><td>{Continuous}</td></tr>
</table>
</td></tr>
<tr><td><b>Data Source(s)</b></td><td>NPSAS:16 Student Records</td></tr>
<tr><td><b>Component Derived Variables</b></td><td>Not applicable</td></tr>
<tr><td><b>Applies to</b></td><td>All study members</td></tr>
<tr><td><b>Availability</b></td><td>NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16</td></tr>
<tr><td><b>Change History</b></td><td>Unchanged from NPSAS:96 through NPSAS:16.</td></tr>
<tr><td><b>N:16 Usage Statistics</b></td><td>Number of uses: 110 (UG); 0 (GR)</td></tr>
<tr><td><b>Programming Narrative</b></td><td>Sum institutional loans from student records (type=6 and grad type=2):
1. Graduate aid section:
a. If CFAGRTYP = 2 and CFAGRAMT>0
b. Else if CFAGRTYP = 2 and CFAGRAMT = missing, then set minimum imputation flag to 1, to impute positive value.
2. Institutional aid section:
a. If CFAITYP = 6 and CFAIAMT>0
b. Else if CFAITYP = 6 and CFAIAMT = missing, then set minimum imputation flag to 1, to impute positive value.
3. Other aid section:
a. If CFAOTHSRC = 1 and CFAOTHTYP = 6 and CFAOTHAMT>0
b. Else if CFAOTHSRC = 1 and CFAOTHTYP = 6 and CFAOTHAMT = missing, then set minimum imputation flag to 1, to impute positive value.</td></tr>
<tr><td><b>SAS Code</b></td><td><pre>
* Source variable cfagrtyp01-cfagrtyp03="Graduate aid program type";
* Source variable cfagramt01-cfagramt03="Graduate aid program amount";
* Source variable cfaityp01-cfaityp03="Institution aid program type";
* Source variable cfaiamt01-cfaiamt03="Institution aid program amount";
* Source variables cfa1othsrc-cfa3othsrc="Other aid source";
* Source variables cfa1othtyp-cfa3othtyp="Other aid type";
* Source variables cfa1othamt-cfa3othamt="Other aid amount";
* Source variable cfacvans="Received any financial aid";
***;
  array gtyp   (3) cfagrtyp01 cfagrtyp02 cfagrtyp03;
  array gtemp   (3) loang1   loang2   loang3;
  array gamt   (3) cfagramt01 cfagramt02 cfagramt03;
  array ityp   (3) cfaityp01 cfaityp02 cfaityp03;
  array itemp   (3) loani1   loani2   loani3;
  array iamt   (3) cfaiamt01 cfaiamt02 cfaiamt03;
  array otemp   (3) loano1   loano2   loano3;
  array othsrc (3) cfa1othsrc cfa2othsrc cfa3othsrc;
  array othtype (3) cfa1othtyp cfa2othtyp cfa3othtyp;
  array oamt   (3) cfa1othamt cfa2othamt cfa3othamt;
</pre></td></tr>
</table>

```
do i = 1 to 3;
  gtemp(i) = 0;
  if cfagraid ~= 0 then
   do;
    if gtyp(i) = 2 then
     do;
      gtemp(i) = gamt(i);
      if missing(gtemp(i)) = 1 then gtemp(i) = .m;
     end;
    else if cfagraid = .x then gtemp(i) = .x;
   end;

  itemp(i) = 0;
  if cfainstaid ~= 0 then
   do;
    if ityp(i) = 6 then
     do;
      itemp(i) = iamt(i);
      if missing(itemp(i)) = 1 then itemp(i) = .m;
     end;
    else if cfainstaid = .x then itemp(i) = .x;
   end;

  otemp(i) = 0;
  if cfaothaid ~= 0 then
   do;
    if othsrc(i) = 1 & othtype(i) = 6 then
     do;
      otemp(i) = oamt(i);
      if missing(otemp(i)) = 1 then otemp(i) = .m;
     end;
    else if cfaothaid = .x then otemp(i) = .x;
   end;
end;
array forInlnamt(9) loang1 loang2 loang3 loani1 loani2 loani3 loano1 loano2 loano3;
do i = 1 to 9;
  if forInlnamt(i) > 0 then
   do j = 1 to 9;
    if forInlnamt(j) = .x then forInlnamt(j) = 0;
   end;
end;
inlnamt = loang1 + loang2 + loang3 + loani1 + loani2 + loani3 + loano1 + loano2 + loano3;
inlnamt_minimp = sum(of loang1 loang2 loang3 loani1 loani2 loani3 loano1 loano2 loano3);
if inlnamt_minimp = . then inlnamt_minimp = 0;
do i = 1 to 9;
  if forInlnamt(i) = .m then Inlnamt_minimp = Inlnamt_minimp + 1;
end;
if missing(inlnamt) = 1 then
 do;
  inlnamt = .x;
 end;
```

<table>
<tr><td><b>Variable Name</b></td><td><b>INSMERIT</b></td></tr>
<tr><td><b>Variable Label</b></td><td>Institutional merit-only grants amount</td></tr>
<tr><td><b>Description</b></td><td>Total amount of institutional merit-only grants and scholarships received during the 2015–16 academic year.</td></tr>
<tr><td><b>Assigned Values</b></td><td><table><tr><th>Value</th><th>Value label</th></tr><tr><td><b>0</b></td><td>{Zero}</td></tr><tr><td><b>c</b></td><td>{Continuous}</td></tr></table></td></tr>
<tr><td><b>Data Source(s)</b></td><td>NPSAS:16 Student Records</td></tr>
<tr><td><b>Component Derived Variables:</b></td><td>INATHAMT</td></tr>
<tr><td><b>Applies to</b></td><td>All undergraduate study members</td></tr>
<tr><td><b>Availability</b></td><td>NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16</td></tr>
<tr><td><b>Change History</b></td><td>Unchanged from NPSAS:96 through NPSAS:16.</td></tr>
<tr><td><b>N:16 Usage Statistics</b></td><td>Number of uses: 259 (UG); 0 (GR)</td></tr>
<tr><td><b>Programming Narrative</b></td><td>Sum of imputed interim variable (meritNoAth) and derived variable INATHAMT.
1. Create INATHAMT.
2. Create meritNoAth by summing the merit-only scholarships from student records (type=2):
   a. Institutional aid section:
      i. if CFAITYP = 2 and CFAIAMT >0
      ii. Else if CFAITYP = 2 and CFAIAMT = missing, then set minimum imputation flag to 1 (to impute positive value).
   b. Other aid section:
      i. If CFAOTHSRC = 1 and CFAOTHTYP = 2 and CFAOTHAMT>0
      ii. Else if CFAOTHSRC = 1 and CFAOTHTYP = 2 and CFAOTHAMT = missing, then set minimum imputation flag to 1 (to impute positive value).
3. Impute in batch 3, both INATHAMT and meritNoAth.
4. Final variable is sum of INATHAMT and meritNoAth.</td></tr>
<tr><td><b>SAS Code</b></td><td>* Source variable inathamt="Athletic scholarships";
* Source variable meritNoAth="Institutional merit-only grants except for athletic scholarships";
  **;
if stypelst > 1 then do; insmerit = .y; end;
  else if stypelst = 1 then
   do;
    insmerit = inathamt + meritNoAth;
    if missing(insmerit) = 1 then
     do;
      insmerit = .x;
     end;</td></tr>
</table>

Variable Name	**INSMILAMT**
Variable Label	Institution military/armed forces grants amount
Description	Total amount of institution military/armed forces grants received in 2015–16 academic year

Assigned Values

Value	Value label
0	{Zero}
c	{Continuous}

Data Source(s)	NPSAS:16 Student Records
Component Derived Variables	Not applicable
Applies to	All study members
Availability	NPSAS:16
Change History	New variable in NPSAS:16
N:16 Usage Statistics	Number of uses: 0 (UG); 0 (GR)
Programming Narrative	Sum student records institution military grants: 1. ROTC/Armed forces a. cfaiamt if cfaityp= 14 b. cfaothamt if cfaothsrc = 1 & cfaothtyp = 14

SAS Code

```
* Source variable cfaityp01-cfaityp03="Institution aid program type";
* Source variable cfaiamt01-cfaiamt03="Institution aid program amount";
* Source variables cfa1othsrc-cfa3othsrc="Other aid source";
* Source variables cfa1othtyp-cfa3othtyp="Other aid type";
* Source variables cfa1othamt-cfa3othamt="Other aid amount";
* Source variable cfacvans="Received any financial aid";
  ************************************************************;
  array ityp    (3) cfaityp01  cfaityp02  cfaityp03;
  array itemp   (3) mili1      mili2      mili3;
  array iamt    (3) cfaiamt01  cfaiamt02  cfaiamt03;
  array otemp   (3) milo1      milo2      milo3;
  array othsrc  (3) cfa1othsrc cfa2othsrc cfa3othsrc;
  array othtype (3) cfa1othtyp cfa2othtyp cfa3othtyp;
  array oamt    (3) cfa1othamt cfa2othamt cfa3othamt;

  do i = 1 to 3;
    itemp(i) = 0;
    if cfainstaid ~= 0 then
     do;
      if ityp(i) = 14 then
       do;
        itemp(i) = iamt(i);
        if missing(itemp(i)) = 1 then itemp(i) = .m;
       end;
      else if cfainstaid = .x then itemp(i) = .x;
     end;

    otemp(i) = 0;
    if cfaothaid ~= 0 then
     do;
      if othsrc(i) = 1 & othtype(i) = 14 then
       do;
        otemp(i) = oamt(i);
        if missing(otemp(i)) = 1 then otemp(i) = .m;
       end;
      else if cfaothaid = .x then otemp(i) = .x;
     end;
  end;
  array forlnsmilamt(6) mili1 mili2 mili3 milo1 milo2 milo3;
  do i = 1 to 6;
    if forlnsmilamt(i) > 0 then
     do j = 1 to 6;
      if forlnsmilamt(j) = .x then forlnsmilamt(j) = 0;
     end;
  end;
  insmilamt = mili1 + mili2 + mili3 + milo1 + milo2 + milo3;
  if missing(insmilamt) = 1 then
   do;
    insmilamt = .x;
   end;
```

Variable Name	**INSTAMT**
Variable Label	Institutional aid amount
Description	Total amount of institutional aid received during the 2015–16 academic year.

Assigned Values	**Value** | **Value label**
	0 | {Zero}
	c | {Continuous}

Data Source(s)	NPSAS:16 Student Records, NPSAS:16 Interview
Component Derived Variables	INGRTAMT, INLNAMT, INSTWRK, GRASTAMT
Applies to	All study members
Availability	NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16
Change History	Unchanged from NPSAS:96 through NPSAS:16.
N:16 Usage Statistics	Number of uses: 125 (UG); 23 (GR)
Programming Narrative	Equal to the sum of institutional grants and fellowships (INGRTAMT), institutional loans (INLNAMT), institution-sponsored work-study (INSTWRK), and graduate student assistantships (GRASTAMT) for graduate students.

SAS Code

```
* Source variable ingrtamt="Institution grants total";
* Source variable instwrk="Institutional work-study";
* Source variable inlnamt="Institutional loans";
* Source variable grastamt="Graduate assistantships";
**************************************************************************;
* Sum total institutional grants, institutional work-study, institutional loans
* and graduate assistantships.

if stypelst = 1 then
   do;
    instamt = ingrtamt + instwrk + inlnamt;
    if missing(instamt) = 1 then
```

```
          do;
            instamt = .x;
    else if stypelst > 1 then
      do;
        instamt = ingrtamt + instwrk + inlnamt + grastamt;
        if missing(instamt) = 1 then
          do;
            instamt = .x;
          end;
```

Variable Name	**INSTNEED**
Variable Label	Institutional need-based grants amount
Description	Total amount of institutional need-based grants received during the 2015–16 academic year.

Assigned Values	**Value** **Value label**
	0 {Zero}
	c {Continuous}

Data Source(s)	NPSAS:16 Student Records
Component Derived Variables	Not applicable
Applies to	All undergraduate study members
Availability	NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16
Change History	Unchanged from NPSAS:2000 through NPSAS:16. This variable does not include state grants for California public institutions that are funded by state dollars and allocated by the institutions (e.g., Community College Board of Governors Grants, California State University Grants, and Educational Opportunity Program grants). These were classified as state need-based grants (STATNEED) in NPSAS:16, which differs from prior NPSAS studies. To compare estimates of institutional need-based grants over time, use this variable and INSTNEED2 from prior NPSAS studies.
N:16 Usage Statistics	Number of uses: 291 (UG); 0 (GR)
Programming Narrative	Sum need-based and both need and merit-based institutional scholarships in student records: 1. Institutional aid section: a. if CFAITYP in (1 3) and CFAIAMT >0 b. Else if CFAITYP in (1 3) and CFAIAMT = missing, then set minimum imputation flag to 1 (to impute positive value). 2. Other aid section: a. If CFAOTHSRC = 1 and CFAOTHTYP in (1 3) and CFAOTHAMT>0 b. Else if CFAOTHSRC = 1 and CFAOTHTYP in (1 3) and CFAOTHAMT = missing, then set minimum imputation flag to 1 (to impute positive value). 3. Cap edit: For values greater than 73,000, cap at $73,000.

SAS Code	* Source variable cfaityp01-cfaityp03="Institution aid program type";
	* Source variable cfaiamt01-cfaiamt03="Institution aid program amount";
	* Source variables cfa1othsrc-cfa3othsrc="Other aid source";
	* Source variables cfa1othtyp-cfa3othtyp="Other aid type";
	* Source variables cfa1othamt-cfa3othamt="Other aid amount";
	* Source variable cfacvans="Received any financial aid";
	***;
	array ityp (3) cfaityp01 cfaityp02 cfaityp03;

```
array itemp   (3) ineedi1    ineedi2    ineedi3;
array iamt    (3) cfaiamt01  cfaiamt02  cfaiamt03;
array otemp   (3) ineedo1    ineedo2    ineedo3;
array othsrc  (3) cfa1othsrc cfa2othsrc cfa3othsrc;
array othtype (3) cfa1othtyp cfa2othtyp cfa3othtyp;
array oamt    (3) cfa1othamt cfa2othamt cfa3othamt;

do i = 1 to 3;
  itemp(i) = 0;
  if cfainstaid ~= 0 then
   do;
    if ityp(i) in (1, 3) then
     do;
      itemp(i) = iamt(i);
      if missing(itemp(i)) = 1 then itemp(i) = .m;
     end;
    else if cfainstaid = .x then itemp(i) = .x;
   end;

  otemp(i) = 0;
  if cfaothaid ~= 0 then
   do;
    if othsrc(i) = 1 & othtype(i) in (1, 3) then
     do;
      otemp(i) = oamt(i);
      if missing(otemp(i)) = 1 then otemp(i) = .m;
     end;
    else if cfaothaid = .x then otemp(i) = .x;
   end;
end;

array forInstneed(6) ineedi1 ineedi2 ineedi3 ineedo1 ineedo2 ineedo3;
do i = 1 to 6;
  if forInstneed(i) > 0 then
   do j = 1 to 6;
    if forInstneed(j) = .x then forInstneed(j) = 0;
   end;
end;

* Applies to: All undergraduate study members;
if stypelst > 1 then do; instneed = .y; instneed_minimp = .y; instneed_yn = .y; end;
else if stypelst = 1 then
 do;
  instneed = ineedi1 + ineedi2 + ineedi3 + ineedo1 + ineedo2 + ineedo3;
  if missing(instneed) = 1 then
   do;
    instneed = .x;
   end;
```

<table>
<tr><td><b>Variable Name</b></td><td><b>INSTNOND</b></td></tr>
<tr><td><b>Variable Label</b></td><td>Institutional non-need & merit grants amount</td></tr>
<tr><td><b>Description</b></td><td>Total amount of institutional grants and scholarships not based on financial need received during the 2015–16 academic year.</td></tr>
<tr><td><b>Assigned Values</b></td><td><table><tr><th>Value</th><th>Value label</th></tr><tr><td><b>0</b></td><td>{Zero}</td></tr><tr><td><b>c</b></td><td>{Continuous}</td></tr></table></td></tr>
<tr><td><b>Data Source(s)</b></td><td>NPSAS:16 Student Records</td></tr>
<tr><td><b>Component Derived Variables</b></td><td>INSTNOND1, INSMERIT, INSMILAMT, INSVETAMT</td></tr>
<tr><td><b>Applies to</b></td><td>All undergraduate study members</td></tr>
<tr><td><b>Availability</b></td><td>NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16</td></tr>
<tr><td><b>Change History</b></td><td>Unchanged from NPSAS:2000 through NPSAS:16.</td></tr>
<tr><td><b>N:16 Usage Statistics</b></td><td>Number of uses: 93 (UG); 0 (GR)</td></tr>
<tr><td><b>Programming Narrative</b></td><td>For undergraduates only (STYPELST = 1): Sum of merit-based grants (INSMERIT), institution non-need grants (INSTNOND1), military/armed forces grants (INSMILAMT), and institutional veterans education benefits (INSVETAMT). INATHAMT, EMPLYWAIV, and INSWAIV are included in one or more of the variables above.</td></tr>
<tr><td><b>SAS Code</b></td><td><pre>
* Source variable insmerit="Institutional merit-only grants";
* Source variable instnond1="Institution non-need grants";
* Source variable insmilamt="Institutional military/armed forces grants";
* Source variable insvetamt="Institutional Veterans' education benefits";
  **;
  * Applies to: All undergraduate study members;
if stypelst > 1 then do; instnond = .y; end;
  else if stypelst = 1 then
   do;
    instnond = insmerit + instnond1 + insmilamt + insvetamt;
    if missing(instnond) = 1 then
     do;
      instnond = .x;
     end;
</pre></td></tr>
</table>

<table>
<tr><td><b>Variable Name</b></td><td><b>INSTNOND1</b></td></tr>
<tr><td><b>Variable Label</b></td><td>Institutional non-need grants amount</td></tr>
<tr><td><b>Description</b></td><td>Total amount of institutional non-need-based grants and tuition waivers during the 2015–16 academic year.</td></tr>
<tr><td><b>Assigned Values</b></td><td><table><tr><th>Value</th><th>Value label</th></tr><tr><td><b>0</b></td><td>{Zero}</td></tr><tr><td><b>c</b></td><td>{Continuous}</td></tr></table></td></tr>
<tr><td><b>Data Source(s)</b></td><td>NPSAS:16 Student Records</td></tr>
<tr><td><b>Component Derived Variables</b></td><td>INSWAIV, EMPLWAIV</td></tr>
</table>

Applies to	All undergraduate study members
Availability	NPSAS:16
Change History	New variable in NPSAS:16
N:16 Usage Statistics	Number of uses: 43 (UG); 0 (GR)
Programming Narrative	Sum of institutional categorical grants (instCatGrt) and institutional tuition waivers (INSWAIV). 1. Create INSWAIV (see spec). Includes EMPLWAIV. 2. Create instCatGrt. Sum of unclassified or non-need, non-merit grants from student records (type=4, 13, or 17). a. Institutional aid section: i. If CFAITYP in (4, 13, 17) and CFAIAMT>0 ii. Else if CFAITYP in (4, 13, 17) and CFAIAMT = missing, then set minimum imputation flag to 1, to impute positive value. b. Other aid section: i. If CFAOTHSRC = 1 and CFAOTHTYP in (4, 13, 17) and CFAOTHAMT>0 ii. Else if CFAOTHSRC = 1 and CFAOTHTYP in (4, 13, 17) and CFAOTHAMT = missing, then set minimum imputation flag to 1, to impute positive value. 3. Create the final variable after imputation and after creating INSWAIV. Sum of INSWAIV and instCatGrt.

SAS Code

```
* Source variable instCatGrt="Institutional categorical grants";
* Source variable waivNoEmp="Institutional tuition & fee waivers except for those to staff";
* Source variable emplwaiv="Institutional tuition waivers for staff";
  *************************************************************;
if stypelst > 1 then do; instnond1 = .y; end;
  else if stypelst = 1 then
   do;
    instnond1 = instCatGrt + waivNoEmp + emplwaiv;
    if missing(instnond1) = 1 then
     do;
      instnond1 = .x;
     end;
```

Variable Name	**INSTWRK**
Variable Label	Institutional work-study amount
Description	Total amount of all institutionally sponsored work-study received during the 2015–16 academic year.
Assigned Values	see table below
Data Source(s)	NPSAS:16 Student Records, NPSAS:16 Interview
Component Derived Variables	Not applicable
Applies to	All study members
Availability	NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16
Change History	Unchanged from NPSAS:96 through NPSAS:16.
N:16 Usage Statistics	Number of uses: 5 (UG); 0 (GR)

Value	Value label
0	{Zero}
c	{Continuous}

Programming Narrative

Sum of student records institution work-study, graduate institution work-study, and other aid source=institution; type=work-study (type=7, graduate type=10).

cfaiamt if cfaityp=7 +
cfagramt if cfagrtyp = 10 +
cfaothamt if cfaothsrc = 1 & cfaothtyp = 7

Set to missing if the institution aid section gate is missing (cfainstaid) or the graduate aid section gate is missing (cfagraid), or the other aid section gate is missing (cfaothaid). Any time there was indication of receipt of this aid (e.g., source and type are correct), but the amount is missing, set a minimum imputation floor as the sum of any other amounts of the same type and source, else set this floor to 1, so that we impute a positive amount.

Edits:
1. No cap for undergraduate (No need).
2. Edit to the student's cost of attendance (BUDGETAJ) if STYPELST>1.

SAS Code

```
* Source variable cfacvans="Received any financial aid";
* Source variables cfaiamt01-cfaiamt03="Institution aid program 1 amount";
* Source variables cfaityp01-cfaityp03="Institution aid program 1 type";
* Source variables cfagramt01-cfagramt03="Graduate aid program amount";
* Source variables cfagrtyp01-cfagrtyp03="Graduate aid program type";
* Source variables cfa1othamt-cfa3othamt="Other aid amount";
* Source variables cfa1othsrc-cfa3othsrc="Other aid source";
* Source variables cfa1othtyp-cfa3othtyp="Other aid type";
**************************************************************************;
 array iamt(3)     iamt1-iamt3;
  array sriamt(3)   cfaiamt01-cfaiamt03;
  array srityp(3)   cfaityp01-cfaityp03;
  array gramti(3)   gramti1-gramti3;
  array srgramt(3)  cfagramt01-cfagramt03;
  array srgrtyp(3)  cfagrtyp01-cfagrtyp03;
  array othamti(3)  othamti1-othamti3;
  * array srothamt(3) cfa1othamt cfa2othamt cfa3othamt;
  * array srothsrc(3) cfa1othsrc cfa2othsrc cfa3othsrc;
  * array srothtyp(3) cfa1othtyp cfa2othtyp cfa3othtyp;
  do i = 1 to 3;
   iamt(i) = 0;
   if cfainstaid ~= 0 then
    do;
     if srityp(i) = 7 then
      do;
       iamt(i) = sriamt(i);
       if missing(iamt(i)) = 1 then iamt(i) = .m;
      end;
     else if cfainstaid = .x then iamt(i) = .x;
    end;

   gramti(i) = 0;
   if cfagraid ~= 0 then
    do;
     if srgrtyp(i) = 10 then
      do;
```

```
       gramti(i) = srgramt(i);
       if missing(gramti(i)) = 1 then gramti(i) = .m;
      end;
     else if cfagraid = .x then gramti(i) = .x;
    end;

   othamti(i) = 0;
   if cfaothaid ~= 0 then
    do;
     if srothsrc(i) = 1 & srothtyp(i) = 7 then
      do;
       othamti(i) = srothamt(i);
       if missing(othamti(i)) = 1 then othamti(i) = .m;
      end;
     else if cfaothaid = .x then othamti(i) = .x;
    end;
  end;
  array forInstwrk(9) iamt1 iamt2 iamt3 gramti1 gramti2 gramti3 othamti1 othamti2 othamti3;
  do i = 1 to 9;
    if forInstwrk(i) > 0 then
     do j = 1 to 9;
      if forInstwrk(j) = .x then forInstwrk(j) = 0;
     end;
  end;

  instwrk = iamt1 + iamt2 + iamt3 + gramti1 + gramti2 + gramti3 + othamti1 + othamti2 +
othamti3;


  if missing(instwrk) = 1 then
   do;
    instwrk = .x;
   end;
```

<table>
<tr><td><b>Variable Name</b></td><td><b>INSVETAMT</b></td></tr>
<tr><td><b>Variable Label</b></td><td>Institutional veterans education benefits amount</td></tr>
<tr><td><b>Description</b></td><td>Total amount of institutional veterans education benefits received in 2015–16 academic year.</td></tr>
<tr><td><b>Assigned Values</b></td><td><table><tr><th>Value</th><th>Value label</th></tr><tr><td><b>0</b></td><td>{Zero}</td></tr><tr><td><b>c</b></td><td>{Continuous}</td></tr></table></td></tr>
<tr><td><b>Data Source(s)</b></td><td>NPSAS:16 Student Records</td></tr>
<tr><td><b>Component Derived Variables</b></td><td>Not applicable</td></tr>
<tr><td><b>Applies to</b></td><td>All study members</td></tr>
<tr><td><b>Availability</b></td><td>NPSAS:16</td></tr>
<tr><td><b>Change History</b></td><td>New variable in NPSAS:16</td></tr>
</table>

N:16 Usage Statistics	Number of uses: 22 (UG); 0 (GR)
Programming Narrative	Sum institution veterans education benefits from Student Records (type=15) and institution veterans education benefits (type=15) reported in other sources (source=3). 1. cfaiamt if cfaityp = 15 2. cfaothamt if cfaothsrc = 3 & cfaothtyp = 15

SAS Code

```
* Source variable cfaityp01-cfaityp03="Institution aid program type";
* Source variable cfaiamt01-cfaiamt03="Institution aid program amount";
* Source variables cfa1othsrc-cfa3othsrc="Other aid source";
* Source variables cfa1othtyp-cfa3othtyp="Other aid type";
* Source variables cfa1othamt-cfa3othamt="Other aid amount";
* Source variable cfacvans="Received any financial aid";
  **************************************************************;
  array ityp    (3) cfaityp01  cfaityp02  cfaityp03;
  array itemp   (3) veti1      veti2      veti3;
  array iamt    (3) cfaiamt01  cfaiamt02  cfaiamt03;
  array otemp   (3) veto1      veto2      veto3;
  array othsrc  (3) cfa1othsrc cfa2othsrc cfa3othsrc;
  array othtype (3) cfa1othtyp cfa2othtyp cfa3othtyp;
  array oamt    (3) cfa1othamt cfa2othamt cfa3othamt;

  do i = 1 to 3;
    itemp(i) = 0;
    if cfainstaid ~= 0 then
     do;
      if ityp(i) = 15 then
       do;
        itemp(i) = iamt(i);
        if missing(itemp(i)) = 1 then itemp(i) = .m;
       end;
      else if cfainstaid = .x then itemp(i) = .x;
     end;

    otemp(i) = 0;
    if cfaothaid ~= 0 then
     do;
      if othsrc(i) = 1 & othtype(i) = 15 then
       do;
        otemp(i) = oamt(i);
        if missing(otemp(i)) = 1 then otemp(i) = .m;
       end;
      else if cfaothaid = .x then otemp(i) = .x;
     end;
  end;
  array forInsvetamt(6) veti1 veti2 veti3 veto1 veto2 veto3;
  do i = 1 to 6;
    if forInsvetamt(i) > 0 then
     do j = 1 to 6;
      if forInsvetamt(j) = .x then forInsvetamt(j) = 0;
     end;
  end;
```

	insvetamt = veti1 + veti2 + veti3 + veto1 + veto2 + veto3;
	if missing(insvetamt) = 1 then
	do;
	insvetamt = .x;
	end;

Variable Name	**INSWAIV**
Variable Label	Institutional tuition & fee waivers amount
Description	Total amount of institutional tuition waivers received during the 2015–16 academic year.

Assigned Values	**Value** **Value label**
	0 {Zero}
	c {Continuous}

Data Source(s)	NPSAS:16 Student Records
Component Derived Variables	EMPLWAIV
Applies to	All study members
Availability	NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16
Change History	Unchanged from NPSAS:96 through NPSAS:16.
N:16 Usage Statistics	Number of uses: 26 (UG); 40 (GR)
Programming Narrative	This variable is the sum of an interim and imputed variable, waivNoEmp and the derived variable EMPLWAIV. 1. Create EMPLWAIV (see specs). 2. Create waivNoEmp. Sum non-faculty and staff tuition waivers from student records. 3. Create the final variable after imputing both source variables. INSWAIV = waivNoEmp + EMPLWAIV

SAS Code	***; * Source variable waivNoEmp="Institutional tuition & fee waivers except for those to staff";
	* Source variable emplwaiv="Institutional tuition waivers for staff";
	***;
	inswaiv = waivNoEmp + emplwaiv;
	if missing(inswaiv) = 1 then do;
	inswaiv = .x;
	end;

Variable Name	**MFT**
Variable Label	Number of full-time months enrolled
Description	Number of months enrolled full-time between July 2015 and June 2016.

Assigned Values	**Value** **Value Label**
	0 {Zero}
	c {Continuous}

Data Source(s)	NPSAS:16 Student Records, NPSAS:16 Interview, NSC:16, NSLDS:16

Component Derived Variables	ENR01-ENR12 Note: ENR01-ENR12 are not included in this report. Information on these variables may be found in the NPSAS:16 undergraduate codebook, https://nces.ed.gov/datalab/powerstats/codebook.aspx?dataset=121&type=subject#ENR01, and the NPSAS:16 graduate student codebook, https://nces.ed.gov/datalab/powerstats/codebook.aspx?dataset=122&type=subject#ENR01.
Applies to	All study members
Availability	NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16
Change History	Unchanged from NPSAS:96 through NPSAS:16; NSC has only been a data source since NPSAS:12
N:16 Usage Statistics	Number of uses: 1 (UG); 0 (GR)
Programming Narrative	Sum the number of months enrolled full-time from July 2015 to June 2016

SAS Code	* Source variable enr1-enr12="Monthly enrollment status";
	***;
	 array enr{12} enr01-enr12;
	 array em{0:2} enlen mft mpt;
	
	   do i = 0 to 2;
	     em{i} = 0;
	   end;
	
	   do i = 1 to 12;
	     * Count total months enrolled;
	     if enr{i} > 0 then enlen + 1;
	     do j = 1 to 2;
	     * Count months enrolled full-time and part-time;
	       if enr{i} = j then em{j} + 1;
	     end;
	   end;

Variable Name	**MILTYPE**
Variable Label	Military type
Description	Student's type of military service or veteran status during the 2015–16 academic year.
Assigned Values	**Value** **Value label** **0** No military service **1** Active duty **2** Reserves or National Guard **3** Veteran
Data Source(s)	VBA:16, NPSAS:16 Interview
Component Derived Variables	AGE
Applies to	All study members
Availability	NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16

Change History	Unchanged from NPSAS:04 through NPSAS:08; NPSAS:12 introduced the National Guard category, which was combined with Reserves in NPSAS:16. In NPSAS:16, changed to incorporate VBA in place of CPS.
N:16 Usage Statistics	Number of uses: 614 (UG); 12 (GR)
Programming Narrative	1. Take VBA indicators 2. Take student interview

SAS Code

```
/*------------- MILTYPE: Military service type ------------*/
/* VBA */
IF actduty_vba=1 THEN MILVBA=1;       /* take "active duty" */
ELSE IF veteran_vba=1 THEN MILVBA=3;  /* then "veteran" */
ELSE IF reserves_vba=1 THEN MILVBA=2; /* then "reserves" */

/* student interview */
IF N16FMILITC=1 THEN MILSI=2;       /* take "reserves" */
ELSE IF N16FMILITD=1 THEN MILSI=4;  /* then "national guard" */

/* first take VBA then student interview */
IF MILVBA=1 THEN MILTYPE=1;
ELSE IF MILVBA=3 THEN MILTYPE=3;
ELSE IF MILVBA=2 OR MILSI=2 THEN MILTYPE=2;
ELSE IF MILSI=4 THEN MILTYPE=4;
ELSE IF N16FMILITN>=0 THEN MILTYPE=0; /* WHY: anything non-missing, implying
they answered the form which contains N16FMILITA-N16FMILITN. We trust
that VBA is the best and only source for active duty and Veterans, but
not for Reserves & National Guard. So, if the student answered and didn’t
check either of those boxes, then we assume they have no military service
for NPSAS purposes, because we already have our ï¿½trueï¿½ answers for Active
duty and Veteran status from the VBA */

/* The Reserves and Nat. Guard have an age limit that varies by branch,
   but the oldest is 42 and the youngest is always 17 */
IF 0<AGE<17 OR AGE>42 AND MILTYPE IN (.,2,4) THEN DO; MILTYPE = 0;
MILTYPE_imp=1; END;
```

Variable Name	**MILTYPE2**
Variable Label	Military type (for dependency)
Description	Student’s type of military service or veteran status (for dependency and federal aid) during the 2015–16 academic year.
Assigned Values	**Value** **Value label** **0** No military service **1** Active duty **2** Reserves or National Guard **3** Veteran
Data Source(s)	FAFSA:16, VBA:16, NPSAS:16 Interview
Component Derived Variables	AGE
Applies to	All study members

Availability	NPSAS:16
Change History	New variable in NPSAS:16
N:16 Usage Statistics	Number of uses: 183 (UG); 7 (GR)
Programming Narrative	This variable measures military service with priority given to CPS. It is used to calculate undergraduates' dependency status (DEPEND). It is more directly comparable than MILTYPE to previous NPSAS studies. 1. If the student was on active duty according to CPS, take this value first (c16actduty = 1). 2. Else if the student was a veteran according to CPS, take this value second (c16veteran = 1). 3. Else if both the active duty and veteran flags from CPS are no, set to "no military service." 4. Else if the student was on active duty according to the VBA (actduty_vba), take this value next. 5. Else if the student was a veteran according to the VBA (veteran_vba), take this value next (MILTYPE2 = 3). 6. If the student was in the reserves according to the VBA (reserves_vba) and not already listed as on active duty or a veteran, take this value next (MILTYPE2 = 2) even if the value is "no military service" from step 3. The reason is that reserves training is not a dependency trigger unless the student had been put on active duty. 7. Since the VBA will not contain service records for reservists or National Guard members who were never on active duty or who never received benefits, use the student interview to fill in these values. a. If N16FMILITC = 1 and MILTYPE2 is currently in (0 or missing), then MILTYPE2 = 2 b. Else if N16FMILITD = 1 and MILTYPE2 is currently in (0 or missing), then MILTYPE2 = 4 8. We still need to impute respondents who didn't respond to the interview into the National Guard or Reserves and No military service categories. The assumption is VBA has full coverage of active duty service. But this will NOT include service in the reserves or National Guard if the student never gets put on active duty. The interview is our only source for activation, but we don't have 100% response. 9. Set MILTYPE2 to zero if any of the following conditions are met: a. The student responded to the military status question in the interview (N16FMILITN>=0), but MILTYPE is still missing. b. When MILTYPE2 is either still missing, Reserves, or National Guard (in missing, 2, or 4), and the student is either younger than age 17, or over the age of 42, set MILTYPE2 to zero. (The Reserves and National Guard have age limits that vary by branch, but the oldest is 42 and the youngest is always 17.) 10. For Imputation: Only impute students into categories of either no military service or the Reserves or National Guard. The reason is that we have complete coverage of active duty military and prior active duty service (i.e., veterans) from VBA. We cannot identify students in the Reserves or National Guard in the NPSAS year who were never on active duty.

