# Section 5 – The Policy Modules

**Date last modified: 14 September 2017**

## Introduction

The CAPITA policy code is broken up into 12 separate modules to make it more manageable. These are called from a central file, *RunCAPITA.sas*, where users input the details of each run. *RunCAPITA compare.sas* is similar to *RunCAPITA* but executes two sequential policy code runs. The user can vary either parameters or policy code between the two runs to incorporate policy changes in the model. The first run, the base world, generally represents current policy settings. The second run, the simulation world, usually includes proposed policy changes.

The policy modules themselves all have a common structure. They begin by defining an overarching macro, with the same name as the module, which steps through the major steps contained within each module. This provides the user with an overview of the code before encountering the details. The overview macro calls a number of macros specified further on within the module where the detailed operations are performed. The overarching macro is called at the conclusion of each module as all SAS macros must be defined in the code before they can be called.

This document focuses on walking the user through the CAPITA code rather than explaining the policy in detail. The variable register is a useful resource to refer to whilst reading this section of the documentation. While sections can be read in isolation, for coding brevity concepts are only described in detail the first time they occur.

For further information regarding the operation of the tax and transfer system please refer to the following resources:

* [A guide to Australian Government payments](https://www.humanservices.gov.au/corporate/publications-and-resources/guide-australian-government-payments)
* [Guide to Social Security Law](http://guides.dss.gov.au/guide-social-security-law)
* [ATO website](https://www.ato.gov.au/)
* Relevant legislation

## RunCAPITA

Section 2 of this documentation described how to use *RunCAPITA* to generate the distributional outcomes of the personal tax and transfer system, including the settings for directories and period of analysis, and details on the outfile which is generated as a result of *RunCAPITA*. The paragraphs below discuss some of the more technical aspects of the *RunCAPITA* code.

#### Step 0: Specify date of interest, period of analysis, and whether to model the grandfathered removal of the Energy Supplement

If RunCameo is set to Y, then the **DateFlag** macro tells *RunCAPITA* to source the Year, Quarter and Duration specifications from the *Cameo Code*. If the RunBenchmarkFlag is Y, this tells RunCAPITA that it is being called as part of the *Benchmarking* module in the basefiles construction process, and so the Year is set equal to the BmYear from the loop in the *Benchmarking* module. Finally, the DateFlag is created by attaching a ‘1Jul’ prefix to the year for an annual run or ‘1&Quarter’ prefix to the year for a quarterly run.

If RunEs is set to G, the grandfathered removal of the Energy Supplement will be modelled (using random variables contained on the basefile in conjunction with the grandfathering proportions by transfer payment type read in in the Param dataset described below). This option however is not available in the publicly available version of CAPITA. Setting RunEs to Y retains the Energy Supplement for all eligible recipients, whilst a setting of N removes the Energy Supplement from all recipients. The policy modules contain conditional macro statements where the Energy Supplement parameters are assigned depending on the setting for RunEs.

#### Step 1: Specify basefile

If RunCameo is set to Y, the Basefile is set equal to the **Capita\_InputFile**, which is the name of the basefile produced by the *Cameo Code*. If RunBenchmarkFlag is set to Y, the Basefile is set equal to the basefile for the appropriate year produced in the *Benchmarking* module of the *BasefileCallingProgram*. If neither of these settings are Y, then the basefile is sourced from the Basefiles folder in the CAPITA directory (i.e. the location where the basefiles are produced when the *BasefileCallingProgram* is run).

#### Step 2: Specify location of master parameters data set

The parameter directories are defined in this step, as discussed in Section 2.

#### Step 3: Specify locations of policy modules

The locations of the policy modules are specified using %LET statements. Note that this does not call the modules; it simply defines their locations as macro variables.

#### Step 4: Specify setup for macro-free version of policy modules

This step is optional and its purpose is to aid the debugging of policy modules. To use this functionality the user should set GenMacroFree to Y, which instructs SAS to write to a separate SAS program, located at the directory specified by FileMacroFree, a compiled version of the policy code. By setting RunMacroFree to Y, SAS then runs the SAS file, which occurs in Step 7. GenMacroFree and RunMacroFree are separate options so that the user can run a compiled version of policy code repeatedly, without also having to generate the compiled program each time as well. It is important to note that each time the GenMacroFree option is run, the existing SAS program is appended to, rather than overwritten, so the user should always remove any existing file before running with this option.

#### Step 5: Read in parameters for the specified date and period

This step reads in the parameters from the parameter folders defined in Step 2 above, and stores them in a dataset called **Param**. This will contain either annual or quarterly parameters, in accordance with the specifications made in Step 0. The WHERE statement ensures that only parameters which are relevant to the specified period of analysis are extracted, through the use of the Date and Enddate variables (these originate in the CPS and so are also contained in the parameter datasets). The parameter datasets read in are the AllParms\_Q and AllParms\_A datasets generated by *RunParameters* from the CPS.

#### Step 6: Run policy modules

This step calls each of the policy modules sequentially to form the **Capita\_Outfile**. This is done in a DATA step. First, the **Param** dataset is used to attach the parameter values to each income unit on the basefile. Then, the policy modules are included sequentially, which runs each policy module in the appropriate ordering. Note that the *Childcare* module is only called when *RunCAPITA* is called via the *Cameo Code* (i.e. when RunCameo is set to Y).

#### Step 7: Run macro-free version of policy modules

Related to Step 4, this is the data step that is run if RunMacroFree is set to Y. It executes the new SAS program created in Step 4 that is the compiled form of the policy modules. Its purpose is to generate a more detailed SAS log containing the information that is useful for debugging.

## RunCAPITACompare

There are some notable differences between *RunCAPITA* and *RunCAPITACompare*. The paragraphs below describe the additional code contained in *RunCAPITACompare*.

#### Step 1: Specify year and quarter of interest, period of analysis and whether Standard Output is required

When RunStandardOutput is set to Y, SAS will run the *Standard Output* code[[1]](#footnote-2) as part of the *RunCAPITACompare* run to generate the distributional comparison tables and charts which were discussed in Section 2. The SOFolder specifies the location of the folder which contains the *Standard Output* code. Finally, the ExcelOut setting tells SAS whether or not to output the distributional comparison tables and charts – a setting of N means that only the **Capita\_Compare** outfile is generated.

The **DateFlag** macro is the same as in *RunCAPITA*, except that the RunBenchmarkFlag is no longer featured, as *RunCAPITACompare* is not run as part of the basefiles construction process (since only the base world is required to construct the basefiles).

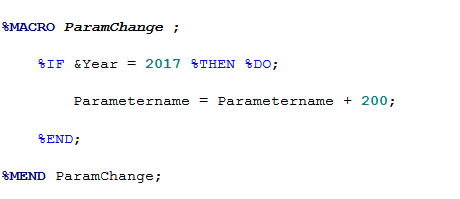
#### Step 2: Specify basefiles

Most components of this step operate in the same way as in *RunCAPITA*, except that CompareType is an additional feature. In most cases the user would set CompareType to Policy and the base world and sim world outfile would use the same basefile. The user can also set CompareType to Version, in which case the base world outfile uses the existing basefile, and the sim world outfile would use a new version of basefile. This is useful for comparing, for example, the differences between basefiles from different years, or after a new SIH update.

#### Step 3: Specify location of master parameters data set

As described in Section 2, the parameters must have already been generated using the *RunParameters* module (see below for the Guide to the *RunParameters* module). The ParmBaseDrive and the ParmSimDrive specify the locations of the base world and sim world parameter datasets respectively.

The **ParamChange** macro is one of the most important features of *RunCAPITACompare*. This macro allows the user to change the sim world parameters directly, without needing to change the parameters in the CPS and then re-generate the parameter datasets. It is important to note that year conditions will need to be used in most cases, as a sim world parameter change will usually depend on the year of analysis. For example, see figure 5.1 for the relevant code to increase a hypothetical parameter value by $200 in 2017-18.



**Figure 5.1: Example code in the ParamChange macro to change a hypothetical parameter.**

Finally, note that the **ParamChange** macro is not actually called until Step 5 below.

#### Step 4: Specify locations of policy modules

This step mirrors the corresponding step in *RunCAPITA*, except that it defines macro variables as locations for both the base and sim worlds. Note that PolicyBase and PolicySim can be used to direct the code to the folders containing the appropriate policy modules.

#### Step 5: Read in parameters for the specified date and period

This step mirrors the corresponding step in *RunCAPITA*, except that the BaseOrSim macro variable has been added so that the DATA step can be efficiently called for each of the base and sim worlds. This also allows the **ParamChange** macro to only be called for the sim world parameter dataset.

#### Step 6: Run policy modules

This step also mirrors the corresponding step in *RunCAPITA*, except that the DATA step is now conducted within the **Outfile** macro, which enables it to be generalised to apply to both the base and sim worlds. Note that the PolicySfx macro variable needs to be either ‘\_Base’ or ‘\_Sim’, so that the module names are generated with underscores, as required from Step 4.

#### Step 7: Combine base and simulated policy outfiles

Steps 7, 8 and 9 are specific to the *RunCAPITACompare* code.

Step 7 defines two macros, **RenameVar** and **Merge**. **RenameVar** writes a DATA step which attaches either ‘\_Base’ or ‘\_Sim’ suffixes to each of the variables on the outfile, depending on whether the outfile is created from a base or sim world run respectively. The **Merge** macro then writes another step which merges these two renamed outfiles together into the **CAPITA\_Compare** dataset.

#### Step 8: Execute Steps 6 and 7 according to type of comparison run

If CompareType is set to Policy, then the macros from steps 6 and 7 are called sequentially to form the **CAPITA\_Compare** dataset. First, the **Outfile** macro calls the appropriate policy modules and parameters to generate either the base or sim world. Then, the **RenameVar** macro attaches the suffixes to each of the variables and the **Merge** module creates **CAPITA\_Compare**.

If CompareType is set to Version, then the Outfile macro uses the new basefile that is defined by Step 2, and creates the sim world outfile from it.

#### Step 9: Produce Standard Output if required

Provided *RunCAPITACompare* is not called via the *Cameo Code*, and provided RunStandardOutput was set to Y in Step 1, the *Standard Output* code is called to generate the distributional charts and tables discussed in Section 2.

## RunParameters

Most of the functional aspects of the *RunParameters* code were discussed in Section 3. The paragraphs below describe some features of the code in more detail.

#### Step 1: Specify Excel workbook

As discussed in Section 3, in most cases, you will not need to change any of the specifications in the RunParameters code. However, if you are using a separate copy of the CPS to make simulated policy changes which are to be reflected in the simulation world of CAPITA, the ParamWkBk should be directed to this separate copy of the CPS, and the AllParmDrive should be directed to the Parameter (Sim) folder, rather than the Parameter folder. This ensures that RunParameters generates the simulated parameter datasets into the Parameter (Sim) folder.

The WkShtListQ and WkShtListA macro variables contain lists of the tab names from the CPS spreadsheet containing the parameters to be read-in. If additional parameters are added into new tabs, the names of these new tabs will need to be added to these lists.

The ParamStartDate macro variable specifies the starting date from which the parameters should be read in. Note that this must correspond with the survey year, since the basefiles construction process requires parameters for the survey year as part of the benchmarking module.

#### Step 2: Import parameters from Excel workbook

This step defines a macro called **ImportXls** which imports the parameters from the CPS spreadsheet into the Parameter or Parameter (Sim) folder of the CAPITA directory. Note that the WHERE statement contained in the PROC IMPORT step ensures that only the years from the survey year to the end of the forward estimates period are read in.

The PROC SQL combines together each of the separate datasets that have been read in into one. This is the parameter dataset that is used by the RunCapita and RunCapitaCompare. PROC SQL is used instead of a data step merge due to the more efficient way it processes many-to-one merges.

The final DATA step initialises any missing values it finds for numeric variables in the CPS to zero, to ensure that the log does not produce missing values for these parameters. It also drops parameters from the CPS that begin with ‘not\_used’ as well as empty, unnamed variables created by SAS (F1, F2 etc.). This is done to reduce the number of variables in the CAPITA output file.

## The Standard Output Module

The Standard Output module is an optional add-on module to RunCapitaCompare.sas which can produce a number of datasets designed to allow CAPITA users to quickly and conveniently compare two different output datasets. Broadly, the code has three functions:

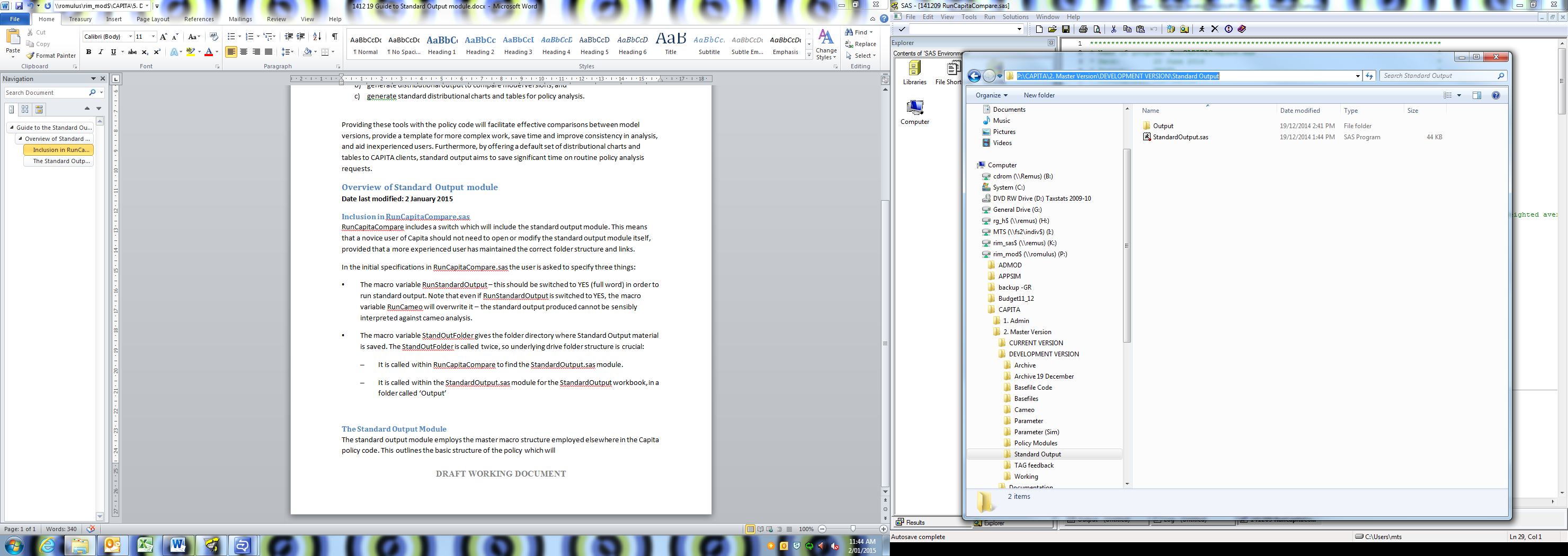
1. provide utilities to aid analysis;
2. generate distributional output to compare model versions; and
3. generate standard distributional charts and tables for policy analysis.

Providing these tools with the policy code will facilitate effective comparisons between model versions, provide a template for more complex work, save time and improve consistency in analysis, and aid inexperienced users. Furthermore, by offering a default set of distributional charts and tables to CAPITA clients, standard output aims to save significant time on routine policy analysis requests.

RunCapitaCompare includes a switch which will include the standard output module.

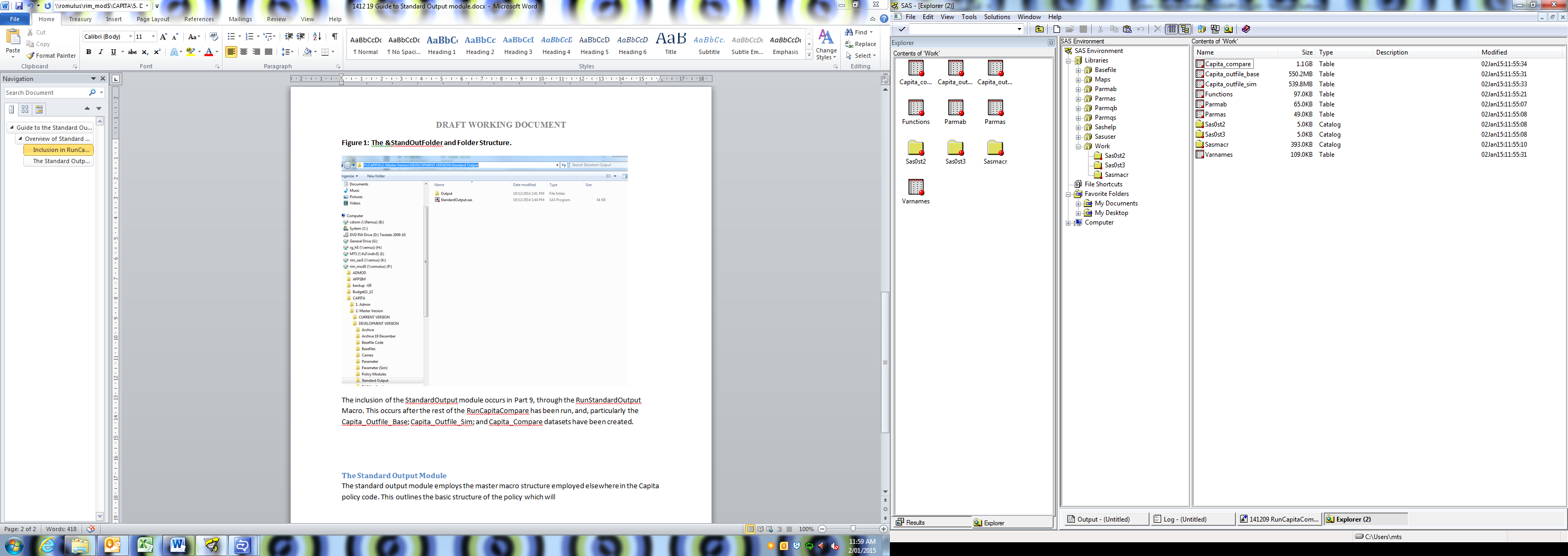
In the initial specifications in RunCapitaCompare.sas the user is asked to specify three things:

* The global macro variable RunStandardOutput – this should be switched to Y in order to run standard output. Note that even if RunStandardOutput is switched to Y, the macro variable RunCameo will overwrite it – the standard output produced cannot be sensibly interpreted against cameo analysis.
* The global macro variable StandOutFolder gives the folder directory where Standard Output material is saved. The StandOutFolder is called twice, so the underlying drive folder structure is crucial:
  + It is called within RunCapitaCompare to find the StandardOutput.sas module.
  + It is called within the StandardOutput.sas module for the StandardOutput workbook, in a folder called ‘Output’
* The global macro variable ExcelOut determines whether StandardOutput tables are exported to Excel (and a linked workbook) or just output in SAS. This should be switched to YES for a linked Excel workbook.



**Figure 5.2: The &StandOutFolder and Folder Structure.**

The inclusion of the StandardOutput module occurs in Part 9, through the RunStandardOutput Macro. This occurs after the rest of the RunCapitaCompare has been run, and, particularly the Capita\_Outfile\_Base; Capita\_Outfile\_Sim; and (where a policy, rather than version change is modelled) Capita\_Compare datasets have been created in the Work folder.



**Figure 5.3: Contents of the Work Folder before StandardOutput.sas runs.**

The standard output module uses the master macro structure employed elsewhere in the Capita policy code. This outlines the basic structure of the policy which will be implemented as the final macro call in the module.

### 1. Define Lists of variables and helpful macros

In the first step a number of lists are created. Creating the lists means that the code can operate by looping through Do loops, minimising the need for repetition in the code and for users to require multiple changes.

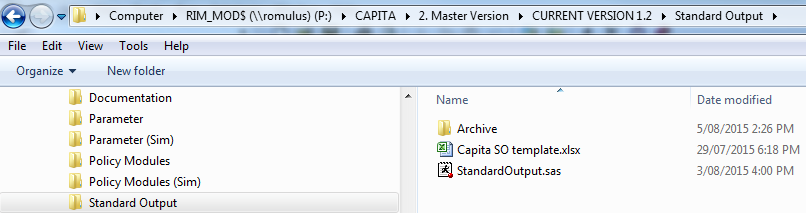
* The global macro variable idlist refers to the list of ID variables which can apply to an individual in Capita. These are household ID (HHID); family ID (FAMID); income unit ID (IUID); and a variable created within the code for person identification (which will be r, s, 1-5 depending on their location in the income unit.
* The global macro variables varlistHinc and varlistinc refer to the different types of income which we want to output at either the person or household levels. Only annualised private and disposable incomes are meaningful at the household level (IncPrivA and IncDispA); while at the individual level annualised Taxable Income and Adjusted Taxable Income are also included (TaxIncA and AdjTaxIncA).
* The next variable list global macro variables have a specific form – varlistUNIT where unit is either fam, coup, sps or ref, depending on which group of individuals are individually recorded in the Capita\_Outfile. For instance the policy code only ever allocates Wife Pension to a spouse (that is, only WifePenBasicAs exists as a variable on the outfile); so it is recorded in varlistsps. By comparison, the total tax paid or refund allocated, PayOrRefAmntA is recorded for each working age individual in the income unit, so it I recorded in varlistfam.
* The global macro variable VarlistHH records totals for entire households, which are not allocated to a particular individual.
* The global macro variable VarlistAll creates a summary of all variables which are being analysed.
* The global macro variables personsUNIT gives a list of the suffixes which are to be appended to the respective unit variable lists. For instance, we will want to find WifePenBasicAs, so the personssps macro variable enables us to tell sas to append the letter s to the varlistsps variable WifePenBasicA, when we are executing a Do Loop. Similarly we will append r,s,1-5 to PayOrRefAmntA so that we consider each of the variables PayOrRefAmntAr, PayOrRefAmntAs, PayOrRefAmntA1, PayOrRefAmntA2, PayOrRefAmntA3, PayOrRefAmntA4 and PayOrRefAmntA5.
* The global macro variable VarCompList gives the list of variables which we want SAS to specifically compare across individuals. This is used in winner and loser analysis.

