EXAMPLE 4:
DISTRIBUTING HOUSEHOLD-LEVEL INFORMATION TO RESPONDENTS

EXAMPLE RESEARCH QUESTION(S): What characteristics are associated with the probability of being poor?

DESCRIPTION: In this example, household level information is linked to individual level observations. The resulting file is at the individual level, with household level variables.

FILES: w**hhresp**, w**indall**

WAVES: 1

STEPS:
1. Get household level variables (w**hhresp**). Inspect data, derive equivalised household income, save file
2. Get individual level file (w**indall**)
3. Merge household level variables onto individual level file
4. Derive poverty indicator

NEW COMMANDS:
- `return list` and using macros for automated calculations
- generating binary indicators using logical statements

4.1 HOUSEHOLD LEVEL VARIABLES

First we will read in the household level variables from **ahhresp**:

```
use “$dir1/ahhresp”, clear
```

Task: Inspect this dataset. How many observations does it contain? How many variables? Which variable identifies the households?

Next let’s inspect the household income variables:

```
describe afihh*
```

Note the use of * in place of one or more characters in the variable name (which means all variables which begin with **afihh** will be described)

In the following we will be working with **afihhmn**. This variable is derived by summing up all components of income in the month before the interview, for all members of the household. It includes imputed data.

Imputation is a method of dealing with missing data. It is used in the special case where a sample member gave an interview, but did not answer certain questions. If the respondents who refuse to answer questions on income tend to have higher or lower incomes than those who do answer, then ignoring the missing data will lead to biased estimates. Imputation is used to estimate the likely value of the missing variable and thereby reduce the risk of non-response bias. In the BHPS imputation is used for few variables, one of which is household income.

The imputation flag variable **fihhmni** takes a value 0 if there was no imputation, 1 if some component of an individual household member’s income was imputed, and 2 if the whole income of one or more household members was imputed.

Question: In which proportion of households was income imputed for at least one household member?

Question: What does the distribution of household monthly income look like? What are the mean, median, minimum and maximum values?

When comparing household incomes, we will use a conversion factor to adjust for the effects of household size and composition on the needs of the household. We will use the conversion factor provided in the BHPS data, which is the McClements scale (**afieqfcb**). To get an idea of how this conversion factor works, let’s tabulate the scale by household size:

```
table ahhsize, contents(mean afieqfcb min afieqfcb max afieqfcb) ///
format(%9.2f)
```

Note that `table` is a different command from `tabulate` (which you can abbreviate to `tab`). `tabulate` does not allow the option `contents`.

Remember that /// tells Stata that the command continues in the following line. This only works if you run it from a do-file. To run the `table` command interactively, you need to write the entire command in a single line, without ///.

Household incomes cannot be compared across households because households vary by size. Suppose there are two households with household incomes of £1000 each. One consists of just one person, the other a couple. The person in the first household can consume all she can buy with £1000 while each person in the second household can consume some proportion (perhaps half) of the goods that they buy with £1000. So, the effective income that each person in this household has access to is £500. However, some things can be used by one or more persons at no extra cost such as apartment, television, dining table, etc. If we take that into account then each person in this two-person household may enjoy a standard of living that a single person with household income £750 does. In other words, single person household income of £750 is *equivalent* to two-person household income of £1000. There are different methods of calculating these equivalence scales and one such is the **McClements Scale.** In the BHPS we calculate equivalised household income by dividing the monthly household income by the McClements Scale equivalence scale. That is, the monthly income of one-person households is divided by 0.61; the income of an 11-person household is divided by 2.81. This reflects the fact that a one person household with a, say, £2000 monthly income is better off financially than an 11 person household with the same amount of income:

```
gen aefihhmn = afihhmn/afieqfcb
lab var aefihhmn "equivalised HH income"
inspect aefihhmn
```

Two other household level variables we will use are household size (**ahhsize**) and household type (**ahhtype**).