SAS Code	/*------------ MILTYPE2: Military service type (for dependency) ---------*/
	/* CPS */
	IF C16ACTDUTY=1 THEN MILCPS=1; /* Active duty */
	ELSE IF C16VET=1 THEN MILCPS=3; /* veteran */
	ELSE IF C16ACTDUTY=2 AND C16VET=2 THEN MILCPS=0; /* none */
	
	/* VBA */
	IF actduty_vba=1 THEN MILVBA2=1; /* Active duty */
	ELSE IF veteran_vba=1 THEN MILVBA2=3; /* veteran */
	
	/* take CPS -> VBA -> student interview */

```
IF MILCPS>=0 THEN MILTYPE2=MILCPS;
ELSE IF MILVBA2>0 THEN MILTYPE2=MILVBA2;
IF reserves_vba=1 AND MILTYPE2<1 THEN MILTYPE2=2;
IF MILTYPE2 IN (.,0) THEN DO;
 IF N16FMILITC=1 THEN MILTYPE2=2;
 ELSE IF N16FMILITD=1 THEN MILTYPE2=4;
 ELSE IF N16FMILITN>=0 THEN MILTYPE2=0;
END;

/* The Reserves and Nat. Guard have an age limit that varies by branch,
   but the oldest is 42 and the youngest is always 17 */
IF 0<AGE<17 OR AGE>42 AND MILTYPE2 IN (.,2,4) THEN DO; MILTYPE2=0;
MILTYPE2_imp=1; END;

/* for veteran, set to zero if CPS EFC is dependent */
IF MILTYPE2=3 AND C16EFCPT IN (1,4) THEN DO; MILTYPE2=0; MILTYPE2_imp=1; END;
```

Variable Name	**NETCST1**
Variable Label	Student budget minus all aid
Description	Represents the estimated “out-of-pocket” expense to students remaining after all financial aid received for the 2015–16 academic year.

Assigned Values	**Value** **Value label**
	-3 {Skipped}
	0 {Zero}
	c {Continuous}

Data Source(s)	NPSAS:16 Student Records, IPEDS:16, NSLDS:16, NPSAS:16 Interview, FAFSA:16, VBA:16, NSC:16
Component Derived Variables	BUDGETAJ, TOTAID, STUDMULT
Applies to	Study members who attended one institution (STUDMULT=1)
Availability	NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16
Change History	Unchanged from NPSAS:96 through NPSAS:16; NSC has only been a data source since NPSAS:12
N:16 Usage Statistics	Number of uses: 156 (UG); 4 (GR)
Programming Narrative	1. Subtract total aid (TOTAID) from the total budget (BUDGETAJ). 2. Set negative values to zero 3. Set study members who attended multiple institutions to skip (STUDMULT>1).

SAS Code

```
/* Net cost */
NETCST1=BUDGETAJ-TOTAID;
/* for the following, set negative value to 0 (indication of no need) */
ARRAY A SNEED1--NETCST44;
        DO OVER A;
          IF A<0 THEN A=0;
        END;

/* set skips: skip everyone who went to more than one institution */
ARRAY B LOANCST2--NETCST44;
        DO OVER B; IF STUDMULT>1 THEN B=-3; END;
```

<table>
<tr><td><b>Variable Name</b></td><td><b>NETCST3</b></td></tr>
<tr><td><b>Variable Label</b></td><td>Student budget minus all grants</td></tr>
<tr><td><b>Description</b></td><td>Total net price after all grants for the 2015–16 academic year.</td></tr>
<tr><td><b>Assigned Values</b></td><td><table><tr><th>Value</th><th>Value label</th></tr><tr><td><b>-3</b></td><td>{Skipped}</td></tr><tr><td><b>0</b></td><td>{Zero}</td></tr><tr><td><b>c</b></td><td>{Continuous}</td></tr></table></td></tr>
<tr><td><b>Data Source(s)</b></td><td>NPSAS:16 Student Records, IPEDS:16, NPSAS:16 Interview, NSLDS:16, NSC:16</td></tr>
<tr><td><b>Component Derived Variables</b></td><td>BUDGETAJ, TOTGRT, STUDMULT</td></tr>
<tr><td><b>Applies to</b></td><td>Study members who attended one institution (STUDMULT=1)</td></tr>
<tr><td><b>Availability</b></td><td>NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16</td></tr>
<tr><td><b>Change History</b></td><td>Unchanged from NPSAS:96 through NPSAS:16; NSC has only been a data source since NPSAS:12</td></tr>
<tr><td><b>N:16 Usage Statistics</b></td><td>Number of uses: 515 (UG); 136 (GR)</td></tr>
<tr><td><b>Programming Narrative</b></td><td>1. Subtract total grants (TOTGRT) from the total budget (BUDGETAJ).
2. Set negative values to zero
3. Set study members who attended multiple institutions to skip (STUDMULT>1).</td></tr>
<tr><td><b>SAS Code</b></td><td>/* Net cost */
NETCST3=BUDGETAJ-TOTGRT;
/* for the following, set negative value to 0 (indication of no need) */
ARRAY A SNEED1--NETCST44;
        DO OVER A;
          IF A<0 THEN A=0;
        END;

/* set skips: skip everyone who went to more than one institution */
ARRAY B LOANCST2--NETCST44;
        DO OVER B; IF STUDMULT>1 THEN B=-3; END;</td></tr>
</table>

<table>
<tr><td><b>Variable Name</b></td><td><b>NETCST9</b></td></tr>
<tr><td><b>Variable Label</b></td><td>Tuition and fees minus all grants</td></tr>
<tr><td><b>Description</b></td><td>Tuition and fees minus all grants for the 2015–16 academic year.</td></tr>
<tr><td><b>Assigned Values</b></td><td><table><tr><th>Value</th><th>Value label</th></tr><tr><td><b>-3</b></td><td>{Skipped}</td></tr><tr><td><b>0</b></td><td>{Zero}</td></tr><tr><td><b>c</b></td><td>{Continuous}</td></tr></table></td></tr>
<tr><td><b>Data Source(s)</b></td><td>NPSAS:16 Student Records, IPEDS:16, NPSAS:16 Interview, NSLDS:16, NSC:16</td></tr>
<tr><td><b>Component Derived Variables</b></td><td>TUITION2, TOTGRT, STUDMULT</td></tr>
<tr><td><b>Applies to</b></td><td>Study members who attended one institution (STUDMULT=1).</td></tr>
</table>

Availability	NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16
Change History	Unchanged from NPSAS:96 through NPSAS:16; NSC has been a data source only since NPSAS:12.
N:16 Usage Statistics	Number of uses: 965 (UG); 177 (GR)
Programming Narrative	1. Subtract total grants (TOTGRT) from tuition (TUITION2). 2. Set negative values to zero. 3. Set students who attended multiple institutions to skip (STUDMULT>1).

SAS Code	NETCST9=TUITION2-TOTGRT;
	
	/* for the following, set negative value to 0 (indication of no need) */
	ARRAY A SNEED1--NETCST44;
	        DO OVER A;
	          IF A<0 THEN A=0;
	        END;
	
	/* set skips: skip everyone who went to more than one institution */
	ARRAY B LOANCST2--NETCST44;
	        DO OVER B; IF STUDMULT>1 THEN B=-3; END;

Variable Name	**NFEDCUM1**
Variable Label	Cumulative non-federal loan amount for undergraduate education
Description	Cumulative non-federal/private loan amount borrowed for undergraduate education through June 30, 2016.

Assigned Values	**Value** | **Value label**
	0 | {Zero}
	c | {Continuous}

Data Source(s)	NPSAS:16 Interview, NSLDS:16, NPSAS:16 Student Records
Component Derived Variables	BORAMT1, FEDCUM1
Applies to	All study members
Availability	NPSAS:08, NPSAS:12, NPSAS:16
Change History	Unchanged from NPSAS:08 through NPSAS:16.
N:16 Usage Statistics	Number of uses: 98 (UG); 1 (GR)

Programming Narrative

1. Primary source is interview item, n16cugprvt.
2. Triangulate using federal loan amounts (FEDCUM1), and total amount borrowed for undergraduate education from the interview (n16cugln):
    a. If cumulative federal loan amount for undergrad (FEDCUM1) is greater than the cumulative amount borrowed for undergrad from the interview (n16cugln) and the private amount is greater than zero (n16cugprvt > 0), then set private amount to missing and impute.
    b. Else if the private loan amount value is greater than the total minus the federal amount (private > total – federal), adjust private loan amount downwards such that private amount = total – federal.
3. Check: for undergraduates (STYPELST=1), ensure the cumulative amount is greater than or equal to the private amount borrowed in the NPSAS year (n16cugprvt >= PRIVLOAN). Else, set the cumulative amount value to missing.
4. Use value from the interview, as per criteria above. Else if missing, impute using private amount categorical estimate if possible [n16cugprvest]).

SAS Code

```
* Start with student self-reported cumulative private loan amount from interview;
    if n16cugprvt >= 0 then do; nfedcum1 = n16cugprvt; end;
    * EDIT 1: Set loans less than 50 to zero;
    if 1 <= nfedcum1 <= 49 then
      do;
        nfed1r = 1;
        nfedcum1 = 0;
      end;
* EDIT 2: Set NFEDCUM1 to zero if the self-reported UG total is within 5 percent of the NSLDS total for UG loans (FEDCUM1).;
    if nfedcum1 > 0 & n16cugln > 0 & fedcum1 > 0 then nfedzero = (100 * (abs(n16cugln - fedcum1) / n16cugln) < 5);
    if nfedzero = 1 then
      do;
        nfed1r = 2;
        nfedcum1 = 0;
      end;

* EDIT 3: Adjust NFEDCUM1 downward if UGTOT is greater than the self-reported total N16CUGLN;
* In THEORY, if the student and NSLDS are perfectly reported N16CUGLN = NFEDCUM1 + FEDCUM1;
* Calculate total UG borrowed using unedited NFEDCUM1 and FEDCUM1;
    if missing(nfedcum1) = 0 & missing(fedcum1) = 0 then ugtot = nfedcum1 + fedcum1;

* Calculate the difference between self-reported TOTAL UG loans borrowed and the NSLDS FEDCUM1 (note, will often be negative);
* Do not include those we already flagged and fixed as double counted;
    if ugtot > n16cugln & nfedcum1 > 0 & ugtot > 0 & n16cugln > 0 & nfed1r ~= 2 then nfedcum1_new = n16cugln - fedcum1;
    if missing(nfedcum1_new) = 0 then nfed1r = 3;
    if nfedcum1_new < 0 & missing(nfedcum1_new) = 0  then
      do;
        nfedcum1 = 0;
       end;
    if nfedcum1_new >= 0 & missing(nfedcum1_new) = 0 then
      do;
        nfedcum1 = nfedcum1_new;
    end;

* EDIT 4: For Undergrads (STYPELST = 1): If PRIVLOAN+STLNAMT+INLNAMT is greater
than NFEDCUM1, adjust;
  if missing(nfedcum1) = 0 then n1 = (nfedcum1 > 0);
  if missing(privloan) = 0 then pl = (privloan > 0);
  if missing(stlnamt) = 0  then sl = (stlnamt > 0);
  if missing(inlnamt) = 0  then il = (inlnamt > 0);

* Sum of NPSAS year non-federal loans (missing counted as zero);
  nfedNpsas = sum(of privloan inlnamt stlnamt);
  if missing(nfedNpsas) = 1 then nfedNpsas = 0;

* REPLACE NFEDCUM1 with npsas year non-federal loans IF:;
* 1. The total is less than 50,000 (nfedNpsas);
* 2. The total is less than the self-reported total amount of UG borrowed (n16cugln);
* 3. NFEDCUM1 is currently less than the NPSAS year non-federal total;
* 4. The student is an undergraduate (STYPELST = 1);
* 5. 1 of the following is true ;
*    a. PRIVLOAN is positive, and came from student records (ZPRIVLON = 2);
*    b. State loans are positive (STLNAMT);
*    c. Institutional loans are positive (INLNAMT);
  if nfedNpsas < 50000 & missing(nfedNpsas) = 0 & nfedcum1 < nfedNpsas < n16cugln &
    missing(nfedcum1) = 0 & stypelst = 1 &
    ((zprivlon = 2 & privloan > 0) | sl = 1 | il = 1) then
      do;
        nfed1r = 4;
        nfedcum1 = nfedNpsas
      end;

* REPLACE NFEDCUM1 with npsas year non-federal loans IF:;
* 1. Federal cumulative UG loans is zero (FEDCUM1 = 0);
* 2. The total NPSAS year non-federal amount is less than the self-reported total amount of
UG borrowed (n16cugln);
* 3. The student is an undergraduate (STYPELST = 1);
* 4. 1 of the following is true ;
*   a. PRIVLOAN is positive, and came from the interview (ZPRIVLON = 1);
*   b. State loans are positive (STLNAMT);
*   c. Institutional loans are positive (INLNAMT)    if fedcum1 = 0 & nfedcum1 < nfedNpsas
&;
  if fedcum1 = 0 & nfedcum1 < nfedNpsas &
    missing(nfedcum1) = 0 & stypelst = 1 &
    ((zprivlon = 1 & privloan > 0) | sl = 1 | il = 1) then
      do;
        nfed1r = 4;
        nfedcum1 = nfedNpsas;
      end;

* EDIT 5: Compare those without valid value for total UG loans to FEDCUM1. If within 10%,
then set to zero (double count);
  if missing(n16cugln) = 1 & fedcum1 > 0 & nfedcum1 > 0 &
```

```
    missing(nfedcum1) = 0 then double1 = (100 * (abs(nfedcum1 - fedcum1) / fedcum1) <=
10);
  if double1 = 1 then
   do;
    nfed1r = 5;
    nfedcum1 = 0;
   end;

* EDIT 6: The NFEDCUM1 values from abbreviated interviews, or where the student left the
total UGLN missing;
* If the value is the same or nearly the same as the NPSAS year non-federal loan total, then
we'll keep it, else set it to missing and impute;
  if stypelst = 1 & nfedcum1 > 0 & nfedNpsas > 0 &
    missing(n16cugln) = 1 then match = (100 * (abs(nfedcum1 - nfedNpsas) / nfedNpsas)
<= 10);
  if match = . & stypelst = 1 & nfedcum1 > 0 & (nfednpsas = 0 | missing(nfednpsas) = 1) &
missing(n16cugln) = 1 then match = 0;
  if match = 0 then
   do;
    nfed1r = 6;
    nfedcum1 = .x;
   end;
  if match = 1 then
   do;
    nfedcum1 = nfedNpsas;
   end;

* EDIT 7: Use the self-reported total UG loan from the interview to calculate NFEDCUM1 if
possible;
  if n16cugln > 0 & missing(nfedcum1) = 1 & n16cugln < 100000 then
   do;
    nfed1r = 7;
    if n16cugln >= fedcum1 & missing(fedcum1) = 0 then
     do;
      nfedcum1 = n16cugln - fedcum1;
     end;
    if n16cugln < fedcum1 then
     do;
      nfedcum1 = 0;
      end;
    end;

* EDIT 8: Replace NFEDCUM1 with missing and set a minimum imputation flag if the
student said they did not receive any non-federal loans for UG education, but student
records said they did in the NPSAS year (private, state, or institution loans).;
  if nfedcum1 = 0 & nfedNpsas > 0 & stypelst = 1 &
    ((zprivlon = 2 & privloan > 0) | sl = 1 | il = 1) & nfedNpsas < 100000 then
    do;
     nfed1r = 8;
     nfedcum1 = .x;
     nfedcum1_minimp = nfedNpsas;
    end;
```

```
* EDIT 9: Student reported Private loans in NPSAS year, but none for Cumulative amount
UG. (only for UG). Need to EDIT these but only if the Total they reported (N16CUGLN)
minus FEDCUM1 is greater than the Private loan amount they reported.;
  if nfedcum1 = 0 & stypelst = 1 & (zprivlon = 1 & privloan > 0) & (n16cugln - fedcum1 >
privloan) then
    do;
      nfed1r = 9;
      nfedcum1 = .x;
      nfedcum1_minimp = nfedNpsas;
    end;

 * EDIT 10: Re-edit values less than $50;
  if 1 <= nfedcum1 <= 49 then
    do;
      nfed1r = 1;
      nfedcum1 = 0;
    end;

  if nfedcum1 = .          then nfedcum1 = .x;
  if (pl = 1 | sl = 1 | il = 1) & stypelst = 1 & missing(nfedcum1) = 1 &
missing(nfedcum1_minimp) = 1 then nfedcum1_minimp = nfedNpsas;
  if missing(nfedcum1) = 0 then nfedcum1_minimp = nfedcum1;
  if nfedcum1_minimp = .   then nfedcum1_minimp = 0;
```

<table>
<tr><td><b>Variable Name</b></td><td><b>NFEDCUM2</b></td></tr>
<tr><td><b>Variable Label</b></td><td>Cumulative nonfederal loan amount for graduate education</td></tr>
<tr><td><b>Description</b></td><td>Cumulative nonfederal/private loan amount borrowed for graduate education through June 30, 2016.</td></tr>
<tr><td><b>Assigned Values</b></td><td><table><tr><th>Value</th><th>Value label</th></tr><tr><td><b>0</b></td><td>{Zero}</td></tr><tr><td><b>c</b></td><td>{Continuous}</td></tr></table></td></tr>
<tr><td><b>Data Source(s)</b></td><td>NPSAS:16 Interview, NSLDS:16, NPSAS:16 Student Records</td></tr>
<tr><td><b>Component Derived Variables</b></td><td>STYPELST, BORAMT2, FEDCUM2</td></tr>
<tr><td><b>Applies to</b></td><td>All graduate study members</td></tr>
<tr><td><b>Availability</b></td><td>NPSAS:08, NPSAS:12, NPSAS:16</td></tr>
<tr><td><b>Change History</b></td><td>Unchanged from NPSAS:08 through NPSAS:16.</td></tr>
<tr><td><b>N:16 Usage Statistics</b></td><td>Number of uses: 0 (UG); 8 (GR)</td></tr>
</table>

Programming Narrative

1. Since this is a graduate level variable, set to missing for undergraduates.
2. Primary source is interview item, n16cgrprvt.
3. Triangulate using federal loan amounts (FEDCUM2), and total amount borrowed for graduate education from the interview (n16cgrln):
   a. If cumulative federal loan amount for grad (FEDCUM2) is greater than the cumulative amount borrowed for grad from the interview (n16cgrln) and the private amount is greater than zero (n16cgrprvt > 0), then set private amount to missing and impute.
   b. Else if the private loan amount value is greater than the total minus the federal amount (private > total – federal), adjust private loan amount downwards such that private amount = total – federal.
4. Check: for graduates (STYPELST > 1), ensure the cumulative amount is greater than or equal to the private amount borrowed in the NPSAS year (n16cgrprvt >= PRIVLOAN). Else, set the cumulative amount value to missing.
5. Use value from the interview, as per criteria above. Else if missing, impute using private amount categorical estimate if possible [n16cgrprvest]).

SAS Code

```
  if n16cgrprvt >= 0 & stypelst > 1 then do; nfedcum2 = n16cgrprvt; end;
* EDIT 1: Set loans less than 50 to zero;
  if 1 <= nfedcum2 <= 49 then
   do;
    nfed2r = 1;
    nfedcum2 = 0;
   end;

* EDIT 2: Set nfedcum2 to zero if the self-reported GR total is within 5 percent of the NSLDS total for UG loans (FEDCUM2).;
  if nfedcum2 > 0 & n16cgrln > 0 & fedcum2 > 0 then nfedzero2 = (100 * (abs(n16cgrln - fedcum2) / n16cgrln) < 5);
  if nfedzero2 = 1 then
   do;
    nfed2r = 2;
    nfedcum2 = 0;
   end;

* EDIT 3: Adjust nfedcum2 downward if GRTOT is greater than the self-reported total n16cgrln;
* In THEORY, if the student and NSLDS are perfectly reported n16cgrln = nfedcum2 + fedcum2;
  * Calculate total GR borrowed using unedited nfedcum2 and fedcum2.;
  if missing(nfedcum2) = 0 & missing(fedcum2) = 0 then grtot = nfedcum2 + fedcum2;
* Calculate the difference between self-reported TOTAL UG loans borrowed and the NSLDS fedcum2 (note, will often be negative).;
* Do not include those we already flagged and fixed as double counted;
  if grtot > n16cgrln & nfedcum2 > 0 & grtot > 0 & n16cgrln > 0 & nfed2r ~= 2 then nfedcum2_new = n16cgrln - fedcum2;
  if missing(nfedcum2_new) = 0 then nfed2r = 3;
  if nfedcum2_new < 0 & missing(nfedcum2_new) = 0  then
   do;
    nfedcum2 = 0;
   end;
  if nfedcum2_new >= 0 & missing(nfedcum2_new) = 0 then
   do;
    nfedcum2 = nfedcum2_new; used*/
    end;

* EDIT 4: For Grads (STYPELST > 1): If PRIVLOAN+STLNAMT+INLNAMT is greater than
nfedcum2, adjust;
  if missing(nfedcum2) = 0 then n12 = (nfedcum2 > 0);
  if missing(privloan) = 0 then pl2 = (privloan > 0);
  if missing(stlnamt) = 0  then sl2 = (stlnamt > 0);
  if missing(inlnamt) = 0  then il2 = (inlnamt > 0);

* Sum of NPSAS year non-federal loans (missing counted as zero);
  nfedNpsas2 = sum(of privloan inlnamt stlnamt);
  if missing(nfedNpsas2) = 1 then nfedNpsas2 = 0;
* REPLACE nfedcum2 with npsas year non-federal loans IF:;
* 1. The total is less than 50,000 (nfedNpsas2);
* 2. The total is less than the self-reported total amount of GR borrowed (n16cgrln);
* 3. nfedcum2 is currently less than the NPSAS year non-federal total;
* 4. The student is an graduate (STYPELST > 1);
* 5. 1 of the following is true ;
*   a. PRIVLOAN is positive, and came from student records (ZPRIVLON = 2);
*   b. State loans are positive (STLNAMT);
*   c. Institutional loans are positive (INLNAMT);
  if nfedNpsas2 < 50000 & missing(nfedNpsas2) = 0 & nfedcum2 < nfedNpsas2 < n16cgrln &
    missing(nfedcum2) = 0 & stypelst > 1 &
    ((zprivlon = 2 & privloan > 0) | sl2 = 1 | il2 = 1) then
     do;
       nfed2r = 4;
       nfedcum2 = nfedNpsas2;
     end;

* REPLACE nfedcum2 with npsas year non-federal loans IF:;
* 1. Federal cumulative GR loans is zero (fedcum2 = 0);
* 2. The total NPSAS year non-federal amount is less than the self-reported total amount of
GR borrowed (n16cgrln);
* 3. The student is an graduate (STYPELST > 1);
* 4. 1 of the following is true ;
*   a. PRIVLOAN is positive, and came from the interview (ZPRIVLON = 1);
*   b. State loans are positive (STLNAMT);
*   c. Institutional loans are positive (INLNAMT);
  if fedcum2 = 0 & nfedcum2 < nfedNpsas2 &
    missing(nfedcum2) = 0 & stypelst > 1 &
    ((zprivlon = 1 & privloan > 0) | sl2 = 1 | il2 = 1) then
     do;
       nfed2r = 4;
       nfedcum2 = nfedNpsas2;
     end;

* EDIT 5: Compare those without valid value for total GR loans to fedcum2. If within 10%, then
set to zero (double count).;
  if missing(n16cgrln) = 1 & fedcum2 > 0 & nfedcum2 > 0 &
    missing(nfedcum2) = 0 then double2 = (100 * (abs(nfedcum2 - fedcum2) / fedcum2) <=
10);
  if double2 = 1 then
```

```
    do;
      nfed2r = 5;
      nfedcum2 = 0;
    end;

* EDIT 6: The NFEDCUM2 values from abbreviated interviews, or where the student left the
total GRLN missing;
* If the value is the same or nearly the same as the NPSAS year non-federal loan total, then
we'll keep it, else set it to missing and impute;
  if stypelst > 1 & nfedcum2 > 0 & nfedNpsas2 > 0 &
     missing(n16cgrln) = 1 then match2 = (100 * (abs(nfedcum2 - nfedNpsas2) / nfedNpsas2)
<= 10);
  if match2 = . & stypelst > 1 & nfedcum2 > 0 & (nfednpsas2 = 0 | missing(nfednpsas2) = 1) &
missing(n16cgrln) = 1 then match2 = 0;
  if match2 = 0 then
    do;
      nfed2r = 6;
      nfedcum2 = .x;
    end;
  if match2 = 1 then
    do;
      nfedcum2 = nfedNpsas2;
    end;

* EDIT 7: Use the self-reported total UG loan from the interview to calculate nfedcum2 if
possible;
  if n16cgrln > 0 & missing(nfedcum2) = 1 & n16cgrln < 150000 & stypelst > 1 then
    do;
      nfed2r = 7;
      if n16cgrln >= fedcum2 & missing(fedcum2) = 0 then
        do;
          nfedcum2 = n16cgrln - fedcum2;
        end;
      if n16cgrln < fedcum2 then
        do;
          nfedcum2 = 0;
        end;
    end;

* EDIT 8: Replace nfedcum2 with missing, and set a minimum imputation flag if the student
said they did not receive any non-federal loans for UG education, but student records said
they did in the NPSAS year (private, state, or institution loans).;
  if nfedcum2 = 0 & nfedNpsas2 > 0 & stypelst > 1 &
     ((zprivlon = 2 & privloan > 0) | sl2 = 1 | il2 = 1) & nfedNpsas2 < 100000 then
     do;
       nfed2r = 8;
       nfedcum2 = .x;
       nfedcum2_minimp = nfedNpsas2;
     end;
```

```
* EDIT 9: Student reported Private loans in NPSAS year, but none for Cumulative amount GR.
(only for GR). Need to EDIT these but only if the Total they reported (n16cgrln) minus
fedcum2 is greater than the Private loan amount they reported.;
  if nfedcum2 = 0 & stypelst > 1 & (zprivlon = 1 & privloan > 0) & (n16cgrln - fedcum2 >
privloan) then
    do;
      nfed2r = 9;
      nfedcum2 = .x;
      nfedcum2_minimp = nfedNpsas2;
    end;

* EDIT 10: Re-edit values less than $50;
  if 1 <= nfedcum2 <= 49 then
    do;
      nfed2r = 1;
      nfedcum2 = 0;
    end;

  if stypelst = 1 then
    do;
      nfedcum2 = .y;
      nfedcum2_minimp = .y;
    end;
  else
    do;
      if nfedcum2 = .          then nfedcum2 = .x;
      if (pl2 = 1 | sl2 = 1 | il2 = 1) & missing(nfedcum2) = 1 & missing(nfedcum2_minimp) = 1
then nfedcum2_minimp = nfedNpsas2;
      if missing(nfedcum2) = 0 then nfedcum2_minimp = nfedcum2;
      if nfedcum2_minimp = .   then nfedcum2_minimp = 0;
    end;
```

<table>
<tr><td><b>Variable Name</b></td><td><b>OTHFDGRT</b></td></tr>
<tr><td><b>Variable Label</b></td><td>Other federal grants (not Title IV) amount</td></tr>
<tr><td><b>Description</b></td><td>Total amount of grants from various small federal grant programs received during the 2015–16 academic year.</td></tr>
<tr><td><b>Assigned Values</b></td><td><table><tr><th>Value</th><th>Value label</th></tr><tr><td><b>0</b></td><td>{Zero}</td></tr><tr><td><b>c</b></td><td>{Continuous}</td></tr></table></td></tr>
<tr><td><b>Data Source(s)</b></td><td>NPSAS:16 Student Records</td></tr>
<tr><td><b>Component Derived Variables</b></td><td>Not applicable</td></tr>
<tr><td><b>Applies to</b></td><td>All study members</td></tr>
<tr><td><b>Availability</b></td><td>NPSAS:96, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16</td></tr>
</table>

Change History	Unchanged from NPSAS:96, NPSAS:04-NPSAS:16. Prior to NPSAS:12, OTHFDGRT included additional federal programs such as the Academic Competitiveness Grant, SMART (National Science and Mathematics Access to Retain Talent) grants, and Robert C. Byrd honors scholarships. These grants were discontinued prior to 2011–12.
N:16 Usage Statistics	Number of uses: 0 (UG), 2 (GR)
Programming Narrative	If study member has a value for total amount of aid from student records (cfacvans) then sum the amounts received in non-Title IV federal grants as follows: 1. Bureau of Indian Affairs = CFAGOVAMT; Where CFAGOVTYP = 6 2. DC tuition assistance grant = CFAGOVAMT ; Where CFAGOVTYP = 8 3. Federal fellowship = CFAGRAMT; Where CFAGRTYP = 3 4. Other aid (federal grants) = CFAOTHAMT; Where CFAOTHSRC = 3 & CFAOTHTYP in (1, 2, 3, 4, 17) 5. Iraq & Afghanistan Service grant = CFAIRAQ 6. Federal traineeship = CFAGRAMT; Where CFAGRTYP = 4 7. Other aid (federal traineeship) = CFAOTHAMT; Where CFAOTHSRC = 3 & CFAOTHTYP = 13

SAS Code

```
* Source variable cfacvans="Received any financial aid";
* Source variable cfairaq="Iraq & Afghanistan Service Grant";
* Source variables cfaothgov="Received any private or other government aid";
* Source variables cfa1govamt-cfa3govamt="Private aid program amount";
* Source variables cfa1govtyp-cfa3govtyp="Private aid program type";
* Source variables cfagraid="Received any graduate aid";
* Source variables cfagramt01-cfagramt03="Graduate aid program amount";
* Source variables cfagrtyp01-cfagrtyp03="Graduate aid program type";
* Source variables cfaothaid="Received any other aid";
* Source variables cfa1othamt–cfa3othamt="Other aid amount";
* Source variables cfa1othsrc-cfa3othsrc="Other aid source";
* Source variables cfa1othtyp-cfa3othtyp="Other aid type";
****************************************************************************;
array govamt(3)   govamt1-govamt3;
  array srgovamt(3) cfa1govamt cfa2govamt cfa3govamt;
  array srgovtyp(3) cfa1govtyp cfa2govtyp cfa3govtyp;
  array gramt(3)    gramt1-gramt3;
  array srgramt(3)  cfagramt01 cfagramt02 cfagramt03;
  array srgrtyp(3)  cfagrtyp01 cfagrtyp02 cfagrtyp03;
  array othamt(3)   othamt1-othamt3;
  array srothamt(3) cfa1othamt cfa2othamt cfa3othamt;
  array srothsrc(3) cfa1othsrc cfa2othsrc cfa3othsrc;
  array srothtyp(3) cfa1othtyp cfa2othtyp cfa3othtyp;
  do i = 1 to 3;
   * Bureau of Indian Affairs = CFAGOVAMT where CFAGOVTYP = 6;
   * DC tuition assistance grant = CFAGOVAMT where CFAGOVTYP = 8;
   govamt(i) = 0;
   if cfaothgov ~= 0 then
    do;
     if srgovtyp(i) in (6,8) then
      do;
       govamt(i) = srgovamt(i);
       * There are people who have other federal grants in the interim
       * variables OTHAMT, but the amounts themselves are
       * missing (-9). For these we should add up the amounts they do
       * have from other component variables (if any) and treat that
       * value as the floor and impute a higher positive value for this
       * variable;
       if missing(govamt(i)) = 1 then govamt(i) = .m;
      end;
     else if cfaothgov = .x then govamt(i) = .x;
    end;
   * Federal fellowship = CFAGRAMT where CFAGRTYP= 3;
   * Federal traineeship = CFAGRAMTwhere CFAGRTYP = 4;
   gramt(i) = 0;
   if cfagraid ~= 0 then
    do;
     if srgrtyp(i) in (3,4) then
      do;
       gramt(i) = srgramt(i);
       if missing(gramt(i)) = 1 then gramt(i) = .m;
      end;
     else if cfagraid = .x then gramt(i) = .x;
    end;
   * Other aid (federal grants) = CFAOTHAMT where CFAOTHSRC = 3 & CFAOTHTYP in
(1, 2, 3, 4, 17);
   * Other aid (federal traineeship) = CFAOTHAMT where CFAOTHSRC = 3 & CFAOTHTYP
= 13;
   othamt(i) = 0;
   if cfaothaid ~= 0 then
    do;
     if srothsrc(i) = 3 & srothtyp(i) in (1,2,3,4,13,17) then
      do;
       othamt(i) = srothamt(i);
       if missing(othamt(i)) = 1 then othamt(i) = .m;
      end;
     else if cfaothaid = .x then othamt(i) = .x;
    end;
  end;
  * If study member has a value for total amount of aid from student records
  * (cfacvans) then sum up the amount of federal grants that don't have
  * their own derived variable:;
  array forOthfdgrt(9) govamt1 govamt2 govamt3 gramt1 gramt2 gramt3 othamt1 othamt2
othamt3;
  othfdgrt = govamt1 + govamt2 + govamt3 + gramt1 + gramt2 + gramt3 + othamt1 + othamt2
+ othamt3 + cfairaq;
  othfdgrt_minimp = sum(of govamt1 govamt2 govamt3 gramt1 gramt2 gramt3 othamt1
othamt2 othamt3 cfairaq);
  if othfdgrt_minimp = . then othfdgrt_minimp = 0;
  do i = 1 to 9;
    if forOthfdgrt(i) = .m then Othfdgrt_minimp = Othfdgrt_minimp + 1;
  end;
  if missing(othfdgrt) = 1 then
   do;
    othfdgrt = .x;
   end;
```

<table>
<tr><td><b>Variable Name</b></td><td><b>OTHGTAMT</b></td></tr>
<tr><td><b>Variable Label</b></td><td>Outside grants (private & employer) amount</td></tr>
<tr><td><b>Description</b></td><td>Total amount of grants received during the 2015–16 academic year from outside private sources (PRIVAID) or employers (EMPLYAM3).</td></tr>
<tr><td><b>Assigned Values</b></td><td><table><tr><th>Value</th><th>Value label</th></tr><tr><td><b>0</b></td><td>{Zero}</td></tr><tr><td><b>c</b></td><td>{Continuous}</td></tr></table></td></tr>
<tr><td><b>Data Source(s)</b></td><td>NPSAS:16 Student Records, NPSAS:16 Interview</td></tr>
<tr><td><b>Component Derived Variables</b></td><td>PRIVAID, EMPLYAM3</td></tr>
<tr><td><b>Applies to</b></td><td>All study members</td></tr>
<tr><td><b>Availability</b></td><td>NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16</td></tr>
<tr><td><b>Change History</b></td><td>Unchanged from NPSAS:96 through NPSAS:16.</td></tr>
<tr><td><b>N:16 Usage Statistics</b></td><td>Number of uses: 75 (UG); 1 (GR)</td></tr>
<tr><td><b>Programming Narrative</b></td><td>The sum of private sources grants and employer aid (student and parents)</td></tr>
<tr><td><b>SAS Code</b></td><td>*Source variable privaid="Private sources grants";
*Source variable emplyam3="Employer aid (student & parents) ";
*Applies to: all students;
othgtamt = privaid + emplyam3;
if missing(othgtamt) = 1 then othgtamt = .x;</td></tr>
</table>

<table>
<tr><td><b>Variable Name</b></td><td><b>OTHRSCR</b></td></tr>
<tr><td><b>Variable Label</b></td><td>Outside sources aid amount</td></tr>
<tr><td><b>Description</b></td><td>Total amount of aid from outside sources received during the 2015–16 academic year.</td></tr>
<tr><td><b>Assigned Values</b></td><td><table><tr><th>Value</th><th>Value label</th></tr><tr><td><b>0</b></td><td>{Zero}</td></tr><tr><td><b>c</b></td><td>{Continuous}</td></tr></table></td></tr>
<tr><td><b>Data Source(s)</b></td><td>NPSAS:16 Student Records, NPSAS:16 Interview, VBA:16</td></tr>
<tr><td><b>Component Derived Variables</b></td><td>OTHGTAMT, PRIVLOAN, VADODAMT</td></tr>
<tr><td><b>Applies to</b></td><td>All study members</td></tr>
<tr><td><b>Availability</b></td><td>NPSAS:96, NPSAS:2000; NPSAS:04, NPSAS:08; NPSAS:12; NPSAS:16</td></tr>
<tr><td><b>Change History</b></td><td>Unchanged from NPSAS:96 through NPSAS:12; NPSAS:16 includes VBA data.</td></tr>
<tr><td><b>N:16 Usage Statistics</b></td><td>Number of uses: 2 (UG); 1 (GR)</td></tr>
<tr><td><b>Programming Narrative</b></td><td>Sum outside grants from private sources and employers (OTHGTAMT), private commercial or alternative loans (PRIVLOAN), and federal veterans education benefits and military tuition aid (VADODAMT).</td></tr>
</table>

SAS Code

```
* Source variable othgtamt="Outside grants (private & employer)";
* Source variable privloan="Private (alternate loans)";
* Source variable vadodamt="Veterans' benefits and Department of Defense";
   ***************************************************************;
* Sum of outside grants, private commercial or alternative loans, and federal Veterans' benefits
and military tutition aid ;
   othrscr = sum(othgtamt,privloan,vadodamt);
   if missing (othrscr) = 1 then do ;
      othrscr = .x;
     end;
```

Variable Name	**OTHTYPE**
Variable Label	Total other type of aid (Direct PLUS, job training, VA) amount
Description	Total amount of aid received during the 2015–16 academic year not classified by type as grants, loans to students, or work-study.