Next a number of formats used within the output are created.

* Income\_fmt gives income bands in $10,000 increments from 0 to $300,000.
* iutypeA gives a general breakdown into six different income unit types, that will be specified later in the code.
* iutypeB gives a more specific break down for working age recipients into low, medium and high income (which will be relative to the whole population).
* Incthr gives the low, medium and high breakdown, but also separates out seniors.
* Iupaymenttype is used to label the payment type category which the income unit is deemed to fall into.
* DispIncChangePROP and DispIncChangeAbs break down the changes in disposable income into 15 categories each, depending on whether the absolute change or the percentage change is considered.

The Macros StandardOutputWorkBook and OutPutExcel are the final parts of part one.

* The Macro StandardOutputWorkBook exists to set the global macro variable StandOutE, the file location of the relevant linked excel workbook which is contained. The code outputs to a time stamped copy of Capita SO.xlsx and allows for different templates for Policy and Version runs (these currently use the same template).



**Figure 5.4 – Standard Output/Output/ folder structure:**

* The Macro OutputExcel is called to output datasets from the work folder to a specific tab in the Excel Workbook. It makes use of the Export Procedure and mirrors code from the Cameo Output.

### 2. Create Person and Household Level Data Sets

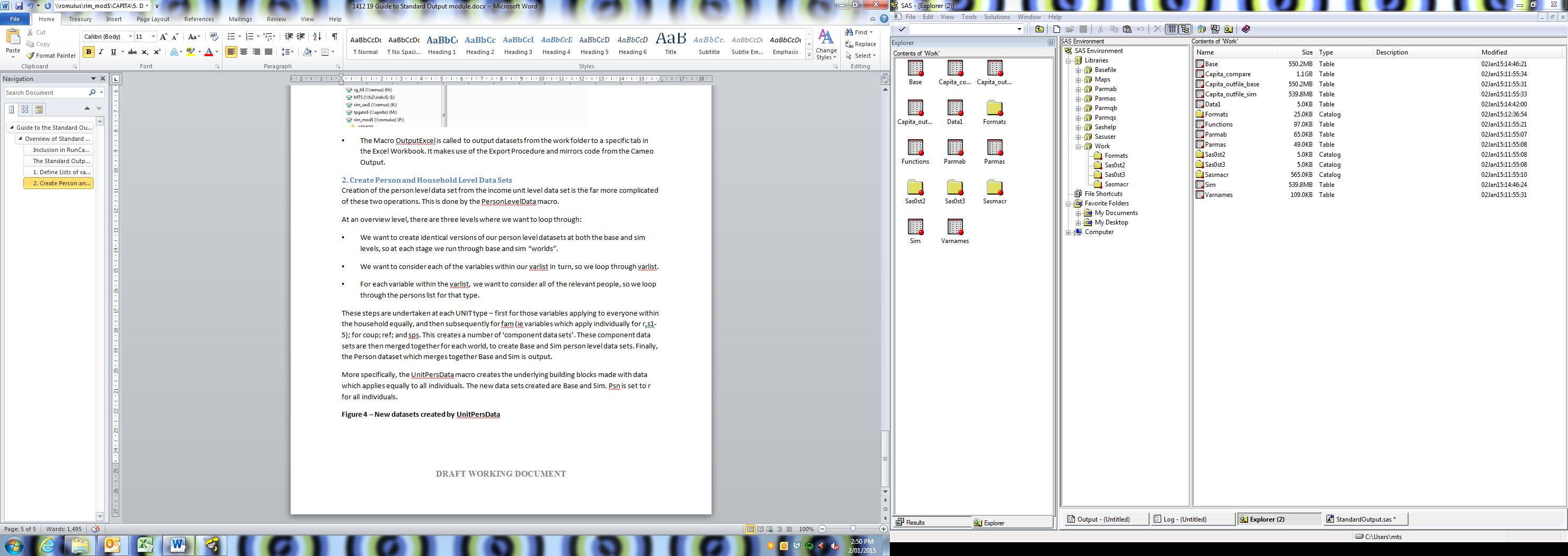
Creation of the person level data set from the income unit level data set is the far more complicated of these two operations. This is done by the PersonLevelData macro.

At an overview level, there are three levels where we want to loop through:

* We want to create identical versions of our person level datasets at both the base and sim levels, so at each stage we run through base and sim “worlds”.
* We want to consider each of the variables within our varlist in turn, so we loop through varlist.
* For each variable within the varlist, we want to consider all of the relevant people, so we loop through the persons list for that type.

These steps are undertaken at each UNIT type – first for those variables applying to everyone within the household equally, and then subsequently for fam (ie variables which apply individually for r,s1-5); for coup; ref; and sps. This creates a number of ‘component data sets’. These component data sets are then merged together for each world, to create Base and Sim person level data sets. Finally, the Person dataset which merges together Base and Sim is output.

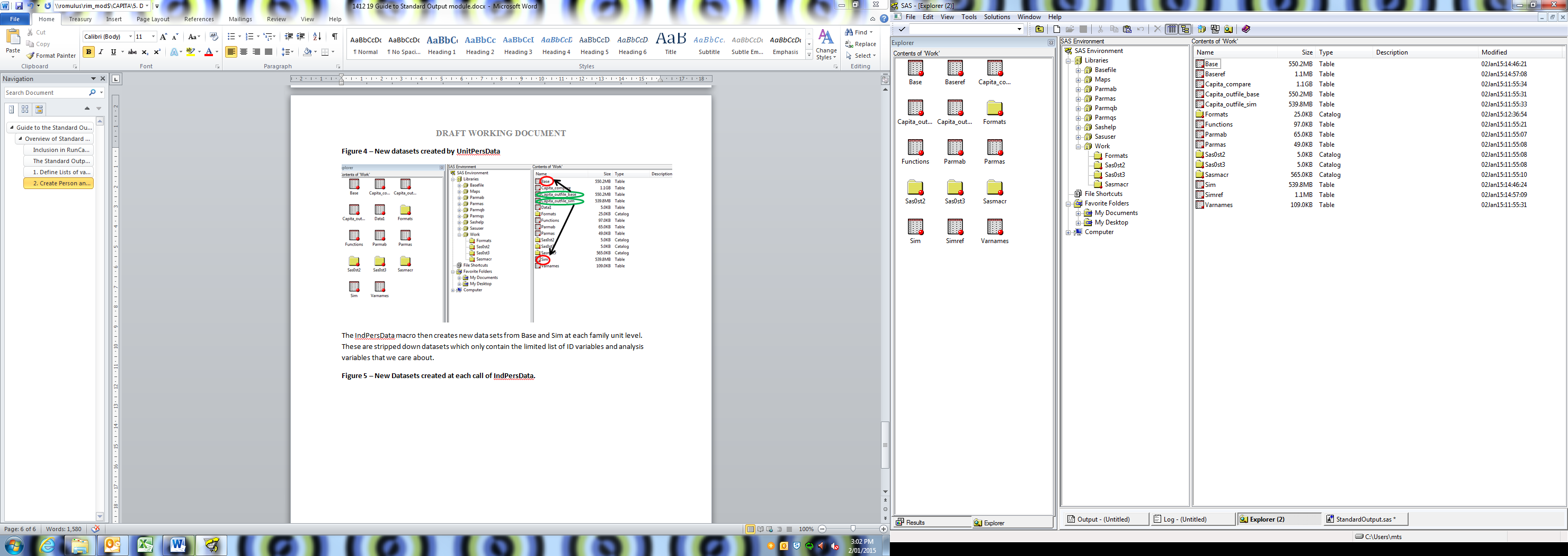
More specifically, the UnitPersData macro creates the underlying building blocks used to analyse. The new data sets created are Base and Sim. Psn is set to r for all individuals; the HHID\_base and HHID\_sim and IUID\_base and IUID\_sim variables are renamed HHID and IUID respectively, to facilitate merging (if necessary) later. Finally, it introduces the global macro variables keeplist\_base and keeplist\_sim which will build the list of variables we want to retain.

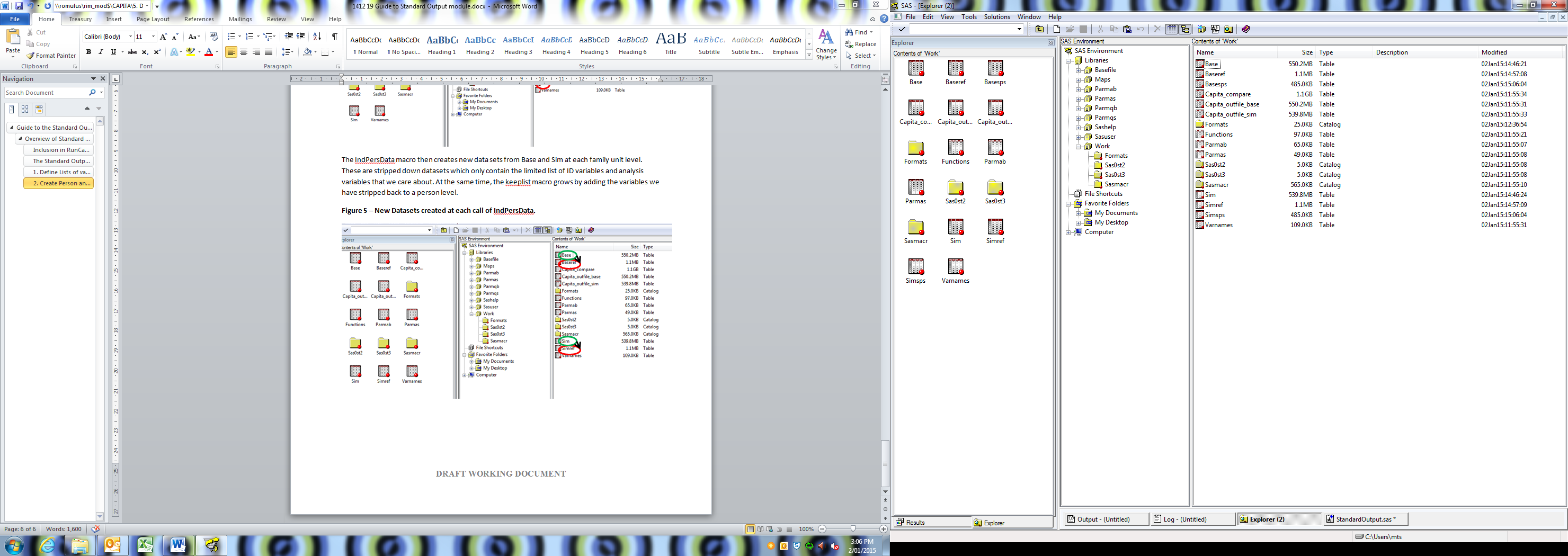


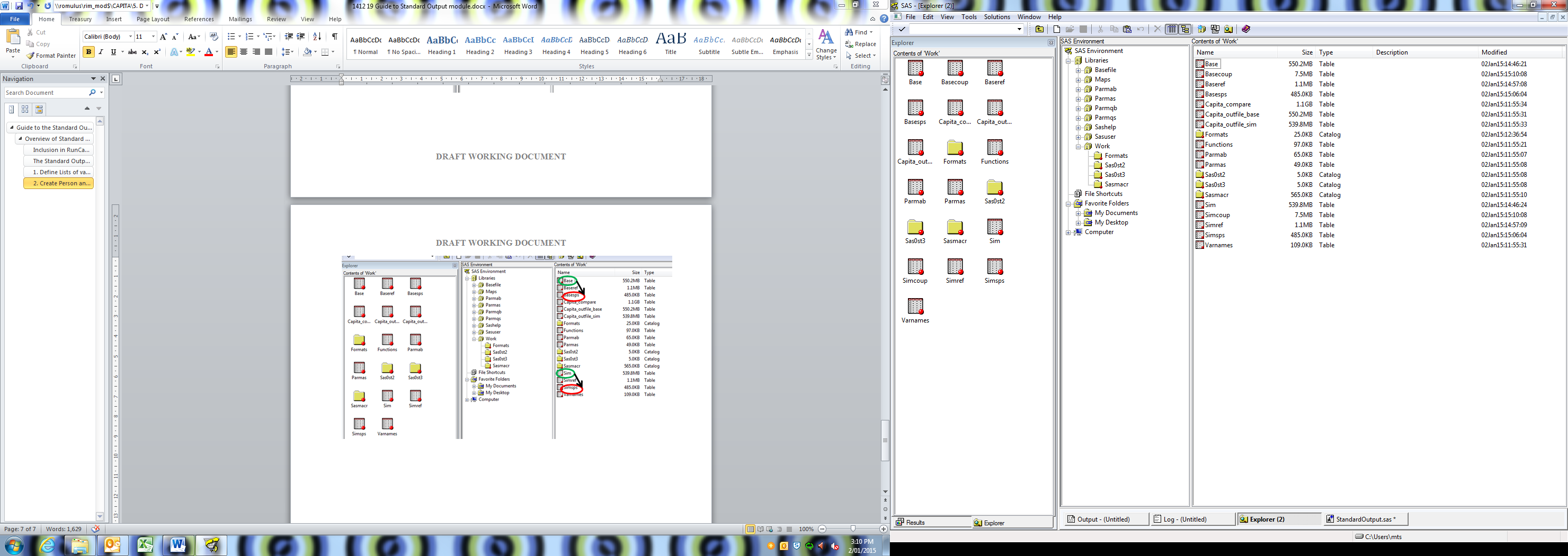
**Figure 5.5 – New datasets created by UnitPersData**

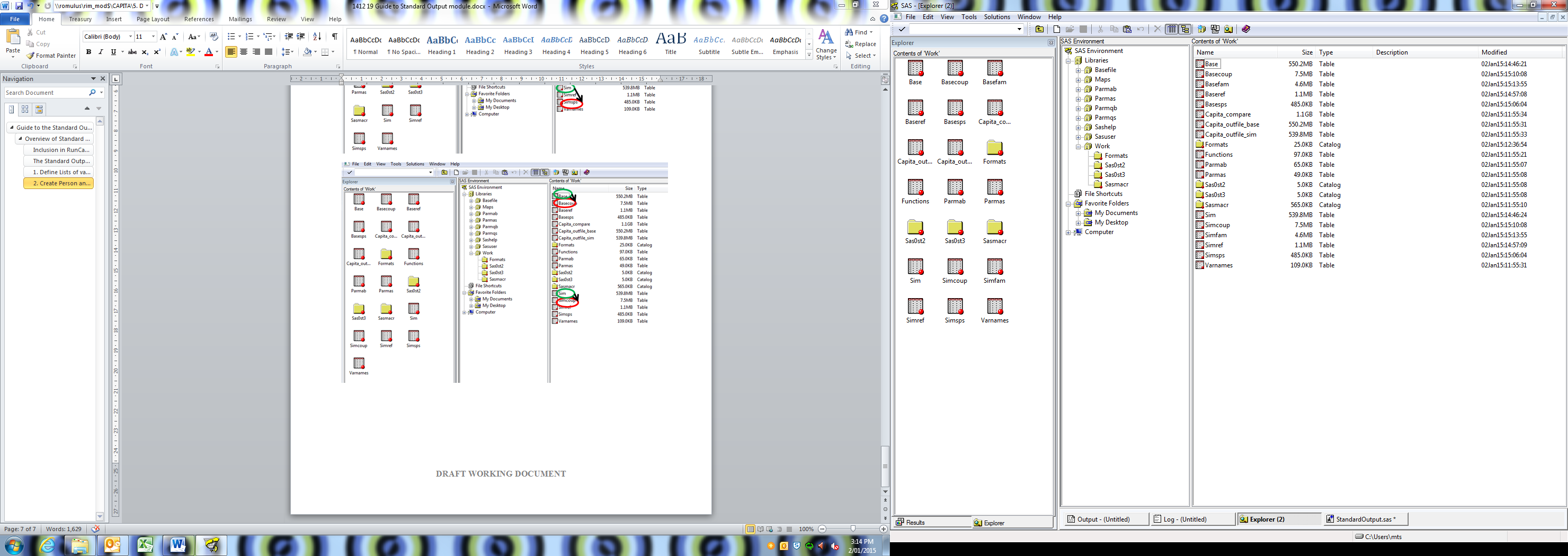
The IndPersData macro then creates new data sets from Base and Sim at each family unit level. These are stripped down datasets which only contain the limited list of ID variables and analysis variables that we care about (note that, for instance the SimRef data set is much smaller than the Sim data set, this is because it has only a tiny fraction of the total variables). On the other hand, because they OUTPUT for each person and each variable, they create more observations than are in the underlying Base and Sim variables.

At the same time, the keeplist macro grows by adding the variables we have stripped back to a person level.





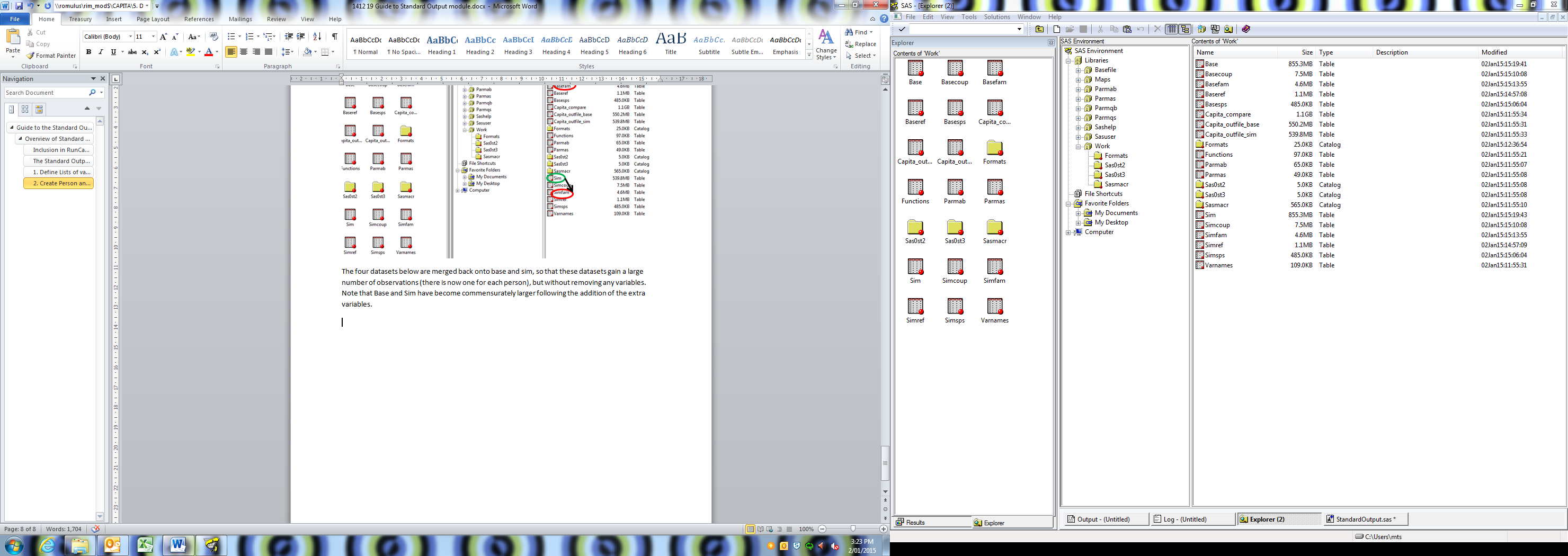


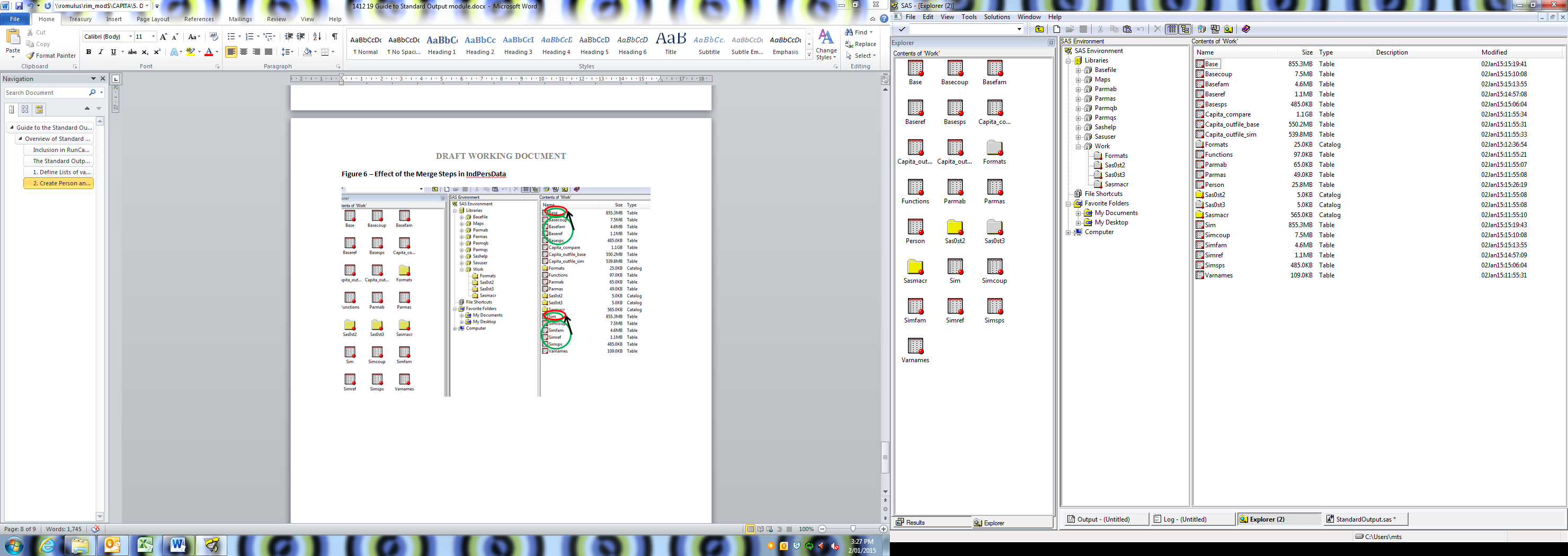


**Figure 5.6 – New Datasets created at each call of IndPersData.**

The four datasets below are merged back onto base and sim, so that these datasets gain a large number of observations (there is now one for each person), but without removing any variables. Note that Base and Sim have become commensurately larger following the addition of the extra variables.