Question: What proportion of households are one-person households? What proportion are couples without children?

Now we will clear up and keep only the variables we will need later, sort the file and save it:

```
keep ahid aefihhmn ahhsize ahhtype
compress
sort ahid
save ahhrespjunk, replace
```

4.2 INDIVIDUAL LEVEL FILE

Now open the individual level file, which contains observations for all household members:

```
use "$dir1/aindall", clear
```

Task: Check the number of observations. Check that the file contains all household members.

*Question: How many children younger than 16 are represented? (Tip: **aivfio**)*

Clear up, keep only those variables we will be using later, and sort in preparation for merging. (**axewght** is the cross-sectional weight for all enumerated household members, which we will use to calculate the poverty line. More on weights in Example 7.)

```
keep pid ahid axewght
sort ahid
```

4.3 MERGE HOUSEHOLD LEVEL VARIABLES ONTO INDIVIDUAL LEVEL FILE

Next we merge the household level variables onto the individual level file which is currently open, using the **ahid** household identifier as the linkage variable. As the individual level file is the one open and we are merging the household level file onto it and multiple individuals will match onto one household (in case of households with more than one member) this will be a many-to-one merge:

```
merge m:1 ahid using ahhrespjunk
```

Check that all observations were successfully merged:

```
tab _merge
drop _merge
```

Inspect the resulting data file:

```
sort ahid
list in 1/50, sepby(ahid)
```

And delete the file we no longer need. Note that the command `erase` requires the file extension **.dta**. This is to safeguard against inadvertently deleting datasets:

```
erase ahhrespjunk.dta
```

The file creation is now completed.

4.4 DERIVATION OF POVERTY INDICATOR

We will define the poverty line as 60% of median household income. To calculate this line, we will weight observations to adjust for non-response. Note that if you add the option `detail` to the `summarize` command, Stata displays additional summary statistics:

```
summ aefihhmn [aw = axewght], detail
```

`[aw = axewght]` specifies the weighting variable and type. More on weighting in Exercise 7.

The output from the above command tells us the median household income is 1114.299. We could calculate the poverty line as .6*1114.299. But this would be error prone. It is better to automate the calculation by retrieving the values automatically stored by Stata. With each estimation command Stata executes, it automatically stores a range of statistics which can later be retrieved and used. For the `summarize` command, Stata stores the following: `r(N)` contains the number of observations, `r(mean)` contains the mean, `p(90)` the 90th percentile, etc. To check which statistics Stata has stored, type:

```
return list
```

You can refer to any of the stored values and use them in your calculations, as we will do here. Remember that the values stored by Stata will be overwritten each time another command is run. Hence, we will calculate the poverty line and store it in a global macro, which we name **apovline**; we will therefore be able to access this value at any time by calling up the macro:

```
global apovline = .60 * r(p50)
display "The poverty line = " $apovline
```

Note how we've used the display command to echo information to the output window. The text in inverted commas **"…"** is displayed as is. We also retrieve the content of **apovline**, by prefixing the name of the global macro we defined with the dollar symbol, **$**. This tells Stata that we are referring to a global macro.

Next we create a binary indicator variable called **apov** and label the variable and its values. Here we create this indicator in one step using a logical expression (**aefihhmn < $apovline**). For each observation, Stata will set **apov** to 1 if the logical expression is true, i.e. if the equivalised household income of that observation is below the poverty line; it will set **apov** to 0 if this statement is false; and to missing for observations where equivalised household income is missing:

```
generate apov = (aefihhmn < $apovline) if aefihhmn < .
label variable apov "Poverty status"
label define pov 1 "Poor" 0 "Not poor"
label values apov pov
```

Now we can estimate the proportion of people living in poor households:

```
tab apov [aw = axewght], missing
```

Finally, let's clear up and save the file, so that we can continue using it in later examples:

```
compress
sort ahid
save ajunk, replace
```