Assigned Values	**Value** — **Value label**
	0 — {Zero}
	c — {Continuous}

Data Source(s)	NPSAS:16 Student Records, NPSAS:16 Interview, NSLDS:16, VBA:16
Component Derived Variables	PLUSAMT, VOCHELP, VADODAMT, GRASTAMT (GR only), STYPELST
Applies to	All study members
Availability	NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16
Change History	Unchanged from NPSAS:04 through NPSAS:16. TOTOTHR from NPSAS:96-NPSAS:2000.
N:16 Usage Statistics	Number of uses: 6 (UG); 0 (GR)
Programming Narrative	1. If undergraduate study member (STYPELST = 1), then other type of aid is equal to the sum of Direct PLUS Loans to parents (PLUSAMT), state vocational rehabilitation and job training grants (VOCHELP), and federal veterans education benefits and military tuition aid (VADODAMT). 2. If graduate study member, then other type of aid is equal to the sum of total graduate assistantship amount (GRASTAMT) and federal veterans education benefits and military tuition aid (VADODAMT).

SAS Code

```
* Source variable plusamt="Direct PLUS Loans to parents";
* Source variable vochelp="Vocational rehabilitation and training";
* Source variable vadodamt="Veterans' benefits and Department of Defense";
* Source variable grastamt="Total assistantships amount";
  ***************************************************************;
* If undergraduate study member (STYPELST = 1), then other type of aid is
* equal to the sum of Direct PLUS Loans to parents (PLUSAMT), state
* vocational rehabilitation and job training grants (VOCHELP) and federal
* Veterans' benefits and military tuition aid (VADODAMT).
* If graduate study member, then other type of aid is equal to the sum of
* total graduate assistantship amount (GRASTAMT) and federal veterans'
* benefits and military tuition aid (VADODAMT);
  if stypelst = 1 then
   do;
    othtype = plusamt + vochelp + vadodamt;
    if missing(othtype) = 1 then
     do;
      othtype = .x;
     end;
   end;
  else if stypelst > 1 then
   do;
    othtype = grastamt + vadodamt;
    if missing(othtype) = 1 then
     do;
      othtype = .x;
     end;
   end;
```

<table>
<tr><td><b>Variable Name</b></td><td><b>PELLAMT</b></td></tr>
<tr><td><b>Variable Label</b></td><td>Federal Pell grant amount</td></tr>
<tr><td><b>Description</b></td><td>Total amount of federal Pell grants received at all institutions attended during the 2015–16 academic year.</td></tr>
<tr><td><b>Assigned Values</b></td><td><table><tr><th>Value</th><th>Value label</th></tr><tr><td>0</td><td>{Zero}</td></tr><tr><td>c</td><td>{Continuous}</td></tr></table></td></tr>
<tr><td><b>Data Source(s)</b></td><td>NSLDS:16</td></tr>
<tr><td><b>Component Derived Variables</b></td><td>STYPELST</td></tr>
<tr><td><b>Applies to</b></td><td>All undergraduate study members</td></tr>
<tr><td><b>Availability</b></td><td>NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16</td></tr>
<tr><td><b>Change History</b></td><td>Unchanged from NPSAS:96 through NPSAS:12; NPSAS:16 uses only NSLDS rather than NSLDS and Student Records.</td></tr>
<tr><td><b>N:16 Usage Statistics</b></td><td>Number of uses: 3,141 (UG); 0 (GR)</td></tr>
</table>

Programming Narrative	1. Data management a. Keep only Pell Grants from NSLDS grant dataset (type = "PE") b. Subset to grants that were awarded for the 2016 federal award year only c. Drop grants with an amount paid to date of zero (amt_paid_to_dt = 0). d. Identify duplicate grants using the following variables: student, award year, transaction number, disbursement date, and disbursement amount. Of these duplicates, set one of the total amounts paid to date and percent eligibility used to zero (amt_paid_to_dt = 0 & seg_elig_used = 0) to avoid duplicate counting. 2. Variable derivation a. If study member is a graduate student, set variable to skip (PELLAMT = -3). b. Else if undergraduate student, sum the amount paid to date for all Pell Grants in 2016. c. Annual limit check: compute the annual percent eligibility used (seg_elig_used) by grant recipient, and examine those with greater than 100% used in 2016 (if any). Note: There are only 2 such recipients in a BPS:11 dataset spanning 10 years of potential Pell eligibility.

SAS Code	* Source variable ndpell="Federal Pell grant (NSLDS)";
	* Source variable stypelst="Student type indicator";
	***;
	* If study member is a graduate student set variable to skip (PELLAMT = -3);
	if stypelst > 1 then do; pellamt = .y; end;
	* Else if undergraduate student, sum up the amount paid to date for Pell
	* Grants in 2016;
	else if stypelst = 1 then
	do;
	pellamt = ndpell;
	end;
	* Data edit: Set cases who received $5,776 in Pell grants to the max,
	* which is $5,775;
	if pellamt = 5776 then pellamt = 5775;
	end;

Variable Name	**PERKAMT**
Variable Label	Perkins loan amount
Description	Total amount of federal Perkins loans received at all institutions attended during the 2015–16 academic year.

Assigned Values	**Value** | **Value label**
	0 | {Zero}
	c | {Continuous}

Data Source(s)	NSLDS:16, NPSAS:16 Student Records
Component Derived Variables	STYPELST
Applies to	All study members
Availability	NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16
Change History	Unchanged from NPSAS:96 through NPSAS:16.
N:16 Usage Statistics	Number of uses: 60 (UG), 22 (GR)

Programming Narrative

1. Data management:
   a. Drop loans for students who had ALL loans cancelled.
   b. Subset to loans with a total disbursed amount greater than or equal to $1.
2. Variable derivation:
   a. Determine the federal award year for the period beginning and ending date (begyear & endyear) using per_beg_dt and per_end_dt.
   b. For Perkins loans (loan_type = "PU", "EU", or "NU") with both the period beginning and ending date in the 2016 federal award year (July 1, 2015 - June 30, 2016), add the full "tot_dis" to PERKAMT.
   c. For all other Perkins loans, use the disbursement dates to determine if they fell in the NPSAS year. If dis_dt* is between 01jul2015 and 30jun2016, then add the "amt_diff*" associated with that disbursement date to PERKAMT. Note, this step accommodates situations in which institutions report multiple years of Perkins loans while NSLDS includes only one record.
   d. For students without Perkins Loans in NSLDS, check student records (CPERKINS). Only use this amount if the student also has a CPS record (filed a FAFSA). If this value is greater than the maximum for undergraduates (STYPELST=1 & cperkins>5500) don't use it. If this value is greater than the maximum for graduates (STYPELST>1 & cperkins>8000) don't use it.
   e. Cap this variable at $5,500 for undergraduates (7 people) and at $8,000 for graduates (3 people).

Note: The Perkins Loan program was discontinued in 2015, briefly resurrected, then discontinued again in 2017.

SAS Code

```
* Source variable ndperk="Perkins loan (NSLDS)";
* Source variable cperkins="Perkins Loan (Student Records)";
* Source variable incps="CPS record available";
***************************************************************************;
perkamt = ndperk;
  * For students without Perkins Loans in NSLDS, check student records
  * (CPERKINS). Only use this amount if the student also has a CPS record
  * (filed a FAFSA);
  if perkamt = 0 & cperkins > 0 & incps = 1 then
   do;
    * If this value is greater than the maximum for undergraduates
    * (STYPELST=1 & cperkins>5500) don't use it;
    if stypelst = 1 & cperkins <= 5500 then do; perkamt = cperkins; end;
    * If this value is greater than the maximum for graduates
    * (STYPELST>1 & cperkins>8000) don't use it;
    if stypelst > 1 & cperkins <= 8000 then do; perkamt = cperkins; end;
   end;
  * Cap this variable at $5,500 for undergraduates (7 people) and at $8,000
  * for graduates (3 people);
  if stypelst = 1 & perkamt > 5500 then perkamt = 5500;
  if stypelst > 1 & perkamt > 8000 then perkamt = 8000;
   end;
```

Variable Name	**PLUSAMT**
Variable Label	Direct PLUS Loans to parents amount
Description	Total amount of federal Direct PLUS loans to parents received at all institutions attended during the 2015–16 academic year.

Assigned Values	**Value** **Value label**
	0 {Zero}
	c {Continuous}

Data Source(s)	NSLDS:16
Component Derived Variables	STYPELST
Applies to	All undergraduate study members
Availability	NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16
Change History	Unchanged from NPSAS:2000 through NPSAS:16; PLUSAMT3 in NPSAS:96.
N:16 Usage Statistics	Number of uses: 1,118 (UG); 0 (GR)
Programming Narrative	1. Data management: a. Subset to loans with a total disbursed amount greater than or equal to $1. b. Subset to loans whose enrollment period coincided with the 2016 federal award year (July 1, 2015 - June 30, 2016). 2. Set independent undergraduates (STYPELST = 1 & DEPEND = 2) to zero (PLUSAMT = 0). Independent students' parents cannot get Parent Plus Loans. Set graduate students to skip (STYPELST>1), since no grad students can receive these loans as independents either. 3. Sum the total amount disbursed of Federal Parent PLUS Loans to undergraduate students (loan_type = "D4"). Note, FFELP loans were not awarded after 2010, so they are not included in this variable specification (loan_type = "PL").

SAS Code	`**********************************************************************;`
	`*Source variable depend="Dependency status";`
	`*Source variable ndplus="Direct PLUS Loans to parents (NSLDS)";`
	`*Source variable stypelst="Student type indicator";`
	`**********************************************************************;`
	`  * Set independent undergraduates (STYPELST = 1 & DEPEND = 2) to zero`
	`  * (PLUSAMT = 0). Independent students' parents cannot get Parent Plus Loans;`
	`  if stypelst = 1 & depend = 2 then do; plusamt = 0; end;`
	`  * Set graduate students to skip (STYPELST>1), since no grad students can`
	`  * receive these loans as independents either;`
	`  else if stypelst > 1 then do; plusamt = .y; end;`
	`  else`
	`   do;`
	`    plusamt = ndplus;`
	`   end;`

Variable Name	**PRIVAID**
Variable Label	Private source grants amount
Description	Total amount of grants and scholarships from private outside sources received during the 2015–16 academic year.

Assigned Values	**Value** **Value label**
	0 {Zero}
	c {Continuous}

Data Source(s)	NPSAS:16 Student Records, NPSAS:16 Interview

Component Derived Variables	Not applicable
Applies to	All study members
Availability	NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16
Change History	Unchanged from NPSAS:96 through NPSAS:16.
N:16 Usage Statistics	Number of uses: 205 (UG); 4 (GR)
Programming Narrative	1. Use student records: a. Sum scholarships/grants from private organizations, custom aid: other need-based grant, custom aid: other merit-based grant and custom aid: other grant, both need and merit. b. If this sum is 0 or less use private organization aid amount from the interview. 2. Post-imputation edits: a. Remove double counting: i. If abs(INGRTAMT - PRIVAID)/INGRTAMT<=.05 and (INGRTAMT + PRIVAID>BUDGETAJ and STUDMULT = 1 or INGRTAMT + PRIVAID > $40,000 and STUDMULT>1), then set PRIVAID = 0. ii. If abs(EMPLYAM3 - PRIVAID)/EMPLYAM3=.05 and (EMPLYAM3 + PRIVAID>BUDGETAJ and STUDMULT = 1 or EMPLYAM3 + PRIVAID > $40,000 and STUDMULT>1), then set PRIVAID = 0. b. Cap edits: i. Edit values greater than $200,000 (e.g., such suspect values as $999,999) to zero. ii. Cap at $40,000 if CONTROL = 1, and the PRIVAID amount is greater than the BUDGETAJ (and STUDMULT = 1). Else if STUDMULT>1 and CONTROL = 1, cap at $40,000. iii. Cap at $60,000 if CONTROL>1 and is greater than the BUDGETAJ (and STUDMULT = 1). Else if STUDMULT>1 and CONTROL > 1, cap at $60,000. c. TOTAID and COA edit to EMPLYAM3 and PRIVAID. i. Flag cases where OTHGTAMT (PRIVAID + EMPLYAM3) plus all other aid except private loans and VADODAMT is greater than the cost of attendance by 125%. This is prior to the hard edits to BUDGETAJ, where the non-tuition expenses (BUDNONAJ) are raised to the total aid amount. ii. Reduce both PRIVAID and EMPLYAM3 by the ratio of the difference between total aid (minus PRIVLOAN & VADODAMT) and the BUDGETAJ divided by OTHGTAMT, if the case was flagged in the previous step.
SAS Code	* Source variables cfa1govamt–cfa3govamt="Private aid program amount"; * Source variables cfa1govtyp-cfa3govtyp="Private aid program type"; * Source variables cfaothgov="Received any private or other government aid"; * Source variables cfa1othamt-cfa3othamt="Other aid amount"; * Source variables cfa1othsrc-cfa3othsrc="Other aid source"; * Source variables cfa1othtyp-cfa3othtyp="Other aid type"; * Source variables cfaothaid="Received any other aid"; * Source variable n16caidprv1="Undergraduate aid 2015–16: scholarship from private organization"; * Source variable n16caidprv2="Graduate aid 2015–16: scholarship from private organization"; * Source variable n16cprvgtamt="Aid amount: scholarships from private organizations"; **; array othamta(3) othamta1-othamta3; * array srothamt(3) cfa1othamt cfa2othamt cfa3othamt; * array srothsrc(3) cfa1othsrc cfa2othsrc cfa3othsrc;

```
  * array srothtyp(3) cfa1othtyp cfa2othtyp cfa3othtyp;
  array govamta(3) govamta1-govamta3;
  * array srgovamt(3) cfa1govamt cfa2govamt cfa3govamt;
  * array srgovtyp(3) cfa1govtyp cfa2govtyp cfa3govtyp;
  do i = 1 to 3;
    othamta(i) = 0;
    if cfaothaid ~= 0 then
     do;
      if srothsrc(i) = 4 & srothtyp(i) in (1, 2, 3, 4, 5, 17) then
       do;
        othamta(i) = srothamt(i);
        if missing(othamta(i)) = 1 then othamta(i) = .m;
       end;
      else if cfaothaid = .x then othamta(i) = .x;
     end;

    govamta(i) = 0;
    if cfaothgov ~= 0 then
     do;
      if srgovtyp(i) = 2 then
       do;
        govamta(i) = srgovamt(i);
        if missing(govamta(i)) = 1 then govamta(i) = .m;
       end;
      else if cfaothgov = .x then govamta(i) = .x;
     end;
  end;

  array forPrivaid(6) othamta1 othamta2 othamta3 govamta1 govamta2 govamta3;
  do i = 1 to 6;
    if forPrivaid(i) > 0 then
     do j = 1 to 6;
      if forPrivaid(j) = .x then forPrivaid(j) = 0;
     end;
  end;

  * Use student records;
if othamta1 + othamta2 + othamta3 + govamta1 + govamta2 + govamta3 > 0 then
   do;
    privaid = othamta1 + othamta2 + othamta3 + govamta1 + govamta2 + govamta3;
   end;
  * If this sum is 0 or less use private organization aid amount from the interview;
  else if n16cprvgtamt > 0 then
   do;
    privaid = n16cprvgtamt;
    end;
  else if sum(of othamta1 othamta2 othamta3 govamta1 govamta2 govamta3) > 0 | othamta1
= .m | othamta2 = .m | othamta3 = .m |  govamta1 = .m | govamta2 = .m | govamta3 = .m then
   do;
    privaid = othamta1 + othamta2 + othamta3 + govamta1 + govamta2 + govamta3;
   if missing(privaid) = 1 then do; privaid = .x; end;
  end;
  else if othamta1 + othamta2 + othamta3 + govamta1 + govamta2 + govamta3 = 0 then
   do;
    privaid = 0;
    end;
  else if n16cprvgtamt = 0 | n16caidprv1 = 0 | n16caidprv2 = 0 then
   do;
    privaid = 0;
   end;
  else
   do;
    privaid = othamta1 + othamta2 + othamta3 + govamta1 + govamta2 + govamta3;
    if missing(privaid) = 1 then do; privaid = .x; end;
   end;
```

Variable Name	**PRIVLOAN**
Variable Label	Private (alternative) loans amount
Description	Total amount of private or alternative loans received during the 2015–16 academic year.

Assigned Values	**Value** **Value label**
	0 {Zero}
	c {Continuous}

Data Source(s)	NPSAS:16 Student Records, NPSAS:16 Interview
Component Derived Variables	Not applicable
Applies to	All study members
Availability	NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16
Change History	Unchanged from NPSAS:2000 through NPSAS:12. Name change from OTHLNAMT (NPSAS:96) to PRIVLOAN (NPSAS:2000).
N:16 Usage Statistics	Number of uses: 619 (UG); 122 (GR)
Programming Narrative	1. Raw interview data were passed through a program to identify double-counted values, including private loans. 2. Raw student records data were passed through a similar program to identify double-counted values. 3. First use student records if student records value is positive 4. If student records value is zero or missing, use student interview

SAS Code

```
* Source variable privloan_sr="Private (alternative) loans (Student Records)";
* Source variable iloandup="Duplicate flag: Institutional loan";
* Source variable sloandup="Duplicate flag: State loan";
* Source variable n16cprvln="Took out private loans in academic year";
* Source variable n16cprvamt="Amount borrowed in private loans in 2015–16 academic year";
************************************************************;
  * If privloan_sr>0 then PRIVLOAN = privloan_sr;
  if privloan_sr > 0 then
   do;
    privloan = privloan_sr;
    privloan_minimp = privloan_sr_minimp;
  end;
* Else if n16cprvamt>0 & iloandup ne 1 and sloandup ne 1 then PRIVLOAN = n16cprvamt;
else if n16cprvamt > 0 & iloandup ~= 1 & sloandup ~= 1 then
  do;
    privloan = n16cprvamt;
    privloan_minimp = n16cprvamt;
  end;
* Else if sum of student records values is positive or at least one value
* was missing (.m) indicating we need to impute a positive value, then
* PRIVLOAN = privloan_sr, and set the minimum imputation flag accordingly
* (add 1 for every .m value);
else if privloan_sr_minimp > 0 then
  do;
    privloan = privloan_sr;
    privloan_minimp = privloan_sr_minimp;
    if missing(privloan) = 1 then do; privloan = .x; end;
  end;
* Else if n16cprvamt = 0 OR n16cprvln = 0, then PRIVLOAN = 0;
else if n16cprvamt = 0 | n16cprvln = 0 then
  do;
    privloan = 0;
    privloan_minimp = 0;
  end;
* Else if either of the duplicate flags is one (iloandup=1 or sloandup=1)
* and the student records private loan value is zero or missing
* (privloan_sr in [.,0]), then PRIVLOAN = 0;
else if (iloandup = 1 | sloandup = 1) & (missing(privloan_sr) = 1 | privloan_sr = 0) then
  do;
    privloan = 0;
    privloan_minimp = 0;
  end;
else
  do;
    privloan = .x;
    privloan_minimp = 0;
  end;
```

<table>
<tr><td><b>Variable Name</b></td><td><b>SEOGAMT</b></td></tr>
<tr><td><b>Variable Label</b></td><td>Federal Supplemental Educational Opportunity Grant (SEOG) amount</td></tr>
<tr><td><b>Description</b></td><td>Total amount of Supplemental Educational Opportunity Grant (SEOG) received during the 2015–16 academic year.</td></tr>
<tr><td><b>Assigned Values</b></td><td><table><tr><th>Value</th><th>Value label</th></tr><tr><td><b>0</b></td><td>{Zero}</td></tr><tr><td><b>c</b></td><td>{Continuous}</td></tr></table></td></tr>
<tr><td><b>Data Source(s)</b></td><td>NPSAS:16 Student Records</td></tr>
<tr><td><b>Component Derived Variables</b></td><td>Not applicable</td></tr>
</table>

Applies to	All undergraduate study members
Availability	NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16
Change History	Unchanged from NPSAS:96 through NPSAS:16.
N:16 Usage Statistics	Number of uses: 94 (UG); 0 (GR)
Programming Narrative	1. Set graduate students to a skip (SEOGAMT = -3). 2. Use student records to determine the amount of SEOGAMT in the 2015–16 academic year (cfafseog) 3. If missing student records information and amount of Pell Grants is 2015–16 is zero, set SEOGAMT to zero.

SAS Code	* Source variable cfafseog="Federal SEOG grant";
	* Source variable pellamt="Federal Pell grant";
	* Source variable stypelst="Student type indicator";
	***;
	* Set graduate students to a skip (SEOGAMT = -3);
	if stypelst > 1 then do; seogamt = .y; end;
	else if stypelst = 1 then
	do;
	* Use student records to determine the amount of SEOGAMT in the
	* 2015–16 academic year (cfafseog);
	if missing(cfafseog) = 0 then
	do;
	seogamt = cfafseog;
	end;
	* If missing student records information and amount of Pell Grants in
	* 2015–16 is zero, set SEOGAMT to zero.;
	else if missing(cfafseog) = 1 then
	do;
	seogamt = .x;
	end;
	end;

Variable Name	**SMARITAL**
Variable Label	Marital status
Description	Student’s marital status during the 2015–16 academic year.
Assigned Values	**Value** — **Value label** **1** — Single, divorced, or widowed **2** — Married **3** — Separated
Data Source(s)	FAFSA:16, NPSAS:16 Interview, NPSAS:16 Student Records
Component Derived Variables	Not applicable
Applies to	All study members
Availability	NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16
Change History	Unchanged from NPSAS:96 through NPSAS:16
N:16 Usage Statistics	Number of uses: 876 (UG); 18 (GR)

Programming Narrative	1. Take values in the order listed in source field. 2. CPS: a. If C16MARR in (1, 2, 3) then SMARITAL = C16MARR b. Else if C16MARR = 4 then SMARITAL = 1 (divorced or widowed). 3. Interview: a. If N16AMARR in (1, 2, 3) then SMARITAL = N16AMARR b. Else if N16AMARR in (4, 5, 6) then SMARITAL = 1 (divorced, widowed, living with partner in marriage-like relationship). 4. Student records: a. Take as is. AMARITAL (1 - 3).
SAS Code	/*---------- SMARITAL: Student's marital status (3-category) ----------*/ /* variable used: C16MARR (C16MAR) N16AMARR AMARITAL */ /* categories: 1=Single, divorced, or widowed 2=Married 3=Separated */ IF C16MARR IN (1,2,3) THEN SMARITAL=C16MARR; /* CPS */ ELSE IF C16MARR=4 THEN SMARITAL=1; /* CPS */ ELSE IF N16AMARR IN (1,2,3) THEN SMARITAL=N16AMARR; /* student interview */ ELSE IF N16AMARR IN (4,5,6) THEN SMARITAL=1; /* student interview */ ELSE IF AMARITAL>0 THEN SMARITAL=AMARITAL; /* student records */

<table>
<tr><td><b>Variable Name</b></td><td><b>STAFFAMT</b></td></tr>
<tr><td><b>Variable Label</b></td><td>Direct Subsidized and Unsubsidized Loans amount</td></tr>
<tr><td><b>Description</b></td><td>Total amount of Federal Direct Loans (also known as Stafford Loans) received at all institutions attended during the 2015–16 academic year.</td></tr>
<tr><td><b>Assigned Values</b></td><td><table><tr><th>Value</th><th>Value label</th></tr><tr><td>0</td><td>{Zero}</td></tr><tr><td>c</td><td>{Continuous}</td></tr></table></td></tr>
<tr><td><b>Data Source(s)</b></td><td>NSLDS:16</td></tr>
<tr><td><b>Component Derived Variables</b></td><td>STAFSUB, STAFUNSB</td></tr>
<tr><td><b>Applies to</b></td><td>All study members</td></tr>
<tr><td><b>Availability</b></td><td>NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16</td></tr>
<tr><td><b>Change History</b></td><td>Unchanged from NPSAS:96 through NPSAS:12; NPSAS:16 uses only NSLDS rather than NSLDS and Student Records.</td></tr>
<tr><td><b>N:16 Usage Statistics</b></td><td>Number of uses: 649 (UG); 81 (GR)</td></tr>
</table>

Programming Narrative	1. Data management for component variables found in STAFSUB and STAFUNSB. 2. Derive Direct Subsidized Loans (STAFSUB) and Direct Unsubsidized Loans (STAFUNSB). Note that STAFSUB should not apply to graduate students (all zero). 3. If undergraduate (STYPELST =1), add the amount of Direct Subsidized Loans to the amount of Direct Unsubsidized Loans (STAFFAMT = STAFSUM + STAFUNSB) 4. If graduate student (STYPELST > 1), equals Direct Unsubsidized Loans (STAFFAMT = STAFUNSB).

SAS Code	* Source variable stafsub="Direct Subsidized Loans";
	* Source variable stafunsb="Direct Unsubsidized Loans";
	* Source variable stypelst="Student type indicator";
	**;
	* If undergraduate (STYPELST =1), add the amount of Direct Subsidized
	* Loans to the amount of Direct Unsubsidized Loans
	* (STAFFAMT = STAFSUM + STAFUNSB);
	   if stypelst = 1 then do; staffamt = stafsub + stafunsb; end;
	   * If graduate student (STYPELST > 1), equals Direct Unsubsidized Loans
	   * (STAFFAMT = STAFUNSB).;
	   else if stypelst > 1 then do; staffamt = stafunsb; end;

<table>
<tr><td><b>Variable Name</b></td><td><b>STAFSUB</b></td></tr>
<tr><td><b>Variable Label</b></td><td>Direct Subsidized Loans amount</td></tr>
<tr><td><b>Description</b></td><td>Total amount of Federal Direct Subsidized Loans (also known as subsidized Stafford loans) received during the 2015–16 academic year.</td></tr>
<tr><td><b>Assigned Values</b></td><td><table><tr><th>Value</th><th>Value label</th></tr><tr><td><b>0</b></td><td>{Zero}</td></tr><tr><td><b>c</b></td><td>{Continuous}</td></tr></table></td></tr>
<tr><td><b>Data Source(s)</b></td><td>NSLDS:16</td></tr>
<tr><td><b>Component Derived Variables</b></td><td>Not applicable</td></tr>
<tr><td><b>Applies to</b></td><td>All study members in NPSAS:96 through NPSAS:12; all undergraduate study members in NPSAS:16</td></tr>
<tr><td><b>Availability</b></td><td>NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16</td></tr>
<tr><td><b>Change History</b></td><td>Unchanged from NPSAS:96 through NPSAS:12; NPSAS:16 uses only NSLDS rather than NSLDS and Student Records.</td></tr>
<tr><td><b>N:16 Usage Statistics</b></td><td>Number of uses: 250 (UG); 0 (GR)</td></tr>
<tr><td><b>Programming Narrative</b></td><td>1. Data management:
a. Subset to loans with a total disbursed amount greater than or equal to $1.
b. Subset to loans whose enrollment period coincided with the 2016 federal award year (July 1, 2015 - June 30, 2016).
2. Set all graduate students (STYPELST > 1) to a skip. Graduate and professional students were not eligible for subsidized loans in 2015–16 academic year.
3. Sum the total amount disbursed of Federal Direct Subsidized loans (loan_type = "D1"). Note, no FFELP loans were originated past 2010, so they are not included in this variable specification.</td></tr>
</table>

SAS Code	**;
	* Source variable ndstafs="Direct Subsidized Loans (NSLDS)";
	* Source variable stypelst="Student type indicator";
	**;
	* Set all graduate students (STYPELST > 1) to a skip. Graduate and professional students were not eligible for subsidized loans in 2015–16 academic year.;
	if stypelst > 1 then do; stafsub = .y; end;
	else if stypelst = 1 then
	do;
	stafsub = ndstafs;
	end;

<table>
<tr><td><b>Variable Name</b></td><td><b>STAFUNSB</b></td></tr>
<tr><td><b>Variable Label</b></td><td>Direct Unsubsidized Loans amount</td></tr>
<tr><td><b>Description</b></td><td>Total amount of Federal Direct Unsubsidized Loans (also known as unsubsidized Stafford loans) received during the 2015–16 academic year.</td></tr>
<tr><td><b>Assigned Values</b></td><td><table><tr><th>Value</th><th>Value label</th></tr><tr><td><b>0</b></td><td>{Zero}</td></tr><tr><td><b>c</b></td><td>{Continuous}</td></tr></table></td></tr>
<tr><td><b>Data Source(s)</b></td><td>NSLDS:16</td></tr>
<tr><td><b>Component Derived Variables</b></td><td>Not applicable</td></tr>
<tr><td><b>Applies to</b></td><td>All study members</td></tr>
<tr><td><b>Availability</b></td><td>NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16</td></tr>
<tr><td><b>Change History</b></td><td>Unchanged from NPSAS:96 through NPSAS:12; NPSAS:16 uses only NSLDS rather than NSLDS and Student Records.</td></tr>
<tr><td><b>N:16 Usage Statistics</b></td><td>Number of uses: 175 (UG); 106 (GR)</td></tr>
<tr><td><b>Programming Narrative</b></td><td>1. Data management:
a. Subset to loans with a total disbursed amount greater than or equal to $1.
b. Subset to loans whose enrollment period coincided with the 2016 federal award year (July 1, 2015 - June 30, 2016).
2. Sum the total amount disbursed of Federal Direct Unsubsidized loans (loan_type = "D2"). Note, no FFELP loans were originated past 2010, so they are not included in this variable specification.</td></tr>
<tr><td><b>SAS Code</b></td><td>**;
* Source variable ndstafu="Direct Unsubsidized Loans (NSLDS)";
**;
stafunsb = ndstafu;
end;</td></tr>
</table>

Variable Name	**STATEAMT**
Variable Label	State aid amount
Description	Total amount of state aid received during the 2015–16 academic year.

Assigned Values

Value	Value label
0	{Zero}
c	{Continuous}

Data Source(s)	NPSAS:16 Student Records, NPSAS:16 Interview
Component Derived Variables	STGTAMT, STLNAMT, STWKAMT, VOCHELP
Applies to	All study members
Availability	NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16
Change History	Unchanged from NPSAS:96 through NPSAS:16.
N:16 Usage Statistics	Number of uses: 78 (UG); 24 (GR)
Programming Narrative	1. If study member is an undergraduate (STYPELST=1): Sum of state grants, state loans, state work-study, and vocational rehabilitation and training. 2. Else if graduate student: Sum of state grants, state loans, and state work study.

SAS Code

```
* Source variable stgtamt="State grants total";
* Source variable stlnamt="State loans";
* Source variable stwkamt="State work-study";
* Source variable vochelp="Vocational rehabilitation and training";
* Source variable stypelst="Student type indicator";
***************************************************************;
* If study member is an undergraduate (STYPELST=1): Sum of state grants,
* state loans, state work-study and vocational rehabilitation and training.
    if stypelst = 1 then
      do;
        stateamt = stgtamt + stlnamt + stwkamt + vochelp;
        if missing(stateamt) = 1 then
          do;
            stateamt = .x;
          end;
     * Else if graduate student: Sum of state grants, state loans, and state work study.;
    else if stypelst > 1 then
      do;
        stateamt = stgtamt + stlnamt + stwkamt;
        if missing(stateamt) = 1 then
          do;
            stateamt = .x;
          end;
       end;
```

Variable Name	**STATNEED**
Variable Label	State need-based grants amount
Description	Total amount of state need-based grants received during the 2015–16 academic year.