Finally base and sim are merged onto a stripped down dataset Person, making use of the maintained keeplists from above (note that although combining Base and Sim, it is still much smaller).

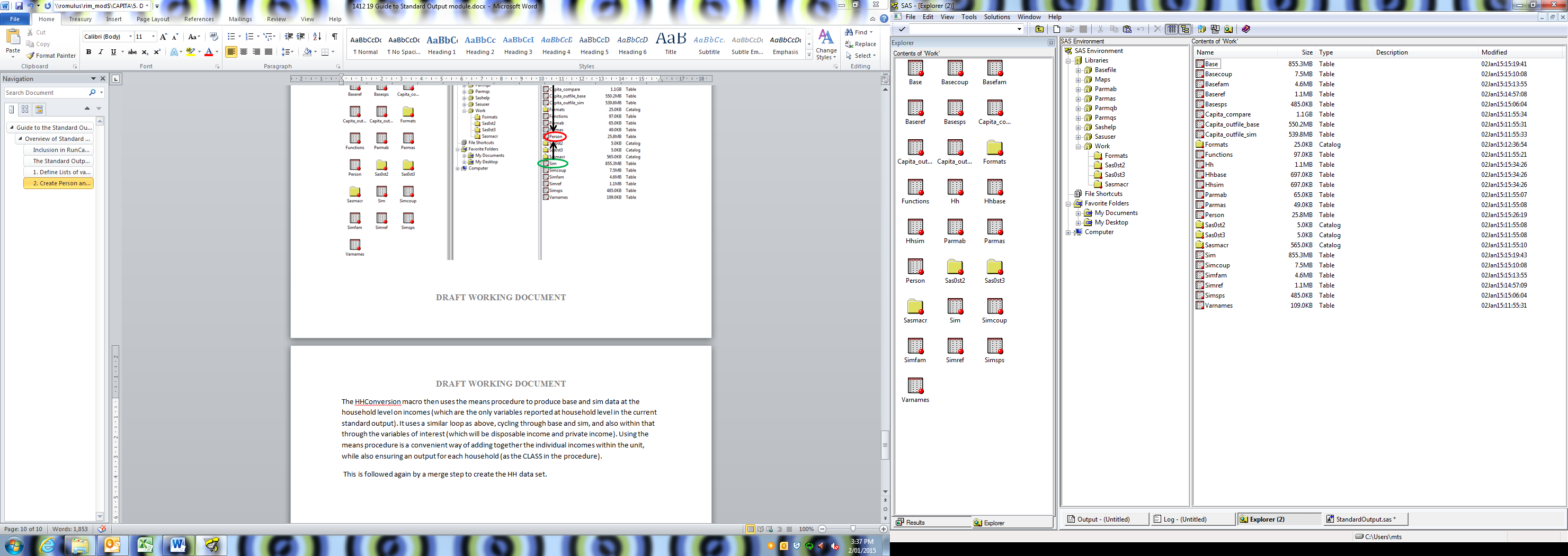




**Figure 5.7 – Effect of the Merge Steps in IndPersData macro**

The HHConversion macro then uses the means procedure to produce base and sim data at the household level on incomes (which are the only variables reported at household level in the current standard output). It uses a similar loop as above, cycling through base and sim, and also within that through the variables of interest (which will be disposable income and private income). Using the means procedure is a convenient way of adding together the individual incomes within the unit, while also ensuring an output for each household (as the CLASS in the procedure).

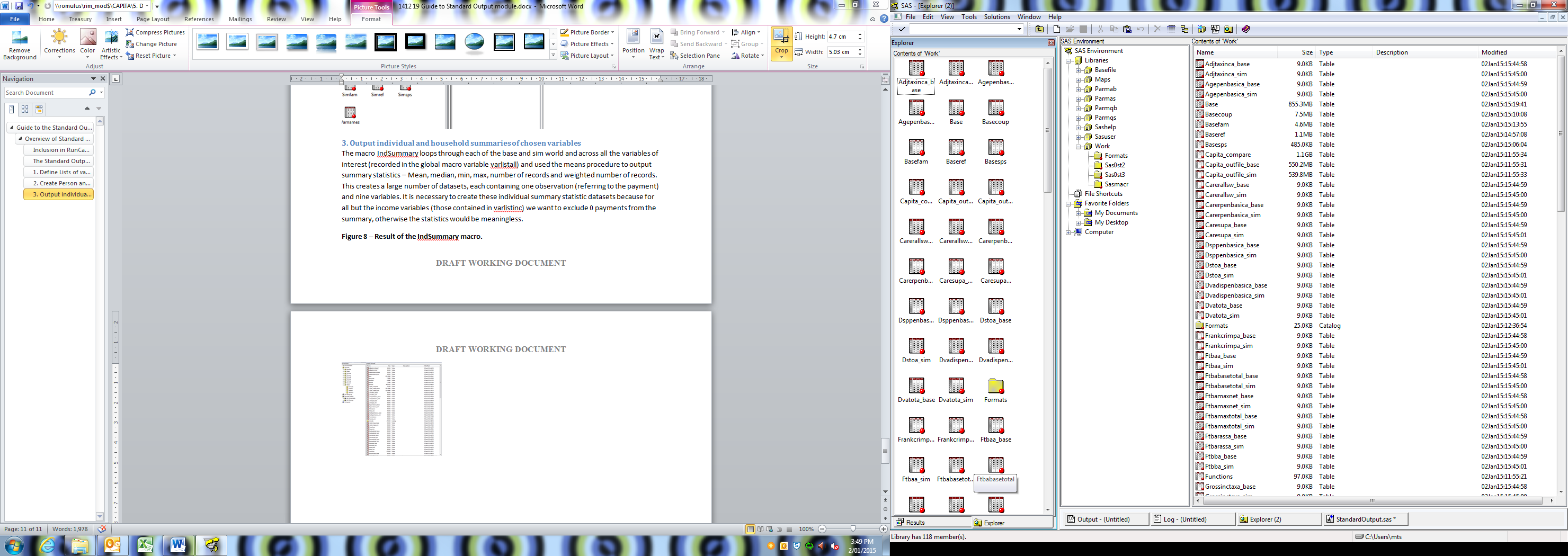
This is followed again by a merge step to create the HH data set.



**Figure 5.8 – Result of the HHConversion macro.**

### 3. Output individual and household summaries of chosen variables

The macro IndSummary loops through each of the base and sim world and across all the variables of interest (recorded in the global macro variable varlistall) which are stored in the Person dataset and uses the means procedure to output summary statistics – Mean, median, min, max, number of records and weighted number of records. This creates a large number of datasets, each containing one observation (referring to the payment) and nine variables. It is necessary to create these individual summary statistic datasets because for all but the income variables (those contained in varlistinc) we want to exclude 0 payments from the summary, otherwise the statistics would be meaningless.



**Figure 5.9 – Result of the IndSummary macro.**

These are then merged together to create the actual output of interest which is the Summary Dataset. It should have two lines for each payment (one base and one sim) and contain each of the 5 summary statistics.

The HHSummary dataset does much the same using the means procedure for the incomes at the household level from the HH dataset. The only difference is that since there is no longer the need to ignore cases where the variable is zero (since all deals with incomes) it does not create separate datasets for each variable of interest. Furthermore, it does not then merge these two datasets back to a single summary. Instead it outputs SummHHBase and SummHHSim directly.

## The Cameo Module

### Overview

The Cameo module produces cameos which illustrate the impact of policy changes on hypothetical families. It can also produce cameos which illustrate the effective marginal tax rates (EMTRs) of a family: the proportion of each additional dollar earned that is removed due to either tax or the withdrawal of transfer payments. The process for setting up and running cameos and EMTRs has been discussed earlier in Section 3.

The module imports family and income data from the cameo input spreadsheet, creates cameo basefile data, runs the Capita policy modules, and exports the results to the cameo spreadsheet.

### 1. Import family and income data from cameo spreadsheet, generate income ranges

Family and income data for selected hypothetical families on the cameo input spreadsheet are read in and converted into SAS data set.

For cameo runs, the CameoList specifies the variables required for the cameo output results.

For EMTR runs, the list of required variables are defined in the EmtrVarList. Each variable has the following four components:

* The name of the variable (including the person suffix).
* The income measure that the variable is calculated against (for example, private income, taxable income).
* A negative or positive designation, to indicate if the variable increases (positive) or decreases (negative) with a change in income.
* A plain English label, to be displayed in the EMTR Output spreadsheet.

Finally, the **RenameEmtrVarList** is defined. This macro assigns each variable its plain English label.

The **IncomeLists** macro assigns the income list (range of incomes) corresponding to each selected hypothetical family for cameo runs.

EMTR charts differ from cameos in that they concern marginal dollar-by-dollar impacts, and so incomes are not imported from the Cameo Input spreadsheet. The **EmtrIncome** macro calculates the income distributions for both the reference and spouse by first generating a vector of income values for the reference person according to the ‘Start’ and ‘Stop’ specifications in the Cameo Input spreadsheet, and then calculating the spouses income distribution based on the reference person’s according to whether the spouse’s income has been set to ‘Fixed’ or ‘Variable’ in the EMTR Input spreadsheet.

### 2. Create cameo basefile

The **CameoInitialise** macro initialises and incorporates into the cameo base data the basefile variables created from the SIH dataset used for distributional analysis. This is required since both the cameo functionality and distributional functionality use the same policy modules. The macro ensures that no errors occur due to non-existing or uninitialised variables when running cameos.

The receipt flags for Youth Allowance, NSA, and Age pension (*YouthAllSW*, *NsaSW* and *AgePenSW*, *WrkForceIndep* and *WrkForceExp*) are initialised to 1 to ensure eligibility of family members for these payments.

The ages of reference and spouse are checked to determine their relationship status and position in the family. If both ages are greater than zero, *Coupleu* is set to 1 (to indicate that the family consists of a couple) and family positions “REF” and “SPOUSE” are assigned to the reference and spouse respectively. Otherwise, *Coupleu* is set to zero (to indicate that the reference is a single person) and the family position “REF” is assigned to the reference.

If there are age data for children in the Cameo Input spreadsheet, the age of the youngest child is obtained which is used for determining eligibility and payment rates for Parenting Payment and FTBB. Otherwise, it is set to 99, to indicate that there are no children in the selected family type.

The **KidsAge** macro counts and categorises by age the number of children aged 0 to 14 years old.

The **KidAssign** macro removes from person1 to person4 children aged under 15 years old who are too young to be students for Youth Allowance. It is assumed that half the children are male and the other half are female. Children are assigned a family position of DEPCHILD.

The **IncomeAssign** macro assigns income from the Cameo Input spreadsheet to wages and salaries or service income if the person is under age pension age. Otherwise, income is assigned to superannuation income.

It is assumed that the private health insurance status of the reference person (i.e., with private health insurance if *PrivHealthCameo* takes the value 1, and 0 if otherwise) applies to all members of the family.

The weekly rent data from the Cameo Input spreadsheet is converted to a fortnightly amount which is the frequency used in the policy modules for calculating rent assistance.

If DSP or CARE has been specified for payment in the Cameo Input spreadsheet, the flags for DSP or Carer Payment and Allowance and number of dependants cared for are set to 1 to ensure the person’s eligibility for the specified payment.

For eligible couples, the Parenting Payment receipt flag is assigned to the spouse if that person is not eligible for Youth Allowance, which is a higher payment for most of the income distribution. Therefore, the Parenting Payment receipt flag is assigned to the spouse if that person meets the following additional conditions:

* If the person is in full time study, then they must be 25 years of age or over (since the upper age limit to receive Youth Allowance (Student) is 24) to receive the ParPaySWs = 1 flag; and
* If the person is not studying full time, then they must be 22 years of age or over (since current Government policy sets the limit to receive Youth Allowance (Other) at 22 years of age) to receive the ParPaySWs = 1 flag.

Otherwise, Parenting Payment is assigned to the reference person. For eligible sole parent families, Parenting Payment is assigned to the reference.

### 3. Run policy code

The **RunPolicy** macro calls either the *RunCAPITA* module (used for running the base world scenario) if RunCompare is set to “N”, or the *RunCAPITACompare* module (used for comparing impact of policy changes) if RunCompare is set to “Y”. Note that the RunCompare switch has no effect on the output for EMTR runs.

### 4. Export output to Excel

The **CreateOutfile** macro creates a time stamped copy of the output file (Cameo output.xlsx or EMTR output.xlsb).

The macro **Turning\_Point(psn)** creates labels against income levels which are known to cause EMTR ‘jumps’, for example, ‘start of first income tax bracket’. These labels are exported as part of the output to the EMTR Output spreadsheet.

The macro **RemoveZeroVar** creates a list of the variables that only have non-zero values and drops them from the final output. This makes it clear which variables are relevant for each cameo.

The **ExportCameos** macro exports the cameo results from SAS to the output spreadsheet. For EMTR runs it calculates the EMTR at each income level and calls the Turning\_point macro.

For cameos, if running the base world scenario only, the results will be the values of variables under the base world scenario. Otherwise, the results will include the values of the variables under the base world scenario, their values after the policy change (the sim world scenario), and their difference.

If the DropZeroVars switch is set to Y then the RemoveZeroVar macro is called.

## The Initialisation Module

### Overview

The initialisation module assigns default values (either 0 or “” for numeric or character variables, respectively) to all variables created in the policy code. Many of these will be overwritten later in the policy code. The module uses a series of lists, one for each group of variables to be initialised. The variables are grouped into lists depending on the *type* of each variable (i.e whether it is character or numeric) and the *members of each income unit* that require the variable. The members of the income unit are the reference person (r), their spouse (s) and up to four dependant students aged over 15 (1,2,3,4).

### 1. Define variable lists

The variable lists are defined using %LET statements that generate macro variables. When the macro variable is resolved, the macro call will be replaced by the contents of the list. The first group of lists defined in the module are sets of suffixes indicating which *members of an income unit* require the variable. The second series of lists define the *base variable names*, these are grouped according to: the suffix list that is to be applied to them, and whether they are numeric or character variables. The variables are sorted alphabetically. For example the variables in VarListAll are numeric and are required for all members in an income unit (r, s, 1, 2, 3, 4) contained in SuffixListAll. These variables will all be initialised to zero. The variables in VarListCharCoup are all character and required for the reference and spouse. These will be initialised to “” (blank).

### 2. Initialise numeric variables

The **InitNumVar** macro is called for each corresponding pair of numeric variable and suffix lists. Income unit variables that do not require a suffix and are initialised using the **InitialiseNum** macro.

### 3. Initialise character variables

The **InitCharVar** macro is called for each corresponding pair of character variable and suffix lists. Income unit variables that do not require a suffix are initialised using the **InitialiseChar** macro.

### The macros

#### InitNumVar

Inputs: VarList – list of numeric variables, delimited by a hyphen (‘-‘).

SuffixList – list of members of the income unit for which the variable is required, delimited by a hyphen.

The NumVar macro variable counts the number of variables (words) in the VarList using the SAS function: %SYSFUNC(COUNTW()). The hyphens in the list are used as delimiters. Similarly NumSuff counts the number of suffixes in the SuffixList. Then the macro loops through the variable list, using the %SCAN function to extract the *i*th variable in the list. For each variable, it also loops through each of the suffixes, setting each combination of variable and suffix to zero. E.g. *AdjTaxIncAr* = 0.

##### Eventually, all numeric variables in the VarList are initialised to 0 for each of the suffixes in SuffixList.

#### InitialiseNum

Inputs: VarList – list of numeric variables, delimited by a hyphen.

##### All numeric income unit variables in the VarList are initialised to 0.

#### InitCharVar

Inputs: VarList – list of character variables, delimited by a hyphen.

SuffixList – list of members of the income unit for which the variable is required, delimited by a hyphen.

##### All variables in the VarList will be initialised to “” (blank) for each of the suffixes in SuffixList.

#### InitialiseChar

Inputs: VarList – list of character variables, delimited by a hyphen.

All income unit character variables in the VarList will be initialised to “” (blank).

## The Income1 Module

### Overview

Income1 calculates income and assets according to the definitions used by the social security system by summing components. CAPITA has a second module (Income2), later in the model, which calculates income according to the definitions used for Family Tax Benefit and the personal tax system, as these require information generated in earlier CAPITA policy modules. Details on the income definitions can be found online in the resources given on page 1 of this document.

### Incomes calculated

* Private income
* Taxable private income
* Non-taxable private income
* Ordinary income
* Income eligible for work bonus
* Private income less work bonus
* Previous financial year taxable income
* Parental income
* Maintenance income

### 1. Calculate Private Income

The **PrivIncome** macro calculates a person’s private income.

### 2. Calculate Ordinary Income and Assets

The **IncomeTest** macro calculates income used to assess entitlement for pensions and allowances.

The **AssetsTest** macro calculates assets used to assess entitlement for pensions.

### 3. Calculate Combined Parental Income

The **ParenIncTestAll** macro uses the previous year taxable income to calculate parental income to assess entitlement for Youth Allowance.

Maintenance income for the parental unit is calculated as the sum of maintenance income received by the reference person and the spouse. This amount is retained for own income units within the same family.

### 4. Calculate Private and Ordinary Income for Dependants

The **PrivIncome** and **IncomeTest** macros are used to calculate private income and the income used to assess entitlement for pensions and allowances for dependants 1 to 4.

### The macros

##### Unless otherwise specified, all macros in the Income1 module have the same inputs:

##### Inputs: psn – person. (One of: r, s, 1, 2, 3, 4, r\_)

The Income1 macros all sum up the components of income.

#### PrivIncome

The components of private income are summed to give an individual’s taxable and non-taxable components of private income. Some income components are only available for r and s. Conditional code %IF… is used to include these income components in the sum only if the relevant data exist (e.g. for persons r and s) and exclude them otherwise (e.g. for persons 1 – 4). This avoids errors arising from summing missing values. The use of r\_ as a macro input is associated with the “look-ahead” method described in detail in the Appendix of this document. The r\_ represents the reference person non-parental income units in the family. To obtain private income, the taxable and non-taxable components are added. The annual figure is then converted to a fortnightly figure by dividing by 26.

#### DeemIncome

There are no inputs to this macro. **DeemIncome** calculates deemed income (assumed financial investment income for the pension income test) based on the value of an income unit’s financial assets, the relevant deeming threshold and deeming rates. The variable *DeemedCalcF* is the adjustment required to assessable income to properly apply the income test (replace actual financial investment income with deemed).

***WorkIncome***

#### Inputs: ***psn – person. (One of r, s)***

Work bonus eligibility is assigned to individuals over age pension age (or veterans over the DVA pension age) unless they are in receipt of parenting payment single. The work bonus reduces a person’s ordinary income, which is used for means testing pensions. For eligible individuals, the work bonus is the greater of the legislated work bonus amount (WorkBonF) and sum of their income from wages, salaries and business (*IncWBF*). The work bonus is subtracted from private income to give IncPrivLessWBF. For those ineligible for the work bonus, IncPrivLessWBF is simply their private income. Private income less the work bonus is used to determine ordinary income in the next macro.

***OrdIncome***

Ordinary income is calculated by summing its components for the reference and spouse. A special definition of ordinary income is used for DVA payments, which excludes income from DVA disability and war widow pensions. A separate definition of ordinary income is used for persons 1-4 as all their components of ordinary income, other than private income, are zero and persons 1-4 cannot be eligible for the work bonus as they must be under age pension age so CAPITA simply sets their ordinary income equal to private income.

#### IncomeTest

##### The WorkIncome then OrdIncome macros are called to calculate ordinary income. DeemIncome is called to calculate the adjustment for financial investment income deeming in the income test. Because income is shared between couples for the calculation of pensions and DVA pensions the IncDVATestF and IncPenTestF (and their deeming-adjusted versions, IncDeemDvaTestF and IncDeemPenTestF) halve the couple’s ordinary income. For allowances, means tests use the individual’s income.

#### TaxInvPrevYr

##### If data from the previous financial year is available (indicated by DataScopeType = "PrevYrAvail"), this is used to calculate taxable income. The current year’s tax deductions are used as a proxy for the previous year’s as these are not reported on the SIH.

##### In some cases, the previous financial year data is deemed unreliable by the ABS due to significant changes within the household that may impact on the comparability of current year and previous year estimates. Examples include recent arrival in Australia or a change in marital status. In this case, the current year’s private income is used as a proxy for the previous year’s taxable income.

#### ParenIncTestAll

##### Parental income is only calculated if relevant dependants exist. The statement IF ActualAge1 > 0 is used to determine if student dependants over 15 exist in the parental income unit. The statement IF FIRST.FamID AND NumIUu > **1** checks if there are dependants in their own income unit. Due to the way the data is sorted, the first income unit encountered by SAS (FIRST.FamID) is the parental income unit. NumIUu indicates if there is more than one income unit in the family.

##### The value of parental income used in the test is the couple’s combined (if applicable) previous years’ parental income which includes maintenance received and excludes maintenance paid.

## The Dependants1 Module

### Overview

The parental income unit contains information about the parents, children aged less than 15 and full-time students less than 25 living with their parents. Other income units in the family may contain dependants aged 15 or over living with their parents but not studying full-time. The Dependants1 module counts the number of dependants in each income unit, according to the various income definitions used in the social security system, and assigns dependant flags to those over 15. CAPITA has a second module (Dependants2), later in the model, which counts the number of dependants, according to the definitions used in the personal tax system, as these require information generated in the intervening CAPITA policy modules. More detail around the different type of dependants can be found online in the resources given on page 1 of this document.

### Dependant definitions applied

* DepsFtbSec16\_18
* DepsFtbSec19
* DepsFtbaOwnIU
* DepsFtbbOwnIU
* Kids15u
* DepsUnder5
* DepsUnder13
* Deps13\_15u
* DepsUnder15
* DepsPrinCare
* DepsFtbPr
* DepsFtbSec
* DepsSec5
* DepsSSTotal
* DepsFtba
* DepsFtbb
* DepsFtbaMaint
* DepsYaMaint
* DepsMaint

### 1. Create lists

Using the **RenameList** macro, lists of variables for use in the look-ahead method are generated.

### 2. Tally up dependants

For each family, the **SocialSecDependants** macro tallies the number of dependants according to the definitions used in the social security system.

### The macros

#### RenameList

This macro scans KeepList, which contains a set of variable names delimited by hyphens. The macro variable RenameList contains the text required to rename all the variables in the set to add an underscore to their names. E.g. *ActualAger* in KeepList will lead to ‘*ActualAger* = *ActualAger\_*’ in RenameList. When RenameList is resolved in conjuction with a RENAME statement, all the variables in KeepList will be renamed. The **RenameList** macro also produces DropList, which contains all the renamed variables, i.e. all the variables in KeepList with an ‘\_’ suffix added. This is used with a DROP statement to drop these variables after use. The use of lists allows the user to add additional variables in one place rather than at several points in the code.

#### SocialSecDependants

##### Student dependants in the parental income unit

When SAS encounters the parental income unit in each family (FIRST.*FamID*), the code loops through dependants 1 to 4 to test whether they are a dependant under section 5 of the *Social Security Act 1991* (*Sec5Dep*) or the expanded definition (*SSPlusDep*) using the **Sec5DepTest** and the **SSPlusDepTest** macros, respectively and assigns flags, e.g. *Sec5DepFlag1* = 1, if the person satisfies the dependency criteria. It then tallies the number of secondary school FTBA dependants (*DepsFtbSec16\_18*, for 16 to 18 year olds and *DepsFtbSec19* for 19 year olds) in the income unit.

Similarly the code also loops through dependants 1 to 4 to test whether a dependant is eligible to receive child support, check if they are receiving Youth Allowance using the **MaintDepTest**, assign flags e.g. DepsYaMaintIFlag1 if the person satisfies the criteria, and tally the number of dependants in the income unit. *Depsmaint* and *DepsYaMaintI*are retained for own income units within the family.

##### Dependants in their own income unit

If there are other dependants in the family, in their own income unit, this is indicated by *NumIUu* > 1. CAPITA draws their information into the parental income unit using the look-ahead method described in detail in the Appendix of this document. In summary, the look-ahead method uses information from subsequent rows in the basefile containing the information on other income units in the same family and attaches this information to the parental income unit. The SET and POINT statements reads in data from subsequent rows. The variables in KeepList are then renamed using the list stored in the RenameList macro variable, described above, which adds a ‘\_’ suffix to the variable names to avoid a conflict between the dependant’s data and the data stored in the same variable for the parental income unit.

CAPITA loops through each of the non-parental income units in the family and calculates private income and ordinary income using the **PrivIncome** and **OrdIncome** macros. While this information will be calculated for these income units when SAS reaches them, the line by line processing used in SAS means that, when processing the parental income unit record, the subsequent records have not yet been processed so the information is not yet available. The work bonus is not applied for dependants as they are clearly under age pension age.

Post 1 July 2014,

From 1 July 2014, only 0 to 15 year olds and 16 to 19 year olds who are studying full-time are eligible for FTB. So the only people in their own income unit, eligible for FTB are 15 year olds.

15 year olds: If the dependant is eligible for FTB, i.e they are not receiving an allowance on the SIH, then the tally of the number of FTB dependants in their own income unit is incremented both for FTBA (DepsFtbaOwnIU) and FTBB (DepsFtbbOwnIU).

For all years, 15 year olds are then tested to determine if they are a section 5 dependant. If so, they receive a flag and the tally of *DepsSec5OwnIU* in the parental income unit is incremented.

Similarly, all dependants aged 15, 16 or 17, in their own income unit, are tested to determine if they satisfy the extended social security definition and the *DepsSSPlusOwnIU* tally is incremented, if required.

In addition, individuals are tested to see whether they are eligible to receive maintenance income; and whether they receive Youth Allowance (DepsMaintTest). If so, they receive a flag and the tally is incremented.

The variables pertaining to the dependant in their own income unit are then dropped before repeating the process for the next dependant in their own income unit if applicable.

##### Dependants in the parental income unit

Summary variables are created by combining other variables.

The number of children aged 15 in the income unit (these must all be dependants 1 to 4) is determined using the Boolean expression ActualAge1 = 15. If dependant1 is aged 15 this will resolve to 1 and contribute to the sum. Otherwise it will resolve to zero.

The number of children under age five is required for FTBB as they receive a higher payment rate.

The number of children under age 13 and from 13 to 15 is required for FTBA as they receive different payment rates.

The number of children under age 15 is required to determine those that satisfy the section 5 definition.

A principal carer flag is assigned to the income unit if they have a child under age 15 or a student aged 16. This is used to assign the single principal carer flag required to give the person a higher taper rate for NSA and widow allowance.

The number of primary school children is calculated by assuming children aged 5 to 11 are primary school age and the number of secondary school children are those aged from 12 to 15 plus secondary students aged from 16 to 19. Note: dependants in their own income unit cannot be students by definition so are not considered here.

Totals for section 5 dependants are calculated by adding the number of dependants under age 15 to the flags for those aged 15 and over (these are set to 1 if they are eligible and zero otherwise) and the number of section 5 dependants in their own income unit. *DepsSSTotal* *, DepsYaMaint* and *Depsmaint* are calculated similarly.

The number of FTBA dependants are those aged 15 and under, those not studying but in their own income unit, and those aged 16 to 19 year old secondary school students.

For single parent families, FTBB dependants are those aged 15 and under (including those in their own income unit) and 16 year old secondary school students; for couple families, FTBB dependants are those aged under 13.

#### Sec5DepTest

The Sec5DepTest macro determines if a person is a dependant under section 5 of the *Social Security Act 1991*. This definition is used for most allowances.

Under 15 year olds: we check if they are studying full time, earning less than the dependent child income limit for under 16 year olds (the amount is weekly so it must be multiplied by 2 when comparing it with the fortnightly ordinary income variable), and not eligible for Youth Allowance, Austudy, DSP or NSA.

For 15 to 22 year olds: we check if they are studying full time (at secondary school or not at school), if their income is under the annual limit for 16 and over and that they are not receiving Youth Allowance, Austudy, DSP or NSA on the SIH (as a proxy for the choice they make between Youth Allowance and FTB).

If the person is deemed to be a section 5 dependant, their flag *DepsSec5Flag* is set to 1.

#### SSPlusDepTest

The SSPlusDepTest macro determines if a person is aged 15 to 17 and not in receipt of Youth Allowance on the SIH. Their *DepsSSPlusDepFlag* is set to 1.

***MaintDepTest***

The **MaintDepTest** macro determines if a person is eligible to receive maintenance income i.e. a person is aged 16 to 17, or they are aged 18 and are in full-time study. If eligible, their*DepsMaintFlag* is set to 1. In addition, if they receive Youth Allowance, their *DepsYaMaintFlag* is also set to 1.

When calculating the maintenance income free area where there are other Youth Allowance recipients in the family, there is no requirement that the other youth recipient be workforce dependent.

## The DVA Module

### Overview

The DVA module determines eligibility, assigns parameters and calculates final outcomes for Department of Veteran’s Affairs (DVA) payments modelled in CAPITA. Note that some DVA payments are included in the model by uprating incomes reported in the Survey of Income and Housing (i.e. they are not explicitly modelled).

### Payments covered

* DVA Service Pension (modelled)
* DVA Disability Pension (uprated SIH data)
* DVA War Widows Pension (uprated SIH data)

### 1. Determine eligibility

The **DVANotMod** macro determines whether an individual is eligible for the DVA Disability Pension and the DVA War Widow Pension, and sets their payment rate equal to the SIH value (which was uprated as part of the basefiles construction process). The **DvaServPenEligibility** macro determines whether the reference and spouse are eligible for the DVA Service Pension.

### 2. Assign parameters

The **DvaParameters**macro assigns appropriate DVA Service Pension parameters to the reference and spouse, provided they are eligible.

### 3. Assign the maximum rate of rent assistance

The **RentAssistMaxRate** macro determines the maximum possible rate of rent assistance receivable by the income unit. This operation is performed for all income units which do not have FTBA dependants, regardless of whether they receive DVA Service Pension. These parameters are also used in later modules, to calculate rent assistance for recipients of other payments which do not have FTBA dependants. Rent assistance for those with FTBA dependants is calculated in the FTB module.

### 4. Assign rent assistance for families without children

For income units without FTBA dependants, the **DvaRentAssistAlloc** macro assigns the full maximum amount of rent assistance to singles receiving DVA and shares the maximum amount between couples where at least one member of the couple receives DVA. Income units with children will receive their rent assistance with their FTBA payment.

### 5. Calculate the pension reduction amount

The **PenReduction** macro determines the amount the DVA Service Pension is reduced by means testing.

### 6. Calculate the amount of service pension received

The **HarmerPensionCalc** macro reduces the maximum DVA Service Pension by the pension reduction amount calculated in Step 5. It also distributes the pension between the components. It also assigns pension recipients a maximum rate flag if their reduction is zero, or a part rate flag if they receive a non-zero, reduced amount of the pension.

### 7. Aggregate DVA payments

Determine the aggregate amount of DVA that each individual receives, which includes the DVA payments that are uprated and not modelled (DVA Disability Pension and DVA War Widow Pension).

### The macros

#### DvaNotMod

##### The ***DvaNotMod*** macro determines whether an individual is eligible for the DVA Disability Pension and the DVA War Widow Pension, and sets their payment rate equal to the SIH value. Recall that the SIH values were uprated in the *Uprate* module of the *BasefileCallingProgram*. Also note the inclusion of the ‘Nm’ in the DVA Disability Pension and DVA War Widow Pension variable names to indicate that they are not modelled.

#### DvaServPenEligibility

##### A person is deemed eligible for the DVA Service Pension, and their *DvaType* set to ‘SERVICE’, if either they are receiving the DVA Service Pension on the SIH, or their partner is receiving the Service Pension on the SIH and the person is not in receipt of the Age Pension on the SIH. Receipt of the Service Pension on the SIH is used as a proxy for military service either by themselves or their partner.

#### DvaParameters

##### If an individual or either member of a couple is eligible for the DVA Service Pension, the **PenParmAlloc** macro is called to assign either couple or single pension parameters to the income unit as appropriate.

#### PenParmAlloc

##### The macro assigns either couple or single pension parameters as appropriate. This macro is also called in the *Pension* module.

#### RentAssistMaxRate

##### For families without FTBA dependants, the *income unit* rent assistance parameters are assigned depending on whether they are single or a couple. People living in shared accommodation are only assigned two thirds of the maximum rate. For families with FTBA dependants, these parameters are assigned in the FTB module prior to calling this macro.

Renters paying more than the minimum rent to receive rent assistance are assigned a maximum possible amount of rent assistance which is a proportion (currently 75 per cent) of the difference between the rent they pay and the minimum rent required to receive rent assistance.

Those income units not renting, or paying less than the minimum rent, are assigned a zero rate of rent assistance. This macro is also called in the FTB module.

#### DvaRentAssistAlloc

The maximum amount of rent assistance payable to an individual is derived from the income unit parameters. Singles receiving the DVA Service Pension are assigned the maximum rate of rent assistance for the income unit. If either member of the couple is receiving a DVA Service Pension, each member of the couple is assigned half the maximum rate of rent assistance for the income unit.

#### PenReduction

##### Means testing reduces pensions by the larger of: a proportion (the taper rate) of the amount by which an individual’s ordinary income exceeds the pension income test threshold; or a proportion of the amount by which an individual’s assessable assets exceed the assets test threshold. The **PenReduction** macro determines this reduction and it is applied in the **HarmerPenCalc** and the **OldPensionCalc** macros.

##### This macro is also called in the *Pension* module.

#### HarmerPensionCalc

##### If a person has no pension reduction, they receive the full rate of all components and the pension rate type (PenRateType) is flagged as ’Maximum Rate’. Otherwise, if the reduction is less than the total pension, the amount of the reduction is compared with the total cumulative value of the pension components. The components are reduced in order until the reduction has been fully applied (and the person’s PenRateType is flagged as ‘Part Rate’). If no pension is payable, the pension type is set to ‘ ‘.

##### The amounts of the pension components received are summed to give the total pension amount and the components are assigned to variables specific to the particular pension.

##### This macro is also called in the *Pension* module.

#### DVAAggregates

Total amount of DVA received is equal to the sum of the DVA total variable calculated for the payments modelled in CAPITA (DVA Service Pension) and the total of the DVA payments that are uprated but not modelled in CAPITA (DVA Disability Pension and DVA War Widow Pension).

## The Pension Module

### Overview

The *Pension* module determines eligibility, assigns parameters and calculates final outcomes for Department of Social Services (DSS) pensions modelled in CAPITA. There are two types of pensions. ‘Harmer’ pensions are those covered by changes resulting from the 2009 Harmer Pension review, the remainder are described as ‘old’ pensions.

### Payments covered

* Age Pension (Harmer pension)
* Disability Support Pension (Harmer pension if 21 or over, old pension if under 21)
* Carer Payment (Harmer pension)
* Parenting Payment Single (old pension)
* Wife Pension[[2]](#footnote-3) (Harmer pension)

### 1. Determine eligibility

The **PenEligibility** macro determines whether the reference and spouse of an income unit are eligible for the following pensions: Age Pension, Disability Support Pension, Carer Payment, Parenting Payment Single (reference person only), and Wife Pension (spouse only).

### 2. Assign parameters

The **PenParameters** macro assigns the appropriate payment parameters if the reference or spouse is eligible for the Age Pension, Disability Support Pension (aged 21 and over), Carer Payment, or Wife Pension. The macros **DspU21ParmAlloc** and **PpsParmAlloc**, perform the same role for Disability Support Pension (for people aged under 21) and Parenting Payment Single recipients, respectively. Note that the maximum rent assistance parameters have already been assigned in the DVA module.

### 3. Assign Rent Assistance

The **PenRentAssistAlloc** macro assigns the maximum personal rate of rent assistance an individual can receive in a similar way to the **DvaRentAssistAlloc** macro from the *DVA* module.

### 4. Calculate pension reduction amount

The **PenReduction** macro (defined in the DVA module) calculates the total reduction amount that is applied to an individual’s maximum payment rate due to the income and assets test.

### 5. Calculate pension components

Either the **HarmerPensionCalc** (located in the DVA module) or the **OldPensionCalc** macro is called for each pension as appropriate. They reduce the maximum pension by the pension reduction amount calculated in Step 4 to work out the final pension amount that the reference and spouse receive. They also distribute the pension between the components. The macros also assign pension recipients a maximum rate flag if their reduction is zero or a part rate flag if they receive a non-zero, reduced amount of the pension.

### The macros

#### PenEligibility

##### Age Pension (‘AGE’) eligibility is assessed based on the person’s age and their year of arrival in Australia. People are only eligible for a single social security payment, so the macro also checks whether *DVAType* or *PenType* have already been assigned. If people are already receiving the Widow Allowance, DSP or Carer Payment on the SIH, they will not be eligible for the Age Pension. If the person is eligible for the Age Pension, they are assigned a *PenType* of ‘AGE’.

##### Disability Support Pension (‘DSP’ or ‘DSPU21’) uses receipt on the SIH as a proxy for a disability. To receive the full disability pension, a person must be either 21 or over or have a dependent child. These people receive a *PenType* of ‘DSP’. Those under 21 years of age are assigned a *PenType* of ‘DSPU21’. Note that since disability information is not currently collected on the SIH, if the payment were to have more generous income testing, modifications to CAPITA would be required to capture additional recipients.

Carer Payment (‘CARER’) uses receipt on the SIH as a proxy for caring responsibilities, and the presence of a person being cared for in the income unit. As per the Working Age Payment Reforms from the 2017-18 Budget, the Wife Pension will cease on 20 March 2020[[3]](#footnote-4), and those receiving the Wife Pension will transition onto either the Carer Payment, JobSeeker Payment or Age Pension. In CAPITA individuals receiving the Wife Pension and Carer Allowance on the SIH are assigned eligibility for the Carer Payment.

Parenting Payment Single (‘PPS’) eligibility is tested for the reference persons only since, by definition, members of a couple are ineligible. Receipt of Parenting Payment on the SIH is used as a proxy for activity test for people with a youngest child over 6 and the youngest child in the family must be under 8.

Wife Pension (‘WIFE’) is only payable to spouses of Age Pension or Disability Support Pension recipients. This payment is grandfathered and only payable to recipients already receiving the payment in 1995. The reduction in wife pension recipients over time is incorporated in the model through benchmarking, which will reduce the weight of records receiving wife pension over time. The Wife Pension will cease on 20 March 2020.

#### PenParameters

##### Depending on a person’s pension eligibility, the appropriate macro is called to assign either couple or single pension parameters as appropriate. Most pensioners are assigned the Harmer pension parameters using the **PenParmAlloc** macro located in the DVA module. DSPunder21 and PPS are assigned their different parameters using the **DspU21ParmAlloc** and **PpsParmAlloc** macros respectively.

#### DSPU21ParmAlloc

If an individual or either member of a couple is eligible for DSP under 21, the **DspU21ParmAlloc** macro is called to assign either couple or single pension parameters as appropriate in a similar manner to **PenParmAlloc**.

#### PPSParmAlloc

If an individual is eligible for Parenting Payment Single, the **PPSParmAlloc** macro is called to assign pension parameters in a similar manner to **PenParmAlloc**. Unlike other pensions, the means test threshold for this pension depends on the number of children.

#### PenRentAssistAlloc

This macro works in a similar way to **DVARentAssistAlloc**.

#### OldPensionCalc

Similar to the **HarmerPensionCalc** macro, this macro determines a person’s total pension and all the components thereof (including rent assistance) and adds flags to indicate whether the pension is paid at the full-rate or the part-rate. Old pension recipients receive pharmaceutical allowance but not the pension supplement minimum or remaining amounts.