<table>
<tr><td><b>Assigned Values</b></td><td><table><tr><th>Value</th><th>Value label</th></tr><tr><td><b>0</b></td><td>{Zero}</td></tr><tr><td><b>c</b></td><td>{Continuous}</td></tr></table></td></tr>
<tr><td><b>Data Source(s)</b></td><td>NPSAS:16 Student Records</td></tr>
<tr><td><b>Component Derived Variables</b></td><td>STNDMRT, STNDONLY</td></tr>
<tr><td><b>Applies to</b></td><td>All study members</td></tr>
<tr><td><b>Availability</b></td><td>NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16</td></tr>
<tr><td><b>Change History</b></td><td>Unchanged from NPSAS:96 through NPSAS:16. Included in this variable are state grants for California public institutions that are funded by state dollars but allocated by the institutions (e.g., Community College Board of Governors Grants, California State University Grants, and Educational Opportunity Program grants). In past NPSAS studies, these were included as institutional need-based grants (INSTNEED). To compare estimates of state need-based grants over time, use this variable and STATNEED2 from prior NPSAS studies.</td></tr>
<tr><td><b>N:16 Usage Statistics</b></td><td>Number of uses: 81 (UG); 1 (GR)</td></tr>
<tr><td><b>Programming Narrative</b></td><td>Add state need-based aid and other aid state need-based aid. Include aid classified as both need and merit-based aid (STNDMRT + STNDONLY).</td></tr>
<tr><td><b>SAS Code</b></td><td>* Source variable stndmrt="State grants based both on need and merit";
* Source variable stndonly="State need-based grants";
***;
  statneed = stndonly + stndmrt;
  if missing(statneed) = 1 then
   do;
    statneed = .x;
   end;</td></tr>
</table>

<table>
<tr><td><b>Variable Name</b></td><td><b>STATNOND</b></td></tr>
<tr><td><b>Variable Label</b></td><td>State non-need & merit grants amount</td></tr>
<tr><td><b>Description</b></td><td>Total amount of state grants and scholarships not based on need received during the 2015–16 academic year.</td></tr>
<tr><td><b>Assigned Values</b></td><td><table><tr><th>Value</th><th>Value label</th></tr><tr><td><b>0</b></td><td>{Zero}</td></tr><tr><td><b>c</b></td><td>{Continuous}</td></tr></table></td></tr>
<tr><td><b>Data Source(s)</b></td><td>NPSAS:16 Student Records</td></tr>
<tr><td><b>Component Derived Variables</b></td><td>STMERIT, STMILAMT, STVETAMT, STNOND1</td></tr>
<tr><td><b>Applies to</b></td><td>All study members</td></tr>
<tr><td><b>Availability</b></td><td>NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16</td></tr>
<tr><td><b>Change History</b></td><td>Unchanged from NPSAS:96 through NPSAS:16.</td></tr>
<tr><td><b>N:16 Usage Statistics</b></td><td>Number of uses: 12 (UG); 0 (GR)</td></tr>
<tr><td><b>Programming Narrative</b></td><td>Sum of state merit-only grants and state non-need grants, state military, and state veterans education benefits.</td></tr>
</table>

SAS Code	* Source variable stmerit="State merit-only grants";
	* Source variable stmilamt="State military/armed forces grants";
	* Source variable stvetamt="State Veterans' education benefits";
	* Source variable stnond1="State non-need grants";
	***;
	  * Sum of state merit-only grants and state non-need grants;
	  statnond = stmerit + stmilamt + stvetamt + stnond1;
	  if missing(statnond) = 1 then
	   do;
	    statnond = .x;
	   end;

<table>
<tr><td><b>Variable Name</b></td><td><b>STFCUM1</b></td></tr>
<tr><td><b>Variable Label</b></td><td>Cumulative Direct Subsidized and Unsubsidized Loans for undergraduate education</td></tr>
<tr><td><b>Description</b></td><td>Cumulative Direct Subsidized and Unsubsidized Loan (also known as subsidized and unsubsidized Stafford Loans) amount borrowed for undergraduate education through June 30, 2016.</td></tr>
<tr><td></td><td></td></tr>
<tr><td><b>Assigned Values</b></td><td><table><tr><th>Value</th><th>Value label</th></tr><tr><td><b>0</b></td><td>{Zero}</td></tr><tr><td><b>c</b></td><td>{Continuous}</td></tr></table></td></tr>
<tr><td><b>Data Source(s)</b></td><td>NSLDS:16</td></tr>
<tr><td><b>Component Derived Variables</b></td><td>STAFFAMT, BAYEARM, STYPELST
Note: BAYEARM is not included in this report. Information on this variable may be found in the NPSAS:16 graduate student codebook, https://nces.ed.gov/datalab/powerstats/codebook.aspx?dataset=122&type=subject#BAYEARM.</td></tr>
<tr><td><b>Applies to</b></td><td>All study members</td></tr>
<tr><td><b>Availability</b></td><td>NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16</td></tr>
<tr><td><b>Change History</b></td><td>Name changed from STFCUM in NPSAS:96. Unchanged from NPSAS:2000 through NPSAS:16.</td></tr>
<tr><td><b>N:16 Usage Statistics</b></td><td>Number of uses: 94 (UG); 30 (GR)</td></tr>
</table>

Programming Narrative	1. Assign all non-consolidated loans with missing academic levels to undergraduate or graduate based on the following criteria: a. If undergraduate in NPSAS (STYPELST=1), set loans to undergraduate b. For graduate students (STYPELST>1), set to undergraduate if beginning enrollment date of loan is before their bachelor's degree award date (per_beg_dt<BAYEARM); else set loan to graduate-level. 2. Calculate cumulative amount disbursed according to loan types using *tot_dis*, a variable sourced from NSLDS data that represents the total amount disbursed for a loan: a. pastStsbcum1, the cumulative subsidized Stafford loan amount for undergraduate education before the NPSAS year (loan_type in "D1" "SF") if academic level of loan is undergraduate (1<=acad_lvl<=5). b. pastUnsbcum1, the cumulative unsubsidized Stafford loan amount for undergraduate education before the NPSAS year (loan_type in "D2" "SU" "SL") if academic level of loan is undergraduate (1<=acad_lvl<=5). 3. For undergraduates (STYPELST=1): STFCUM1 = pastStsbcum1 + pastUnsbcum1 + STAFFAMT a. STAFFAMT is a derived variable sourced from federalAid1.sas equal to the cumulative loan amount for both subsidized and unsubsidized Stafford loans in the NPSAS year. 4. For graduates (STYPELST>1): STFCUM1 = pastStsbcum1 + pastUnsbcum1. Do not include Stafford loans from the NPSAS year (STAFFAMT).

SAS Code	* Source variable stypelst="Student type indicator";
	* Source variable pastStsbcum1="Cumulative Direct Subsidized Loans for
	* undergrad before NPSAS year";
	* Source variable pastUnsbcum1="Cumulative Direct Unsubsidized Loans for
	* undergrad before NPSAS year";
	* Source variable staffamt="Direct Subsidized and Unsubsidized Loans";
	    **;
	* For undergraduates (STYPELST=1): STFCUM1 = pastStsbcum1 + pastUnsbcum1 + STAFFAMT;
	    if stypelst = 1 then stfcum1 = pastStsbcum1 + pastUnsbcum1 + staffamt;
	* For graduates (STYPELST>1): STFCUM = pastStsbcum1. Do not include
	* Stafford loans from the NPSAS year (STAFFAMT);
	    else if stypelst > 1 then stfcum1 = pastStsbcum1 + pastUnsbcum1;

<table>
<tr><td><b>Variable Name</b></td><td><b>STFCUM2</b></td></tr>
<tr><td><b>Variable Label</b></td><td>Cumulative Direct Subsidized and Unsubsidized Loans for graduate education</td></tr>
<tr><td><b>Description</b></td><td>Cumulative Direct Subsidized and Unsubsidized Loan (also known as subsidized and unsubsidized Stafford Loan) amount borrowed for graduate education through June 30, 2016.</td></tr>
<tr><td><b>Assigned Values</b></td><td><table><tr><th>Value</th><th>Value label</th></tr><tr><td><b>0</b></td><td>{Zero}</td></tr><tr><td><b>c</b></td><td>{Continuous}</td></tr></table></td></tr>
<tr><td><b>Data Source(s)</b></td><td>NSLDS:16</td></tr>
<tr><td><b>Component Derived Variables</b></td><td>STAFFAMT, STYPELST, BAYEARM
Note: BAYEARM is not included in this report. Information on this variable may be found in the NPSAS:16 graduate student codebook, https://nces.ed.gov/datalab/powerstats/codebook.aspx?dataset=122&type=subject#BAYEARM.</td></tr>
</table>

Applies to	All graduate study members
Availability	NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16
Change History	Unchanged from NPSAS:2000 through NPSAS:16.
N:16 Usage Statistics	Number of uses: 0 (UG); 68 (GR)
Programming Narrative	1. Since this is a graduate level variable, set to missing for undergraduates. 2. Assign all non-consolidated loans with missing academic levels to undergraduate or graduate based on the following criteria: a. If undergraduate in NPSAS (STYPELST=1), set loans to undergraduate b. For graduate students (STYPELST>1), set to undergraduate if beginning enrollment date of loan is before their bachelor's degree award date (per_beg_dt<BAYEARM); else set loan to graduate-level. 3. Calculate cumulative amount disbursed according to loan types using tot_dis, a variable sourced from NSLDS data that represents the total amount disbursed for a loan: a. pastStsbcum2, the cumulative subsidized Stafford loan amount for graduate education before the NPSAS year (loan_type in "D1" "SF") if academic level of loan is graduate (6<=acad_lvl<=10). b. pastUnsbcum2, the cumulative unsubsidized Stafford loan amount for graduate education before the NPSAS year (loan_type in "D2" "SU" "SL") if academic level of loan is graduate (6<=acad_lvl<=10). 4. STFCUM2 = pastStsbcum2 + pastUnsbcum2 + STAFFAMT a. STAFFAMT is a derived variable equal to the cumulative loan amount for both subsidized and unsubsidized Stafford loans in the NPSAS year.

SAS Code	* Source variable stypelst="Student type indicator";
	* Source variable pastStsbcum2="Cumulative Direct Subsidized Loans for
	* grad before NPSAS year";
	* Interim variable pastUnsbcum2="Cumulative Direct Unsubsidized Loans for
	* grad before NPSAS year";
	* Source variable staffamt="Direct Subsidized and Unsubsidized Loans";
	***;
	* Applies to: All graduate study members;
	if stypelst = 1 then stfcum2 = .y;
	else if stypelst > 1 then stfcum2 = pastStsbcum2 + pastUnsbcum2 + staffamt;

Variable Name	**STFCUM3**
Variable Label	Cumulative Direct Subsidized & Unsubsidized Loans for undergraduate and graduate education
Description	Cumulative Direct Subsidized and Unsubsidized Loan (also known as subsidized and unsubsidized Stafford Loan) amount borrowed for undergraduate and graduate education through June 30, 2016.

Assigned Values	**Value** — **Value label**
	0 — {Zero}
	c — {Continuous}

Data Source(s)	NSLDS:16
Component Derived Variables	STFCUM1, STFCUM2
Applies to	All graduate study members

Availability	NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16
Change History	Unchanged from NPSAS:2000 through NPSAS:16.
N:16 Usage Statistics	Number of uses: 0 (UG); 0 (GR)
Programming Narrative	Add together STFCUM1, the cumulative Stafford amount for undergraduate education, and STFCUM2, the cumulative Stafford amount for graduate education. Set to missing for undergraduates (STYPELST=1), as it is a graduate level variable. For information on how STFCUM1 and STFCUM2 are derived, see their respective derivations.

SAS Code	* label stfcum3="Cumulative Direct Subsidized & Unsubsidized Loans for undergrad & grad";
	* Source variable stypelst="Student type indicator";
	* Source variable stfcum1="Cumulative Direct Subsidized and Unsubsidized Loans for undergrad";
	* Source variable stfcum2="Cumulative Direct Subsidized and Unsubsidized Loans for grad";
	**;
	* Applies to: All graduate respondents;
	if stypelst = 1 then stfcum3 = .y;
	* Add together STFCUM1, the cumulative Stafford amount for undergraduate
	* education, and STFCUM2, the cumulative Stafford amount for graduate
	* education;
	else if stypelst > 1 then stfcum3 = stfcum1 + stfcum2;

<table>
<tr><td><b>Variable Name</b></td><td><b>STGTAMT</b></td></tr>
<tr><td><b>Variable Label</b></td><td>State grants amount</td></tr>
<tr><td><b>Description</b></td><td>Total amount of state grants, scholarships, and fellowships received during the 2015–16 academic year.</td></tr>
<tr><td><b>Assigned Values</b></td><td><table><tr><th>Value</th><th>Value label</th></tr><tr><td><b>0</b></td><td>{Zero}</td></tr><tr><td><b>c</b></td><td>{Continuous}</td></tr></table></td></tr>
<tr><td><b>Data Source(s)</b></td><td>NPSAS:16 Student Records</td></tr>
<tr><td><b>Component Derived Variables</b></td><td>STATNEED, STATNOND</td></tr>
<tr><td><b>Applies to</b></td><td>All study members</td></tr>
<tr><td><b>Availability</b></td><td>NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16</td></tr>
<tr><td><b>Change History</b></td><td>Unchanged from NPSAS:96 through NPSAS:12. NPSAS:16 eliminated a step that in previous studies had set to zero missing values for students sampled at private for-profit institutions.</td></tr>
<tr><td><b>N:16 Usage Statistics</b></td><td>Number of uses: 687 (UG); 1 (GR)</td></tr>
<tr><td><b>Programming Narrative</b></td><td>Sum state-need based grants and state non-need and merit grants.</td></tr>
</table>

SAS Code	* Source variable statneed="State need-based grants";
	* Source variable statnond="State non-need & merit grants";
	**;
	* Sum state-need based grants and state non-need and merit grants;
	stgtamt = statneed + statnond;
	if missing(stgtamt) = 1 then
	do;
	stgtamt = .x;
	end;

<table>
<tr><td><b>Variable Name</b></td><td><b>STLNAMT</b></td></tr>
<tr><td><b>Variable Label</b></td><td>State loans amount</td></tr>
<tr><td><b>Description</b></td><td>Total amount of state loans received during the 2015–16 academic year.</td></tr>
<tr><td><b>Assigned Values</b></td><td><table><tr><th>Value</th><th>Value label</th></tr><tr><td>0</td><td>{Zero}</td></tr><tr><td>c</td><td>{Continuous}</td></tr></table></td></tr>
<tr><td><b>Data Source(s)</b></td><td>NPSAS:16 Student Records</td></tr>
<tr><td><b>Component Derived Variables</b></td><td>Not applicable</td></tr>
<tr><td><b>Applies to</b></td><td>All study members</td></tr>
<tr><td><b>Availability</b></td><td>NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16</td></tr>
<tr><td><b>Change History</b></td><td>Unchanged from NPSAS:96 through NPSAS:12. NPSAS:16 removed caps on values based on institution control.</td></tr>
<tr><td><b>N:16 Usage Statistics</b></td><td>Number of uses: 67 (UG); 0 (GR)</td></tr>
<tr><td><b>Programming Narrative</b></td><td>Sum student records state loan amounts and Other Aid state loan amounts (type 6)</td></tr>
<tr><td><b>SAS Code</b></td><td>**;
* Source variable cfacvans="Received any financial aid";
* Source variables cf01sttyp-cf03sttyp="State aid program type";
* Source variables c01stamt-c03stamt="State aid program amount";
* Source variables cfa1othamt-cfa3othamt="Other aid amount";
* Source variables cfa1othsrc-cfa3othsrc="Other aid source";
* Source variables cfa1othtyp-cfa3othtyp="Other aid type";
* Source variables sector11="Institution NPSAS sector updated using most
*   current IPEDS - 11 categories";
**;
array stamtd(3) stamtd1-stamtd3;
* array srstamt(3) c01stamt c02stamt c03stamt;
* array srsttyp(3) cf01sttyp cf02sttyp cf03sttyp;
array othamtd(3) othamtd1-othamtd3;
* array srothamt(3) cfa1othamt cfa2othamt cfa3othamt;
* array srothsrc(3) cfa1othsrc cfa2othsrc cfa3othsrc;
* array srothtyp(3) cfa1othtyp cfa2othtyp cfa3othtyp;
do i = 1 to 3;
* Use student records: Sum state loan amounts and Other Aid state loan amounts.
* cstamt if cfsttyp = 6</td></tr>
</table>

```
        * cfaothamt if cfaothsrc = 2 & cfaothtyp = 6;
        stamtd(i) = 0;
        if cfastataid ~= 0 then
          do;
            if srsttyp(i) = 6 then
              do;
                stamtd(i) = srstamt(i);
                if missing(stamtd(i)) = 1 then stamtd(i) = .m;
              end;
            else if cfastataid = .x then stamtd(i) = .x;
          end;

othamtd(i) = 0;
        if cfaothaid ~= 0 then
          do;
            if srothsrc(i) = 2 & srothtyp(i) = 6 then
              do;
                othamtd(i) = srothamt(i);
                if missing(othamtd(i)) = 1 then othamtd(i) = .m;
              end;
            else if cfaothaid = .x then othamtd(i) = .x;
          end;
    end;
    array forStlnamt(6) stamtd1 stamtd2 stamtd3 othamtd1 othamtd2 othamtd3;
    do i = 1 to 6;
        if forStlnamt(i) > 0 then
          do j = 1 to 6;
            if forStlnamt(j) = .x then forStlnamt(j) = 0;
          end;
    end;
    stlnamt = stamtd1 + stamtd2 + stamtd3 + othamtd1 + othamtd2 + othamtd3;
    end;
  if missing(stlnamt) = 1 then
      do;
        stlnamt = .x;
      end;
```

<table>
<tr><td><b>Variable Name</b></td><td><b>STMERIT</b></td></tr>
<tr><td><b>Variable Label</b></td><td>State merit-only grants amount</td></tr>
<tr><td><b>Description</b></td><td>Total amount of state merit-only grants and scholarships received during the 2015–16 academic year.</td></tr>
<tr><td><b>Assigned Values</b></td><td><table><tr><th>Value</th><th>Value label</th></tr><tr><td><b>0</b></td><td>{Zero}</td></tr><tr><td><b>c</b></td><td>{Continuous}</td></tr></table></td></tr>
<tr><td><b>Data Source(s)</b></td><td>NPSAS:16 Student Records</td></tr>
<tr><td><b>Component Derived Variables</b></td><td>Not applicable</td></tr>
</table>

Applies to	All study members
Availability	NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16
Change History	Unchanged from NPSAS:96 through NPSAS:08; Available for graduate students starting in NPSAS:12. NPSAS:16 removed caps on values for undergraduates and for graduate students.
N:16 Usage Statistics	Number of uses: 108 (UG); 0 (GR)
Programming Narrative	Sum student records state aid: merit-based grant (type=2) and state merit-based aid (type=2) reported in other aid field (source=2)
SAS Code	See code below

```
* Source variable cfacvans="Received any financial aid";
* Source variable cf01sttyp-cf03sttyp="State aid program type";
* Source variable c01stamt-c03stamt="State aid program amount";
* Source variables cfa1othamt-cfa3othamt="Other aid amount";
* Source variables cfa1othsrc-cfa3othsrc="Other aid source";
* Source variables cfa1othtyp-cfa3othtyp="Other aid type";
  *************************************************************;
  array stamtb(3) stamtb1-stamtb3;
  * array srstamt(3) c01stamt c02stamt c03stamt;
  * array srsttyp(3) cf01sttyp cf02sttyp cf03sttyp;
  array othamtb(3) othamtb1-othamtb3;
  * array srothamt(3) cfa1othamt cfa2othamt cfa3othamt;
  * array srothsrc(3) cfa1othsrc cfa2othsrc cfa3othsrc;
  * array srothtyp(3) cfa1othtyp cfa2othtyp cfa3othtyp;
  do i = 1 to 3;
   * Sum student records state aid: merit-based grant and other aid: state merit-based grant.
   * cstamt if cfsttyp = 2
   * cfaothamt if cfaothsrc = 2 & cfaothtyp = 2;
   stamtb(i) = 0;
   if cfastataid ~= 0 then
    do;
     if srsttyp(i) = 2 then
      do;
       stamtb(i) = srstamt(i);
       if missing(stamtb(i)) = 1 then stamtb(i) = .m;
      end;
     else if cfastataid = .x then stamtb(i) = .x;
    end;

   othamtb(i) = 0;
   if cfaothaid ~= 0 then
    do;
     if srothsrc(i) = 2 & srothtyp(i) = 2 then
      do;
       othamtb(i) = srothamt(i);
       if missing(othamtb(i)) = 1 then othamtb(i) = .m;
      end;
     else if cfaothaid = .x then othamtb(i) = .x;
    end;
  end;
  array forStmerit(6) stamtb1 stamtb2 stamtb3 othamtb1 othamtb2 othamtb3;
  do i = 1 to 6;
```

```
    if forStmerit(i) > 0 then
     do j = 1 to 6;
      if forStmerit(j) = .x then forStmerit(j) = 0;
     end;
end;
stmerit = stamtb1 + stamtb2 + stamtb3 + othamtb1 + othamtb2 + othamtb3;
if missing(stmerit) = 1 then
 do;
  stmerit = .x;
 end;
```

Variable Name	**STMILAMT**
Variable Label	State military/armed forces grants amount
Description	Total amount of state military/armed forces grants received in 2015–16 academic year.
Assigned Values	

Value	Value label
0	{Zero}
c	{Continuous}

Data Source(s)	NPSAS:16 Student Records
Component Derived Variables	Not applicable
Applies to	All study members
Availability	NPSAS:16
Change History	New variable in NPSAS:16
N:16 Usage Statistics	Number of uses: 0 (UG); 0 (GR)
Programming Narrative	Sum student records state military grants (type 14)
SAS Code	

```
* Source variable cfacvans="Received any financial aid";
* Source variables cf01sttyp-cf03sttyp="State aid program type";
* Source variables c01stamt-c03stamt="State aid program amount";
* Source variables cfa1othamt-cfa3othamt="Other aid amount";
* Source variables cfa1othsrc-cfa3othsrc="Other aid source";
* Source variables cfa1othtyp-cfa3othtyp="Other aid type";
*****************************************************************;
  array stamtg(3) stamtg1-stamtg3;
  * array srstamt(3) c01stamt c02stamt c03stamt;
  * array srsttyp(3) cf01sttyp cf02sttyp cf03sttyp;
  array othamtg(3) othamtg1-othamtg3;
  * array srothamt(3) cfa1othamt cfa2othamt cfa3othamt;
  * array srothsrc(3) cfa1othsrc cfa2othsrc cfa3othsrc;
  * array srothtyp(3) cfa1othtyp cfa2othtyp cfa3othtyp;
  do i = 1 to 3;
    * Sum student records state military grants:
    * ROTC/Armed forces (state)
    * cstamt if cfsttyp = 14
    * cfaothamt if cfaothsrc = 2 & cfaothtyp = 14;
  stamtg(i) = 0;
  if cfastataid ~= 0 then
   do;
    if srsttyp(i) = 14 then
     do;
      stamtg(i) = srstamt(i);
      if missing(stamtg(i)) = 1 then stamtg(i) = .m;
     end;
    else if cfastataid = .x then stamtg(i) = .x;
   end;

  othamtg(i) = 0;
  if cfaothaid ~= 0 then
   do;
    if srothsrc(i) = 2 & srothtyp(i) = 14 then
     do;
      othamtg(i) = srothamt(i);
      if missing(othamtg(i)) = 1 then othamtg(i) = .m;
     end;
    else if cfaothaid = .x then othamtg(i) = .x;
   end;
end;
array forStmilamt(6) stamtg1 stamtg2 stamtg3 othamtg1 othamtg2 othamtg3;
do i = 1 to 6;
  if forStmilamt(i) > 0 then
   do j = 1 to 6;
    if forStmilamt(j) = .x then forStmilamt(j) = 0;
   end;
end;
stmilamt = stamtg1 + stamtg2 + stamtg3 + othamtg1 + othamtg2 + othamtg3;
if missing(stmilamt) = 1 then
 do;
  stmilamt = .x;
 end;
```

Variable Name	**STNDMRT**
Variable Label	State grants based both on need and merit amount
Description	Total amount of state grants based both on need and merit during the 2015–16 academic year.
Assigned Values	
Data Source(s)	NPSAS:16 Student Records
Component Derived Variables	Not applicable
Applies to	All study members
Availability	NPSAS:16

Value	Value label
0	{Zero}
c	{Continuous}

Change History	New variable in NPSAS:16
N:16 Usage Statistics	Number of uses: 1 (UG); 0 (GR)
Programming Narrative	Add together state grants that are both need and merit-based and other aid with source "state" and type grants that are both need and merit-based.

SAS Code

```
* Source variable cfacvans="Received any financial aid";
* Source variable cf01sttyp-cf03sttyp="State aid program type";
* Source variable c01stamt-c03stamt="State aid program amount";
* Source variables cfa1othamt-cfa3othamt="Other aid amount";
* Source variables cfa1othsrc-cfa3othsrc="Other aid source";
* Source variables cfa1othtyp-cfa3othtyp="Other aid type";
  ****************************************************************;
  array stamtf(3) stamtf1-stamtf3;
  array srstamt(3) c01stamt c02stamt c03stamt;
  array srsttyp(3) cf01sttyp cf02sttyp cf03sttyp;
  array othamtf(3) othamtf1-othamtf3;
  array srothamt(3) cfa1othamt cfa2othamt cfa3othamt;
  array srothsrc(3) cfa1othsrc cfa2othsrc cfa3othsrc;
  array srothtyp(3) cfa1othtyp cfa2othtyp cfa3othtyp;
  do i = 1 to 3;
    * Add state need-based aid and other aid state need-based aid.
    * Include aid classified as both need and merit-based aid.
    * cstamt if cfsttyp in (3)
    * cfaothamt if cfaothsrc = 2 & cfaothtyp in (3);
    stamtf(i) = 0;
    if cfastataid ~= 0 then
     do;
      if srsttyp(i) = 3 then
       do;
        stamtf(i) = srstamt(i);
        if missing(stamtf(i)) = 1 then stamtf(i) = .m;
       end;
      else if cfastataid = .x then stamtf(i) = .x;
     end;

    othamtf(i) = 0;
    if cfaothaid ~= 0 then
     do;
      if srothsrc(i) = 2 & srothtyp(i) = 3 then
       do;
        othamtf(i) = srothamt(i);
        if missing(othamtf(i)) = 1 then othamtf(i) = .m;
       end;
      else if cfaothaid = .x then othamtf(i) = .x;
     end;
  end;
  array forStndmrt(6) stamtf1 stamtf2 stamtf3 othamtf1 othamtf2 othamtf3;
  do i = 1 to 6;
   if forStndmrt(i) > 0 then
    do j = 1 to 6;
     if forStndmrt(j) = .x then forStndmrt(j) = 0;
    end;
  end;
  stndmrt = stamtf1 + stamtf2 + stamtf3 + othamtf1 + othamtf2 + othamtf3;
  if missing(stndmrt) = 1 then
   do;
    stndmrt = .x;
   end;
```

Variable Name	**STNDONLY**
Variable Label	State need-based only grants amount
Description	Total amount of state grants based only on need during the 2015–16 academic year.

Assigned Values	**Value** **Value label**
	0 {Zero}
	c {Continuous}

Data Source(s)	NPSAS:16 Student Records
Component Derived Variables	Not applicable
Applies to	All study members
Availability	NPSAS:16
Change History	New variable in NPSAS:16
N:16 Usage Statistics	Number of uses: 8 (UG); 0 (GR)
Programming Narrative	Add together state grants that are need-based only and other aid with source "state" and type grants that are need-based only.

SAS Code

```
* Source variable cfacvans="Received any financial aid";
* Source variable cf01sttyp-cf03sttyp="State aid program type";
* Source variable c01stamt-c03stamt="State aid program amount";
* Source variables cfa1othamt-cfa3othamt="Other aid amount";
* Source variables cfa1othsrc-cfa3othsrc="Other aid source";
* Source variables cfa1othtyp-cfa3othtyp="Other aid type";
**************************************************************;
  array stamta(3) stamta1-stamta3;
  * array srstamt(3) c01stamt c02stamt c03stamt;
  * array srsttyp(3) cf01sttyp cf02sttyp cf03sttyp;
  array othamta(3) othamta1-othamta3;
  * array srothamt(3) cfa1othamt cfa2othamt cfa3othamt;
  * array srothsrc(3) cfa1othsrc cfa2othsrc cfa3othsrc;
  * array srothtyp(3) cfa1othtyp cfa2othtyp cfa3othtyp;
  do i = 1 to 3;
    * Add state need-based aid and other aid state need-based aid.
    * cstamt if csttyp = 1
    * cfaothamt if cfaothsrc = 2 & cfaothtyp = 1;
    stamta(i) = 0;
    if cfastataid ~= 0 then
     do;
      if srsttyp(i) = 1 then
       do;
      stamta(i) = srstamt(i);
      if missing(stamta(i)) = 1 then stamta(i) = .m;
     end;
    else if cfastataid = .x then stamta(i) = .x;
   end;

  othamta(i) = 0;
  if cfaothaid ~= 0 then
   do;
    if srothsrc(i) = 2 & srothtyp(i) = 1 then
     do;
      othamta(i) = srothamt(i);
      if missing(othamta(i)) = 1 then othamta(i) = .m;
     end;
    else if cfaothaid = .x then othamta(i) = .x;
   end;
end;
array forStndonly(6) stamta1 stamta2 stamta3 othamta1 othamta2 othamta3;
do i = 1 to 6;
  if forStndonly(i) > 0 then
   do j = 1 to 6;
    if forStndonly(j) = .x then forStndonly(j) = 0;
   end;
end;
stndonly = stamta1 + stamta2 + stamta3 + othamta1 + othamta2 + othamta3;
if missing(stndonly) = 1 then
 do;
  stndonly = .x;
 end;
```

Variable Name	**STNOND1**
Variable Label	State non-need grants amount
Description	Total amount of state grants received in 2015–16 academic year that were based neither on need nor merit.
Assigned Values	<table><tr><th>Value</th><th>Value label</th></tr><tr><td>**0**</td><td>{Zero}</td></tr><tr><td>**c**</td><td>{Continuous}</td></tr></table>
Data Source(s)	NPSAS:16 Student Records
Component Derived Variables	Not applicable
Applies to	All study members
Availability	NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16
Change History	Unchanged from NPSAS:04 through NPSAS:08; Available for graduate students starting in NPSAS:12.
N:16 Usage Statistics	Number of uses: 3 (UG); 0 (GR)

Programming Narrative

1. Sum student records (state aid: grant - neither need nor merit) + (non-faculty/staff tuition waivers) + (unclassifiable grants/scholarships)—cstamt if cfsttyp in (4, 5, 17)
2. Also include other aid: state grant - neither need nor merit, non-faculty/staff state tuition waivers, and unclassifiable grants/scholarship—cfaothamt if cfaothsrc = 2 & cfaothtyp in (4, 5, 17)
3. Also include grants/scholarships from "outside state agency" found in the government/private aid section of student records—cfagovamt if cfagovtyp = 7

SAS Code

```
* Source variable cfacvans="Received any financial aid";
* Source variables cf01sttyp-cf03sttyp="State aid program type";
* Source variables c01stamt-c03stamt="State aid program amount";
* Source variables cfa1othamt-cfa3othamt="Other aid amount";
* Source variables cfa1othsrc-cfa3othsrc="Other aid source";
* Source variables cfa1othtyp-cfa3othtyp="Other aid type";
* Source variables cfa1govamt–cfa3govamt="Government/Private aid program amount";
* Source variables cfa1govtyp-cfa3govtyp="Government/Private aid program type";
  **************************************************************;
  array stamtc(3) stamtc1-stamtc3;
  * array srstamt(3) c01stamt c02stamt c03stamt;
  * array srsttyp(3) cf01sttyp cf02sttyp cf03sttyp;
  array othamtc(3) othamtc1-othamtc3;
  * array srothamt(3) cfa1othamt cfa2othamt cfa3othamt;
  * array srothsrc(3) cfa1othsrc cfa2othsrc cfa3othsrc;
  * array srothtyp(3) cfa1othtyp cfa2othtyp cfa3othtyp;
  array govamtc(3) govamtc1-govamtc3;
  array srgovamt(3) cfa1govamt cfa2govamt cfa3govamt;
  array srgovtyp(3) cfa1govtyp cfa2govtyp cfa3govtyp;
  do i = 1 to 3;
* Sum student records state aid: grant - neither need nor merit,
* non-faculty/staff tuition waivers, and unclassifiable
* grants/scholarships. Also include other aid: state grant - neither
* need nor merit, non-faculty/staff state tuition waivers, and
* unclassifiable grants/scholarships. Also include grants/scholarships
* from "outside state agency" found in the government/private aid
* section of student records
* cstamt if cfsttyp in (4, 5, 17)
* cfaothamt if cfaothsrc = 2 & cfaothtyp in (4, 5, 17)
* cfagovamt if cfagovtyp = 7;
    stamtc(i) = 0;
    if cfastataid ~= 0 then
     do;
      if srsttyp(i) in (4,5,17) then
       do;
        stamtc(i) = srstamt(i);
        if missing(stamtc(i)) = 1 then stamtc(i) = .m;
       end;
      else if cfastataid = .x then stamtc(i) = .x;
     end;

    othamtc(i) = 0;
    if cfaothaid ~= 0 then
     do;
```

```
      if srothsrc(i) = 2 & srothtyp(i) in (4,5,17) then
       do;
        othamtc(i) = srothamt(i);
        if missing(othamtc(i)) = 1 then othamtc(i) = .m;
       end;
      else if cfaothaid = .x then othamtc(i) = .x;
     end;

    govamtc(i) = 0;
    if cfaothgov ~= 0 then
     do;
      if srgovtyp(i) = 7 then
       do;
        govamtc(i) = srgovamt(i);
        if missing(govamtc(i)) = 1 then govamtc(i) = .m;
       end;
      else if cfaothgov = .x then govamtc(i) = .x;
     end;
  end;
  array forStnond1(9) stamtc1 stamtc2 stamtc3 othamtc1 othamtc2 othamtc3 govamtc1
govamtc2 govamtc3;
  do i = 1 to 9;
    if forStnond1(i) > 0 then
     do j = 1 to 9;
      if forStnond1(j) = .x then forStnond1(j) = 0;
     end;
  end;
  stnond1 = stamtc1 + stamtc2 + stamtc3 + othamtc1 + othamtc2 + othamtc3 + govamtc1 +
govamtc2 + govamtc3;
```

Variable Name	**STUDMULT**
Variable Label	Number of institutions attended
Description	Number of institutions attended during the 2015–16 academic year. May be sequential or simultaneous.
Assigned Values	**Value** **Value label** c {Continuous}
Data Source(s)	NPSAS:16 Interview, NSLDS:16, NSC:16
Component Derived Variables	Not applicable
Applies to	All study members
Availability	NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16
Change History	Unchanged from NPSAS:96 through NPSAS:16; NSC has only been a data source since NPSAS:12
N:16 Usage Statistics	Number of uses: 101 (UG); 1 (GR)

Programming Narrative	1. Determine number of enrollment strings in NSC data 2. Determine the maximum number of institutions at which the student: a. Had an enrollment record during the NPSAS year in NSLDS b. Had a Pell grant in the NPSAS year; and c. Took out loans in the NPSAS year; or 3. If the number of valid enrollment strings from the interview is greater than the maximum number of institutions at which the student took out loans (from NSLDS), set STUDMULT equal to the number of valid enrollment strings from the interview 4. Else, if the maximum number of institutions at which the student took out loans is greater than 1, then STUDMULT is equal to the maximum number of institutions at which the student took out loans, else let STUDMULT=1 5. If STUDMULT=1, but the number of enrollment strings in NSC data is greater than 1, then do the following: a. If the number of enrollment strings in NSC data is equal to 2, then set STUDMULT=2 b. If the number of enrollment strings in NSC data is greater than 2, then set STUDMULT=3

SAS Code	* Source variable enstct="Number of valid enrollment strings in interview";
	* Source variable variable y="Maximum number of institutions taken out of Pell and
	* NSLDS loan data";
	  **;
	* if # of valid enrollment strings from interview is greater than max # of
	* institutions took out loan, set STUDMULT equal # of valid enrollment strings
	* from interview;
	  if enstct > 0 & enstct > y then studmult = enstct;
	* else if max # of institutions took out loan greater than 1 then STUDMULT
	* equal max # of institution took out loan;
	   else if y > 1 then studmult = y;
	* else let STUDMULT=1;
	   else studmult = 1;
	* if STUDMULT=1, but # of enrollment strings in NSC data greater than 1,
	* then set STUDMULT to the # of enrollment strings in NSC;
	  if studmult = 1 & schnumch > 1 then studmult = schnumch;

<table>
<tr><td><b>Variable Name</b></td><td><b>STVETAMT</b></td></tr>
<tr><td><b>Variable Label</b></td><td>State veterans education benefits amount</td></tr>
<tr><td><b>Description</b></td><td>Total amount of state veterans education benefits received in 2015–16 academic year.</td></tr>
<tr><td><b>Assigned Values</b></td><td><table><tr><th>Value</th><th>Value label</th></tr><tr><td><b>0</b></td><td>{Zero}</td></tr><tr><td><b>c</b></td><td>{Continuous}</td></tr></table></td></tr>
<tr><td><b>Data Source(s)</b></td><td>NPSAS:16 Student Records</td></tr>
<tr><td><b>Component Derived Variables</b></td><td>Not applicable</td></tr>
<tr><td><b>Applies to</b></td><td>All study members</td></tr>
<tr><td><b>Availability</b></td><td>NPSAS:16</td></tr>
<tr><td><b>Change History</b></td><td>New variable in NPSAS:16</td></tr>
</table>

N:16 Usage Statistics	Number of uses: 25 (UG); 0 (GR)
Programming Narrative	Sum student records state veterans education benefits (type 15).

SAS Code

```
* Source variable cfacvans="Received any financial aid";
* Source variables cf01sttyp-cf03sttyp="State aid program type";
* Source variables c01stamt-c03stamt="State aid program amount";
* Source variables cfa1othamt-cfa3othamt="Other aid amount";
* Source variables cfa1othsrc-cfa3othsrc="Other aid source";
* Source variables cfa1othtyp-cfa3othtyp="Other aid type";
  ****************************************************************;
  array stamth(3) stamth1-stamth3;
  * array srstamt(3) c01stamt c02stamt c03stamt;
  * array srsttyp(3) cf01sttyp cf02sttyp cf03sttyp;
  array othamth(3) othamth1-othamth3;
  * array srothamt(3) cfa1othamt cfa2othamt cfa3othamt;
  * array srothsrc(3) cfa1othsrc cfa2othsrc cfa3othsrc;
  * array srothtyp(3) cfa1othtyp cfa2othtyp cfa3othtyp;
  do i = 1 to 3;
    * Sum student records state veterans benefits.
    * Veterans (state)
    * cstamt if cfsttyp = 15
    * cfaothamt if cfaothsrc = 2 & cfaothtyp = 15;
    stamth(i) = 0;
    if cfastataid ~= 0 then
     do;
      if srsttyp(i) = 15 then
       do;
        stamth(i) = srstamt(i);
        if missing(stamth(i)) = 1 then stamth(i) = .m;
       end;
      else if cfastataid = .x then stamth(i) = .x;
     end;

    othamth(i) = 0;
    if cfaothaid ~= 0 then
     do;
      if srothsrc(i) = 2 & srothtyp(i) = 15 then
       do;
        othamth(i) = srothamt(i);
        if missing(othamth(i)) = 1 then othamth(i) = .m;
       end;
      else if cfaothaid = .x then othamth(i) = .x;
     end;
  end;
  array forStvetamt(6) stamth1 stamth2 stamth3 othamth1 othamth2 othamth3;
  do i = 1 to 6;
    if forStvetamt(i) > 0 then
     do j = 1 to 6;
      if forStvetamt(j) = .x then forStvetamt(j) = 0;
     end;
```

<table>
<tr><td></td><td><pre>
end;
stvetamt = stamth1 + stamth2 + stamth3 + othamth1 + othamth2 + othamth3;
if missing(stvetamt) = 1 then
 do;
  stvetamt = .x;
 end;
</pre></td></tr>
</table>

<table>
<tr><td><b>Variable Name</b></td><td><b>STWKAMT</b></td></tr>
<tr><td><b>Variable Label</b></td><td>State work-study amount</td></tr>
<tr><td><b>Description</b></td><td>Total amount of state work-study received during the 2015–16 academic year.</td></tr>
<tr><td><b>Assigned Values</b></td><td>
<table>
<tr><th>Value</th><th>Value label</th></tr>
<tr><td><b>0</b></td><td>{Zero}</td></tr>
<tr><td><b>c</b></td><td>{Continuous}</td></tr>
</table>
</td></tr>
<tr><td><b>Data Source(s)</b></td><td>NPSAS:16 Student Records, NPSAS:16 Interview</td></tr>
<tr><td><b>Component Derived Variables</b></td><td>Not applicable</td></tr>
<tr><td><b>Applies to</b></td><td>All study members</td></tr>
<tr><td><b>Availability</b></td><td>NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16</td></tr>
<tr><td><b>Change History</b></td><td>Unchanged from NPSAS:96 through NPSAS:08; Available for graduate students starting in NPSAS:12.</td></tr>
<tr><td><b>N:16 Usage Statistics</b></td><td>Number of uses: 2 (UG); 0 (GR)</td></tr>
<tr><td><b>Programming Narrative</b></td><td>
1. Sum student records state work-study and other aid state work-study (type=7).
2. Set to missing if state aid section gate is missing (cfastataid) or the other aid section gate is missing (cfaothaid).
3. Any time this aid was received (e.g., source and type are correct), but the amount is missing, set a minimum imputation floor as the sum of any other amounts of the same type and source, else set this floor to 1, so that we impute a positive amount.
4. Prior to imputation, use student interview to determine if student has not received any workstudy. If student has not received workstudy, STWKAMT = zero.
</td></tr>
<tr><td><b>SAS Code</b></td><td><pre>
* Source variable cfacvans="Received any financial aid";
* Source variable cf01sttyp-cf03sttyp="State aid program type";
* Source variable c01stamt-c03stamt="State aid program amount";
* Source variables cfa1othamt-cfa3othamt="Other aid amount";
* Source variables cfa1othsrc-cfa3othsrc="Other aid source";
* Source variables cfa1othtyp-cfa3othtyp="Other aid type";
***;
  array stamt(3) stamt1-stamt3;
  array srstamt(3) c01stamt c02stamt c03stamt;
  array srsttyp(3) cf01sttyp cf02sttyp cf03sttyp;
  array othamts(3) othamts1-othamts3;
  * array srothamt(3) cfa1othamt cfa2othamt cfa3othamt;
  * array srothsrc(3) cfa1othsrc cfa2othsrc cfa3othsrc;
  * array srothtyp(3) cfa1othtyp cfa2othtyp cfa3othtyp;
  do i = 1 to 3;
   stamt(i) = 0;
   if cfastataid ~= 0 then
</pre></td></tr>
</table>

```
        do;
          if srsttyp(i) = 7 then
            do;
              stamt(i) = srstamt(i);
              if missing(stamt(i)) = 1 then stamt(i) = .m;
            end;
          else if cfastataid = .x then stamt(i) = .x;
        end;
othamts(i) = 0;
      if cfaothaid ~= 0 then
        do;
          if srothsrc(i) = 2 & srothtyp(i) = 7 then
            do;
              othamts(i) = srothamt(i);
              if missing(othamts(i)) = 1 then othamts(i) = .m;
            end;
          else if cfaothaid = .x then othamts(i) = .x;
        end;
    end;
    array forStwkamt(6) stamt1 stamt2 stamt3 othamts1 othamts2 othamts3;
    do i = 1 to 6;
        if forStwkamt(i) > 0 then
          do j = 1 to 6;
            if forStwkamt(j) = .x then forStwkamt(j) = 0;
          end;
    end;
    stwkamt = stamt1 + stamt2 + stamt3 + othamts1 + othamts2 + othamts3;
    stwkamt_minimp = sum(of stamt1 stamt2 stamt3 othamts1 othamts2 othamts3);
    end;
     if missing(stwkamt) = 1 then
         do;
            stwkamt = .x;
         end;
```

Variable Name	**STYPELST**
Variable Label	Student type indicator
Description	Student type at the NPSAS sample institution during the 2015–16 academic year.
Assigned Values	**Value** / **Value label** **1** Undergraduate students **2** Other graduate students **3** Doctor’s degree – professional practice
Data Source(s)	NPSAS:16 Interview, NPSAS:16 Student Records, FAFSA:16, NSLDS:16
Component Derived Variables	Not applicable
Applies to	All study members
Availability	NPSAS: 96, NPSAS:2000, NPSAS:04; NPSAS:08; NPSAS:12, NPSAS:16

Change History	Unchanged from NPSAS:96 through NPSAS:08. The value “Doctor’s degree – professional practice” replaced “First professional” in NPSAS:12 to reflect a change to how IPEDS collected graduate degrees. The two categories are essentially equivalent except that certain master’s degrees in theology such as M.Div were reclassified from first professional degrees to master’s degrees.
N:16 Usage Statistics	Number of uses: 27 (UG); 2 (GR)
Programming Narrative	1. Determine student type in different sources. a. Interview: use TSTAT; set to professional practice If TDEGREE=10 b. Student Records: Use SRSTYP. If still missing, set to undergrad if FTB flag is 1 (befsttm) c. CPS: set to undergraduate if student had a CPS record indicating undergraduate stats (C16DEG IN (1,2,3,4,5,6,7,9) and 1<C16GRDLVL<=6); set to graduate if student was working on a graduate degree (C16DEG=8), indicated their level was graduate (C16GRDLVL in [7, 8]), and they indicated they were working on a master's or in a doctorate program (C16MSDOC=1). Distinguish between other grad student and professional graduate student using the student sampled type from student records (if STTYPE = 3, and TYPECPS = 2 then TYPECPS = 3). However, set back to UG if student was FTB (C16GRDLVL=1). d. Student records (sampled type): use STTYPE e. Set student type in this order: Interview, Student Records, FAFSA, Sampled type 2. Edits a. If student type in Student Records & CPS matched, but different from the Interview, use Student Records & CPS. b. Fix potential conflict where undergraduate receives graduate aid: i. Determine whether student received grad aid: "yes" if Student Records (CFAGRAID=1) and NSLDS (LNLVLHI>5) indicated student received grad aid and student attended a 4-year college (level=1). ii. If student received grad aid, set to "grad.” iii. If CPS primary EFC type (C16EFCPT in [1, 4]) indicates dependent student or not attending a 4-year college (level>1), set to "undergraduate."
SAS Code	/*------------------------- STYPELST: Student type -------------------------*/ /* variable used: TSTAT TDEGREE BEFSTTM SRSTYP CFAGRAID INCPS C16MSDOC C16FIRBA C16DEG C16GRDLVL C16EFCPT BENADVDG LNLVLHI LEVEL FPOFFER 1=undergraduate 2=graduate 3=Professional practices */ /*--------- FIRST: determinate student type in different sources ---------*/ /* student interview: TYPESI */ IF TSTAT IN (1,3) THEN TYPESI=1; /* UG */ IF TSTAT IN (2,4) THEN TYPESI=2; /* Grad */ IF TDEGREE=10 THEN TYPESI=3; /* professional practice */ /* student records: TYPESR */ TYPESR=SRSTYP; /* use student type as determined in student records this based mainly on BENLADEG */ IF BEFSTTM=1 and SRSTYP=-9 THEN TYPESR=1; /* but set to "UG" if FTB flag is 1 and the student records type is missing*/ /* CPS: TYPECPS */

```
IF (C16DEG IN (1,2,3,4,5,6,7,9) AND 1<C16GRDLVL<=6) THEN TYPECPS=1; /* had a record indicating UG stats */
ELSE IF C16DEG=8 AND C16GRDLVL IN (7,8) AND C16MSDOC=1 THEN TYPECPS=2; /* working on a graduate degree */

IF STTYPE=3 AND TYPECPS=2 THEN TYPECPS=3; /* professional graduate students */

IF C16GRDLVL=1 THEN TYPECPS=1; /* set back to UG if student is FTB */

/* SECOND: Set student type in this order: Interview, Student Records, FAFSA, Sampled type */
IF TYPESI>0 THEN STYPELST=TYPESI;
ELSE IF TYPESR>0 THEN STYPELST=TYPESR;
ELSE IF TYPECPS>0 THEN STYPELST=TYPECPS;
ELSE IF STTYPE>0 THEN STYPELST=STTYPE;

/*----- THIRD: set student type to student records and CPS if they matched ---*/
/* if student records and CPS matched, but different from student interview, then use student records & CPS */

IF TYPESR>0 AND TYPECPS>0 AND TYPESR=TYPECPS THEN STYPELST=TYPESR;

/*--------------------------- FOURTH: fix conflicts: ------------------------*/
/* set to "grad" if received grad aid */
/* GRADAID: received grad aid */
/* set to "1" if student records AND NSLDS indicated student received grad aid and student attending a 4-year college */
IF CFAGRAID=1 AND LNLVLHI>5 AND LEVEL=1 THEN GRADAID=1;
IF STYPELST=1 AND GRADAID=1 THEN STYPELST=2;

/* set to "UG" if CPS EFC indicated student is dependent or not attending a
4-year college */
IF C16EFCPT IN (1,4) OR LEVEL>1 THEN STYPELST=1;
```

<table>
<tr><td><b>Variable Name</b></td><td><b>T4LNAMT1</b></td></tr>
<tr><td><b>Variable Label</b></td><td>Title IV loans (excludes Parent PLUS Loans) amount</td></tr>
<tr><td><b>Description</b></td><td>Total amount of federal Title IV loans (excluding Parent PLUS Loans) received during the 2015–16 academic year.</td></tr>
<tr><td><b>Assigned Values</b></td><td><table><tr><th>Value</th><th>Value label</th></tr><tr><td><b>0</b></td><td>{Zero}</td></tr><tr><td><b>c</b></td><td>{Continuous}</td></tr></table></td></tr>
<tr><td><b>Data Source(s)</b></td><td>NSLDS:16, NPSAS:16 Student Records</td></tr>
<tr><td><b>Component Derived Variables</b></td><td>STAFFAMT, PERKAMT, GPLUSAMT</td></tr>
<tr><td><b>Applies to</b></td><td>All study members</td></tr>
<tr><td><b>Availability</b></td><td>NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16</td></tr>
</table>

Change History	Unchanged from NPSAS:96 through NPSAS:16
N:16 Usage Statistics	Number of uses: 228 (UG); 123 (GR)
Programming Narrative	1. Undergraduate: sum Stafford total (subsidized and unsubsidized) and Perkins loans. 2. Graduate: sum Stafford total (subsidized, which is zero, and unsubsidized), Perkins loans, and Graduate PLUS loan.

SAS Code	* Source variable gplusamt="Direct PLUS Loans to graduate students";
	* Source variable perkamt="Perkins loan";
	* Source variable staffamt="Direct Subsidized and Unsubsidized Loans";
	* Source variable stypelst="Student type indicator";
	***;
	* UG: sum Stafford total (subsidized and unsubsidized) and Perkins loans;
	  if stypelst = 1 then t4lnamt1 = perkamt + staffamt;
	* GR: sum Stafford total (subsidized, which is zero, and
	* unsubsidized), Perkins loans and GR PLUS loan;
	  else if stypelst > 1 then t4lnamt1 = perkamt + staffamt + gplusamt;

Variable Name	**T4LNAMT2**
Variable Label	Title IV loans (includes Parent PLUS Loans) amount
Description	Total amount of federal Title IV loans (including Parent PLUS Loans) received during the 2015–16 academic year.

Assigned Values	**Value** — **Value label**
	0 — {Zero}
	c — {Continuous}

Data Source(s)	NSLDS:16, NPSAS:16 Student Records
Component Derived Variables	PERKAMT, STAFFAMT, PLUSAMT, STYPELST
Applies to	All undergraduate study members
Availability	NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16
Change History	Unchanged from NPSAS:96 through NPSAS:16
N:16 Usage Statistics	Number of uses: 46 (UG); 0 (GR)
Programming Narrative	1. Sum Stafford total (subsidized and unsubsidized), Perkins loans, and Parent PLUS loan. 2. Set graduate students to skip.

SAS Code	* Source variable perkamt="Perkins loan";
	* Source variable staffamt="Direct Subsidized and Unsubsidized Loans";
	* Source variable plusamt="Direct PLUS Loans to parents";
	* Source variable stypelst="Student type indicator";
	    **;
	    if stypelst > 1 then do; t4lnamt2 = .y; end;
	    * Sum Stafford total (subsidized and unsubsidized), Perkins loans, and
	    * Parent PLUS loan;
	    else if stypelst = 1 then
	      do;
	        if depend = 2 then t4lnamt2 = perkamt + staffamt;
	                    else t4lnamt2 = perkamt + staffamt + plusamt;

<table>
<tr><td><b>Variable Name</b></td><td><b>TEACHGRT</b></td></tr>
<tr><td><b>Variable Label</b></td><td>Federal TEACH grant amount</td></tr>
<tr><td><b>Description</b></td><td>Amount of TEACH (Teacher Education Assistance for College and Higher Education) Grants received during the 2015–16 academic year.</td></tr>
<tr><td><b>Assigned Values</b></td><td><table><tr><th>Value</th><th>Value label</th></tr><tr><td><b>0</b></td><td>{Zero}</td></tr><tr><td><b>c</b></td><td>{Continuous}</td></tr></table></td></tr>
<tr><td><b>Data Source(s)</b></td><td>NSLDS:16</td></tr>
<tr><td><b>Component Derived Variables</b></td><td>Not applicable</td></tr>
<tr><td><b>Applies to</b></td><td>All study members</td></tr>
<tr><td><b>Availability</b></td><td>NPSAS:16</td></tr>
<tr><td><b>Change History</b></td><td>New variable in NPSAS:16</td></tr>
<tr><td><b>N:16 Usage Statistics</b></td><td>Number of uses: 77 (UG); 28 (GR)</td></tr>
<tr><td><b>Programming Narrative</b></td><td>1. Subset the NSLDS Pell grant dataset to TEACH Grants (type = "TG").
2. Subset to grants with an award year of 2016.
3. Sum up the total amount Federal TEACH Grants in the 2016 award year.
4. Check values that exceed maximum values ($3,708 for grants disbursed from July 1, 2015 through September 30, 2015 and $3,728 for grants disbursed from October 1, 2015 through June 30, 2016).</td></tr>
<tr><td><b>SAS Code</b></td><td>* Source variable ndteach="Federal TEACH grant amount (NSLDS)";
**;
teachgrt = ndteach;</td></tr>
</table>

Variable Name	**TFEDAID**
Variable Label	Total federal aid (excludes Veterans'/DOD) amount
Description	Total amount of federal aid (excluding federal veterans education benefits & DOD) received during the 2015–16 academic year.

Assigned Values	**Value** **Value label**
	0 {Zero}
	c {Continuous}

Data Source(s)	NPSAS:16 Student Records, NSLDS:16, NPSAS:16 Interview
Component Derived Variables	TFEDGRT, TFEDLN, TFEDLN2, TFEDWRK
Applies to	All study members
Availability	NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16
Change History	Unchanged from NPSAS:96 through NPSAS:16.
N:16 Usage Statistics	Number of uses: 363 (UG); 50 (GR)
Programming Narrative	1. If undergraduate study member (STYPELST = 1): Sum of total federal grants (TFEDGRT), total work study (TFEDWRK), and total federal loans including PLUS Loans to parents (TFEDLN2). 2. If graduate study member (STYPELST > 1): Sum of total federal grants (TFEDGRT), total work study (TFEDWRK), and total federal loans excluding PLUS Loans to parents (TFEDLN).

SAS Code

```
* Source variable tfedgrt="Total federal grants";
* Source variable tfedwrk="Federal work-study";
* Source variable tfedln2="Total federal loans (includes Direct PLUS Loans
* to parents)";
* Source variable tfedln="Total federal loans (excludes Direct PLUS Loans
*     to parents)";
  *****************************************************************;
* If undergraduate study member (STYPELST = 1): Sum of total federal grants
* (TFEDGRT), total work study (TFEDWRK), and total federal loans including
* PLUS Loans to parents (TFEDLN2). If graduate study member (STYPELST > 1):
* Sum of total federal grants (TFEDGRT), total work study (TFEDWRK), and
* total federal loans excluding PLUS Loans to parents (TFEDLN);
if stypelst = 1 then
   do;
    tfedaid = tfedgrt + tfedwrk + tfedln2;
    if missing(tfedaid) = 1 then
     do;
      tfedaid = .x;
     end;
else if stypelst > 1 then
   do;
    tfedaid = tfedgrt + tfedwrk + tfedln;
    if missing(tfedaid) = 1 then
     do;
      tfedaid = .x;
     end;
```

Variable Name	**TFEDGRT**
Variable Label	Total federal grants (excludes Veterans'/DOD) amount
Description	Total amount of federal grants received during the 2015–16 academic year excluding federal veterans education benefits and Department of Defense aid.

Assigned Values	**Value** – **Value label**
	0 – {Zero}
	c – {Continuous}

Data Source(s)	NPSAS:16 Student Records, NSLDS:16
Component Derived Variables	STYPELST, PELLAMT, SEOGAMT, TEACHGRT, OTHFDGRT
Applies to	All study members
Availability	NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16
Change History	Unchanged from NPSAS:96 through NPSAS:16.
N:16 Usage Statistics	Number of uses: 266 (UG); 4 (GR)
Programming Narrative	1. If undergraduate study member (STYPELST = 1): Sum of total Pell grants (PELLAMT), Supplemental Educational Opportunity Grants (SEOGAMT) TEACH grants (TEACHGRT), and other federal grants (OTHFDGRT). 2. If graduate study member (STYPELST > 1): Sum of TEACH grants (TEACHGRT) and other federal grants (OTHFDGRT).

SAS Code	* Source variable stypelst="Student type indicator";
	* Source variable pellamt="Federal Pell grant";
	* Source variable seogamt="Federal Supplemental Educational Opportunity Grant (SEOG)";
	* Source variable teachgrt="Federal TEACH grant amount";
	* Source variable othfdgrt="Other federal grants (not Title IV)";
	**;
	* If undergraduate study member (STYPELST = 1): Sum of total Pell grants
	* (PELLAMT), Supplemental Educational Opportunity Grants (SEOGAMT) TEACH
	* grants (TEACHGRT), and other federal grants (OTHFDGRT).
	* If graduate study member (STYPELST > 1): Equal to TEACH grants (TEACHGRT)
	* and other federal grants (OTHFDGRT);
	if stypelst = 1 then
	do;
	tfedgrt = pellamt + seogamt + teachgrt + othfdgrt;
	if missing(tfedgrt) = 1 then
	do;
	tfedgrt = .x;
	end;
	else if stypelst > 1 then
	do;
	tfedgrt = teachgrt + othfdgrt;
	if missing(tfedgrt) = 1 then
	do;
	tfedgrt = .x;
	end;

<table>
<tr><td><b>Variable Name</b></td><td><b>TFEDLN</b></td></tr>
<tr><td><b>Variable Label</b></td><td>Total federal loans (excludes Parent PLUS Loans) amount</td></tr>
<tr><td><b>Description</b></td><td>Total amount of federal loans excluding Direct PLUS loans to parents received during the 2015–16 academic year. The graduate student version of this variable includes Direct PLUS loans to graduate students.</td></tr>
<tr><td><b>Assigned Values</b></td><td><table><tr><th>Value</th><th>Value label</th></tr><tr><td><b>0</b></td><td>{Zero}</td></tr><tr><td><b>c</b></td><td>{Continuous}</td></tr></table></td></tr>
<tr><td><b>Data Source(s)</b></td><td>NPSAS:16 Student Records, NSLDS:16, NPSAS:16 Interview</td></tr>
<tr><td><b>Component Derived Variables</b></td><td>Undergraduate—STAFFAMT, PERKAMT, PHSLOAN
Graduate–STAFFAMT, PERKAMT, GPLUSAMT, PHSLOAN
Note: PHSLOAN is a restricted-use file variable only. It is not described in this document.</td></tr>
<tr><td><b>Applies to</b></td><td>All study members</td></tr>
<tr><td><b>Availability</b></td><td>NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16</td></tr>
<tr><td><b>Change History</b></td><td>Unchanged from NPSAS:96 through NPSAS:16.</td></tr>
<tr><td><b>N:16 Usage Statistics</b></td><td>Number of uses: 739 (UG); 199 (GR)</td></tr>
<tr><td><b>Programming Narrative</b></td><td>1. See component variables PERKAMT, STAFUNSB, and STAFSUB, and GPLUSAMT for data management steps of federal loan amounts from NSLDS.
2. Derive component variables Perkins amount (PERKAMT), Stafford subsidized and unsubsidized amount (STAFFAMT = STAFUNSB + STAFSUB), federal health professions loans (PHSLOAN), and Graduate PLUS Loans (GPLUSAMT).
3. If study member is an undergraduate (STYPELST = 1): Total federal loans equals the sum of all Stafford loans (STAFFAMT), Perkins loans (PERKAMT), and federal health professions loans (PHSLOAN).
4. Else if study member is a graduate student: Total federal loans equals the sum of all Stafford loans (STAFFAMT), Perkins loans (PERKAMT), federal health professions loans (PHSLOAN), and Graduate PLUS loans (GPLUSAMT).</td></tr>
<tr><td><b>SAS Code</b></td><td><pre>
*Source variable staffamt="Stafford total subsidized+unsubsidized";
*Source variable perkamt="Perkins loan";
*Source variable phsloan="Federal health professions loans";
*Source variable gplusamt [Grad only]="Graduate PLUS loan";
*Source variable stypelst="Student type indicator";
*Applies to: all students;
if stypelst=1 then
     do;
      tfedln = staffamt + perkamt + phsloan;
        if missing(tfedln) = 1 then
               do;
                tfedln = .x;
                end;
    else if stypelst > 1 then
             do;
              tfedln = perkamt + staffamt + phsloan + gplusamt;
              if missing(tfedln) = 1 then
               do;
                tfedln = .x;
               end;
              end;
</pre></td></tr>
</table>

Variable Name	**TFEDLN2**
Variable Label	Total federal loans (includes Parent PLUS Loans) amount
Description	Total amount of federal loans including Parent PLUS Loans received during the 2015–16 academic year.
Assigned Values	

Value	Value label
0	{Zero}
c	{Continuous}

Data Source(s)	NPSAS:16 Student Records, NSLDS: 16
Component Derived Variables	STAFFAMT, PERKAMT, PLUSAMT, STYPELST, PHSLOAN Note: PHSLOAN is a restricted-use file variable only. It is not described in this document.
Applies to	All undergraduate study members
Availability	NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16
Change History	Unchanged from NPSAS:96 through NPSAS:16.
N:16 Usage Statistics	Number of uses: 294 (UG); 0 (GR)
Programming Narrative	1. See component variables, PERKAMT, STAFUNSB, and STAFSUB, and PLUSAMT for data management steps of federal loan amounts from NSLDS. 2. If study member is a graduate student (STYPELST > 1), then set to skip (TFEDLN2 = -3). 3. Derive component variables Perkins amount (PERKAMT), Stafford subsidized and unsubsidized amount (STAFFAMT = STAFUNSB + STAFSUB), federal health professions loans (PHSLOAN), and Direct PLUS loans to parents (PLUSAMT). 4. Sum component variables, including PLUS Loans to parents of undergraduates (TFEDLN2 = PERKAMT + STAFFAMT + PHSLOAN + PLUSAMT).
SAS Code	

```
* Source variable perkamt="Perkins loan";
* Source variable staffamt="Direct Subsidized and Unsubsidized Loans";
* Source variable phsloan="Federal health professions loans";
* Source variable plusamt="Direct PLUS Loans to parents";
* Source variable depend="Dependency status";
* Source variable stypelst="Student type indicator";
  ****************************************************************************;
* If study member is a graduate student (STYPELST > 1), then set
* to skip (TFEDLN2 = -3);
  if stypelst > 1      then do; tfedln2 = .y; end;
* Sum component variables, including PLUS Loans to parents of
* undergraduates (TFEDLN2 = PERKAMT + STAFFAMT + PHSLOAN + PLUSAMT);
  else if stypelst = 1 then
   do;
    if depend = 2 then
     do;
      tfedln2 = perkamt + staffamt + phsloan;
      if missing(tfedln2) = 1 then
       do;
        tfedln2 = .x;
     do;
      tfedln2 = perkamt + staffamt + phsloan + plusamt;
      if missing(tfedln2) = 1 then
       do;
```

	tfedln2 = .x;

<table>
<tr><td><b>Variable Name</b></td><td><b>TFEDWRK</b></td></tr>
<tr><td><b>Variable Label</b></td><td>Federal work-study amount</td></tr>
<tr><td><b>Description</b></td><td>Total amount of federal work-study awarded during the 2015–16 academic year.</td></tr>
<tr><td><b>Assigned Values</b></td><td><table><tr><th>Value</th><th>Value label</th></tr><tr><td><b>0</b></td><td>{Zero}</td></tr><tr><td><b>c</b></td><td>{Continuous}</td></tr></table></td></tr>
<tr><td><b>Data Source(s)</b></td><td>NPSAS:16 Student Records, NPSAS:16 Interview</td></tr>
<tr><td><b>Component Derived Variables</b></td><td>Not applicable</td></tr>
<tr><td><b>Applies to</b></td><td>All study members</td></tr>
<tr><td><b>Availability</b></td><td>NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16</td></tr>
<tr><td><b>Change History</b></td><td>Unchanged from NPSAS:96 through NPSAS:16.</td></tr>
<tr><td><b>N:16 Usage Statistics</b></td><td>Number of uses: 108 (UG); 9 (GR)</td></tr>
<tr><td><b>Programming Narrative</b></td><td>1. Use student records: sum of federal work study and other specified aid (CFAOTHAMT, CFAOTHSRC, and CFAOTHTYP) if applicable. This relevance is determined by source and types of aid.
2. Prior to imputation, use the interview to determine if a student has not received any workstudy and assign it zero. Any time there is indication of receipt of this aid, but the amount is missing, set minimum imputation floor as sum of other amounts of same type and sources. Else set floor to $1.</td></tr>
<tr><td><b>SAS Code</b></td><td><pre>
* Source variable cfaothaid="Received any other aid";
* Source variable cfatdfws="Federal work-study";
* Source variables cfa1othamt-cfa3othamt="Other aid amount";
* Source variables cfa1othsrc-cfa3othsrc="Other aid source";
* Source variables cfa1othtyp-cfa3othtyp="Other aid type";
**;
  array othamtf(3) othamtf1-othamtf3;
  array srothamt(3) cfa1othamt cfa2othamt cfa3othamt;
  array srothsrc(3) cfa1othsrc cfa2othsrc cfa3othsrc;
  array srothtyp(3) cfa1othtyp cfa2othtyp cfa3othtyp;
  do i = 1 to 3;
   othamtf(i) = 0;
   if cfaothaid ~= 0 then
    do;
     if srothsrc(i) = 3 & srothtyp(i) = 7 then
      do;
       othamtf(i) = srothamt(i);
       if missing(othamtf(i)) = 1 then othamtf(i) = .m;
      end;
     else if cfaothaid = .x then othamtf(i) = .x;
    end;
  end;
  array forTfedwrk(3) othamtf1 othamtf2 othamtf3;
  if cfatdfws > 0 then
</pre></td></tr>
</table>

```
      do j = 1 to 3;
        if forTfedwrk(j) = .x then forTfedwrk(j) = 0;
      end;

* Use student records: sum of federal work study and custom federal
* aid if applicable. This relevance is determined by source and types
* of aid;
    tfedwrk = cfatdfws + othamtf1 + othamtf2 + othamtf3;

    if missing(tfedwrk) = 1 then
      do;
        tfedwrk = .x;
      end;
*****************************************************************;
    * Reconcile work-study variables
  *****************************************************************;
if missing(tfedwrk) = 1 & n16caidwkst1 = 0 then do; tfedwrk = 0; tfedwrk_minimp = 0; end;
```

<table>
<tr><td><b>Variable Name</b></td><td><b>TITIVAMT</b></td></tr>
<tr><td><b>Variable Label</b></td><td>Total federal Title IV aid amount</td></tr>
<tr><td><b>Description</b></td><td>Total amount of federal Title IV financial aid received during the 2015–16 academic year.</td></tr>
<tr><td><b>Assigned Values</b></td><td><table><tr><th>Value</th><th>Value label</th></tr><tr><td><b>0</b></td><td>{Zero}</td></tr><tr><td><b>c</b></td><td>{Continuous}</td></tr></table></td></tr>
<tr><td><b>Data Source(s)</b></td><td>NSLDS:16, NPSAS:16 Student Records, NPSAS:16 Interview</td></tr>
<tr><td><b>Component Derived Variables</b></td><td>T4LNAMT2, PELLAMT, SEOGAMT, TFEDWRK, T4LNAMT1, TEACHGRT</td></tr>
<tr><td><b>Applies to</b></td><td>All study members</td></tr>
<tr><td><b>Availability</b></td><td>NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16</td></tr>
<tr><td><b>Change History</b></td><td>Unchanged from NPSAS:96 through NPSAS:08; TEACH grants were new to NPSAS:12 and were not derived separately until NPSAS:16.</td></tr>
<tr><td><b>N:16 Usage Statistics</b></td><td>Number of uses: 98 (UG); 17 (GR)</td></tr>
<tr><td><b>Programming Narrative</b></td><td>1. Undergraduate: sum Title IV loans (includes Parent PLUS), federal Pell grant, federal Supplemental Educational Opportunity Grant (SEOG), TEACH grants, and federal work-study.
2. Graduate: sum Title IV loans, TEACH grants, and federal work-study.</td></tr>
</table>

SAS Code	*Source variable t4lnamt1="Title IV loans (excludes Parent PLUS)";
	*Source variable t4lnamt2="Title IV loans (includes Parent PLUS)";
	*Source variable pellamt="Federal Pell grant";
	*Source variable seogamt="Federal Supplemental Educational Opportunity Grant (SEOG)";
	*Source variable tfedwrk="Federal work-study";
	*Source variable teachgrt="TEACH grants";
	**;
	*Applies to: all study members;
	if (stypelst eq 1) then do;
	titivamt = t4lnamt2 + pellamt + seogamt + tfedwrk + teachgrt;
	end;
	else do;
	titivamt = t4lnamt1 + tfedwrk + teachgrt;
	end;
	if missing(titivamt)=1 then do; titivamt=.x; end;

Variable Name	**TOTAID**
Variable Label	Total aid amount
Description	Total amount of all financial aid received during the 2015–16 academic year.

Assigned Values	**Value** | **Value label**
	0 | {Zero}
	C | {Continuous}

Data Source(s)	NPSAS:16 Student Records, NSLDS:16, NPSAS:16 Interview, VBA:16
Component Derived Variables	TOTGRT, TOTLOAN, TOTWKST, OTHTYPE
Applies to	All study members
Availability	NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16
Change History	Unchanged from NPSAS:96 through NPSAS:16.
N:16 Usage Statistics	Number of uses: 1,238 (UG); 97 (GR)
Programming Narrative	Sum of total grants, total loans (excluding Parent PLUS), total work-study, and total other type of aid (Parent PLUS, job training, assistantships, VA/DOD).