## The Allowance Module

### Payments covered

* Youth Allowance (Student) and Youth Allowance (Other)
* Austudy
* Parenting Payment Partnered
* Newstart Allowance (JobSeeker Payment after 20 March 2020)
* Widow Allowance
* Special Benefit (uprated only)
* Sickness Allowance (uprated only)
* Partner Allowance (uprated only)
* Abstudy (uprated only)

### Overview

The *Allowance* module determines eligibility, assigns parameters and calculates final outcomes for DSS allowance payments in CAPITA.

The last four allowances listed are not explicitly modelled; rather, their values received on the SIH are uprated. These are all payments with very few or declining numbers of recipients.

### 1. Determine eligibility

The **AllowNotMod, AllowEligibility** and **YaEligibility** macros determine whether the reference or spouse is eligible for any of the allowances listed above. Eligibility for dependants 1 to 4 is determined in Step 7.

### 2. Assign parameters

The **AllowParameters** macro assigns the appropriate payment parameters to the reference and spouse based on their allowance eligibility. These are the maximum amounts for each payment and are subject to means testing in Step 5. Note that the rent assistance parameters have already been assigned in the *DVA* module.

### 3. Calculate rent assistance maximum rates

These are calculated in the *DVA* module (for income units without FTBA dependants) and the *FTB* module (for income units with FTBA dependants).

### 4. Assign Rent Assistance

The **AllowRentAssistAlloc** macro determines the maximum personal rate of rent assistance an individual can receive and is subject to means testing in Step 5.

### 5. Parental income test and maintenance income test

This section calculates the reduction for parental income and incorporates the parental income test and maintenance income test for workforce dependant recipients.

The **YaParIncTestFam** macro collects family level information of parental income and maximum payment of Youth Allowance payments a family can receive in a fortnight (*MaxFamYa*). The macro obtains this information for students 1 to 4 in the parental income unit and for students in separate income units but still subject to the parental income test. The macro is run for the parental (first) income unit in each family and for dependants in their own income unit. The ‘look-ahead’ method is used to read income information from income units within the same family.

The macro also calculates the parental test excess amount (*AllPareIncTestExF*) which is the amount for which parental income exceeds the parental test income free area.

As *MaxFamYaF* and *AllPareIncTestExF* are only recorded against the parental income unit, it retains *MaxFamYaF* and *AllPareIncTestExF* for own income units within the family.

The **YaParIncTestRes** macro calculates the reduction in Youth Allowance due to the parental income test (known as the parental test result from 1 January 2017 onwards) for students 1 to 4 in the parental income unit and for students in their own income units but still subject to the parental income test.

The **YaMaintIncTestRes** macro calculates the fortnightly test result due to the maintenance income test. It is run from 2017 for students 1 to 4 in the parental income unit and for students in their own income units who are workforce dependent.

The **YaParTestRed** macro calculates the person’s fortnightly reduction for parental income. Prior to 2017 and the introduction of the maintenance income test, the fortnightly reduction for parental income is equivalent to the outcome of the parental income test.

From 2017 onwards, the reduction for parental income takes into account the parental income test result and the maintenance income test result. It is run for students 1 to 4 in the parental income unit and for students in their own income units.

### 6. Calculate allowance reduction amount

The **AllowReduction** macro calculates the total amount the individual’s maximum payment rate is reduced by means testing. The reductions calculated are from the personal income test, the partner income test (if applicable) and the parental income test (if applicable). The final reduction amount is the maximum of the personal income test and the parental income test reduction amounts, plus the partner income test reduction.

### 7. Calculate allowance components

If the reference or spouse is eligible for an allowance, the **AllowCalc** macro is called for the type of allowance for which they are eligible. The macro reduces the maximum payment rates by the allowance reduction amount to work out the final allowance outcome and distributes the allowance between the components by sequentially reducing the basic allowance, Rent Assistance, Pharmaceutical Allowance and finally the Energy Supplement.

### 8. Determine allowance outcomes for students 1 to 4

The **DepAllowCalc** determines the allowance outcomes for students 1 to 4.

### The macros

#### AllowNotMod

If a person is receiving Special Benefit, Sickness Allowance or Partner Allowance on the SIH, or if they are receiving Austudy or Abstudy on the SIH but are above the maximum eligibility age for Youth Allowance (meaning that they must be receiving Abstudy), then they are assigned these payments in CAPITA and their payment rate is the uprated amount of their allowance, calculated as part of the basefile construction process. As there is no detail on the individual components of these payments, only the total amount is determined. Note that no new grants of partner allowance have been awarded since 2003 and the payment will cease on 1 January 2022. The Sickness Allowance will cease to exist after 20 March 2020 as per the Working Age Payment Reforms from the 2017-18 Budget. Recipients will transition onto the JobSeeker Payment (previously known as Newstart Allowance).

#### AllowEligibility

The **AllowEligibility**macro determines if an individual is eligible for Austudy, PPP, NSA or widow allowance. Eligible people are assigned an allowance type and a more detailed allowance sub-category used to later allocate the appropriate parameters.

Austudy (‘AUSTUDY’): A person is deemed eligible for Austudy and given ‘AUSTUDY’ for their allowance type if they are over the minimum age, studying full-time but not still in school, and not receiving a pension or other allowance. Receipt of Austudy on the SIH is used to proxy the assets test and liquid assets waiting period. The specific sub-category of Austudy (for example, ‘SINGNODEPS’ for singles without children) is determined by the **YaAusType** macro.

Parenting payment partnered (‘PPP’): A person is deemed eligible for Parenting Payment Partnered, if: their youngest child is under the maximum PPP age; they are in a couple; and they are not receiving another pension or allowance. Receipt on the SIH of either a Parenting Payment or NSA is also used to test eligibility for PPP in CAPITA. Receipt of NSA is used as individuals tend to report receiving NSA on the SIH when they are actually receiving a parenting payment and vice versa, this confusion comes about because the payment rates are the same and many individuals switch between the payments depending on their family situation. As only one member of a couple can receive PPP, we test which member of the couple has the lower income and they will be eligible for the PPP. The allowance sub-category ‘COUPLE’ is assigned as there is no other possibility for Parenting Payment Partnered recipients.

Newstart allowance (‘NSA’)/ JobSeeker Payment after 20 March 2020: A person is deemed eligible for Newstart Allowance if they are: over the age of independence; under age pension age; not studying full-time; or receiving a pension or other allowance. Receipt of NSA or Youth Allowance on the SIH is used as a proxy for the activity test and liquid assets waiting period as both payments have similar conditions. This allows recipients to transition between the two payments when the age of independence changes. A person can also be eligible for NSA in CAPITA if they are reporting receipt of Parenting Payment on the SIH (for the reason explained in the PPP paragraph). As a result of the Working Age Payment reforms individuals receiving Wife Pension on the SIH without Carer Allowance, and those receiving Sickness Allowance are also eligible to receive the JobSeeker Payment after 2020 (formerly Newstart Allowance). The sub-category of Newstart is determined using the **NsaWidType** macro.

Widow Allowance (‘WIDOW’): To be deemed eligible for Widow Allowance a person must be a single female over the minimum widow allowance age with no recent workforce experience or pension or allowance. Receipt of widow allowance on the SIH is used as a proxy for the assets test and the person’s widow status. In CAPITA, individuals receiving Widow Allowance on the SIH but above the minimum widow allowance age are eligible to receive Newstart Allowance (JobSeeker Payment after 2020).

As for Newstart allowance, the sub-category of widow allowance is flagged by calling the **NsaWidType** macro as the payment rates of Widow Allowance are the same as the rates of Newstart Allowance. Widow Allowance will cease to exist on 1 January 2022.

#### YaEligibility macro

The **YaEligibility** macro determines whether a person is eligible for Youth Allowance (Full-Time Student) or Youth Allowance (Other). Youth Allowance eligibility is defined in a separate macro to other allowances as, in addition to applying for the reference and spouse, it also applies to dependants (1 to 4).

This macro has two purposes. Under normal use (*SihCondition* is set to blank) it determines whether a person is eligible for Youth Allowance. In this case, receipt of either NSA or Youth Allowance on the SIH is used as a proxy for the activity, liquid assets and assets tests. However, the **YaEligibility** macro can also be used to determine whether a person is *theoretically* eligible for Youth Allowance by waiving the receipt on the SIH condition (*SihCondition* = ‘NoReceiptSih’) to determine the number of *potential* Youth Allowance recipients for the parental means test. This has the effect of ignoring the assets and activity tests and the choice made by the recipient if they may also be eligible for FTB.

A person is deemed eligible for Youth Allowance (Student) and receives and allowance type ‘YASTUD’ if they are under the upper age limit for Youth Allowance, studying full time, and not receiving any other pensions or allowances. In addition, in normal use of the macro, receipt of Youth Allowance or NSA on the SIH is used as a proxy for activity and assets tests and the decision to receive Youth Allowance over FTB. When being used to identify *potential* Youth Allowance recipients for determining the parental income test, the receipt on the SIH condition is waived. These dependants need to be flagged as Youth Allowance eligible in order to calculate the family’s maximum amount of Youth Allowance, required to determine the Youth Allowance parental income test reduction.

A person is deemed eligible for Youth Allowance (Other) if they are: the right age; not studying at secondary school or full-time; and not receiving any other pensions or allowances. Again, in normal use of the macro, receipt of Youth Allowance or NSA on the SIH is used as a proxy for activity and assets tests but when being used to identify *potential* Youth Allowance recipients for determining the parental income test, the receipt on the SIH condition is waived.

The specific sub-category of Youth Allowance is determined by the **YaAusType** macro.

#### YaAusType macro

The **YaAusType**macro determines the rate of Youth Allowance or Austudy a person can receive and allocates the appropriate allowance sub-category. The rates are based on: the age of the recipient, whether they are single or in a relationship, whether they live with their parents or away‑from‑home (tested using the *FamPos*) and whether or not they have children. For the away-from-home dependants, a proportion of them are randomly assigned to be away-from-home ‘with cause’ and ‘without cause’, and will therefore, receive different rates. The information required to determine this is not available on the SIH and has been imputed by setting the proportion to 0.5 in the CPS.

The flags are:

* ‘YNGAH’ – single 16-17 year olds living at home or living away from home without a cause
* ‘OLDAH’ – over 18 year old singles living at home or away from home without a cause
* ‘SINGNODEPS’ – singles living away from home with a cause
* ‘SINGDEPS’ – single with dependent children
* ‘COUPNODEPS’ – members of a couple without children
* ‘COUPDEPS’ – members of a couple with children

#### NsaWidType macro

The **NsaWidType**macro determines the rate of Newstart allowance or widow allowance a person receives and allocates the appropriate allowance sub-category. The rates are based on the age of the recipient, whether they are single or in a relationship, and whether or not they have children.

For single people aged 60 years and over, a proxy is required to determine whether they have been unemployed for the required nine month period. This is estimated by using the SIH variable giving the duration of unemployment as less than 1 year.

The flags are:

* ‘OLDLTR’ – single, over 60 and a long term unemployment benefit recipient
* ‘SINGNODEPS’ – singles without dependants
* ‘SINGDEPS’ – single with dependent children
* ‘COUP’ – members of a couple (with or without children)

#### AllowParameters macro

**AllowParameters** macro calls **AllParmAlloc** to assign either couple or single allowance parameters appropriate to the type of allowance they are receiving. There are three categories of allowance, Unemp (unemployed), YngUnemp (unemployed under the age of independence) and Stud (students), each with different payment parameters.

#### AllParmAlloc

**AllParmAlloc** assigns each person their correct parameters based on their *AllType* and *AllSubType* variables, as well as their couple or single status. For example, a reference person with *AllType* = ‘NSA’ and *AllSubType* = ‘SINGNODEPS’, the code will resolve to:

AllBasicMaxFr = NsaSingNoDepsBasicMaxFr

In terms of the Energy Supplement, if the option to model grandfathering is chosen (described on page 3 of this document) the ES is assigned randomly to recipients based on the probability of being a grandfathered customer. Please note this option is not available in the publically released version of CAPITA. If the option RunEs = Y is chosen in RunCAPITA, all allowees eligible for the payment will receive it, and if the option RunEs = N is chosen, no allowees will receive it.

Pharmaceutical allowance parameters are only assigned for single principal carers receiving either NSA or Youth Allowance (Other) and people over 60 who have been receiving income support for more than nine months (‘OLDLTR’ )[[4]](#footnote-5).

The taper rates for NSA and Widow Allowance are over-written for single principal carers who have a lower taper rate (40%) than standard allowance recipients.

#### ***AllowRentAssistAlloc***

**AllowRentAssistAlloc** macro works in a similar way to **DVARentAssistAlloc**.

#### AllowReduction

**AllowReduction** determines and reference’s and spouse’s total reduction from the personal income, partner income and parental income tests. A similar calculation for dependants 1 to 4 is performed later in **DepAllowCalc**.

For members of a couple, whose partner is receiving a pension or DVA payment, the income used for allowance test that was calculated in the income module is reestimated and the value of IncAllTestF is overwritten. Instead of using their individual income, half the couple’s combined income is used, similar to the pension income test.

The individual income test is applied for all reference persons and spouses, if they exist, using the **IndivIncTest macro**.

For couples, where the spouse does not receive a pension or DVA service pension, the reduction due to the partner income test is calculated using the **PartIncTest** macro. We do not run this bit of code if the partner receives a pension or DVA entitlement as the individual income test on the spouse will not need to be performed as they will not be receiving an allowance. Additionally, the partner of a person receiving a pension or DVA entitlement is exempted from the allowance partner income test[[5]](#footnote-6).

Otherwise, if the person is not workforce independent, the reduction from the parental income test is determined. If the family’s maximum amount of Youth Allowance is zero, then the parental income test reduction amount is set to zero as it will not be used. Otherwise the allowance reduction amount is equal to the parents excess income (determined in the **YaParIncTestReduc** macro), multiplied by the proportion of the family’s maximum Youth Allowance amount that that person contributes. This value is rounded to the nearest 10 cents. A person will never be subject to both the partner and parental income tests as, if they have a partner, they are considered to be workforce independent regardless of their age. Note: the parental income test macro, **YaParIncTestReduc,** is called later in the allowance module as it only applies to reference persons and spouses in their own income unit (i.e. the second or subsequent family income unit), which will be processed after the parental income unit is complete, or dependants in the parental income unit (their Youth Allowance is calculated later by **DepAllowCalc**).

The final allowance reduction for the reference person is equal to the maximum of the personal and parental income test reduction added to the partner income test reduction. In this equation, only one of the partner and parent reductions will be more than zero (for the reason stated in the previous paragraph). The maximum of the personal and parental income test is used because a dependant is subject to the income test which reduces their payment by the most.

The final allowance reduction for the spouse is the addition of their personal and partner income test reductions. This is because the spouse can never be workforce dependent (for the reason stated above).

#### ***IndivIncTest***

**IndivIncTest** determines the allowance reduction amount from the personal income test. This is used to determine the total allowance reduction amount for reference and spouse in the **AllowReduction** macro or for dependent students 1 to 4 in the **DepAllowCalc** macro.

If income for the allowance test is under the first threshold there is no personal income test reduction.

Otherwise, if income is between the first and second thresholds, the personal income test reduction is equal to the amount of income above the first threshold multiplied by the first taper rate.

Otherwise, if income is above the second threshold, the personal income test reduction is equal to the amount of income above the second threshold multiplied by the second taper rate added to the difference between the first and second threshold multiplied by the first taper rate.

#### PartIncTest

The **PartIncTest** macro determines the allowance reduction amount from the partner income test. It calls the **PartIncTestThresh** macro to determine the threshold used for the partner income test. The idea of the partner income test is that once a person’s allowance has been reduced to zero, any additional income they earn affects their partner’s income.

**Scenario 1:**

If both the reference and spouse receive some allowance amount after applying their personal income test reductions then the partner income test does not apply and *AllRedPartIncF* is set to zero for both the reference and spouse. Note: the test here includes amounts received from pharmaceutical allowance and rent assistance.

**Scenario 2:**

If reference person’s payment is reduced to zero by the personal income test but the spouse receives at least some payment, then the partner income test applies. The reference’s allowance type flag is set to “” and rent assistance is reassigned so that the spouse gets the couple’s full allotment (as their reference person is not using their allotment). The **PartIncTestThresh** macro determines the partner income test threshold for use in applying the reduction on the spouse resulting from the reference person’s excess income. The spouse’s reduction resulting from the reference’s excess income is determined by taking the maximum of zero and the reference person’s income less the reference person’s income threshold, multiplied by the spouse’s partner income test taper rate.

**Scenario 3:**

Conversely, if spouse’s payment is reduced to zero by the personal income test and the reference person still receives some payment, then the partner income test applies. This is the reverse of Scenario 2.

**Scenario 4:**

If both the reference and spouse do not receive any allowance after their personal income test reductions are applied to their maximum payment rates, the partner income test has no affect.

#### PartIncTestThresh

The **PartIncTestThresh** macro determines the partner’s income threshold used in the partner income test. This is the income at which the allowance they are “theoretically” entitled to reduces to zero.

Before calculating the partner income test threshold a flag is created to determine if the person is receiving the Energy Supplement, and therefore whether the notional amount of Energy Supplement should be included in their partner’s “theoretical” allowance entitlement. The flag (AllEsFlag) is equal to 1 if the person is receiving the ES and 0 otherwise. In the calculations that follow, the ES amount is then multiplied by the flag, which will resolve to the relevant ES amount or 0 depending on the flag. This feature of the policy is related to the removal/grandfathering of the ES for income support recipients – the current start date of this measure is 20 September 2017.

If a person receives an allowance, the partner income test threshold is equal to partner’s total payment less the difference between their second and first personal income test thresholds multiplied by the first personal income test taper. This amount is then divided by the second personal income test taper. The second personal income test threshold is then added to this amount. This value is not actually used in the code as these people will not affect their partner’s income but it is provided for information.

If a person does not receive an allowance:

If they are younger than the age of independence, their “theoretical” payment is Youth Allowance (Other). If they have dependent children then the partner income test threshold is equal to the Couple *with* Dependents Basic Youth Allowance Rate, less, the difference between their second and first personal income test threshold multiplied by the first personal income test taper. This amount is then divided by the second personal income test taper. The second personal income test threshold is then added to this amount. If they do not have children the Couple *without* Dependents[[6]](#footnote-7) Basic Youth Allowance Rate is used.

If they are over the age of independence and under the age pension age, the partner income test threshold is equal to the Couple Basic Newstart Rate, less the difference between their second and first personal income test threshold multiplied by the first personal income test taper. This amount is then divided by the second personal income test taper. The second personal income test threshold is then added to this amount.

If they are over the Age Pension age, then the partner income test threshold is equal to the sum of the Couple Basic Newstart Rate and the Couple rate of the Pension Supplement, less, the difference between their second and first personal income test threshold multiplied by the first personal income test taper. This amount is then divided by the second personal income test taper.

Finally, the CEIL function is used to round-up the partner income test threshold to the nearest dollar.

#### YaParIncTestFam

This macro calculates the parent’s maximum fortnightly payment from Youth Allowance for use in the Youth Allowance parental income test for workforce dependent youths, calls the **DepAllowElig** macro to determine eligibility and assigns parameters for dependent students 1 to 4. The macro also calculates the number of Youth Allowance dependants in the family.

Variables used in the parental income test are initialised here rather than in the Initialisation module. This is so they are not overwritten for subsequent income units in the family but are overwritten when the next family unit is processed.

In the parental unit, if the actual age of student dependants 1 to 4 is greater than zero (if that dependant exists), the **DepAllowElig** macro is run for that dependant.

The variables used to determine the parental income test reduction amount from the parent’s income unit observation are retained. These include the family’s maximum Youth Allowance allocation and the parental income test excess amount.

If there is more than one income unit in the family, the eligibility for Youth Allowance of dependants in their own income unit is determined and parameters are assigned based on their allowance type.

The variables from the current observation needed to determine the Youth Allowance parental income test reduction are retained and renamed by adding a ‘\_’ after the variable name. For example, the variable YaSihIncWr (the amount of Youth Allowance received on the SIH) would be renamed as YaSihIncWr\_. This also includes the random numbers required to perform the Energy Supplement grandfathering test for allowances. For example RandNsaEsGfthr is renamed to RandNsaEsGfthr\_. However we condition the code to only include these variables when running the distributional code and not the cameo code (as we do not want to use the random numbers to model grandfathering since cameos are selected to either receive the ES or not (see Section 3 – Using CAPITA).

The person’s maximum Youth Allowance components (basic rate and ES) are then added to the family’s total maximum Youth Allowance variable.

This process is repeated for each additional income unit in the family.

The parent’s maximum rate of FTB-A (which includes standard rate, Energy Supplement, end of year supplement, newborn supplement, and rent assistance) will be added to the maximum Youth Allowance pool for those with FTB-A siblings (2015-16 Budget measure).

After processing all the income units within the family, the parental income excess amount (*AllPareIncTestExA*) is calculated and then converted to a fortnightly amount (AllPareIncTestExF\_).