SAS Code	* Source variable totgrt="Total grants";
	* Source variable totloan="Total loans (excluding Direct PLUS Loans to parents)";
	* Source variable totwkst="Total work-study";
	* Source variable othtype="Total other type of aid (Direct PLUS, job training, VA)";
	**;
	* Sum total grants, total loans (excluding Parent PLUS), total work-study
	* and total other type of aid (Parent PLUS, job training, assistantships,
	* VA/DOD);
	totaid = totgrt + totloan + totwkst + othtype;
	if missing(totaid) = 1 then
	do;
	totaid = .x;
	end;

<table>
<tr><td><b>Variable Name</b></td><td><b>TOTGRT</b></td></tr>
<tr><td><b>Variable Label</b></td><td>Total grants amount</td></tr>
<tr><td><b>Description</b></td><td>Total amount of all grants and scholarships received during the 2015–16 academic year.</td></tr>
<tr><td><b>Assigned Values</b></td><td><table><tr><th>Value</th><th>Value label</th></tr><tr><td>0</td><td>{Zero}</td></tr><tr><td>c</td><td>{Continuous}</td></tr></table></td></tr>
<tr><td><b>Data Source(s)</b></td><td>NPSAS:16 Student Records, NPSAS:16 Interview, NSLDS:16</td></tr>
<tr><td><b>Component Derived Variables</b></td><td>TFEDGRT, STGTAMT, INGRTAMT, OTHGTAMT</td></tr>
<tr><td><b>Applies to</b></td><td>All study members</td></tr>
<tr><td><b>Availability</b></td><td>NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16</td></tr>
<tr><td><b>Change History</b></td><td>Unchanged from NPSAS:96 through NPSAS:16.</td></tr>
<tr><td><b>N:16 Usage Statistics</b></td><td>Number of uses: 1,291 (UG); 125 (GR)</td></tr>
<tr><td><b>Programming Narrative</b></td><td>Sum of all federal grants (TFEDGRT), state grants (STGTAMT), institutional grants (INGRTAMT), and outside grants (OTHGTAMT).</td></tr>
<tr><td><b>SAS Code</b></td><td>* Source variable tfedgrt="Total federal grants";
* Source variable stgtamt="State grants total";
* Source variable ingrtamt="Institution grants total";
* Source variable othgtamt="Outside grants (private & employer)";
    **;
* Sum of all federal grants (TFEDGRT), state grants (STGTAMT),
* institutional grants (INGRTAMT), and outside grants (OTHGTAMT);
    totgrt = tfedgrt + stgtamt + ingrtamt + othgtamt;
    if missing(totgrt) = 1 then
      do;
        totgrt = .x;</td></tr>
</table>

<table>
<tr><td><b>Variable Name</b></td><td><b>TOTLOAN</b></td></tr>
<tr><td><b>Variable Label</b></td><td>Total loans (excluding Parent PLUS Loans) amount</td></tr>
<tr><td><b>Description</b></td><td>Total loans (excluding Direct PLUS Loans to parents).</td></tr>
<tr><td><b>Assigned Values</b></td><td><table><tr><th>Value</th><th>Value label</th></tr><tr><td>0</td><td>{Zero}</td></tr><tr><td>c</td><td>{Continuous}</td></tr></table></td></tr>
<tr><td><b>Data Source(s)</b></td><td>NPSAS:16 Student Records, NSLDS:16, NPSAS:16 Interview</td></tr>
<tr><td><b>Component Derived Variables</b></td><td>TFEDLN, STLNAMT, INLNAMT, PRIVLOAN</td></tr>
<tr><td><b>Applies to</b></td><td>All study members</td></tr>
<tr><td><b>Availability</b></td><td>NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16</td></tr>
<tr><td><b>Change History</b></td><td>Unchanged from NPSAS:96 through NPSAS:16.</td></tr>
<tr><td><b>N:16 Usage Statistics</b></td><td>Number of uses: 1,587 (UG); 218 (GR)</td></tr>
</table>

Programming Narrative	For all study members: sum of all federal loans to students (TFEDLN), state loans (STLNAMT), institutional loans (INLNAMT), and other private commercial or alternative loans (PRIVLOAN).

SAS Code	* Source variable tfedln="Total federal loans (excludes Direct PLUS Loans to parents)";
	* Source variable stlnamt="State loans";
	* Source variable inlnamt="Institutional loans";
	* Source variable privloan="Private (alternative) loans";
	**;
	* For all study members: Sum of all federal loans to students (TFEDLN),
	* state loans (STLNAMT), institutional loans (INLNAMT), and other private
	* commercial or alternative loans (PRIVLOAN);
	totloan = tfedln + stlnamt + inlnamt + privloan;
	if missing(totloan) = 1 then
	do;
	totloan = .x;

Variable Name	**TOTWKST**
Variable Label	Total work-study amount
Description	Total amount of all work-study awards received during the 2015–16 academic year.

Assigned Values	**Value** | **Value label**
	0 | {Zero}
	c | {Continuous}

Data Source(s)	NPSAS:16 Student Records, NPSAS:16 Interview
Component Derived Variables	TFEDWRK, STWKAMT, INSTWRK
Applies to	All study members
Availability	NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16
Change History	Unchanged from NPSAS:96 through NPSAS:16.
N:16 Usage Statistics	Number of uses: 572 (UG); 5 (GR)
Programming Narrative	Sum federal work-study (TFEDWRK), state work-study (STWKAMT), and institutional work-study (INSTWRK).

SAS Code	* Source variable tfedwrk="Federal work-study";
	* Source variable stwkamt="State work-study";
	* Source variable instwrk="Institutional work-study";
	**;
	* Sum federal work-study (TFEDWRK), state work-study (STWKAMT) and
	* institutional work-study (INSTWRK);
	totwkst = tfedwrk + stwkamt + instwrk;
	if missing(totwkst) = 1 then
	do;
	totwkst = .x;

<table>
<tr><td><b>Variable Name</b></td><td><b>TUITION2</b></td></tr>
<tr><td><b>Variable Label</b></td><td>Tuition and fees paid</td></tr>
<tr><td><b>Description</b></td><td>Tuition and fees at the sampled NPSAS institution for students who attended only one institution (STUDMULT=1) during the 2015–16 academic year.</td></tr>
<tr><td><b>Assigned Values</b></td><td><table><tr><th>Value</th><th>Value label</th></tr><tr><td><b>-3</b></td><td>{Skipped}</td></tr><tr><td><b>c</b></td><td>{Continuous}</td></tr></table></td></tr>
<tr><td><b>Data Source(s)</b></td><td>NPSAS:16 Student Records, IPEDS:16</td></tr>
<tr><td><b>Component Derived Variables</b></td><td>SECTOR11, ATTNSTAT, INJURIS, STUDMULT
Note: SECTOR11 and INURIS are not included in this report. Information on these variables may be found in the NPSAS:16 undergraduate codebook, https://nces.ed.gov/datalab/powerstats/codebook.aspx?dataset=121&type=subject#SECTOR11 and https://nces.ed.gov/datalab/powerstats/codebook.aspx?dataset=121&type=subject#INJURIS, and the NPSAS:16 graduate student codebook, https://nces.ed.gov/datalab/powerstats/codebook.aspx?dataset=122&type=subject#SECTOR11 and https://nces.ed.gov/datalab/powerstats/codebook.aspx?dataset=122&type=subject#INJURIS.</td></tr>
<tr><td><b>Applies to</b></td><td>Study members who attended one institution (STUDMULT=1)</td></tr>
<tr><td><b>Availability</b></td><td>NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16</td></tr>
<tr><td><b>Change History</b></td><td>Unchanged from NPSAS:96 through NPSAS:16</td></tr>
<tr><td><b>N:16 Usage Statistics</b></td><td>Number of uses: 1,378 (UG); 274 (GR)</td></tr>
<tr><td><b>Programming Narrative</b></td><td>1. Student records tuition is the main source (bttuitot). Before using, calculate outliers within pre-determined strata (sector [SECTOR11], in-state/out-of-state [INJURIS], and attendance status [ATTNSTAT]).
a. Since distributions are generally highly right-skewed, use Box-Cox power transformations to normalize tuition within each stratum
b. Flag outliers using the mean +/- 3 SD of the transformed tuition. Set outlier tuition values to missing.
2. Replace missing values in the following order:
a. Average tuition within institution by in-state/out-of state and attendance status
b. Average IPEDS tuition, adjusted for attendance status
i. Take all valid values from student records within an institution and find the median ratio of Full-time full-year tuition to other attendance statuses (part-time, full-year; full-time, part-year; part-time, part-year).
ii. Use median ratio to adjust IPEDS tuition for attendance (tuition_adjusted = IPEDS tuition * ratio)
c. Else use the overall institution average (if N>10), and adjust for attendance the same way as with the IPEDS tuition
d. Else use the overall stratum average, and adjust for attendance the same way as with the IPEDS tuition</td></tr>
</table>

SAS Code

```
/* Merge in the IPEDS vocational tuition/charges dataset */

*Create a flag for those reported tuition/fees for vocational programs;
if b=1 then tuit_voc=1; else tuit_voc=0;
run;

*Create a subset and interim variables;
data step1;
merge X.alldata(keep=ZRID INSTID STYPELST BTTUITOT CTUITION CNPERIOD
      TUITION1-TUITION3 fee1-fee3 becipmaj1 n16bmj1spe IN=A)
      X.Reside (keep=ZRID INJURIS)
      X.Attend(keep=ZRID STUDMULT ATTNSTAT)
      X.Inst(keep=ZRID SECTOR11)
      student_initialwt
      ipeds;
by ZRID;
if A;

/* UG students who attended one institution */
if STYPELST=1 and STUDMULT=1;

/* GR students who attended one institution */
if STYPELST>1 and STUDMULT=1;

/* Take the value of the SR variable */
TUITION=BTTUITOT;
if BTTUITOT=0 then TUITION=-9;

/* Use budgeted tuition if their attendance aligns with the
   budget period*/
if TUITION<0 and CTUITION>0 then ctflag=0;
if ctflag=0 then do;
 if (ATTNSTAT=1 and CNPERIOD=1) then ctflag=1;
 if (ATTNSTAT=3 and CNPERIOD=2) then ctflag=1;
 if (ATTNSTAT=4 and CNPERIOD in (3,5,7)) then ctflag=1;
 if (ATTNSTAT=6 and CNPERIOD in (4,6,8)) then ctflag=1;
end;

if ctflag=1 then TUITION=CTUITION;

run;

data step2;
 set step1;

  *New sector variable for outlier detection;
  if sector11<3 then tuit_sector=1;
   else if 3<=sector11<=5 and injuris ne 2 then tuit_sector=2;
   else if 3<=sector11<=5 and injuris=2 then tuit_sector=3;
   else if sector11 in (6,9) then tuit_sector=4;
   else if sector11 in (7,8) then tuit_sector=5;
   else if sector11 in (10,11) then tuit_sector=6;

  *New attendance variable;
  if attnstat=1 then tuit_attn = 1; *Full time, full year;
   else if attnstat in (3,4) then tuit_attn = 2; *FT/PY or PT/FY;
   else if attnstat=6 then tuit_attn = 3; *Part time, part year;

  /*Group variable by sector and attendance
    values for tuit_group:
              1 "Pub <= 2yr; FT/FY"
              2 "Pub <= 2yr; FT/PY or PT/FY"
              3 "Pub <= 2yr; PT/PY"
              4 "Pub 4yr InS; FT/FY"
              5 "Pub 4yr InS; FT/PY or PT/FY"
              6 "Pub 4yr InS; PT/PY"
              7 "Pub 4yr OutS; FT/FY"
              8 "Pub 4yr OutS; FT/PY or PT/FY"
              9 "Pub 4yr OutS; PT/PY"
              10 "Pri nfp <4yr/Pri for <2y; FT/FY"
              11 "Pri nfp <4yr/Pri for <2y; FT/PY or PT/FY"
              12 "Pri nfp <4yr/Pri for <2y; PT/PY"
              13 "Pri nfp 4yr; FT/FY"
              14 "Pri nfp 4yr; FT/PY or PT/FY"
              15 "Pri nfp 4yr; PT/PY"
              16 "Pri for >=2y; FT/FY"
              17 "Pri for >=2y; FT/PY or PT/FY"
              18 "Pri for >=2y; PT/PY"
  */
   if tuit_sector=1 then do;
    if tuit_attn=1 then tuit_group=1;
    if tuit_attn=2 then tuit_group=2;
              if tuit_attn=3 then tuit_group=3;
   end;
   if tuit_sector=2 then do;
    if tuit_attn=1 then tuit_group=4;
    if tuit_attn=2 then tuit_group=5;
              if tuit_attn=3 then tuit_group=6;
   end;
   if tuit_sector=3 then do;
    if tuit_attn=1 then tuit_group=7;
    if tuit_attn=2 then tuit_group=8;
              if tuit_attn=3 then tuit_group=9;
   end;
   if tuit_sector=4 then do;
    if tuit_attn=1 then tuit_group=10;
    if tuit_attn=2 then tuit_group=11;
              if tuit_attn=3 then tuit_group=12;
   end;
   if tuit_sector=5 then do;
    if tuit_attn=1 then tuit_group=13;
    if tuit_attn=2 then tuit_group=14;
```

```
            if tuit_attn=3 then tuit_group=15;
  end;
  if tuit_sector=6 then do;
   if tuit_attn=1 then tuit_group=16;
   if tuit_attn=2 then tuit_group=17;
            if tuit_attn=3 then tuit_group=18;
  end;

run;

*There's no Ns less than 100 within each tuit_group group;

/*Group variable by degree, control and attendance
   values for tuit_group:
            1 "Master's Public; FT/PY or PT/FY"
            2 "Master's Public; PT/PY"
            3 "Master's Pri nfp; FT/PY or PT/FY"
            4 "Master's Pri nfp; PT/PY"
            5 "Master's For profit; FT/PY or PT/FY"
            6 "Master's For profit; PT/PY"
            7 "Doctoral Public; FT/PY or PT/FY"
            8 "Doctoral Public; PT/PY"
            9 "Doctoral Pri nfp; FT/PY or PT/FY"
            10 "Doctoral Pri nfp; PT/PY"
            11 "Doctoral For profit; FT/PY or PT/FY"
            12 "Doctoral For profit; PT/PY"
            13 "Prof Public; non-FT/FY"
            14 "Prof Pri nfp; non-FT/FY"
            15 "Prof For profit; non-FT/FY"
            16 "Non-deg or Cert; FT/PY or PT/FY"
            17 "Non-deg or Cert; PT/PY"
  */
  if TUIT_ATTN>1 and TUIT_DEG<3 then do;
    if TUIT_DEG = 1 & TUIT_CONTROL = 1 & TUIT_ATTN = 2 then TUIT_GROUP=1;
    if TUIT_DEG = 1 & TUIT_CONTROL = 1 & TUIT_ATTN = 3 then TUIT_GROUP=2;
    if TUIT_DEG = 1 & TUIT_CONTROL = 2 & TUIT_ATTN = 2 then TUIT_GROUP=3;
    if TUIT_DEG = 1 & TUIT_CONTROL = 2 & TUIT_ATTN = 3 then TUIT_GROUP=4;
    if TUIT_DEG = 1 & TUIT_CONTROL = 3 & TUIT_ATTN = 2 then TUIT_GROUP=5;
    if TUIT_DEG = 1 & TUIT_CONTROL = 3 & TUIT_ATTN = 3 then TUIT_GROUP=6;
    if TUIT_DEG = 2 & TUIT_CONTROL = 1 & TUIT_ATTN = 2 then TUIT_GROUP=7;
    if TUIT_DEG = 2 & TUIT_CONTROL = 1 & TUIT_ATTN = 3 then TUIT_GROUP=8;
    if TUIT_DEG = 2 & TUIT_CONTROL = 2 & TUIT_ATTN = 2 then TUIT_GROUP=9;
    if TUIT_DEG = 2 & TUIT_CONTROL = 2 & TUIT_ATTN = 3 then TUIT_GROUP=10;
    if TUIT_DEG = 2 & TUIT_CONTROL = 3 & TUIT_ATTN = 2 then TUIT_GROUP=11;
    if TUIT_DEG = 2 & TUIT_CONTROL = 3 & TUIT_ATTN = 3 then TUIT_GROUP=12;
   end;

   *For professional practice, Only use control and non full-time;
   if TUIT_DEG=3 & tuit_control=1 & tuit_attn ne 1 then TUIT_GROUP=13;
   if TUIT_DEG=3 & tuit_control=2 & tuit_attn ne 1 then TUIT_GROUP=14;
   if TUIT_DEG=3 & tuit_control=3 & tuit_attn ne 1 then TUIT_GROUP=15;
```

```
    *For non-degree certificates, use attendance status;
    if TUIT_DEG=4 & tuit_attn=2 then TUIT_GROUP=16;
    if TUIT_DEG=4 & tuit_attn=3 then TUIT_GROUP=17;

/****** BOX-COX OUTLIER*****/

/* For each group, flag outliers using a box-cox power transformation
of the TUITION values greater than zero */
%macro bc(n);
data g&n(keep=zrid TUITION tuit_group x);
 set step2;
 x=1;
 if TUITION>0 and tuit_group=&n then output g&n;
run;

proc transreg data = work.g&n  outtest=lamb_g&n noprint;
 model BoxCox(TUITION / lambda = -3 to 3 by .01) = identity(x);
run;

data lambda_results_g&n(keep=x value rename=(value=lambg&n)); set lamb_g&n;
 if statistic='Lambda'; x=1;
run;

*Use lambda from BoxCox model to transform tuition: trans = TUITION ^lambda;
data lambda_tuit_g&n; merge g&n lambda_results_g&n;by x; transg&n=TUITION ** lambg&n
;run;

*Get mean and standard deviation;
proc means DATA=lambda_tuit_g&n noprint;
    output OUT=total_tuit_g&n (drop= _TYPE_ _FREQ_)
    mean (transg&n)=mean_g&n std (transg&n)=std_g&n;
run;

data total_tuit2_g&n;set total_tuit_g&n; x=1;run;

*Identify group outliers based on trans value using +/-2.5 times the standard
 deviation of the transformed distribution created above;
data total_tuit3_g&n; merge lambda_tuit_g&n total_tuit2_g&n;by x;
 outlier=0;
 if transg&n > mean_g&n+(std_g&n*2.5) or transg&n < mean_g&n-(std_g&n*2.5) then outlier=1;
run;
%mend;
%bc(1);%bc(2);%bc(3);%bc(4);%bc(5);%bc(6);%bc(7);%bc(8);%bc(9);
%bc(10);%bc(11);%bc(12);%bc(13);%bc(14);%bc(15);%bc(16);%bc(17);%bc(18);

*Combine all 18 files;
data total_tuit4(keep=ZRID outlier);
set total_tuit3_g1 total_tuit3_g2 total_tuit3_g3 total_tuit3_g4
total_tuit3_g5 total_tuit3_g6 total_tuit3_g7 total_tuit3_g8
total_tuit3_g9 total_tuit3_g10 total_tuit3_g11 total_tuit3_g12
```

```
total_tuit3_g13 total_tuit3_g14 total_tuit3_g15 total_tuit3_g16
total_tuit3_g17 total_tuit3_g18;
run;
proc sort data=total_tuit4; by ZRID;run;

/***** End of BOX-COX OUTLIER CODE *****/

/***** Use IPEDS tuition values to fill in missing values *****/
data step3;
 merge step2(rename=(cipcode1=_cipcode1 cipcode2=_cipcode2 cipcode3=_cipcode3
                  cipcode4=_cipcode4 cipcode5=_cipcode5 cipcode6=_cipcode6 tuition2=tuition2_)
                  in=a)
       total_tuit4;
by ZRID;
if a;

 *Set the outliers to "out of range";
 if outlier=1 then TUITION=-6;
 if TUITION>70000 then outlier=1;
 if TUITION>70000 then TUITION=-6;

 *Create a flag for missing TUITION values;
 if TUITION<0 then tuitmiss=1; else tuitmiss=0;
run;

/* Find the institution-level means by TUIT_GROUP */
proc summary data=step3 nway;
class instid tuit_group;
var tuition;
where tuitmiss=0;
output out=instmed mean(tuition)=tuition_inst_mean;
run;

proc sort data=step3; by instid tuit_group;run;

data step4;
 merge step3(in=a)
       instmed(drop=_type_ rename=(_freq_=count));
by instid tuit_group;
if a;

*Create the final derived variable (TUITION2);
if tuitmiss=0 then TUITION2=TUITION;

 *First, fill in missing with the institution-level means;
   if TUITION2=. and count>1 then TUITION2=round(tuition_inst_mean);

   if tuitmiss=1 & TUITION2 ne . then tuitimpute=1;

   array c1 _cipcode1-_cipcode6;
   array c2 cipcode1-cipcode6;
   do over c1;
    c2=c1/10000;
   end;

  *Second, create categorical variable to decide which IPEDS value to use;
    if 1<=tuit_group<=6 and tuit_voc=0 then tuit_ipeds = 1;
    if 7<=tuit_group<=18 and tuit_voc=0 then tuit_ipeds = 2;

               /*VOCATIONAL SCHOOLS. Most common CIP (CIPCODE1). First if SR matches, then
student interview if missing*/
               if tuit_voc=1 & becipmaj1=cipcode1 then tuit_ipeds = 3;
               if tuit_voc=1 & tuit_ipeds=. & becipmaj1="-9" & n16bmj1spe=cipcode1 then tuit_ipeds =
3;

               /*VOCATIONAL SCHOOLS. 2nd most common CIP*/
               if tuit_voc=1 & tuit_ipeds=. & becipmaj1=cipcode2 then tuit_ipeds = 4;
               if tuit_voc=1 & tuit_ipeds=. & becipmaj1="-9" & n16bmj1spe=cipcode2 then tuit_ipeds =
4;

               /*VOCATIONAL SCHOOLS. 3rd most common CIP*/
               if tuit_voc=1 & tuit_ipeds=. & becipmaj1=cipcode3 then tuit_ipeds = 5;
               if tuit_voc=1 & tuit_ipeds=. & becipmaj1="-9" & n16bmj1spe=cipcode3 then tuit_ipeds =
5;

               /*VOCATIONAL SCHOOLS. 4th most common CIP*/
               if tuit_voc=1 & tuit_ipeds=. & becipmaj1=cipcode4 then tuit_ipeds = 6;
               if tuit_voc=1 & tuit_ipeds=. & becipmaj1="-9" & n16bmj1spe=cipcode4 then tuit_ipeds =
6;

               /*VOCATIONAL SCHOOLS. 5th most common CIP*/
               if tuit_voc=1 & tuit_ipeds=. & becipmaj1=cipcode5 then tuit_ipeds = 7;
               if tuit_voc=1 & tuit_ipeds=. & becipmaj1="-9" & n16bmj1spe=cipcode5 then tuit_ipeds =
7;

               /*VOCATIONAL SCHOOLS. 6th most common CIP*/
    if tuit_voc=1 & tuit_ipeds=. & becipmaj1=cipcode6 then tuit_ipeds = 8;
               if tuit_voc=1 & tuit_ipeds=. & becipmaj1="-9" & n16bmj1spe=cipcode6 then tuit_ipeds =
8;

               /*VOCATIONAL SCHOOLS. Everyone else*/
               if tuit_voc=1 & tuit_ipeds=. then tuit_ipeds = 9;


  *Third, replace missing TUITION2 with IPEDS values according to the TUIT_IPEDS;
               if tuition2_>=0 & fee2>=0 & tuit_ipeds=1 & TUITION2<0 then TUITION2 =
tuition2_+fee2;
               if tuition3>=0 & fee3>=0 & tuit_ipeds=2 & TUITION2<0 then TUITION2 = tuition3+fee3;
               if tuit_ipeds=3 & TUITION2<0 then TUITION2 = chg1py3;
               if tuit_ipeds=4 & TUITION2<0 then TUITION2 = ciptuit2;
               if tuit_ipeds=5 & TUITION2<0 then TUITION2 = ciptuit3;
               if tuit_ipeds=6 & TUITION2<0 then TUITION2 = ciptuit4;
               if tuit_ipeds=7 & TUITION2<0 then TUITION2 = ciptuit5;
```

```
        if tuit_ipeds=8 & TUITION2<0 then TUITION2 = ciptuit6;
        if tuit_ipeds=9 & TUITION2<0 then TUITION2 = chg1py3;

        if tuitmiss=1 & tuitimpute ne 1 & TUITION2 ne . & tuit_attn=1 then tuitimpute = 2;
        if tuitmiss=1 & tuitimpute ne 1 & TUITION2 ne . & tuit_attn in (2,3) then tuitimpute = 98;


        if 4<=tuit_group<=6 & instid=140164 & TUITION2<0 then TUITION2 = (2660+1003)*2;
        if 7<=tuit_group<=9 & instid=140164 & TUITION2<0 then TUITION2 = (9388+1003)*2;

        if tuitmiss=1 & tuitimpute not in (1,2,98) & TUITION2 ne . & tuit_attn=1 then tuitimpute =
3;
        if tuitmiss=1 & tuitimpute not in (1,2,98) & TUITION2 ne . & tuit_attn in (2,3) then
tuitimpute = 99;

  x=1;
run;

/**** Find ratio between FTFY and the other two statuses at the  ***/
/**** sector level and then adjust the tuition for attendance    ***/
proc summary data=step4 nway;
class sector11 tuit_attn;
var tuition2;
where tuition2>0 and tuitimpute not in (98,99);
output out=inst_att(drop=_type_ _freq_) mean(tuition2)=tuition_inst_att;
run;

proc summary data=step4 nway;
class sector11 tuit_attn;
var tuition2;
where tuition2>0 and tuitimpute not in (98,99);
weight initialstudwt;
output out=inst_attw(drop=_type_ _freq_) mean(tuition2)=wtuition_inst_att;
run;

proc transpose data=inst_att out=inst_att2 prefix=tuition_inst_att;
  by sector11;
  id tuit_attn;
  var tuition_inst_att;
run;
proc transpose data=inst_attw out=inst_attw2 prefix=wtuition_inst_att;
  by sector11;
  id tuit_attn;
  var wtuition_inst_att;
run;

*Unweighted mean ratio by sector;
data inst_att3;
retain tuit1_ratio1 tuit2_ratio1 tuit3_ratio1 tuit4_ratio1 tuit5_ratio1 tuit6_ratio1
    tuit7_ratio1 tuit8_ratio1 tuit9_ratio1 tuit10_ratio1 tuit11_ratio1
    tuit1_ratio2 tuit2_ratio2 tuit3_ratio2 tuit4_ratio2 tuit5_ratio2 tuit6_ratio2
```

```
       tuit7_ratio2 tuit8_ratio2 tuit9_ratio2 tuit10_ratio2 tuit11_ratio2;
 set inst_att2;
 x=1;
 array r1{11} tuit1_ratio1 tuit2_ratio1 tuit3_ratio1 tuit4_ratio1 tuit5_ratio1 tuit6_ratio1
            tuit7_ratio1 tuit8_ratio1 tuit9_ratio1 tuit10_ratio1 tuit11_ratio1;
 array r2{11} tuit1_ratio2 tuit2_ratio2 tuit3_ratio2 tuit4_ratio2 tuit5_ratio2 tuit6_ratio2
            tuit7_ratio2 tuit8_ratio2 tuit9_ratio2 tuit10_ratio2 tuit11_ratio2;
 do i=1 to 11;
 if sector11=i then r1{i}=round(tuition_inst_att2/tuition_inst_att1,.01);
 if sector11=i then r2{i}=round(tuition_inst_att3/tuition_inst_att1,.01);
 end;
run;
data inst_att4(keep=x tuit1_ratio1--tuit11_ratio1 tuit1_ratio2--tuit11_ratio2); set inst_att3;by x;if
last.x;run;

*Weighted mean ratios by sector;
data inst_attw3;
retain wtuit1_ratio1 wtuit2_ratio1 wtuit3_ratio1 wtuit4_ratio1 wtuit5_ratio1 wtuit6_ratio1
       wtuit7_ratio1 wtuit8_ratio1 wtuit9_ratio1 wtuit10_ratio1 wtuit11_ratio1
       wtuit1_ratio2 wtuit2_ratio2 wtuit3_ratio2 wtuit4_ratio2 wtuit5_ratio2 wtuit6_ratio2
       wtuit7_ratio2 wtuit8_ratio2 wtuit9_ratio2 wtuit10_ratio2 wtuit11_ratio2;
 set inst_attw2;
 x=1;
 array r1{11} wtuit1_ratio1 wtuit2_ratio1 wtuit3_ratio1 wtuit4_ratio1 wtuit5_ratio1 wtuit6_ratio1
             wtuit7_ratio1 wtuit8_ratio1 wtuit9_ratio1 wtuit10_ratio1 wtuit11_ratio1;
 array r2{11} wtuit1_ratio2 wtuit2_ratio2 wtuit3_ratio2 wtuit4_ratio2 wtuit5_ratio2 wtuit6_ratio2
             wtuit7_ratio2 wtuit8_ratio2 wtuit9_ratio2 wtuit10_ratio2 wtuit11_ratio2;
 do i=1 to 11;
 if sector11=i then r1{i}=round(wtuition_inst_att2/wtuition_inst_att1,.01);
 if sector11=i then r2{i}=round(wtuition_inst_att3/wtuition_inst_att1,.01);
 end;
run;
data inst_attw4(keep=x wtuit1_ratio1--wtuit11_ratio1 wtuit1_ratio2--wtuit11_ratio2); set
inst_attw3;by x;if last.x;run;

*Using the weighted mean ratios, adjust the tuition for attendance;
data step5;
 merge step4 inst_att4 inst_attw4; by x;

 array r1{11} wtuit1_ratio1 wtuit2_ratio1 wtuit3_ratio1 wtuit4_ratio1 wtuit5_ratio1 wtuit6_ratio1
             wtuit7_ratio1 wtuit8_ratio1 wtuit9_ratio1 wtuit10_ratio1 wtuit11_ratio1;
 array r2{11} wtuit1_ratio2 wtuit2_ratio2 wtuit3_ratio2 wtuit4_ratio2 wtuit5_ratio2 wtuit6_ratio2
             wtuit7_ratio2 wtuit8_ratio2 wtuit9_ratio2 wtuit10_ratio2 wtuit11_ratio2;
 do i=1 to 11;
 if tuitmiss=1 & tuitimpute in (98,99) & sector11=i & tuit_attn=2 then
TUITION2=round(TUITION2*r1{i});
 if tuitmiss=1 & tuitimpute in (98,99) & sector11=i & tuit_attn=3 then
TUITION2=round(TUITION2*r2{i});
 end;
 if tuitmiss=1 & tuitimpute=98 & tuit_attn=2 then tuitimpute = 4;
 if tuitmiss=1 & tuitimpute=99 & tuit_attn=2 then tuitimpute = 5;
 if tuitmiss=1 & tuitimpute=98 & tuit_attn=3 then tuitimpute = 4;
 if tuitmiss=1 & tuitimpute=99 & tuit_attn=3 then tuitimpute = 5;

 if TUITION2=. then tuitimpute = 6;
run;

/*** Fill in the maining missing cases using the ***/
/*** overall weighted mean TUITION by TUIT_GROUP ***/
proc summary data=step5 nway;
class tuit_group;
var TUITION2;
where TUITION2>0;
weight initialstudwt;
output out=wtuition_tuit_group(drop=_type_ _freq_) mean(TUITION2)=wtuition_tuit_group;
run;

proc sort data=step5;by tuit_group;run;
data step6;
 merge step5 wtuition_tuit_group;by tuit_group;
 if TUITION2=. then TUITION2=round(wtuition_tuit_group);

 ZTUITION=tuitimpute;
 if tuitmiss=0 then ZTUITION=0;
 if instid=211893 then ZTUITION=3;
run;

proc sort data=step6;by zrid;run;

data x.tuition_ug;
merge x.alldata(keep=ZRID STYPELST in=a)
      x.attend(keep=ZRID STUDMULT)
      step6(keep=ZRID TUITION2 ZTUITION initialstudwt);
by ZRID;
if A;

/* Set to skip if UG students didn't attend one institution */
if STUDMULT>1 then do; TUITION2=-3; end;
else if STYPELST=1 then do; TUITION2=TUITION2_ug; end;
else if STYPELST>1 then do; TUITION2=TUITION2_gr; end;
```

<table>
<tr><td><b>Variable Name</b></td><td><b>UGDEG</b></td></tr>
<tr><td><b>Variable Label</b></td><td>Undergraduate degree program</td></tr>
<tr><td><b>Description</b></td><td>Undergraduate student’s degree program during the 2015–16 academic year.</td></tr>
<tr><td><b>Assigned Values</b></td><td>
<table>
<tr><th>Value</th><th>Value label</th></tr>
<tr><td>1</td><td>Certificate</td></tr>
<tr><td>2</td><td>Associate's degree</td></tr>
<tr><td>3</td><td>Bachelor's degree</td></tr>
<tr><td>4</td><td>Not in a degree program or others</td></tr>
</table>
</td></tr>
<tr><td><b>Data Source(s)</b></td><td>NPSAS:16 Interview, NPSAS:16 Student Records, FAFSA:16, NSC:16</td></tr>
</table>

Component Derived Variables	STYPELST
Applies to	All undergraduate study members
Availability	NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16
Change History	Unchanged from NPSAS:04 through NPSAS:16; Name change from DEGLAST in NPSAS:2000; NSC has only been a data source since NPSAS:12
N:16 Usage Statistics	Number of uses: 2,910 (UG); 0 (GR)
Programming Narrative	1. Create individual degree variables for all the sources (interview, student records, and CPS) 2. Set the source degree variables to missing when they conflict with the level of the institution a. If level4 is 1 (less-than 2 yr) but the degree isn't either a certificate or undergrad classes, set to missing b. If level4 is not 4 (4-year doctorate granting), but the degree indicates it is (7,8, or 9), set to missing c. If level4 is 2 (2-year institution), but degree is not (1,2,4,6), set to missing (NEW STEP) 3. First, take the student interview as the degree if it is not just classes (4 or 10) AND the source for STYPELST was the interview (ZSTYPE in (1,3,5,7)) 4. Second, take student records as the degree if it is not just classes (4 or 10) 5. Third, take CPS as the degree if it is not just classes (4 or 10) 6. Fourth, take NSC as the degree. 7. If STYPELST = 3, set the degree to 8 (professional practice). New way should have much fewer replacements here (<20) 8. If degree is still missing and If STYPELST = 1 and any of the source degrees indicate the student just took undergraduate classes (degsi = 4 or degsr = 4 or degcps = 4) set degree to 4 9. If degree is still missing and If STYPELST > 1 and any of the source degrees indicate the student just took graduate classes (degsi = 10 or degsr = 10 or degcps = 10) set degree to 10 10. Fix cases when 2/3 sources say one-degree type, but we took another (like STYPELST) a. If degree from student records matches degree from cps, and both are greater than zero, set degree to the student records degree b. If degree from the student interview AND student records is just undergraduate classes (4) or just graduate classes (10), then use that value, rather than the valid degree from CPS.
SAS Code	/*------------------------ UGDEG: Degree program ----------------------*/ /* variable used: STYPELST LEVEL4 LEVEL TDEGREE BENLADEG C16DEG */ /*------- (1) create variables with same categories in all 3 sources ------- DEGSI: Degree programs from student interview DEGSR: Degree programs from student records DEGCPS: Degree programs from CPS DEGNCS: Degree programs from NSC Values: 1=Certificate 2=Associate's degree 3=Bachelor's degree 4=Not in a undergraduate degree program or others

```
contradicting cases with STYPELST:
-1=UG student who has grad degree
-2=grad student who has UG degree
*/
/*---------------- Undergraduate student ----------------*/
IF STYPELST=1 THEN DO;

/* DEGSI */
  IF TDEGREE=2 THEN DEGSI=1;
  ELSE IF TDEGREE=3 THEN DEGSI=2;
  ELSE IF TDEGREE=4 THEN DEGSI=3;
  ELSE IF TDEGREE=1 THEN DEGSI=4;
  ELSE IF TDEGREE>0 THEN DEGSI=-1;   /* UG status with grad deg */

/* DEGSR */
  IF BENLADEG=2 THEN DEGSR=1;
  ELSE IF BENLADEG=3 THEN DEGSR=2;
  ELSE IF BENLADEG=4 THEN DEGSR=3;
  ELSE IF BENLADEG=1 THEN DEGSR=4;
  ELSE IF BENLADEG>0 THEN DEGSR=-1;  /* UG status with grad deg */

/* DEGCPS */
  IF C16DEG IN (5,6,7) THEN DEGCPS=1;
  ELSE IF C16DEG IN (3,4) THEN DEGCPS=2;
  ELSE IF C16DEG IN (1,2) THEN DEGCPS=3;
  ELSE IF C16DEG=9 THEN DEGCPS=4;
  ELSE IF C16DEG>0 THEN DEGCPS=-1;   /* UG status with grad deg */

/* DEGNSC */
  IF CHDEGTYP=1 THEN DEGNSC=1;
  ELSE IF CHDEGTYP=2 THEN DEGNSC=2;
  ELSE IF CHDEGTYP=3 THEN DEGNSC=3;
  ELSE IF CHDEGTYP>4 THEN DEGNSC=-1; /* UG status with grad deg */

END;

/* more contradicting cases with level:
-3=student enrolled at a lt-2-yr college but
  has degree type other than "certificate" and "no degree"
-4=student who did not enrolled at a doc-granting college
  but has doctoral degree
-5=student who enrolled at a 2-yr institution but has
  degree type other than "certificate", "AS", "UG classes"
  and "Post-BA or post-master's certificate"
*/

ARRAY D DEGSI DEGSR DEGCPS DEGNSC;

DO OVER D;
  IF LEVEL4=1 AND D NOT IN (1,4) AND D>0 THEN D=-3;
  IF LEVEL4~=4 AND D IN (7,8,9) THEN D=-4;
  IF LEVEL4=2 AND D NOT IN (1,2,4,6) THEN D=-5;
END;

/*--- (2) use the 3 sources to create one variable that has all degree programs ---*/
/* DEGREE: Degree programs (all students)
  - by using student interview first, then student records, CPS then NSC;
  - if data is contradicting, treat as missing.
  - at the end if missing, treat as "no program"
*/

/* if missing, take student interview first, then student records, CPS then NSC */
IF DEGSI>0 AND DEGSI NOT IN (4,10) AND ZSTYPE IN (1,3,5,7) THEN DEGREE=DEGSI;
  ELSE IF DEGSR>0 AND DEGSR NOT IN (4,10) THEN DEGREE=DEGSR;
  ELSE IF DEGCPS>0 AND DEGCPS NOT IN (4,10) THEN DEGREE=DEGCPS;
  ELSE IF DEGNSC>0 AND DEGNSC NOT IN (4,10) THEN DEGREE=DEGNSC;

/* set professional practice student to missing, these few cases came from NSC */
IF STYPELST=2 AND DEGREE=8 THEN DEGREE=.;

/* set professional-practice student to "8" */
IF STYPELST=3 THEN DEGREE=8;

/* if DEGREE still missing and one of sources agreed, set to UG/GR classes only */
IF DEGREE=. THEN DO;
  IF STYPELST=1 AND (DEGSI=4 OR DEGSR=4 OR DEGCPS=4 OR DEGNSC=4) THEN
DEGREE=4;
  IF STYPELST>1 AND (DEGSI=10 OR DEGSR=10 OR DEGCPS=10 OR DEGNSC=10)
THEN DEGREE=10;
END;

/*----- (3) Fix cases when 2/3 sources say one degree type, but we took another ---*/
IF DEGSR>0 AND DEGCPS>0 AND DEGSR=DEGCPS THEN DEGREE=DEGSR;
IF (DEGSI=4 AND DEGSR=4) THEN DEGREE=4;
IF (DEGSI=10 AND DEGSR=10) THEN DEGREE=10;

/*--------- (4) create UGDEG (UG) and GRADDEG (GRAD) from DEGREE ---------*/

IF STYPELST=1 THEN DO; UGDEG=DEGREE; GRADDEG=-3; END;
ELSE IF STYPELST>1 THEN DO; GRADDEG=DEGREE-4; UGDEG=-3; END;
```

Variable Name	**VADODAMT**
Variable Label	Federal veterans education benefits and Department of Defense amount
Description	Total amount of federal veterans education benefits and Department of Defense grants (including ROTC) received during the 2015–16 academic year. This does not include state or institutional veterans education benefits or military/armed forces grants.
Assigned Values	

Value	Value label
0	{Zero}
c	{Continuous}

Data Source(s)	VBA:16, NPSAS:16 Student Records
Component Derived Variables	VETBEN, DODAMT
Applies to	All study members
Availability	NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16
Change History	NPSAS:96 through NPSAS:08 also used CPS as a source; NPSAS:2000 through NPSAS:12 used the interview as a source; NPSAS:16 used VBA as a source.
N:16 Usage Statistics	Number of uses: 455 (UG); 8 (GR)
Programming Narrative	Sum VETBEN and DODAMT

SAS Code	* Source variable vetben="Veterans' benefits";
	* Source variable dodamt="Department of Defense (military) aid";
	  ***;
	  * Sum VETBEN and DODAMT;
	  vadodamt = vetben + dodamt;
	  if missing(vadodamt) = 1 then
	   do;
	    vadodamt = .x;
	   end;

Variable Name	**VETBEN**
Variable Label	Federal veterans education benefits amount
Description	Total amount of federal veterans education benefits received during the 2015–16 academic year.

Assigned Values	**Value** **Value label**
	0 {Zero}
	c {Continuous}

Data Source(s)	VBA:16
Component Derived Variables	Not applicable
Applies to	All study members
Availability	NPSAS:08, NPSAS:12, NPSAS:16
Change History	NPSAS:08 used the interview, SR, and CPS as a source, NPSAS:12 used only the interview and SR, and NPSAS:16 moved to only use VBA data.
N:16 Usage Statistics	Number of uses: 241 (UG); 18 (GR)
Programming Narrative	Take the data as is from the VBA file (pd_ay15). In the event no amount exists, set to zero.

SAS Code	* Source variable pd_ay15="Benefits amount: Total (2015–16)";
	  ***;
	  vetben = round(pd_ay15);
	  if missing(vetben) = 1 then vetben = 0;

Variable Name	**VOCHELP**
Variable Label	Vocational rehabilitation and training amount
Description	Total amount of vocational rehabilitation and job training grants received during the 2015–16 academic year.

Assigned Values	**Value** **Value label**
	0 {Zero}
	c {Continuous}

Data Source(s)	NPSAS:16 Student Records
Component Derived Variables	Not applicable
Applies to	All undergraduate study members
Availability	NPSAS:96, NPSAS:2000, NPSAS:04, NPSAS:08, NPSAS:12, NPSAS:16
Change History	Unchanged from NPSAS:96 through NPSAS:16.
N:16 Usage Statistics	Number of uses: 306 (UG); 0 (GR)
Programming Narrative	Sum state Workforce Investment Opportunity Act (WIOA) fund/job training (type=18 and source=state), government WIOA/job training (type=18 and source=government), and Other Aid state and federal WIOA/job training benefits (type=18 and source=other).

SAS Code

```
* Source variable cfacvans="Received any financial aid";
* Source variables cf01sttyp-cf03sttyp="State aid program type";
* Source variables c01stamt-c03stamt="State aid program amount";
* Source variables cfa1othamt-cfa3othamt="Other aid amount";
* Source variables cfa1othsrc-cfa3othsrc="Other aid source";
* Source variables cfa1othtyp-cfa3othtyp="Other aid type";
* Source variables cfa1govamt–cfa3govamt="Private aid program amount";
* Source variables cfa1govtyp-cfa3govtyp="Private aid program type";
***************************************************************************;
    array stamte(3) stamte1-stamte3;
  * array srstamt(3) c01stamt c02stamt c03stamt;
  * array srsttyp(3) cf01sttyp cf02sttyp cf03sttyp;
  array othamte(3) othamte1-othamte3;
  * array srothamt(3) cfa1othamt cfa2othamt cfa3othamt;
  * array srothsrc(3) cfa1othsrc cfa2othsrc cfa3othsrc;
  * array srothtyp(3) cfa1othtyp cfa2othtyp cfa3othtyp;
  array govamte(3) govamte1-govamte3;
  * array srgovamt(3) cfa1govamt cfa2govamt cfa3govamt;
  * array srgovtyp(3) cfa1govtyp cfa2govtyp cfa3govtyp;
  do i = 1 to 3;
    * Use student records: Sum (1) state WIOA/job training, (2) government
    * WIOA/job training, and (3) Other Aid state and federal WIOA/job
    * training benefits.
    * cstamt if cfsttype = 18
    * cfagovamt if cfagovtyp = 5
    * cfothamt if cfaothsrc in (2 3) & cfaothtyp = 18;
    stamte(i) = 0;
    if cfastataid ~= 0 then
     do;
      if srsttyp(i) = 18 then
```

```
       do;
        stamte(i) = srstamt(i);
        if missing(stamte(i)) = 1 then stamte(i) = .m;
       end;
      else if cfastataid = .x then stamte(i) = .x;
     end;

    othamte(i) = 0;
    if cfaothaid ~= 0 then
     do;
      if srothsrc(i) in (2,3) & srothtyp(i) = 18 then
       do;
        othamte(i) = srothamt(i);
        if missing(othamte(i)) = 1 then othamte(i) = .m;
       end;
      else if cfaothaid = .x then othamte(i) = .x;
     end;

    govamte(i) = 0;
    if cfaothgov ~= 0 then
     do;
      if srgovtyp(i) = 5 then
       do;
        govamte(i) = srgovamt(i);
        if missing(govamte(i)) = 1 then govamte(i) = .m;
       end;
      else if cfaothgov = .x then govamte(i) = .x;
     end;
  end;

  array forVochelp(9) stamte1 stamte2 stamte3 othamte1 othamte2 othamte3 govamte1
govamte2 govamte3;
  do i = 1 to 9;
    if forVochelp(i) > 0 then
     do j = 1 to 9;
      if forVochelp(j) = .x then forVochelp(j) = 0;
     end;
  end;

  * Applies to: All undergraduate study members;
  if stypelst > 1 then do; vochelp = .y; vochelp_minimp = .y; end;
  else if stypelst = 1 then
   do;
    vochelp = stamte1 + stamte2 + stamte3 + othamte1 + othamte2 + othamte3 + govamte1
+ govamte2 + govamte3;
    vochelp_minimp = sum(of stamte1 stamte2 stamte3 othamte1 othamte2 othamte3
govamte1 govamte2 govamte3);
    if vochelp_minimp = . then vochelp_minimp = 0;
    do i = 1 to 9;
      if forVochelp(i) = .m then Vochelp_minimp = Vochelp_minimp + 1;
    end;
```

```
if missing(vochelp) = 1 then
  do;
    vochelp = .x;
  end;
```

References

Wine, J., Siegel, P., and Stollberg, R. (2018). *2015–16 National Postsecondary Student Aid Study (NPSAS:16) Data File Documentation* (NCES 2018-482). U.S. Department of Education. Washington, DC: National Center for Education Statistics.

Appendix A – Index of Variables by Variable Label

PAGE

Age as of 12/31/2015 4

Age group as of 12/31/2015 4

Athletic scholarships amount 62

Attendance pattern 7

Cumulative amount borrowed for graduate education 9

Cumulative amount borrowed for undergraduate and graduate education 10

Cumulative amount borrowed for undergraduate education 8

Cumulative Direct Subsidized & Unsubsidized Loans for undergraduate and graduate education 117

Cumulative Direct Subsidized and Unsubsidized Loans for graduate education 116

Cumulative Direct Subsidized and Unsubsidized Loans for undergraduate education 115

Cumulative federal loan amount for graduate education 37

Cumulative federal loan amount for undergraduate and graduate education 38

Cumulative federal loan amount for undergraduate education 36

Cumulative federal loan amount owed for graduate education 44

Cumulative federal loan amount owed for undergraduate education 43

Cumulative federal loan amount owed for undergraduate or graduate education 46

Cumulative federal loan amount owed, principal & interest, graduate education 40

Cumulative federal loan amount owed, principal & interest, undergraduate and graduate education 42

Cumulative federal loan amount owed, principal & interest, undergraduate education 39

Cumulative nonfederal loan amount for graduate education 92

Cumulative non-federal loan amount for undergraduate education 88

Department of Defense (military) aid 19

Dependency status 16

Dependency status (3 categories) 17

PAGE

Dependent students: Parents' income 18

Dependents: Has any dependents 15

Direct PLUS Loans to graduate students 47

Direct PLUS Loans to parents amount 103

Direct Subsidized and Unsubsidized Loans amount 110

Direct Subsidized Loans amount 111

Direct Unsubsidized Loans amount 112

Employer aid (student & parents) 32

Expected Family Contribution 21

Federal campus-based aid (Perkins, SEOG, FWS) 13

Federal Pell grant amount 101

Federal Supplemental Educational Opportunity Grant (SEOG) amount 108

Federal TEACH grant amount 136

Federal veterans education benefits amount 159

Federal veterans education benefits and Department of Defense amount 158

Federal work-study amount 141

Graduate degree program 47

Graduate fellowship amount 54

Graduate research assistantship amount 57

Graduate teaching assistantship amount 60

Independent students: student and spouse's income 67

Institution military/armed forces grants amount 72

Institutional aid amount 74

Institutional grants amount 69

Institutional loans amount 70

Institutional merit-only grants amount 72

Institutional need-based grants amount 75

Institutional non-need & merit grants amount 77

Institutional non-need grants amount 77

Institutional tuition & fee waivers amount 82

Institutional tuition waivers for staff 30

Institutional veterans education benefits amount 80

Institutional work-study amount 78

PAGE

Marital status 109

Military type 83

Military type (for dependency) 84

NPSAS institution type: Graduate (with multiple) 6

NPSAS institution type: Undergraduate (with multiple) 5

Number of full-time months enrolled 82

Number of institutions attended 128

Number of total months enrolled 35

Other federal grants (not Title IV) amount 96

Other graduate assistantship amount 51

Outside grants (private & employer) amount 99

Outside sources aid amount 99

Perkins loan amount 102

Private (alternative) loans amount 107

Private source grants amount 104

State aid amount 112

State grants amount 118

State grants based both on need and merit amount 123

State loans amount 119

State merit-only grants amount 120

State military/armed forces grants amount 122

State need-based grants amount 113

State need-based only grants amount 125

State non-need & merit grants amount 114

State non-need grants amount 126

State veterans education benefits amount 129

State work-study amount 131

Student budget (attendance adjusted) 10

Student budget minus all aid 86

Student budget minus all grants 87

Student type indicator 132

Title IV loans (excludes Parent PLUS Loans) amount 134

Title IV loans (includes Parent PLUS Loans) amount 135

	PAGE
Total aid amount	143
Total assistantships amount	50
Total federal aid (excludes Veterans'/DOD) amount	136
Total federal grants (excludes Veterans'/DOD) amount	138
Total federal loans (excludes Parent PLUS Loans) amount	139
Total federal loans (includes Parent PLUS Loans) amount	140
Total federal Title IV aid amount	142
Total grants amount	144
Total income (categorical)	66
Total income (continuous)	14
Total income by dependency (categorical)	64
Total loans (excluding Parent PLUS Loans) amount	144
Total other type of aid (Direct PLUS, job training, VA) amount	100
Total work-study amount	145
Tuition and fees minus all grants	87
Tuition and fees paid	146
Undergraduate degree program	155
Vocational rehabilitation and training amount	160

Appendix B – Flowcharts Showing Relationships Among Components of the Total Student Aid Variable (TOTAID)

The figures in this appendix visually portray how selected financial aid measures in the 2015–16 National Postsecondary Student Aid Study (NPSAS:16) sum to equal the total amount of aid students received. Every figure is arranged as a flowchart, with each box representing a specific category of aid. The topmost box in each figure represents the largest composite of aid in that figure (that is, the most comprehensive aggregate), such as total aid or total grants. Each level below represents an increasingly granular depiction of smaller components of aid. The boxes are also arranged horizontally to cluster components by aid type (grants, loans, work-study, or other) or source (federal, state, institutional, or other).

The text, background shading, background color, and outline pattern of each box indicate key characteristics of the corresponding aid variable. These characteristics are also indicated by a short code in the lower right corner of each box for readers with vision limitations or who are viewing this report rendered in black and white. Each box begins with the variable name, followed by an abbreviated description. For compactness, this description is often shorter than the variable label presented in the body of this report.

Boxes with variable names in all uppercase letters and solid background shading represent variables available for online analysis on NCES DataLab (PowerStats, QuickStats, and TrendStats) at https://nces.ed.gov/datalab. Boxes with mixed-case variable names and striped background shading represent interim variables that are not available on DataLab, typically because they were awarded to too few students for the amounts to be measured reliably.

The type or source of aid is indicated by both the color of the box and a short code in the bottom right corner. In figures B-1 through B-7, bright green boxes with the code “FA” indicate federal aid, light orange boxes with the code “SA” indicate state aid, olive green boxes with the code “IA” indicate institutional aid, and blue boxes with the code “OSA” indicate other source aid. In figures B-8 through B-14, olive boxes with the code “G” indicate grants, orange boxes with the code “L” indicate loans, blue boxes with the code “WS” indicate work-study, and purple boxes with the code “OT” indicate other types of aid. The topmost composite variables in the figures have a background color of gray fading into white.

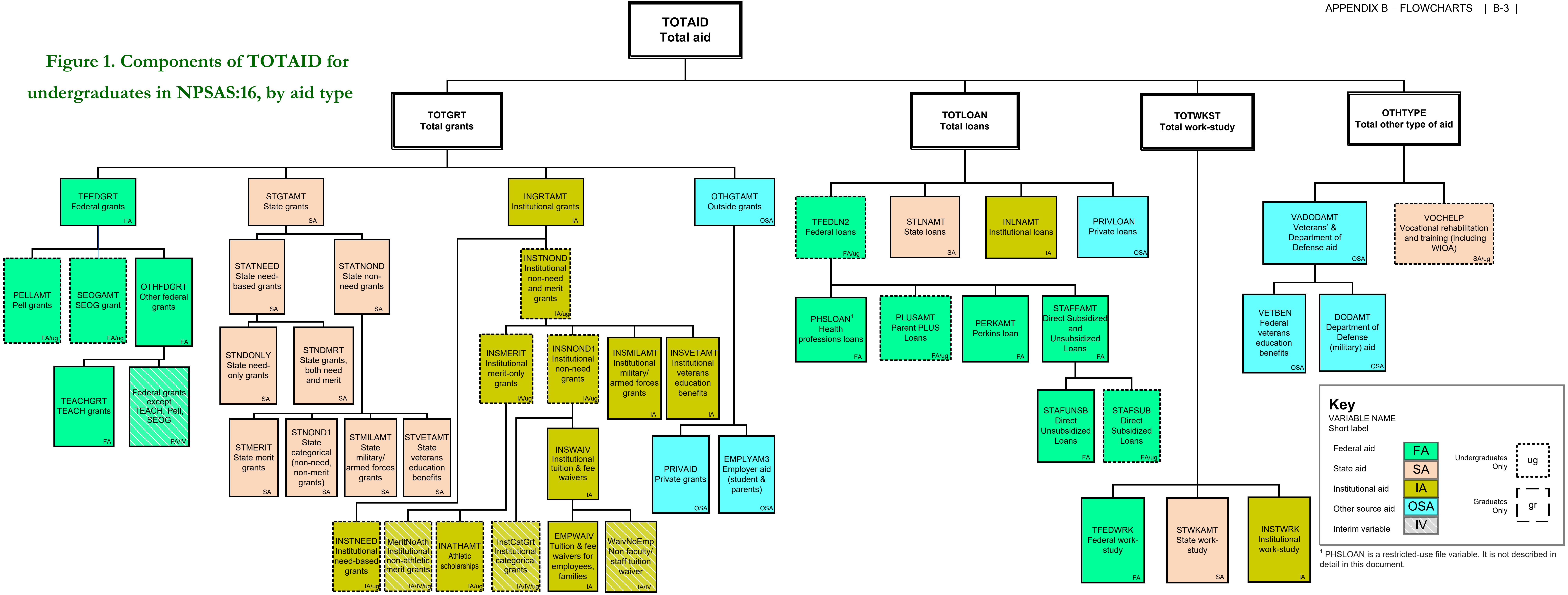
The figures also categorize the NPSAS:16 aid measures by whether they apply to undergraduates, graduate students, or both. Boxes outlined with short dashes and containing the code “ug” in the lower right corner apply to undergraduates only. Boxes outlined with long dashes and containing the code “gr” apply to graduate students only. Boxes with solid outlines and neither the “ug” nor “gr” code apply to both undergraduates and graduate students. Additionally, eight figures are limited to undergraduates only (B-1, B-3, B-8, and B-12) or graduate students only (B-2, B-4, B-9, and B-13).

As with the body of the report, this appendix is not intended to represent the entirety of all aid variables. The NPSAS studies include many variables that are not shown in this report but that represent various combinations of aid. For example, the variable TOTAID4 sums all aid to undergraduates except Direct PLUS loans to parents. Likewise, the variable PRIVAMT consists of all private source grants and private loans for both undergraduates and graduate students. Codebooks containing all variable names and descriptions for each NPSAS study on DataLab, separated by undergraduate or graduate student status, can be viewed at or downloaded from DataLab.

The order of the figures is as follows. Figure B-1 depicts the components of total aid by aid type for undergraduates, and figure B-2 does the same for graduate students. Figures B-3 and B-4 portray the components of total grant aid for undergraduates and graduate students, respectively. Figure B-5 shows the components of total loans for all students (both undergraduates and graduate students). Figures B-6 and B-7 depict the components of total work-study and total other types of aid, respectively, for all students.

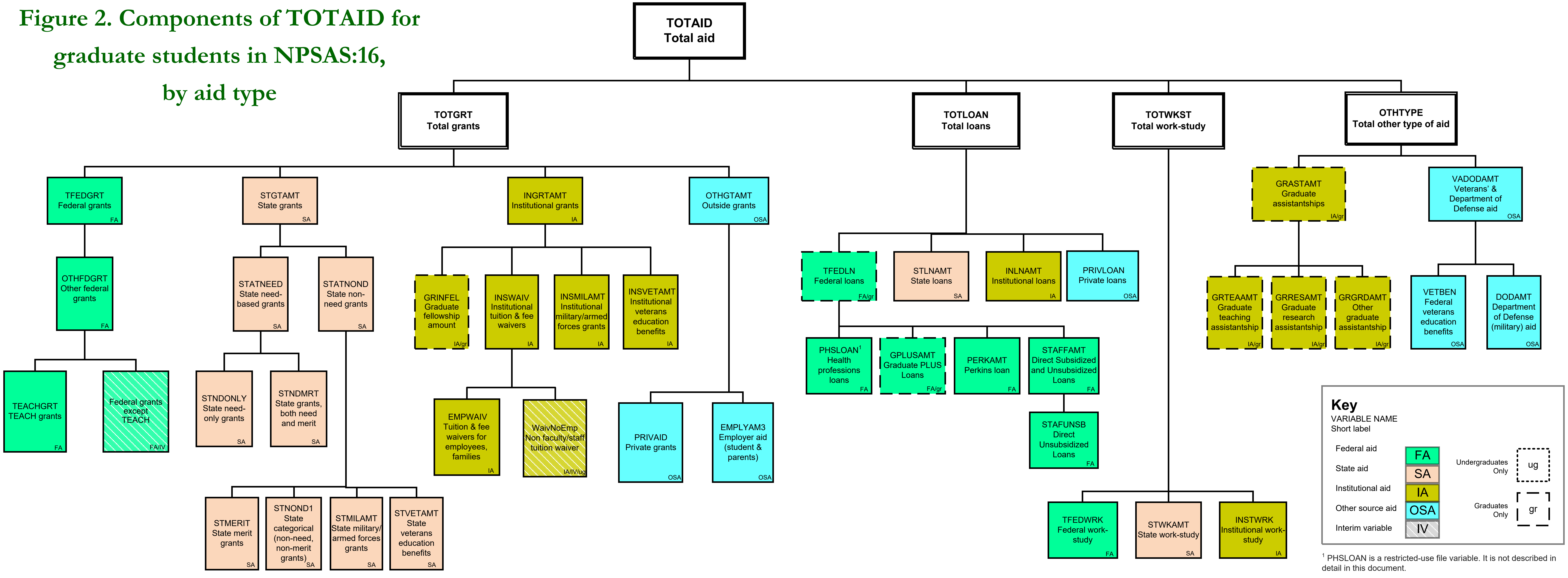
Figures B-8 and B-9 portray the components of total aid by source for undergraduates and graduate students, respectively. Figure B-10 shows the components of total federal aid for all students, and figure B-11 does likewise for total state aid for all students. Figures B-12 and B-13 present the components of institutional aid for undergraduates and graduate students, respectively. Finally, figure B-14 depicts the components of aid from other sources for all students.

Figure 1. Components of TOTAID for undergraduates in NPSAS:16, by aid type



[1] PHSLOAN is a restricted-use file variable. It is not described in detail in this document.

Figure 2. Components of TOTAID for graduate students in NPSAS:16, by aid type



[1] PHSLOAN is a restricted-use file variable. It is not described in detail in this document.

Figure 3. Components of TOTGRT for undergraduates in NPSAS:16, by aid source

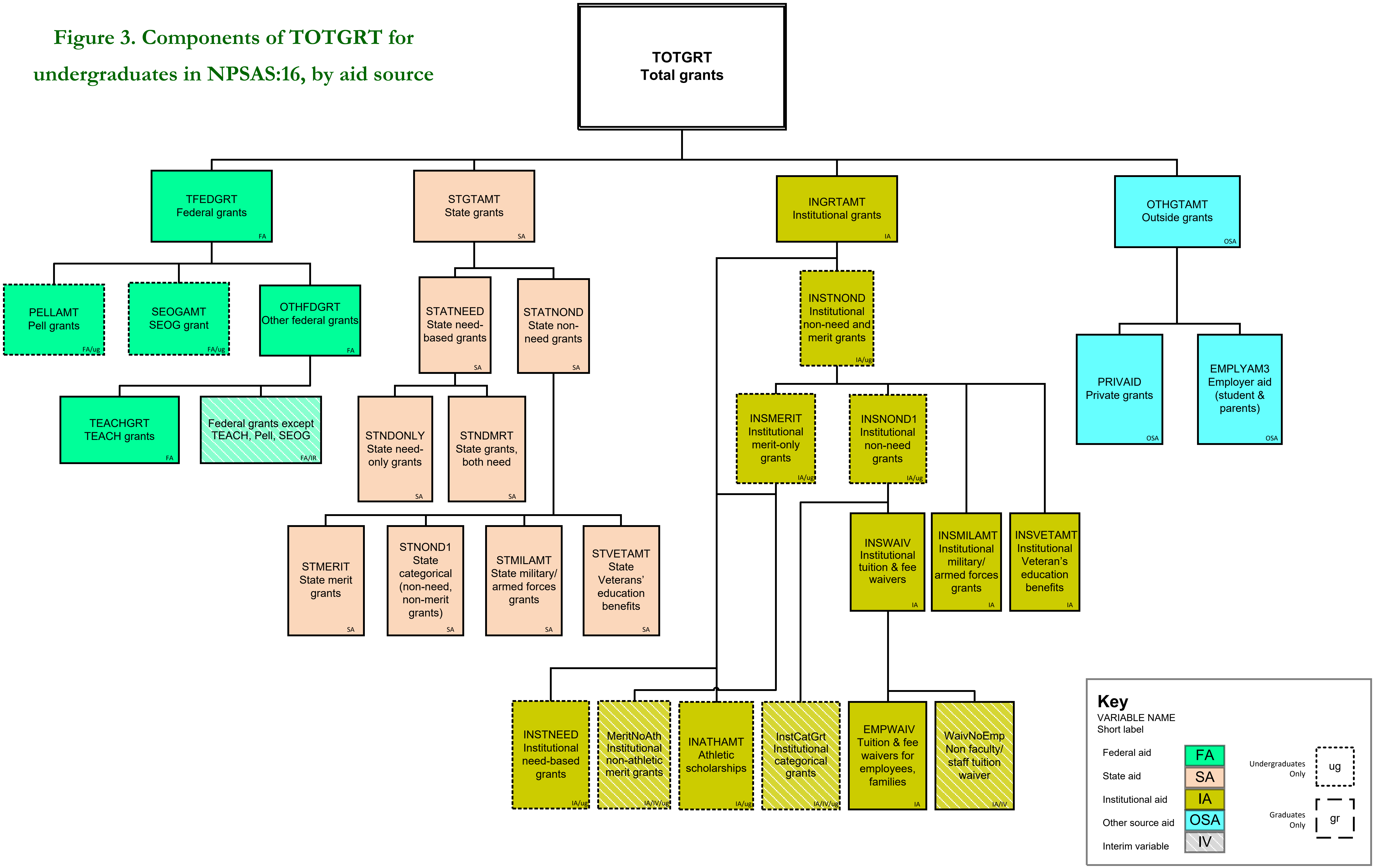


Figure 4. Components of TOTGRT for graduate students in NPSAS:16, by aid source

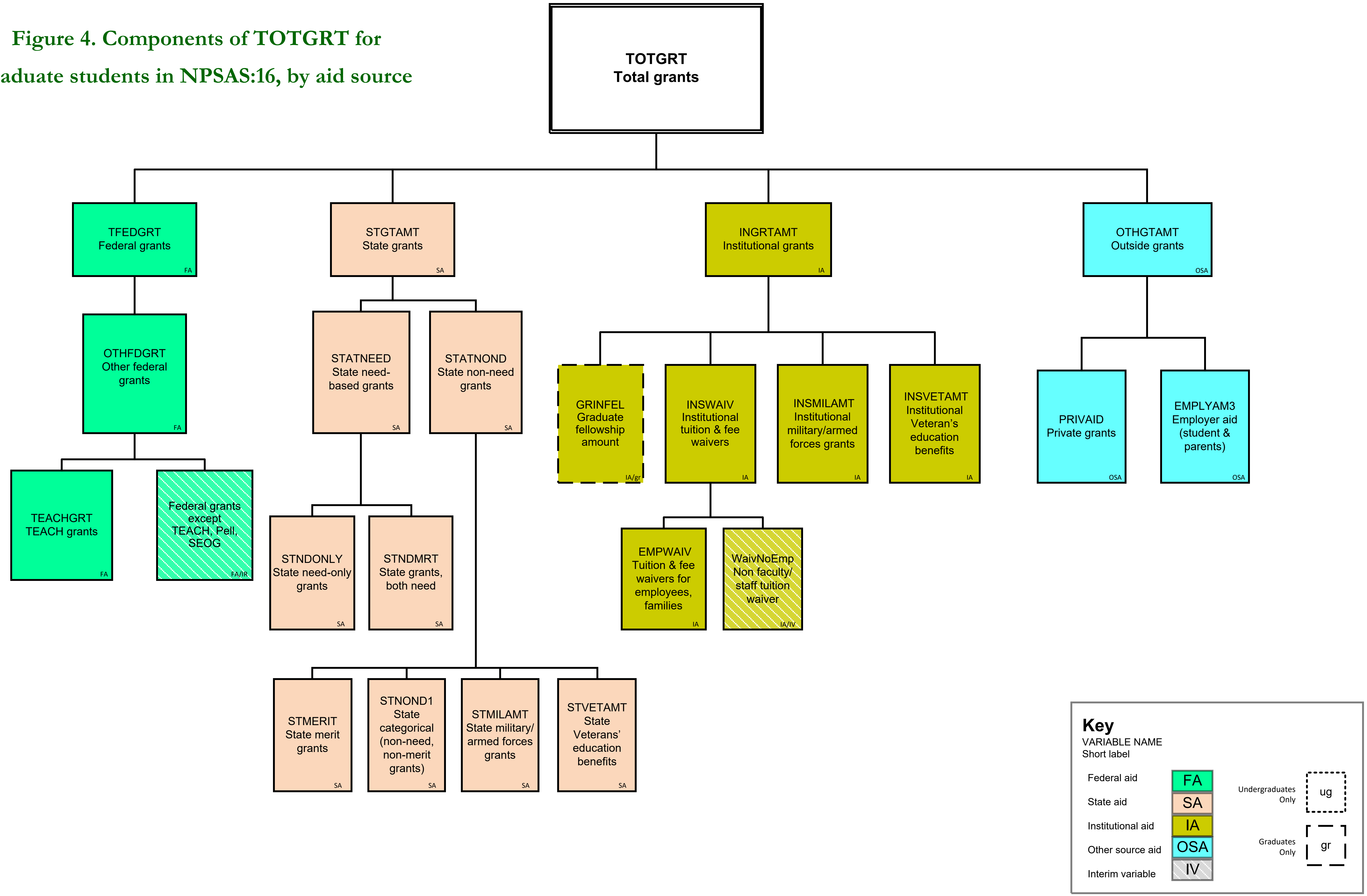
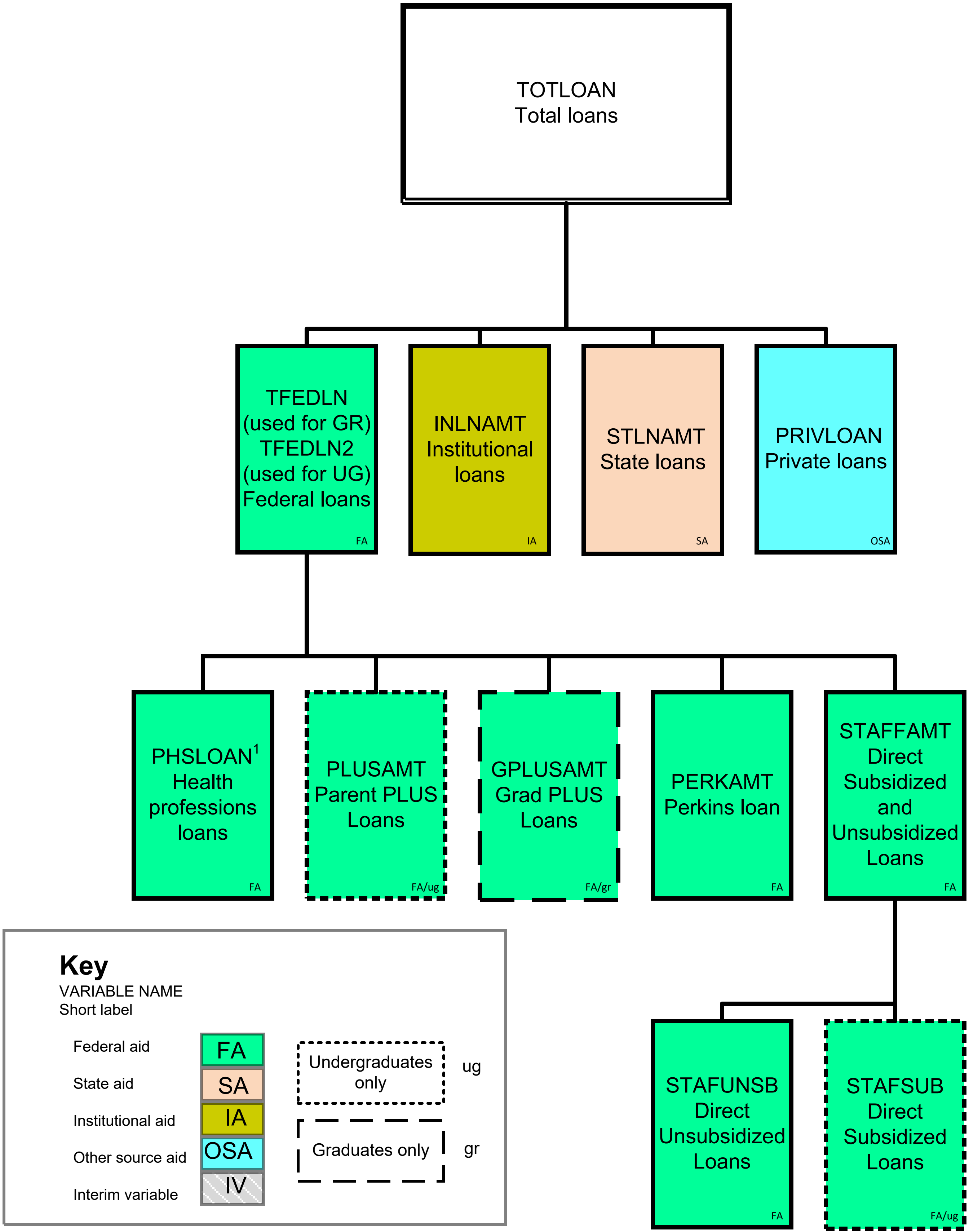


Figure 5. Components of TOTLOAN in NPSAS:16, by source



[1] PHSLOAN is a restricted-use file variable. It is not described in detail in this document.