***FtbaRentAssistAlloc***

The **FtbaRentAssistAlloc** macro is in the Allowance module (instead of the FTB module) because of the addition of the parent’s FTB-A maximum rate to the parental pool for Youth Allowance for those with FTB-A siblings from 1 July 2016 as announced in the 2015-16 Budget.

The macro classifies the renter type based on the relationship status of the parent(s) and the number of FTB-A dependants. It then calls the **AllocateRentParam** macro which allocates the parameters corresponding to the following renter type:

* If a single parent has one or two FTB-A dependants, the renter type is classified as ‘SDeps1\_2’. Otherwise, if there are more than two FTB-A dependants, the renter type is classified as ‘SDepsMany’.
* If a couple has one or two FTB-A dependants, the renter type is classified as ‘CDeps1\_2’. Otherwise, if there are more than two FTB-A dependants, the renter type is classified as ‘CDepsMany’.

***AllocFtbaRentParam***

This macro assigns the parameter for the minimum rent required to receive rent assistance and the parameter for the maximum rent assistance payable, both of which depend on the renter type as classified in the **FtbaRentAssistAlloc** macro.

#### YaParIncTestRes

If a parent receives a pension, allowance or DVA entitlement, the parental income test reduction amount is set equal to zero[[7]](#footnote-8). Otherwise, if the maximum amount the family can receive from Youth Allowance is positive, the parental test result is equal to the fortnightly excess amount multiplied by the individual’s share of the family’s maximum Youth Allowance payment rate and the parental income test taper rate. The annual parental test result is derived by multiplying the fortnightly amount by 26.

#### YaMaintIncTestRes

If a person exists and is eligible to receive maintenance income*,* then their assigned maintenance income is the family total maintenance income divided by the number of children eligible to receive maintenance income in the family. That is, the maintenance income is apportioned equally among children eligible to receive maintenance income.

If the person is eligible for Youth Allowance and receives maintenance income, the maintenance income test result applies (unless a parent is a permanently blind pensioner). The person’s maintenance test income free area is calculated based on the number and type of other maintenance income recipients in the family.

If there are no FTB-A children in the family, the maintenance free area includes the base amount plus the number of any other Youth Allowance recipients who receive maintenance income in the family multiplied by the additional child add-on amount.

If there are FTB-A children in the family, the maintenance free area is only the additional child add-on per person. The base maintenance income free area is not included.

The maintenance income test result is the amount by which the maintenance income assigned to each person exceeds the income free area, multiplied by the taper rate of 50 cents in the dollar.

#### YaParTestRed

Prior to 1 July 2017, the reduction for parental income is equal to the parental income test result.

From 1 July 2017 onwards, the reduction for parental income will be capped by the Maintenance Income Test (MIT) reducible amount which is the difference between the FTB-A maximum rate for secondary school students and the base rate. If the parental income test result is equal to or more than the MIT reducible amount, the person’s reduction for parental income is the parental income test result. Otherwise the maintenance income test applies and the notional amount, which is the sum of the parental income test result and the maintenance income test result, is calculated. The person’s reduction for parental income is equal to the notional amount unless it exceeds the MIT reducible amount, in which case it will equal the MIT reducible amount.

#### AllowCalc

**AllowCalc** uses the total allowance reduction amount calculated in the **AllowReduction** macro for reference and spouse or the **DepAllowCalc** macro for dependent students 1 to 4 to determine the final allowance amounts. Similar to the **HarmerPensionCalc** and **OldPensionCalc** macros, this macro determines a person’s total allowance and all the components thereof and adds flags to indicate whether the pension is paid at the full-rate or the part-rate.

#### DepAllowElig

This macro determines Youth Allowance eligibility (**YaEligiblity** macro) for dependent students 1 to 4, calculates their income (**OrdIncome** macro) for the personal income test, assigns them their Youth Allowance parameters (**AllowParameters** macro) and if appropriate, adds their maximum Youth Allowance outcomes to the family’s maximum Youth Allowance outcome. It is called in the YaParIncTestReduc macro only for dependent students 1 to 4 which exist.

The **YaEligibility** macro is called with the macro inputs, &psn, 0 for DepChild, 0 for Couple and NoReceiptSih. This is because all students 1 to 4 will be single and without dependants. The NoReceiptSih condition is also used so that students who are eligible for Youth Allowance but are means tested off it are included in the MaxFamYA calculation.

If the person is workforce dependent, the person’s maximum Youth Allowance components (basic rate and ES) are then added to the family’s total maximum Youth Allowance variable and the number of Youth Allowance deps in the family is incremented by 1.

The YaEligibility macro is then called again, this time without the NoReceiptSih condition. This is so actual eligibility for Youth Allowance in CAPITA can be re-determined using the receipt on the SIH condition.

If the student is not receiving an Allowance their AllSubType variable is stripped from them[[8]](#footnote-9).

#### DepAllowCalc

This macro performs the personal income test (**IndivIncTest** macro), the parental income test if the student is workforce dependent, calculates the total reduction amount and determines final outcomes for the student dependent (**AllowCalc** macro). It is called in the **RunAllowance** macro only for dependent students 1 to 4 which exist.

To determine the personal income test reduction, the **DepAllowCalc** macro calls the **IndivIncTest** macro.

If the person is workforce dependent, the reduction from the parental income test is determined. The allowance amount is equal to the parent’s excess income (determined in **YaParIncTestReduc** macro) multiplied by the proportion of the family’s maximum Youth Allowance amount contributed by that dependent. This value is rounded to the nearest 10 cents.

The final reduction amount is calculated by taking the maximum of 0, the personal income test reduction and the parental income test reduction.

If the allowance type variable is not blank, the **AllowCalc** macro is called with the inputs YaStud and &psn. This is because dependants 1 to 4 are all full-time students and are therefore only eligible for the full‑time student Youth Allowance payment.

## The Income2 Module

### Overview

The Income2 module calculates income according to the definitions used for Family Tax Benefit and by the personal tax system by summing components. Details on the income definitions can be found online.

### Incomes calculated

* Taxable transfer income
* Taxable income
* Rebate income (for SAPTO)
* Net income from working (for MAWTO)
* Rebatable benefit (for BENTO)
* Income for Medicare Levy Surcharge
* Adjusted taxable income

### 1. Calculate Taxable Transfer Income

The **TaxTranInc** macro calculates taxable transfer income for the reference and spouse (where applicable).

### 2. Calculate Taxable Income

The **TaxInc** macro calculates taxable income for the reference and spouse (where applicable).

### 3. Calculate Rebatable Income

The **RebInc** macro calculates rebate income required to determine SAPTO entitlement for the reference and spouse (where applicable).

### 4. Calculate Net Income from Working

The **NetIncWork** macro calculates net income from working, required for determining MAWTO entitlement (this was abolished on 1 July 2014), for the reference and spouse (where applicable).

### 5. Calculate Rebatable Benefit

The **RebBen** macro calculates the rebatable benefit, required for determinining BENTO entitlement, for the reference and spouse (where applicable).

### 6. Calculate Income for Medicare Levy Surcharge

The **IncMedLevSur** macro calculates income for surcharge purposes, required for determining amount of Medicare levy surcharge the reference and spouse (where applicable) are liable for.

### 7. Calculate Adjusted Taxable Income

The **AdjustedTaxIncome** macro calculates adjusted taxable income, required for determining FTB entitlement and childcare entitlement (in the cameo model only), for the reference and spouse (where applicable).

### The macros

All macros in this module sum up the components of income. Only the first is described in detail here. The code commenting is useful in explaining the other macros in the module.

#### TaxTranInc

The **TaxTranInc** macro sums the components of taxable transfer income. Taxable transfer income consists of most income support payments. However, if the person is below age pension age, their pension income (except for PPS) is non-taxable, so not included in taxable transfer income, but it is included in *adjusted* taxable income. A flag (*AtiFlag*) is set to indicate that a person’s pension is non-taxable. This is used in the Boolean statement (*Atiflag* = 0). If this statement is TRUE, the code in brackets resolves to 1 and the payment is included in the sum. If it resolves to FALSE the payment is not included in the sum. All allowances are taxable. Some payments are only present for r, s or both. Conditional code %IF is used to include these income components in the sum if they exist (e.g. for personas r and s) and exclude them if not (e.g. for persons 1 – 4). This avoids trying to add missing values.

#### TaxInc

The **TaxInc** macro calculates assessable income, the sum of private and taxable transfer income. It then determines taxable income (*TaxInc*), assessable income less any deductions.

#### RebBft

The rebatable benefit is used to work out a person’s BENTO. It includes the basic amount of most allowances. Some of the minor allowances (Partner Allowance, Sickness Allowance and Special Benefit) have not been explicitly modelled in CAPITA. Instead their values from the Survey of Income and Housing have been uprated. In these cases, the entire allowance is assumed to be a rebatable benefit.

#### AdjustedTaxIncome

Adjusted taxable income is used to work out a person’s family payments. In addition to taxable income it includes the basic amount of non-taxable pensions, reportable super contributions and removes expenditure on child maintenance.

## The Dependants2 Module

### Overview

The Dependants2 module determines the number of dependants, according to the definitions used in the personal tax system and the family assistance act.

### Dependant definitions applied

* MedicareDependants
* SifsDependants

### 1. Tally up dependants for Medicare levy and Medicare levy surcharge

For each family, the **MedicareDependants** macro counts dependants for Medicare levy (ML) and Medicare levy surcharge (MLS) purposes.

### 2. Tally up dependants for SIFS

For each family, the **SifsDependants** macro counts dependants for the Single Income Family Supplement (this has been closed to new customers from 1 July 2017).

### The macros

#### MedicareDependants

This macro determines the number of ML dependants and MLS dependants.

##### Student dependants in the parental income unit

When SAS encounters the parental income unit in each family (FIRST.FamID), the code loops through dependants 1 to 4 to test whether they are a dependant under the Income Tax Assessment Act (PartVIIB Medicare levy and Medicare levy surcharge) using the **MLDepTest** macro and **MLSDepTest** macro respectively, and assigns flags, e.g. DepsMlFlag1 = 1, if the person satisfies the dependency criteria.

##### Dependants in their own income unit

As in the *Dependants1* module, if the family has dependants in their own income units (ie NumIUu > 1), CAPITA draws their information into the parental income unit using the look-ahead method. It loops through each of the non-parental income units in the family, calculates adjusted taxable income (proxied by the sum of private income and Youth Allowance income from the SIH), tests whether they are a dependant for ML and MLS purposes, and assigns flags accordingly if the person satisfies the dependency criteria.

The macro then calculates the total number of ML dependants and the total number of MLS dependants in the family. The final tally for the number of ML dependants is the sum of those: under 15; under the ML first age limit with income less than the first income threshold; between the ML first (inclusive) and second age limits and with income less than the second income threshold; and under the ML first age limit with income between the first and second income thresholds to a maximum of one child. For MLS purposes, the number of child dependants is the sum of those under the MLS first age limit and older full time students under the second age limit.

#### MLDepTest

If the child is under the first age limit for ML dependants with an adjusted taxable income below the second ML income threshold, the ML dependant flag is set to 1. If the child’s adjusted taxable income is between the first and second ML income thresholds, the ML dependant flag is set to 2.

If the child is older but under the second age limit, studying full time and with adjusted taxable income between the first and second ML income thresholds, the ML dependant flag is set to 1.

#### MLSDepTest

If the child’s age is below the first age limit for MLS dependant or if the child is older but under the second age limit for MLS dependant and studying full time, the MLS dependant flag is set to 1.

#### SifsDependants

This macro tallies up SIFS dependants by counting non-FTBA children who qualify as SIFS dependants and adding them to the number of FTB-A dependants.

Similar to the **MedicareDependants** macro, the **SifsDependants** macro loops through dependants 1 to 4 in the parental income unit to test whether they are a SIFS qualifying child (i.e., if they satisfy eligibility requirements for FTB-A dependants except for the receipt of government payment) and then tallies those that satisfy the SIFS child criteria.

If the family has dependants in their own income units, the look-ahead method is used to test whether they are a SIFS dependant (i.e., if they would have been FTB-A dependants except for the receipt of government payment). It then counts those who qualify as SIFS dependants.

The total SIFS dependants is derived by adding together the number of SIFS dependants (not eligible for FTB-A) and the number of FTB-A dependants.

The SIFS has been closed to new recipients from 1 July 2017.

## The FTB Module

### Overview

The FTB module determines eligibility, assigns parameters and calculates final outcomes for Family Tax Benefit (FTB) payments modelled in CAPITA.

### Payments covered

* FTB Part A (FTB-A)
* FTB Part B (FTB-B)
* Newborn upfront supplement
* Schoolkids bonus – abolished as of 1 July 2016

### 1. Determine eligibility

Eligibility for FTB-A and FTB-B payments is determined based on the family having dependent children for FTB-A and FTB-B, respectively. These dependants are determined in the Dependants1 module.

### 2. Calculate income for income test

The **FtbIncome** macro checks if any of the parents are receiving income support payments and flags those exempted from the income test. If not exempted, it assigns the income to be used for the income test.

### 3. Assign parameters

The **FtbParameters** macro assigns the appropriate FTB-A and FTB-B parameters if the family is eligible for FTB-A and FTB-B payments.

If the family is receiving maintenance income, the **MaintIncParams** macro assigns the appropriate maintenance income parameters based on the number of income recipients and calculates the total maintenance income threshold for the family.

### 4. Assign rent assistance

The **FtbaRentAssistAlloc** macroclassifies the type of renter based on relationship status and number of FTB-A dependent children. It calls the **AllocateRentParam** macro which assigns the appropriate rent parameters to the type of renter. These macros are located in the Allowance module due to the 2015-16 Budget measure to add the maximum rate for FTB-A dependants to the parental pool for Youth Allowance.

The **RentAssistMaxRate** macro (located in the DVA module) determines the maximum possible rate of rent assistance receivable by the family.

### 5. Calculate FTB-A payment

The **Ftbacalc** macro calculates the FTB-A payments using two methods. In the first method, it calculates the maximum rate of FTB-A, the reduction to this rate, and the amount after reduction. In the second method, it calculates the base rate of FTB-A, the reduction to this rate, and the amount after reduction. It determines the higher payment outcome from these two methods and assigns it to the family. For analytical purposes, the macro further classifies the type of FTB-A payment and apportions the amount of FTB-A across its components.

The **Ftbacalc** macro is called twice – (1) for the period when the newborn supplement (NBS) is payable (13 weeks), and (2) for the period when NBS is not payable. The NBS is paid to families with newborn(s).

The **FtbaFinalPay** macro calculates the amount of FTB-A payable to the family for the year by adding the FTB-A payment outcomes for the period when NBS is payable and for the period when it is not payable. It also disaggregates the FTB-A payment into its components.

### 6. Calculate FTB-B payment

The **FtbbCalc** macro calculates the maximum FTB-B rate, the reduction to this rate, and the amount after the reduction. It also disaggregates the FTB-B payment into its components.

### 7. Calculate other payments

This section calculates the baby upfront payments for families with newborns and Schoolkids bonus (before 1 July 2016) for families with FTB dependants studying in primary and secondary schools

### 8. Calculate summary payments

In this section, the FTB payments are allocated to the lower earner of the couple or to the reference for single parent families. The total annual amount of FTB payments to the family and fortnightly amounts of FTB payments are calculated.

### The macros

***FTBIncome***

Families where one or both parents are receiving income support payments are exempt from the FTB-A and FTB-B income tests.

If not exempt:

* For FTB-A, the income test is based on the total adjusted taxable income(s) of the parent(s) in the family.
* For FTB-B, the adjusted taxable incomes of two-parent families are compared to determine allocation to the income tests applicable. The higher adjusted taxable income is used for the primary earner income test while the lower adjusted taxable income is used for the secondary earner income test. For single parents, their adjusted taxable income is used for both tests. If it does not exceed the primary earner income limit, they receive the maximum FTB-B payment. Otherwise, the payment is reduced under the secondary earner income test.

If the family is receiving maintenance income, it is assumed that all children (FTB-A dependants and non-FTB-A dependants) eligible for child support are receiving maintenance income. This is to avoid making imputations on the SIH data which lack information on children receiving support.

* The maintenance income per child is derived and used as a basis for apportioning the maintenance income for FTB-A based on the number of FTB-A dependants.

***FtbParameters***

This macro calculates the total standard rates and total supplement rates for the family, taking into account the number of FTB dependants and applicable rates by age group.

* For FTB-A Method 1, the standard rates and Energy Supplement rates vary according to the children’s age group.
* For FTB-A Method 2, the standard rate and Energy Supplement rate are the same for all age groups.
* The end of year supplement for both methods is paid at the same rate for all age groups.
* The large family supplement is paid to families where the number of FTB-A children are equal or greater than the required number for receiving this supplement. The large family supplement has been abolished since 1 July 2016.
* If the family is receiving maintenance income, the maintenance income parameters are assigned depending on the number of recipients of maintenance income in the family. The maintenance income threshold for the family is calculated as the sum of the maintenance income free area for the parent(s) receiving child support and the maintenance income free area for each additional FTB-child.
* For FTB-B, the standard rate and Energy Supplement rate are assigned based on the age of the youngest child in the family.

***FtbaCalc***

This macro calculates FTB-A entitlements under Method 1 and Method 2, and assigns the higher payment to the family.

###### Newborn supplement

The newborn supplement (NBS) is payable to families with newborns. A higher rate is paid for the first newborn child and for each newborn child in multiple births. A lower rate is paid to subsequent single birth children.

NBS is paid for a period of up to 13 weeks. A factor for annualising the NBS amount is defined for the means test calculation. The factor is set to 1 if the number of weeks is nil (that is, NBS is not paid in this period). The NBS annualised amount is derived by dividing the NBS rate by the NBS factor.

###### Method 1 calculation

The maximum FTB-A under Method 1 is calculated by adding all the components of the maximum rate. These include the standard maximum rate, Energy Supplement maximum rate, end of year supplement, large family supplement, rent assistance, and NBS annualised amount. The fortnightly rate of rent assistance is converted to annual amount for consistency with the other components.

If the family is exempt from the income test, the reduction to the maximum rate is nil. Otherwise, the income test reduction is calculated by applying the relevant reduction (taper) rate to the income amount above the income threshold. However, if the result is less than zero, the reduction is nil. After 1 July 2018 two taper rates apply, 20% for income between the income free area and the higher income free area, and 30% for income above the higher income free area.

The maintenance income test reduction is estimated by applying the taper rate to the maintenance income amount above the maintenance income threshold. If the result is negative, the reduction is nil.

The total reduction to the FTB-A maximum rate is calculated by adding the income test and maintenance income test reductions.

The net amount of FTB-A is derived by subtracting the total reduction from the maximum FTB-A amount. However, if the result is less than zero, the net FTB-A amount is nil.

###### Method 2 calculation

The maximum FTB-A under Method 2 is calculated by adding all the components of the base rate. These include the standard base rate, Energy Supplement base rate, end of year supplement, large family supplement, and NBS annualised amount. Rent assistance is not included in base rate calculation.

The base rate income threshold includes a basic threshold amount and an additional amount for each child after the first FTB-A child. The income test reduction is calculated by applying the taper rate to the income amount over the income threshold (i.e. if the result is greater than zero, otherwise the reduction is nil).

The net amount of FTB-A base rate is calculated by subtracting the income test reduction from the total FTB amount. If the result is negative, the net FTB-A base rate amount is nil.

###### FTB-A entitlement and components

If the FTB-A amount calculated under Method 1 is higher than that under Method 2, the family is paid the Method 1 amount. If there has been no reduction to the maximum rate, the Ftba type is flagged as ‘Maximum Rate’. If there has been a reduction but the amount payable is not nil, the FtbaType is flagged as ‘Part Rate’. Otherwise, if the amount payable is nil, the FtbaType is set to blank.

* For the usual case where a family with income below the higher income (base rate) threshold is paid the maximum rate, the reduction is apportioned to the components based on the amount the maximum rate components are above the base rate components. This reduction applies only to components that have higher maximum rate amounts than their base rate amounts, namely standard rates, Energy Supplement, and rent assistance. The proportion of each component is calculated as the ratio of the difference between its maximum rate amount and its base rate amount to the difference between the total FTB-A maximum rate amount and the total FTB-A base rate amount. This proportion is multiplied by the total FTB-A maximum rate reduction to determine the reduction for the component. The maximum rate amount of the component is then reduced by the amount determined for the component. However, if the result is less than zero, the post-reduction amount of the component is nil.
* For the special case where a family with income above the higher income threshold is paid the maximum rate, the reduction is apportioned to the components based on the proportion of each component to the FTB-A total maximum rate.

If the FTB-A amount under Method 2 calculation is higher, the base rate is paid to the family. If there has been no reduction to the maximum base rate, the FtbaType is flagged as ‘Base Rate’. If there has been a reduction but the amount payable is not nil, the FtbaType is flagged as ‘Taper Base Rate’. Otherwise, if the amount payable is nil, the FtbaType is set to blank.

* The total base rate reduction is apportioned to the components based on the proportion of each component to the total FTB-A base rate amount.

***FtbaFinalPay***

This macro calculates the final FTB-A amount payable, including the amounts by components. The annual final amounts are estimated as the sum of the amounts for the period when the NBS is payable and the FTB amounts for the period when the NBS is not payable.

In case the FtbaType is not the same for the two periods, the final FtbaType is assigned the FtbaType for the period when NBS is payable. Difference in FTB-A type between the two periods is indicated by setting the FtbaMix flag to 1.

***FtbbCalc***

This macro calculates the FTB-B payment and its components.

The maximum FTB-B payment consists of the standard rate, Energy Supplement, and end of year supplement.

If the family is exempt from the income test, there is no reduction to the maximum FTB-B payment. If not exempt and the income of the primary earner exceeds the FTB-B primary earner income limit, no FTB-B is paid to the family.

If the primary earner’s income does not exceed the primary earner income limit, the reduction to FTB-B is nil for single parent families. For two-parent families that pass the primary earner income limit test, the reduction to FTB-B is calculated by applying the rate of reduction to the secondary earner income amount above the secondary earner income threshold. The final FTB-B payment is then reduced by this amount. If this results in a negative number, the FTB-B payment is nil.

The FTB-B components are reduced in the following order until the reduction has been fully applied: the standard rate is reduced first, then the Energy Supplement, and lastly the end of year supplement.

***BabyUpfrontPayment***

This macro calculates the baby bonus or newborn upfront payment to families with newborns.

From 1 March 2014, the newborn upfront payment and new born supplement replaced the baby bonus. The newborn upfront payment is paid to families with newborns if they are entitled to the newborn supplement which is incorporated in the FTB-A final amount. The newborn payment is included in the FTB-A paid to the family.

## The Supplement Module

### Payments covered

* Carer Allowance
* Carer Supplement
* Income Support Bonus (removed on 31 December 2016)
* Commonwealth Seniors Health Card
* Seniors Supplement
* Pensioner Education Supplement
* Telephone Allowance
* Utilities Allowance
* Single Income Family Supplement (closed to new customers from 1 July 2017)

### Overview

The Supplement Module calculates various supplements in the transfer system.

### 1. Carer Allowance

Determine eligibility and payment rate for Carer Allowance.

### 2. Carer Supplement

Determine eligibility and payment rate for Carer Supplement.

### 3. Income Support Bonus

Determine eligibility and payment rate for Income Support Bonus.

### 4. Commonwealth Seniors Health Card

Determine eligibility for the Commonwealth Seniors Health Card.

### 5. Seniors Supplement

Determine eligibility and payment rate for Seniors Supplement.

### 6. Pensioner Education Supplement

Determine eligibility and payment rate for Pensioner Education Supplement.

### 7. Determine eligibility for Telephone Allowance

Determine eligibility for Telephone Allowance.

### 8. Determine rate for Telephone Allowance

Determine rate for Telephone Allowance.

### 9. Utilities Allowance

Determine eligibility and rate for Utilities Allowance.

### 10. Single Income Family Supplement

Determine eligibility and rate for Single Income Family Supplement (SIFS).

### 11. Supplement Aggregates

Sum up total value of all supplements.

## The Tax Module

### Components covered

* Gross income tax
* Senior Australians and pensioner tax offset (SAPTO)
* Beneficiary tax offset (BENTO)
* Low income tax offset (LITO)
* Mature age workers tax offset (MAWTO) – abolished as of 1 July 2014
* Dependant spouse tax offset (DSTO) – abolished as of 1 July 2014
* Dependant (invalid and carer) tax offset (DICTO)
* Medicare levy
* Medicare levy surcharge
* Temporary Budget Repair Levy (applies in 2014-15, 2015-16 and 2016-17)
* Final tax liabilities

### Overview

The Tax Module applies the personal tax rates to a person’s taxable income, reduces the resulting gross income tax by tax offsets, and adds levies and charges to calculate the final tax liabilities. The Tax Module takes into account the taxable components of the transfer payments (modelled upstream from the module) that form part of the amount of taxable income, and eligibility for certain transfer payments that affect entitlement to certain tax offsets.

The user should refer to the ATO website for further detail on the operation of the policy.

### 1. Calculate gross income tax

Gross income tax is calculated by applying the personal income tax rates to a taxpayer’s taxable income. First the **TaxArray** macro creates an array of the key tax rates and thresholds parameters. Then the **GrossTax** macro is called to calculate the gross income tax.

### 2. Calculate tax offsets

Each tax offset modelled in CAPITA is generally calculated in a separate macro. The tax offsets are calculated in the following order: SAPTO, BENTO, LITO, MAWTO (applies before 2014-15), and DSTO (applies before 2014-15). This follows the order tax offsets and rebates are applied against gross income tax.

**SaptoElig**, **SaptoPsn** and **SaptoCoupTran** calculate the amount of SAPTO. **SaptoElig** determines the taxpayer’s eligibility for SAPTO based on their eligibility for certain pension payments. **SaptoPsn** calculates the amount of SAPTO before transfer of unused SAPTO. **SaptoCoupTran** recalculates the taxpayer’s SAPTO for those who are eligible for transfer of unused SAPTO from their spouse.

The **BENTO** macro calculates the amount of BENTO and ensures that after the amount of SAPTO and BENTO is calculated, only one tax offset is given to the taxpayer.

The **DSTO** macro contains two tax offsets, the dependent spouse tax offset (DSTO) (only applies before 2014-15), and the dependent (invalid and carer) tax offset (DICTO). These two tax offsets are separately identified.

### 3. Calculate Medicare levy

The Medicare levy SAPTO Family Income Threshold parameter is set by the CPS. Unlike the other three Medicare levy thresholds that are generally indexed, its value is calculated. The general principle of the threshold is to ensure a family receiving SAPTO does not pay the Medicare levy if the family is not paying tax, and so it is essentially the effective tax free threshold for the family. It is calculated by the program by solving a series of simultaneous equations. The Medicare levy is calculated by the **MedLevPsn** and **MedLevFamRed** macros. The **MedLevPsn** macro calculates the Medicare levy liability before applying the Family Reduction. The **MedLevFamRed** and the remaining code reduce the Medicare levy liability calculated by **MedLevPsn**, for eligible Medicare levy families, by the Family Reduction amount.

### 4. Calculate Medicare levy surcharge

The **MedLevSur** macro calculates the amount of Medicare levy surcharge liability by taking into account of the effect of dependants on the reference’s and spouse’s surcharge liability.

### 5. Calculate Temporary Budget Repair Levy

The **TempBudgRepLev** macro calculates the amount of Temporary Budget Repair Levy for the 2014‑15, 2015‑16, and 2016‑17 financial years.

### 6. Calculate final tax liabilities

The **FinalTaxLiab** macro calculates key variables including total tax offset, net income tax, amount of refundable tax offset, levies and charges, amount of tax payable or refundable, and the amount of tax offset used.

### 7. Calculate fortnightly amounts

The **Fortnightly** macro converts certain key variables from annual amounts to fortnightly rates.

### 8. Calculate personal income tax liabilities for dependants

Applicable Macros in steps 1 to 7 are called for dependants. For example SAPTO, DSTO, DICTO, and Medicare levy family Reduction Amount do not apply to dependants.

### The macros

#### RunTax

This macro coordinates and calls all other macros. It does this in the order of the eight steps outlined above.

#### TaxArray

This macro declares arrays containing the key income tax parameters: the tax rates, the tax thresholds and the cumulative (lump sum) tax amount at each tax threshold. The tax rates and tax thresholds are used to calculate the cumulative tax amount, whereby the cumulative tax amount at a tax threshold is the sum of the cumulative tax amount at the previous tax threshold, and the tax on the income between these two tax thresholds. The TaxThr and TaxRate arrays’ dimension is set dynamically, depending on how many rate and threshold parameters there are. These two arrays collect parameters that have already been created, so it currently contains four elements because there are currently four steps in the tax scale. In contrast, the CumTax array declares and creates the variables CumTax1 through to CumTax10 that do not already exist on the dataset, with CumTax1 to CumTax4 calculated from the current tax system, and CumTax5 to CumTax10 initialised to 0, for use if the tax system moves to a five or more step system.

This macro is called once for the first income unit and the output variables are retained for all records. Below is a table that illustrates what is contained in each of the three tax arrays.

TaxThr Array:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Array Element** | **1** | **2** | **3** | **4** |
| **Parameter Name** | TaxThr1 | TaxThr2 | TaxThr3 | TaxThr4 |
| **Parameter Value** | Tax free threshold | Second marginal tax threshold | Third marginal tax threshold | Fourth marginal tax threshold |

TaxRate Array:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Array Element** | **1** | **2** | **3** | **4** |
| **Parameter Name** | TaxRate1 | TaxRate2 | TaxRate3 | TaxRate4 |
| **Parameter Value** | First marginal tax rate | Second marginal tax rate | Third marginal tax rate | Fourth marginal tax rate |

CumTax Array:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Array Element** | **1** | **2** | **3** | **4** |
| **Parameter Name** | CumTax1 | CumTax2 | CumTax3 | CumTax4 |
| **Parameter Value** | Cumulative tax up to tax free threshold (zero) | Cumulative tax up to second tax threshold | Cumulative tax up to third tax threshold | Cumulative tax up to fourth tax threshold |

#### GrossTax

Gross income tax can be expressed as the sum of a lump sum and the marginal income tax. The lump sum amount is the cumulative tax paid up to the previous tax threshold, which is contained in the CumTax array. The marginal income tax is the tax payable on the excess income above the previous tax threshold, at the applicable marginal tax rate.

#### SaptoElig

Conditional eligibility for SAPTO is determined by %SaptoElig. An eligibility flag (SaptoType&psn) is created to indicate whether a person is conditionally eligible for SAPTO based on their eligibility or receipt of certain pension payments. **SaptoPsn** and **SaptoCoupTran** updates the eligibility flag to indicate the specific types of eligible recipient.

#### SaptoPsn

The full eligibility criteria also require that an individual’s Rebate Income does not exceed their Rebate Threshold. This condition is included in this macro because it is where the Rebate Threshold is first calculated. If a person’s Rebate Income exceeds their Rebate Threshold, they are ineligible for SAPTO. The eligibility status is reflected in the eligibility flag SaptoType&psn.

After determining the eligibility, **SaptoPsn** calculates each eligible person’s SAPTO entitlement, according to whether they are single or in a couple. It does not yet take into account the transfer of unused SAPTO. For a couple, while the couple combined Rebate Income is used to determine their eligibility for SAPTO, each member’s own Rebate Income is used to calculate their SAPTO entitlement.

In most cases the calculation stops here, where there is no transfer of unused SAPTO between members of a couple.

#### SaptoCoupTran

A person’s eligibility flag may be modified again by the **SaptoCoupTran** macro depending on whether they have transfer of unused SAPTO. After **SaptoPsn** has calculated the individual SAPTO entitlement, **SaptoCoupTran** is called for eligible couples to determine the amount of unused SAPTO that is transferred from one person to another.

To be eligible to transfer unused SAPTO, each member of a couple must be eligible for the SAPTO as a couple determined by **SaptoPsn**; the person giving their unused SAPTO must have more SAPTO than their gross income tax; and the person receiving the unused SAPTO from their partner must have a SAPTO entitlement that is less than the couple SAPTO Rebate Amount (the maximum SAPTO amount). The final condition is equivalent to the person’s Rebate Income being more than their effective tax free threshold (calculated using LITO and SAPTO).

The flag variable is updated to indicate whether the person is receiving or giving unused SAPTO. The amount of unused SAPTO is then calculated using the tax scales. The person giving unused SAPTO has their unused SAPTO entitlement reduced to the amount of their gross income tax, while the person receiving unused SAPTO has their SAPTO Rebate Amount increased by the amount of unused SAPTO. For example it is possible for a spouse on full SAPTO to transfer the full amount to the reference person, so the reference’s SAPTO Rebate Amount after the transfer may be twice the couple Rebate Amount. The increased Rebate Amount also increases the Rebate Threshold and the cut out Threshold. The amount of SAPTO entitlement is recalculated for the person receiving unused SAPTO using the increased Rebate Amount and Rebate Threshold. The amount of SAPTO entitlement is not recalculated for the person giving unused SAPTO because their new Rebate Threshold would always be more than their Rebate Income.

#### The SaptoRebThresh Module

**SaptoPsn** and **SaptoCoupTran** each use the **SolveSaptoRebThr** function that is defined in the SaptoRebThresh module, to dynamically calculate the Rebate Threshold. The **SolveSaptoRebThr** is a user defined function using the equation solver in PROC FCMP to calculate the effective tax free threshold, by finding TaxIncAr such that NetIncTax is zero. The Rebate Threshold equals the effective tax free threshold taking into account SAPTO and LITO, and is calculated dynamically because a person’s amount of SAPTO may change due to transfer of unused SAPTO.

#### Bento

**Bento** calculates the amount of BENTO entitlement. To be eligible for BENTO a person needs to have an amount of Rebatable Benefit, which is the amount of certain eligible Allowance payments. It is created in the Income Module and is derived from the Allowances module. BENTO is effectively the gross income tax payable on the amount of Allowance payments.

Legislation allows an eligible person to receive SAPTO or BENTO but not both. The tax offset that gives the taxpayer the greatest amount of rebate is assigned to the taxpayer. Where both amounts are the same, SAPTO is assigned to the taxpayer because it would be more beneficial to the taxpayer for Medicare levy purposes.

#### Lito

**Lito** calculates the amount of LITO entitlement. A taxpayer is eligible for LITO if their taxable income is less than the LITO cut out threshold. LITO is phased out gradually from the LITO taper threshold.

#### Mawto

**Mawto** is called for the reference and the spouse. MAWTO is phased in gradually at low income levels, and phased out gradually at higher income levels. This was abolished as of 1 July 2014.

#### Dsto

**Dsto** calculates both the DSTO and DICTO, assigning one of these tax offsets to the primary income earner (the reference receives the tax offset in case of equal income levels).

DSTO is calculated first. Where the recipient is the reference, DSTO is calculated using the spouse’s income. This was abolished as of 1 July 2014.

If a person is ineligible for DSTO then DICTO is calculated for the person with respect to their invalid spouse. DICTO is not currently modelled with respect to other invalid dependants or carer dependants due to lack of information on the SIH.

#### SuperTo

**SuperTo** calculates the amount of tax offset assessable superannuation benefit attracts. Which superannuation benefits attracts a tax offset depends on the taxpayer’s age and whether the benefit is an element taxed or an element untaxed in the superannuation fund.

#### MedLevPsn

**MedLevPsn** calculates the individual’s Medicare levy liability depending on whether the person is receiving SAPTO. It does not take into account the family Reduction Amount. A person is only liable for the Medicare levy if their income is above the Threshold Amount, and a flag is assigned to the person to indicate their liability.

#### MedLevFamRed

**MedLevFamRed** is called for each member of a couple or for a single person with Medicare levy dependent children. This macro calculates the amount of family Reduction Amount that applies to members of a family. The relevant Family Income Threshold is assigned, which may be increased by dependent children.

The couple combined income is used in this macro. If the couple combined income is less than the Family Income Threshold then neither the reference nor the spouse is liable for the Medicare levy.

If the couple combined income is more than the Family Income Threshold there may be a Reduction Amount that is calculated according to the formula in legislation. The Reduction Amount may then be apportioned between members of a couple prorated by each person’s share of the couple combined income, and any unused apportioned Reduction Amount may then be transferred between the couple.

#### MedLevSur

**MedLevSur** calculates the amount of Medicare levy surcharge liability. The surcharge thresholds depend on whether a person is in a couple or is single, and how many dependent children they have. Income for surcharge purposes is used to assign the relevant surcharge thresholds, and for a couple the family combined income for surcharge purposes is used. A person in a couple is not liable to pay the surcharge if they are also not liable to pay the Medicare levy.

For the reference and spouse, adequate private health insurance cover requires that each dependent member of the family must have adequate cover. For dependants over 15 years only their own private insurance cover is relevant. The variable for individual private health insurance cover is provided by the SIH, and the variable for family level cover is created in the basefile modules.

After the correct surcharge rates are assigned, the surcharge is applied to the taxable income and reportable fringe benefits.

#### TempBudgRepLevy

**TempBudgRepLevy** calculates the Temporary Budget Repair Levy, which applies for the 2014-15, 2015-16 and 2016-17 financial years.

#### FinalTaxLiab

**FinalTaxLiab** calculates the final aggregates including the final amount of tax payable or refundable, and the aggregates for the amount total tax offset, levies and charges. It also determines the amount of tax offsets actually used.

DSTO, DICTO and MAWTO are combined into one tax offset for the purpose of calculating amount of tax offsets used. This is because no relative priorities are assigned to these three tax offsets to offset against gross income tax.

#### Fortnightly

**Fortnightly** converts key variables generated by the Tax Module from an annual rate to a fortnightly rate. Both the annual and fortnightly rates are available.

## The Childcare Module

### Overview

The Childcare module determines the amount of child care assistance for which a family is eligible. The module is divided into two parts:

* The first part covers the Child Care Rebate and Child Care Benefit, which applies before 1 July 2018;
* The second part covers the Child Care Subsidy, which will replace the current scheme from 1 July 2018.

The module is defined by a master macro called ***RunChildcare***. The year for which *RunCAPITA* is being run determines which collection of macros are called within the master macro, depending on which childcare system applies.

This module is only included in the cameo model of CAPITA, not the distributional model. This is incorporated by the use of a %IF condition in the *RunCAPITA* code, where the *Childcare* module is only called if *RunCAPITA* is being run via the *Cameo Code*.

### I. Current system (operating up to 1 July 2018)

#### Payments covered

* Child Care Benefit (approved care).
* Child Care Benefit (registered care).
* Child Care Rebate.

CAPITA does not model the following: Special Child Care Benefit; Grandparent Child Care Benefit; and Jobs, Education and Training Child Care fee assistance.

#### Overview

Part One of the Childcare module determines the rate of Child Care Benefit (CCB) and Child Care Rebate (CCR) for which a family is eligible. The calculations are based on the prescribed methodology for each type of payment as contained in the legislation*.*

#### 1. Define and determine compliance with the Work/Training/Study test

A family meets the work/training/study test if both the reference person and the spouse (if applicable) have work related commitments and, depending on the type of child care payment, meet the following minimum-hours requirement:

* CCB (approved care): at least 15 hours per week or 30 hours per fortnight (unless both parents have an exemption from that requirement).
* CCB (registered care) and CCR: No minimum number of hours is required.

CAPITA assumes that an individual has work related commitments if he or she is in the labour force (including full time work, part time work, or unemployed) or is studying (including secondary studies, full time non-secondary studies, or part time non-secondary studies).

Macro reference: **CcbElig**

#### 2. Calculation of child care payment - CCB (approved care)

Approved child care services have Australian Government approval to pass on CCB for approved care to families as a reduction in their child care fees. Approved child care can include:

* long day care (LDC);
* outside school hours care (OSHC);
* family day care (FDC);
* in-home care (INC); and
* occasional care (OCC).

For each child, the rate of CCB (approved care) per week =

Eligible hours per week x Standard rate per hour x CCB % x Schooling % x LDC Part-time %

***a. Calculate the eligible number CCB hours***

A limit applies to the number of hours per week for which an individual is eligible for CCB:

* if the work/training/study test is satisfied: up to 50 hours per week; and
* if the work/training/study test is not satisfied: up to 24 hours per week.

There are other circumstances under which an individual would be eligible for up to or more than 50 hours of CCB a week but these are not modelled in CAPITA.

Macro references: **CcbElig**; **CcbHrElig**

***b. Calculate the CCB standard hourly rate***

* The hourly rate for CCB is dependent on the kind of approved care

These hourly rates are indexed every 1 July to movements in the CPI, based on the CPI indices for the December quarters.

Macro reference: **CcbStandHrRate**

***c. Calculate the CCB %***

CCB % = Multiple child % x Taxable income %

An individual’s multiple child % and taxable income % are affected by the number of children the individual has in care of a particular kind. The kinds of care are: care other than occasional (‘other care’) and ‘occasional care’. In CAPITA, the tallying of the number of children in each kind of care is performed in **CcbHrElig**.

|  |  |
| --- | --- |
| * Multiple child % = | Individual’s maximum weekly benefit (‘multiple child rate’) |
| Single child rate of maximum weekly benefit x no. of children in same kind of care |

An individual’s maximum weekly benefit is calculated with respect to the number of children in the same kind of care and whether their combined adjusted taxable income exceeds the upper income threshold.

Individual’s maximum weekly benefit = (Standard hourly rate x 50) x no. of children in same kind of care

+ Additional loading

– Specific taper amount, if ATI > upper threshold and ≥ 2 children

Where: ·

· Specific taper amount = (Upper Income Threshold – Lower Income Threshold) ÷ 52 \* 15%

The additional loading amounts and income thresholds are indexed every 1 July to movements in the CPI, based on the CPI indices for the December quarters.