Figure 6. Components of TOTWKST in NPSAS:16, by source

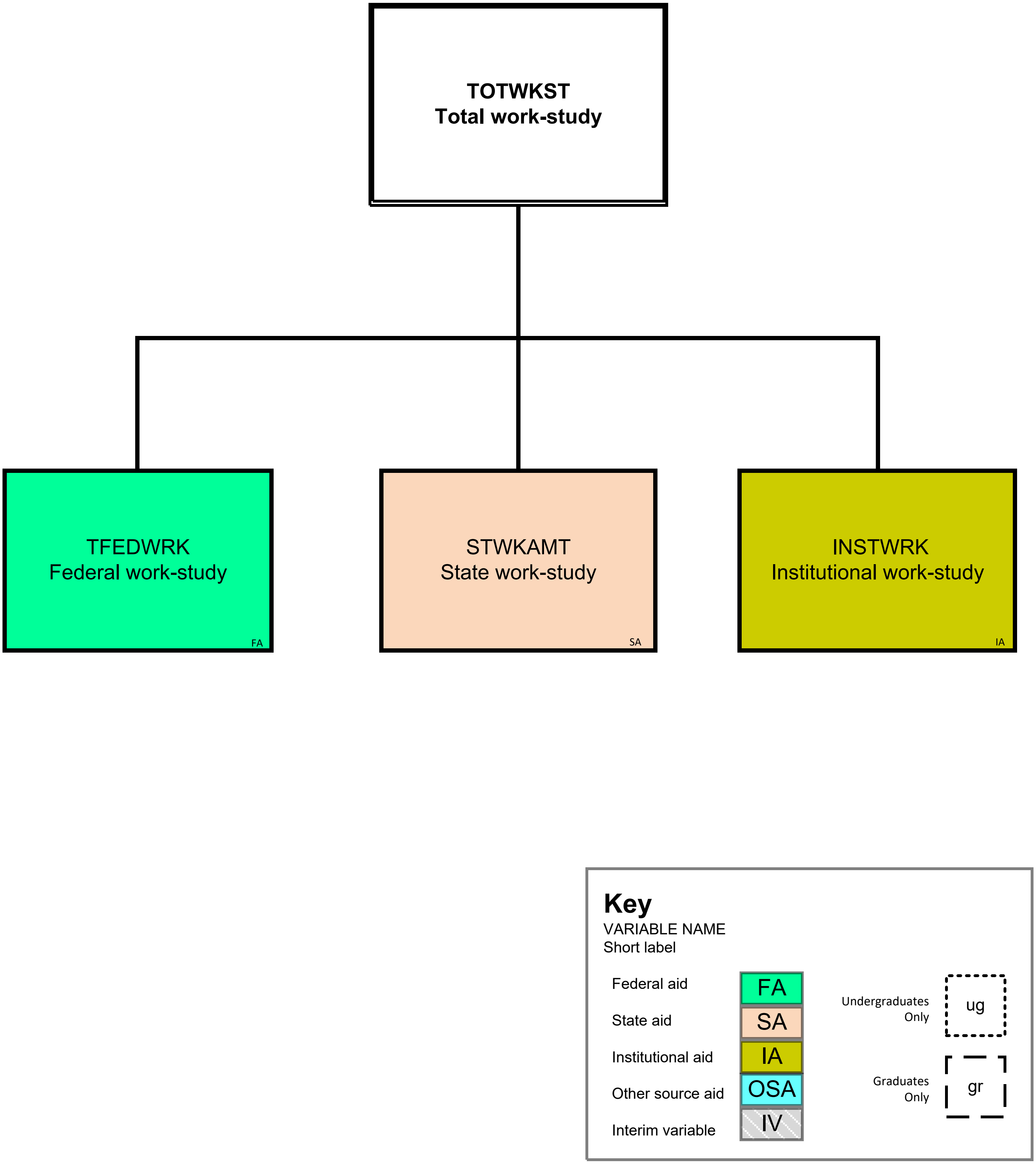


Figure 7. Components of OTHTYPE in NPSAS:16, by type and source

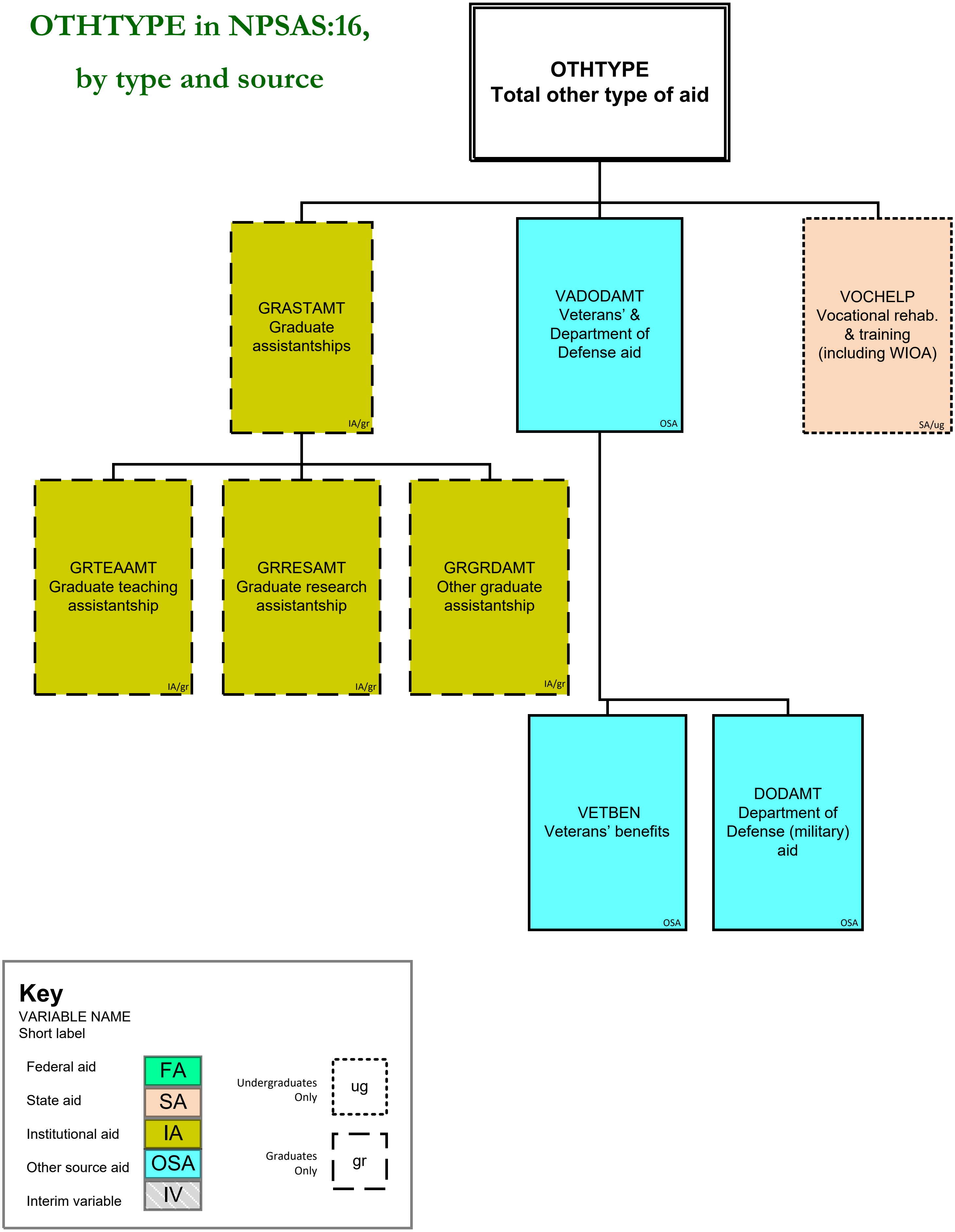
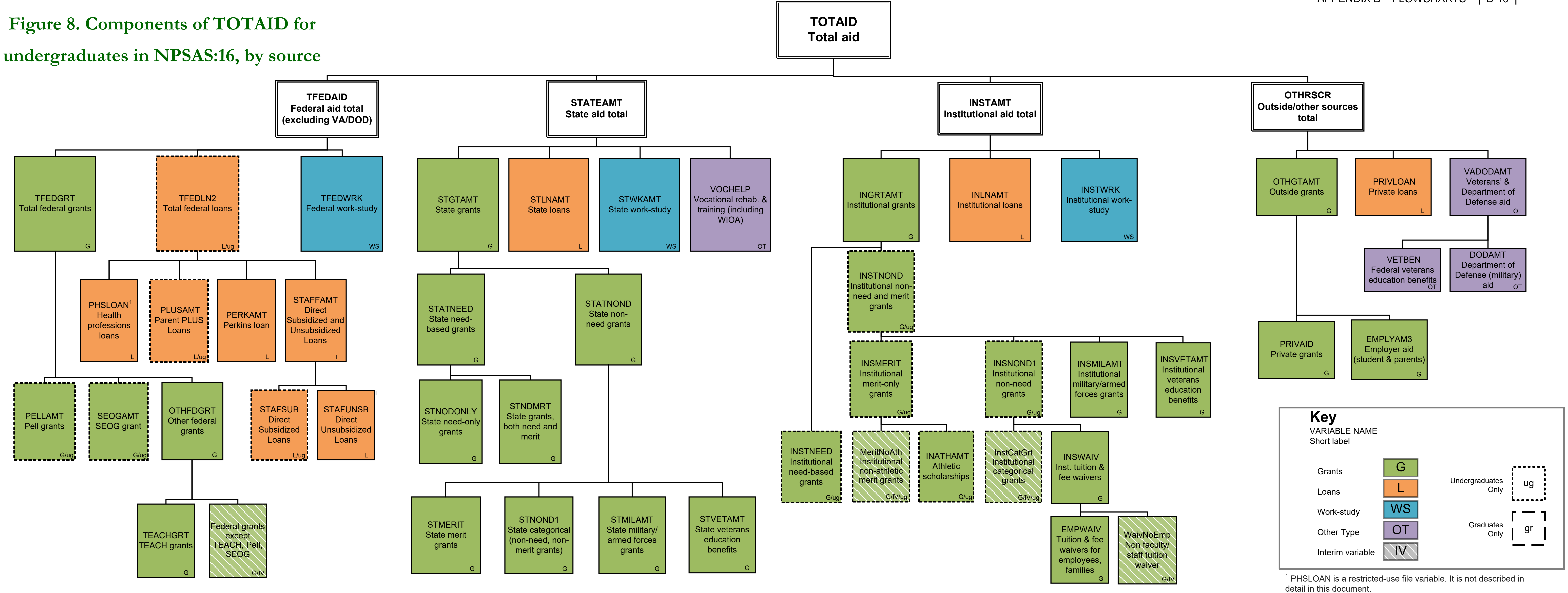
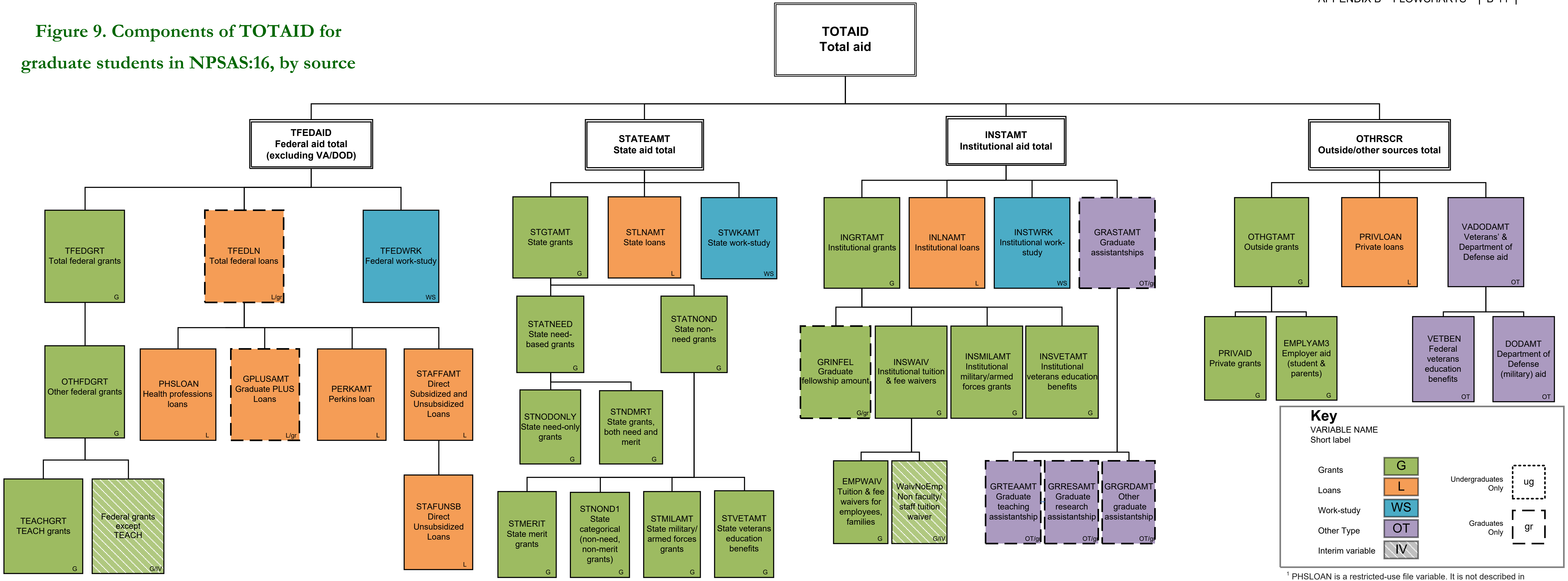


Figure 8. Components of TOTAID for undergraduates in NPSAS:16, by source



[1] PHSLOAN is a restricted-use file variable. It is not described in detail in this document.

Figure 9. Components of TOTAID for graduate students in NPSAS:16, by source



[1] PHSLOAN is a restricted-use file variable. It is not described in detail in this document.

Figure 10. Components of TFEDAID in NPSAS:16, by type

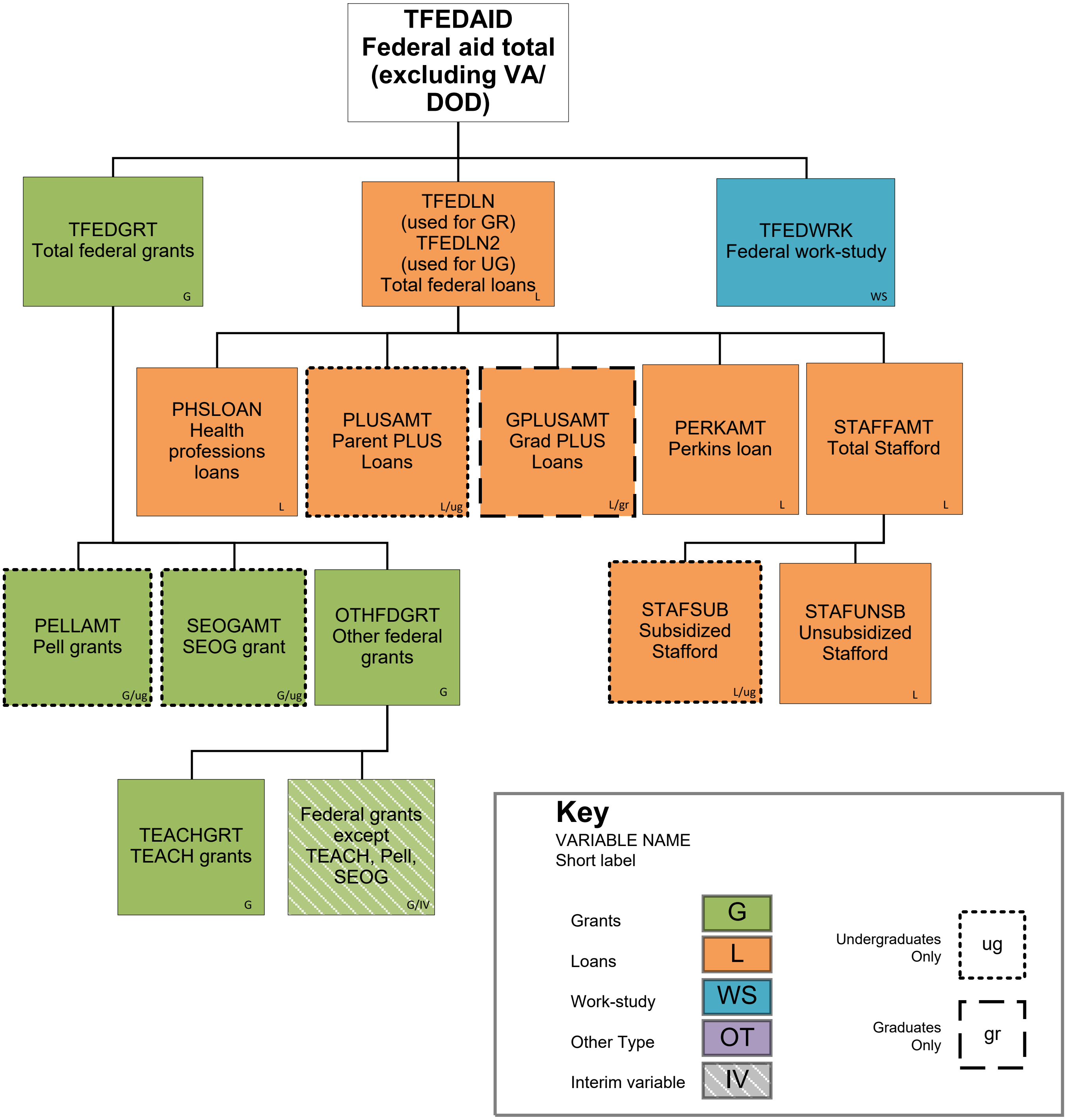


Figure 11. Components of STATEAMT in NPSAS:16, by type

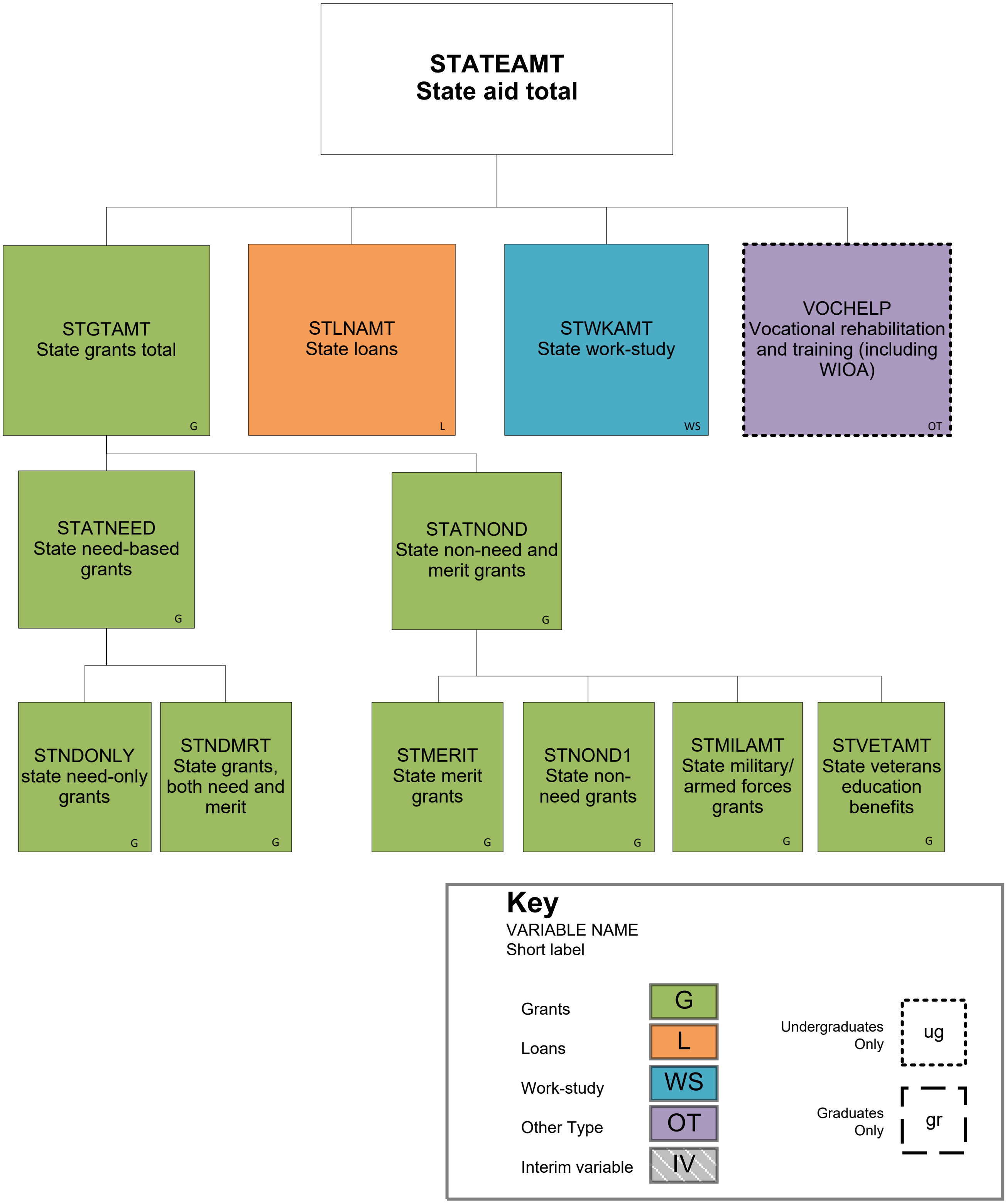


Figure 12. Components of INSTAMT for undergraduates in NPSAS:16, by type

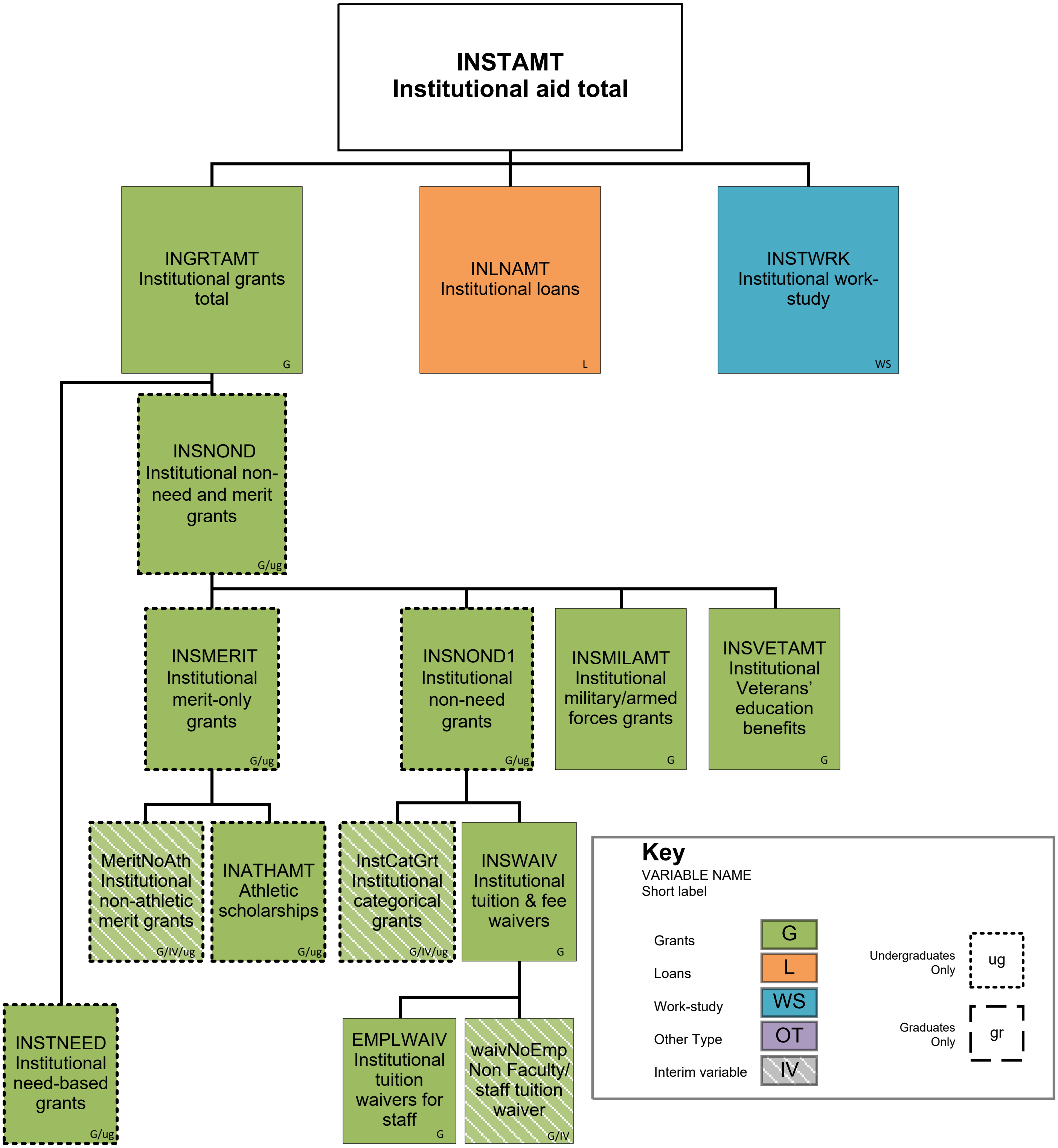

INSTAMT
Institutional aid total
INGRTAMT
Institutional grants total
G
INLNAMT
Institutional loans
L
INSTWRK
Institutional work-study
WS
INSNOND
Institutional non-need and merit grants
G/ug
INSMERIT
Institutional merit-only grants
G/ug
INSNOND1
Institutional non-need grants
G/ug
INSMILAMT
Institutional military/armed forces grants
G
INSVETAMT
Institutional Veterans' education benefits
G
MeritNoAth
Institutional non-athletic merit grants
G/IV/ug
INATHAMT
Athletic scholarships
G/ug
InstCatGrt
Institutional categorical grants
G/IV/ug
INSWAIV
Institutional tuition & fee waivers
G
INSTNEED
Institutional need-based grants
G/ug
EMPLWAIV
Institutional tuition waivers for staff
G
waivNoEmp
Non Faculty/ staff tuition waiver
G/IV
Key
VARIABLE NAME
Short label
Grants
G
Loans
L
Work-study
WS
Other Type
OT
Interim variable
IV
Undergraduates Only
ug
Graduates Only
gr

Figure 13. Components of INSTAMT for graduate students in NPSAS:16, by type

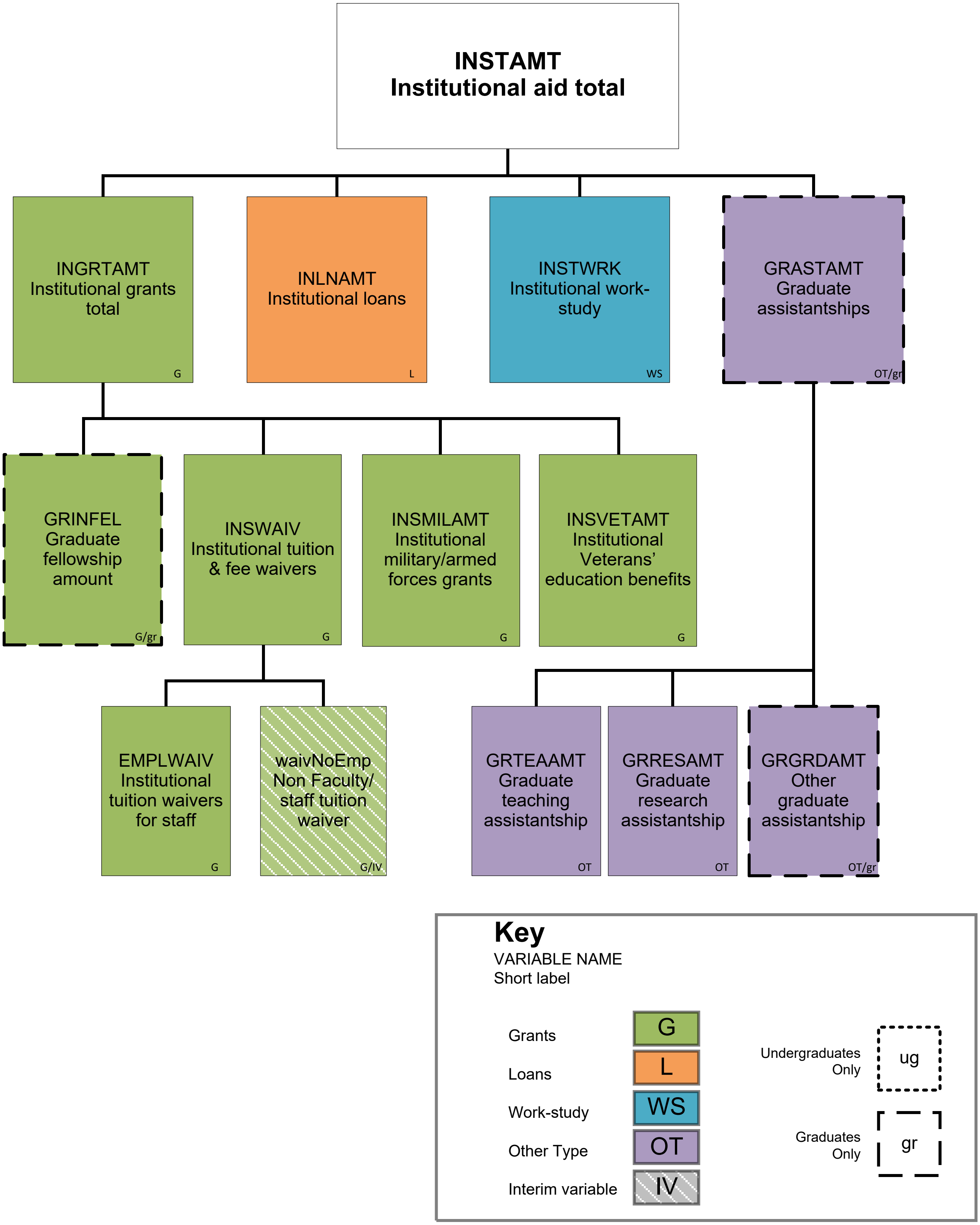
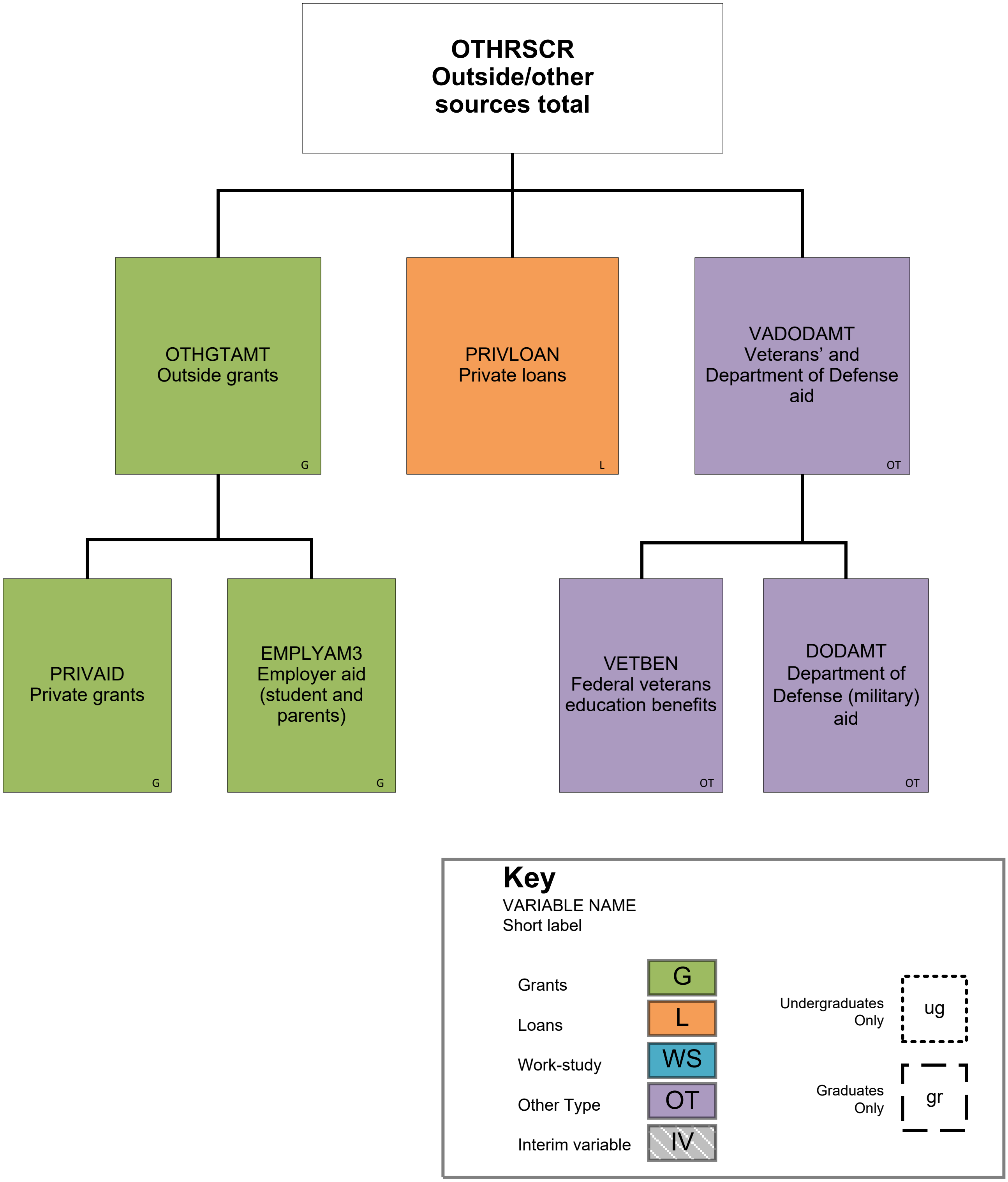


Figure 14. Components of OTHRSCR in NPSAS:16, by type source


OTHRSCR
Outside/other sources total
OTHGTAMT
Outside grants
G
PRIVLOAN
Private loans
L
VADODAMT
Veterans' and Department of Defense aid
OT
PRIVAID
Private grants
G
EMPLYAM3
Employer aid (student and parents)
G
VETBEN
Federal veterans education benefits
OT
DODAMT
Department of Defense (military) aid
OT
Key
VARIABLE NAME
Short label
Grants
G
Loans
L
Work-study
WS
Other Type
OT
Interim variable
IV
Undergraduates Only
ug
Graduates Only
gr