* Taxable income % = 100%, if ATI < lower threshold or either parent is receiving income support, or

|  |  |  |
| --- | --- | --- |
| = 100% – | (ATI – annual threshold) ÷ 52 x Taper % | , otherwise |
| Individual’s maximum weekly benefit |

Where: ·

· Annual threshold = Upper income threshold, if ATI > upper threshold and ≥ 2 children, or

= Lower income threshold, otherwise .

· Taper % = 10% [1 child];

15% [≥ 2 children and ATI ≤ upper threshold];

25% [2 children and ATI > upper threshold]; or

35% [≥3 children and ATI > upper threshold] .

The income thresholds are indexed every 1 July to movements in the CPI, based on the CPI indices for the December quarters .

Macro reference: **CcbPct(CcbType)**

***d. Calculate the Schooling %***

Schooling % = 85%, if the child is a school child, or

= 100%, if the child is not a school child .

**CAPITA assumes that a child aged 5 or over is a school child**.

Macro reference: **CcbSchooling**

***e. Calculate the LDC Part-time %***

A part-time loading is applied to the rate of CCB (approved care) if the child is in long day care and is not a school child. The LDC Part-time % varies depending on the number of hours the child is in long day care per week:

LDC Part-time % = 110%, if LDC hours per week < 34,

= 108%, if 34 ≤ LDC hours per week < 35,

= 106%, if 35 ≤ LDC hours per week < 36,

= 104%, if 36 ≤ LDC hours per week < 37,

= 102%, if 37 ≤ LDC hours per week < 38,

= 100%, otherwise.

Macro reference: **CcbLdcPct**

***f. Calculate the rate of CCB (approved care)***

The rate of CCB (approved care) per week is not to exceed the actual cost incurred by the individual for child care sessions in the week (Part 4, Div 4, Sec 71). The weekly benefit payable for a child is calculated as the minimum of the amount determined by the legislation and the actual cost incurred.

In CAPITA, the cameo model allows the user to specify the number of weeks per year that a child is in approved care. This input is used to determine the annual rate of CCB (approved care) and post-CCB out of pocket costs for a child. The CCB benefits (approved and registered care) and costs post-CCB are summed for all eligible children to determine the total amounts for a family.

Macro reference: **CcbBenefit**

#### 3. Calculation of child care payment - CCB (registered care)

Registered child care is care provided by individuals who are registered as carers. It can include care provided by grandparents, relatives, friends, neighbours or nannies.

The hourly rate of CCB (registered care) per week per child = Minimum hourly amount x Schooling %

The minimum hourly amount is indexed every 1 July to movements in the CPI, based on the CPI indices for the December quarters .

CCB (registered care) is paid for up to 50 hours per child per week if the family meets the work/training/study test or has an exemption from that requirement. The rate of CCB (registered care) per week is not to exceed the actual cost incurred by the individual for child care sessions in the week.

In CAPITA, the cameo model allows the user to specify the number of weeks per year that a child is in registered care for. This input is used to determine the annual rate of CCB (registered care) and　post-CCB out of pocket costs for a child. The CCB benefits (approved and registered care) and costs post-CCB are summed for all eligible children to determine the total amounts for a family.

Macro references: ***CcbElig;* *CcbHrElig;* *CcbStandHrRate; CcbSchooling;*  *CcbBenefit***

#### 4. Calculation of child care payment - CCR

CCR covers 50% of out of pocket child care expenses for approved child care, up to a maximum amount per child per year. A family is eligible for CCR if it meets the work/training/study test. CCR is not income tested.

Macro references: **CcbElig; CcrRebate**

### II. Proposed system (to apply from 1 July 2018 onwards)

#### Payments covered:

* Child care subsidy (CCS)

CAPITA does not model the following: Interim Home Based Carer Subsidy Programme; and Child Care Safety Net.

#### Overview

Part Two of the Childcare module determines the rate of Child Care Subsidy (CCS) for which a family is eligible.

The approved types of child care for CCS differ from CCB and CCR. Approved types of care are limited to:

* long day care (LDC);
* outside school hours care (OSHC); and
* family day care (FDC).

#### 1. Define the activity test and determine the maximum eligible hours of assistance

An activity test is used to calculate the maximum number of hours of assistance per child (which is then stored as the variable CcsMaxHrW). A family meets the activity test if both the reference person and the spouse (if applicable) have work related commitments. The number of hours of assistance is then determined by the number of hours spent in work, as follows:

* if working less than four hours per week, eligible for zero hours of assistance per week;
* if working between four and 8 hours per week, eligible for up to 18 hours of assistance per week;
* if working between 8 and 24 hours per week, eligible for up to 36 hours of assistance per week;
* if working more than 24 hours per week, eligible for up to 50 hours of assistance per week.

Both parents need to be in the same category to qualify for that category’s eligible hours. If one parent is in a lower category than their partner, then the family will only be eligible for the lower category eligible hours.

CAPITA assumes that an individual has work related commitments if he/she is in the labour force (including full time work, part time work, or unemployed) or is studying (including secondary studies, full time non-secondary studies, or part time non-secondary studies).

The family’s adjustable taxable income is calculated by aggregating both parents’ income.

There is a low income exemption to the activity test. If the parents’ combined adjustable taxable income is less than the lower income threshold used for the income test and the family does not meet the activity test, then the family will be eligible for up to 12 hours of assistance per week. Eligibility for this low income exemption is calculated in **CcsActivTest** with **CcsMaxHrElig**specifying the relevant macro variable input based on income.

Macro reference: **CcsMaxHrElig**and**CcsActivTest*.***

2. Calculate the number of eligible hours per child

The eligible weekly hours of childcare that can be claimed per child for assistance is the minimum of:

* The maximum eligible hours as dictated by the activity test (as determined in Step 1 above);
* The actual number of hours that the child has spent in childcare.

This is then stored as the variable *CcsEligHrW&j* (i.e. for each child). The approved child care types are used in the macro to check eligibility across each child.

Macro reference: **CcsHrElig**

#### 3. Set the hourly fee caps

There is an hourly fee cap set for the three types of care. Based on the type of childcare selected in the cameo input spreadsheet, the fee cap for each child is set to the maximum hourly rate. These hourly rates are indexed every 1 July to movements in CPI, based on the CPI indices for the December quarters. The hourly fee cap is stored as the variable *CcsHrFeeCap&j* (i.e. for each child).

Macro reference: **CcsHrFeeCap**

#### 4. Calculation of the subsidy assistance rate

The rate of assistance is based on the family’s adjustable taxable income:

* If the family’s income is less than or equal to the lower income threshold then the family is eligible for the maximum subsidy rate of 85 per cent;
* If the family’s income is above or equal to the lower income threshold, but below the second income threshold, then the family is eligible for a tapered subsidy rate between 80 per cent and 50 per cent;
* If the family’s income is above or equal to the second income threshold ($172,024 in 2018-19), but below the third income threshold, then they are entitled to a subsidy rate of 50 per cent;
* If the family’s income is above the third income threshold ($251,314 in 2018-19), but below the fourth income threshold, then the family is eligible for a tapered subsidy rate between 50 per cent and 20 per cent; and
* If the family’s income is above the fourth income threshold ($341,314 in 2018-19), but below the upper income threshold, then they are entitled to a subsidy rate of 20 per cent.
* If the family’s income is above the upper income threshold then they are not entitled to any subsidy.

However, once their income exceeds the high income threshold ($187,156 in 2018-19), they can only claim a maximum assistance amount per child ($10,220 in 2018-19).

In terms of the tapered subsidy rates, for every $1 above the relevant income threshold, the subsidy rate decreases by a factor of 1/3,000. For example, a family with income $10,000 above the first income threshold will receive a subsidy rate of 81.67% (3.33 percentage points lower than the maximum rate of 85%). The calculation is as below, please note that the subsidy rate is in decimal and not percentage terms (e.g. maximum subsidy rate is 0.85).

* For family income above the lower income threshold but below the second income threshold:

Subsidy rate = Maximum subsidy rate – ((family ATI – lower income threshold)/ (3,000\*100))

* For family income above the third income threshold but below the fourth income threshold:

Subsidy rate = Mid subsidy rate – ((family ATI – lower income threshold)/ (3,000\*100))

The subsidy rate is rounded to the nearest four decimal places (or 0.01%).

Note that this macro does not apply the income test for the high income threshold, as the subsidy amount for incomes above the high income threshold is a fixed dollar cap. The high income threshold is applied in the next step.

The subsidy assistance rate is stored in the variable *CcsRate*.

Macro reference: **CcsRate**

#### 5. Calculation of subsidy amount

This macro calculates the amount of CCS a family can receive through the following steps:

* First, the weekly CCS amount per child is calculated (*CcsAmtW&j*), which is the minimum of either (eligible hours per week x relevant hourly fee cap x subsidy rate) or (eligible hours per week x actual hourly childcare cost x subsidy rate). The childcare amount per hour is rounded to the nearest 2 decimal places;
* Second, the annual CCS amount per child is calculated (*CcsAmtA&j*), which is the weekly CCS amount per child multiplied by the number of weeks per year spent in childcare. However, if the family’s taxable income exceeds the upper income threshold then the maximum assistance amount is applied;
* Thirdly, the weekly and annual actual childcare costs per child are calculated, as well as the annual out of pocket costs per child (i.e. the cost of childcare minus the CCS amount received); and;
* Fourthly, the aggregate (i.e. across all children) annual total CCS amount, actual childcare costs and out of pocket costs for the family are calculated.

Macro reference: **CcsSubsidy**

#### 6. Incorporate child care payment and costs into the calculation of transfer income and disposable income

The CAPITA definition of transfer income in the cameo version of the model is inclusive of any child care payment received by a family. For the purpose of calculating total transfer income for an income unit, CCS amounts are added to the non-taxable transfer income of the reference person.

The CAPITA definition of disposable income is net of any child care costs incurred by a family. For the purpose of calculating total disposable income for an income unit, total child care costs are subtracted from the disposable income of the reference person.

RunCAPITA calls the Childcare policy module before the Finalisation module. The calculation of disposable and transfer income amounts in the Finalisation module is inclusive of fortnightly childcare payments. These fortnightly childcare payments are determined on a pro-rata basis by dividing the annual amounts by 26.

## The Finalisation Module

### Overview

The finalisation module calculates transfer and disposable income totals, for individuals, and income unit totals for a range of income definitions.

### 1. Calculate transfer income

The **TranInc** macro calculates the sum of all non-taxable transfer income and adds this to the sum of all taxable transfer income calculated in the Incomes2 module. The macro then sums these to give total transfer income (*IncTran*).

### 2. Calculate disposable income

The **DispInc** macro calculates disposable income as the sum of private and transfer income less tax payable.

### 3. Calculate summary variables for dependants

If dependants are present in an income unit, then the **TranInc** macro calculates their transfer income and the **DispInc** macro calculates their disposable income.

### 4. Calculate income unit level outcomes

For various income definitions, the **IncomeUnit** macro adds the values for all members of the income unit (r,s,1-4) to give an income unit total.

The macros for this section are straightforward so are not described in detail.

## Appendix – The look-ahead method

CAPITA models allowances using a “look-ahead” method to add family level data onto income unit level records.

**Structure of the CAPITA basefile:**

Number of rows in the basefile that cover people in this family

Characteristics of r and s

Characteristics of student 1

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Famid | NumIU | Ager | Couple | … | Age1 | … | Age2 | … |
| 1 | 1 | ... | … | … |  |  |  |  |
| 2  These three rows need to be considered together | 1 | … | … | … | … | … |  |  |
| 3 | 3 | … | … | … | … | … |  |  |
| 3 | 3 | … | … | … |  |  |  |  |
| 3 | 3 | … | … | … |  |  |  |  |
| 4 | 1 | … | … | … | … | … | … | … |

Note that some families are split across multiple rows in the basefile, with some dependent children qualifying for their own row (for example, if they are aged 15 and over but not studying). The basefile is sorted by “Famid”, so any rows that belong to the same family are together (with the parents’ row coming first).

**How SAS processes the basefile**

A single row from the basefile is read into SAS’s Program Data Vector (PDV). The policy code is then run on the values held in the PDV. Once this is complete, the values in the PDV (including any variables calculated by the policy code) are then written to the output dataset as a single row. At this point, the values in the PDV are set to missing (with some important exceptions), and then a new row from the basefile is read in. This process repeats until every row in the basefile has passed through the PDV. This process is depicted in Diagram 1.

**Diagram 1. “Standard” SAS processing of the basefile (family contained in a single row on the basefile)**

**Outfile data set – rows that have been processed**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Famid | NumIU | Ager | Couple | … | Age1 | … | Age2 | … | Outcomes |
| 1 | 1 | ... | … | … |  |  |  |  | … |

**(3)** Row written from PDV to the outfile once processing is complete.

**(2)** Row is held in the PDV and processed (that is, outcomes calculated according to the policy code).

**SAS’s Program Data Vector (PDV) - row from the basefile currently being processed**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Famid | NumIU | Ager | Couple | … | Age1 | … | Age2 | … | MaxFamYA | Outcomes |
| 2 | 1 | ... | … | … | … | … |  |  |  |  |

**(1)** Single row read from basefile into PDV.

**Basefile data set – rows that have not yet been processed**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Famid | NumIU | Ager | Couple | … | Age1 | … | Age2 | … |
| 3 | 3 |  |  |  |  |  |  |  |
| 3 | 3 | … | … | … |  |  |  |  |
| 3 | 3 | … | … | … |  |  |  |  |
| 4 | 1 | … | … | … | … | … | … | … |

**Simplified CAPITA allowance module program flow**

When families are spread over multiple rows in the basefile, simple row-by-row processing is not sufficient to calculate outcomes: some family-level constructs are needed, based on data from all the different rows within a family. The following steps show how the CAPITA allowance module handles this issue using the ‘look-ahead’ method.

1. A macro is called to determine which (if any) allowance persons r and s are eligible for, using their characteristics on the basefile (Ager, Couple and so on).
2. A separate macro (%YAEligibility) is used to determine whether they are eligible for Youth Allowance. The %YAEligibility macro needs to check whether the person in question is a member of a couple and/or has dependent children (these characteristics are needed to determine what type of Youth Allowance they could be eligible for). There are macro arguments that tell %YAEligibility where to “look” to find out the marital and parental status of the person being processed: for persons r and s the macro is told to check values of the variables “Couple” and “DepChild” to determine these characteristics.
3. Once r and s have their eligibility for allowances determined, their allowance outcomes are computed. The parental income test is only implemented when person r is workforce dependent; this should never be the case for a parental income unit (as by definition having children makes a person workforce independent).
4. If there are any students 1 to 4, the %YAEligibility macro is run to determine if they are eligible for Youth Allowance (there is no need to check if they are eligible for any other allowances, because if they were they should be in their own row, and not sit as a student in their parent’s row in the basefile).
5. When the %YAEligibility macro is called for students 1 to 4, it cannot use the variable “Couple” to determine if the person being processed is a member of a couple – the “Couple” value in the PDV gives the relationship status of the student’s parents. For this reason, when the %YAEligibility macro is called for students 1 to 4, the couple status and number of children are hardcoded to zeros in the macro call (this is fine, as by definition couples or parents should be in their own row in the basefile, and not be a student 1 to 4).
6. The total pool of Youth Allowance that all dependants in the family could receive is needed to implement the Youth Allowance parental income test. For this reason, if any of the students 1 to 4 are judged to be dependent, the maximum value of Youth Allowance they could receive is added to the family’s pool (held in the variable “MaxFamYA”).
7. At this point, the model has computed allowance outcomes for persons r and s, and worked out Youth Allowance eligibility for students 1 to 4. However, the parental income test needs the total pool of Youth Allowance *all* the dependants in the family could qualify for, *including dependants that are only described in the next few rows of the basefile* (the basefile is sorted to keep the rows relating to a single family together, with the parents’ row coming first).
8. For this reason, the first time an observation covering a new family is read into the PDV, the code checks whether the family is spread over multiple rows (that is, whether the variable “NumIU” is greater than 1). If this is the case, the people in these later rows need to be checked to see if they are dependent on their parents and could be eligible for Youth Allowance. To do this the model reads some of the next basefile row into the PDV (only the variables needed to check dependency and Youth Allowance eligibility). Note that the parents’ characteristics are still held in the PDV (indeed the processing of the parents’ row is not yet finished), so the variables are renamed before the additional row is read into the PDV (an underscore is added to the end of their names). This renaming means the information from the additional row is added to the *side* of the PDV, and so the parents’ characteristics are not overwritten by the characteristics from the following row (see Diagram 2).
   1. The %YAEligibility macro is then run on the person whose characteristics have been freshly read into the PDV (these characteristics are stored in the variables ending with an underscore). For these people, the couple status and number of children are hardcoded to zeros in the macro call (this is fine, as by definition couples or parents should have their own distinct family ID rather than sharing one with a parental income unit).
   2. If there are further rows describing people in the same family, these in turn are read into the PDV and their contribution to the family’s total pool of Youth Allowance is calculated. It is straightforward to know how many additional rows need to be considered as the “NumIU” variable encodes the total number of rows from the one family. Note that if multiple additional rows need to be read in to the PDV, the most recently read in additional row will overwrite the characteristics of any previous additional rows – this is not a problem as the total family pool of Youth Allowance will have been updated appropriately before this occurs.
   3. Diagram 2 below gives a schematic representation of this “looking ahead” at future basefile rows.
9. The total dollar amount that parental income reduces the total family pool of Youth Allowance by is also calculated. This will be apportioned between the dependants when their personal outcomes are computed.
10. Finally, now that the relevant family-level data has been collected and the parental income test can be implemented, the outcomes for students 1 to 4 are calculated.
11. The next row of the basefile is read in as normal, and the cycle starts again. Any family-level information (such as the total pool of Youth Allowance) is retained in the PDV, so that final outcomes of people in the same family (but described in different rows in the basefile) can be calculated.
12. Note that the final outcomes of dependants in their own rows are calculated when their own row is being processed (they are person r at this point). While some of their characteristics were already read into the PDV earlier when their parents’ row was being processed, this was only used to help calculate the total pool of Youth Allowance for the family. Now, when the allowance outcomes for person r are being calculated, their workforce dependence will be noted and the parental income test called, using the values of their parents’ income and the family’s total pool of Youth Allowance that were stored when their parents were processed.

**Diagram 2. Processing the basefile when a family is spread over multiple rows in the basefile.**

**Outfile data set – rows that have been processed**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Famid | NumIU | Ager | Couple | … | Age1 | … | Age2 | … | Outcomes |
| 1 | 1 | ... | … | … |  |  |  |  | … |
| 2 | 1 | … | … | … | … | … |  |  | … |

**(2)** Workforce independence and YA eligibility of the person from the additional row is determined, and used to update the total family pool of YA (“MaxFamYA”).

**SAS’s Program Data Vector (PDV) - row from the basefile currently**

**being processed**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Famid | NumIU | Ager | Couple | … | Age1 | … | Age2 | … | Ager\_ | Couple\_ | …\_ | MaxFamYA | Outcomes |
| 3 | 3 | ... | … | … |  |  |  |  |  |  |  |  |  |

**(1)** The characteristics from the next row that are needed to compute YA are read into the PDV (variables are renamed so that parents’ characteristics are not over-written).

**(3)** Steps (1) and (2) repeated for following rows until all rows covering this family are done.

**Basefile data set – rows that have**

**not yet been processed**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Famid | NumIU | Ager | Couple | … | Age1 | … | Age2 | … |
| 3 | 3 | … | … | … |  |  |  |  |
| 3 | 3 | … | … | … |  |  |  |  |
| 4 | 1 | … | … | … | … | … | … | … |

**(4)** Once final outcomes have been calculated for the parents’ row, and that row has been written from the PDV to the output data set, “normal” processing resumes. That is, this observation is read into the PDV as normal (without any variables being renamed) and has its outcomes calculated as normal. The parental income test can be implemented because the family’s total pool of YA (“MaxFamYA”) is still in the PDV (SAS was instructed not to set it to missing when the parents’ row was output).

1. Note that the *Standard Output* code does not need to be run directly. It is called within the *RunCAPITACompare* module. [↑](#footnote-ref-2)
2. Ceases from 20 March 2020. [↑](#footnote-ref-3)
3. The policy change takes effect from 20 March 2020, however in CAPITA we assume it begins at the start of the nearest financial year (1 July 2020), as the model is not capable of capturing part-year effects as mentioned in the Using CAPITA section. [↑](#footnote-ref-4)
4. Pharmaceutical Allowance is only paid to certain types of allowance recipients. [↑](#footnote-ref-5)
5. The partner income test should be applied to people whose partner receives a DVA Disability or War Widow Pension. However, as all recipients of these Pensions in CAPITA will be receiving more than the Energy Supplement rate, they will never cause their partner’s allowance to be reduced. Hence, there is no need to perform the partner income test. [↑](#footnote-ref-6)
6. The Couple without dependants rate is used as this is the only alternative to the couple with dependants rate. [↑](#footnote-ref-7)
7. This is done because a dependant whose parent is receiving one of these payments is exempted from the parental income test. [↑](#footnote-ref-8)
8. This is because it is assigned the first time the YaEligibility macro is run and will not be removed if their eligibility for YA is taken off them the second time it is run. [↑](#footnote-ref-9)