Using Subroutines and Loops to Simplify Submissions Involving Several Datasets

Goal

When doing comparative research, it is natural to want to run the same routine across several different countries or across different waves. This approach can be simplified by the use of macros, subprograms/subroutines, and loops. These are easy ways to repeat program commands without having to retype them each time.

Activity

Compare the median (weighted) labour earnings for the United States and Mexico (2000), and the United Kingdom (1999) by looking separately at gross wage earnings (***pgwage***), net wage earnings (***pnwage***) and self-employment earnings (***pself***). Your sample should only include those with positive labour earnings (from either paid or self-employment) who are 16 years of age or older. For this comparison, you should write a subroutine to run for each dataset.

Now repeat the same analysis for the five new Latin American datasets (BR06, CO04, GT06, PE04 and UY04). In order to shorten the code, loop the subroutine created above over the 5 datasets.

Use the information from your output to answer the following questions:

1. Which country had the highest median wages from employment?

   ____

   ____

   ____

   ____

2. Are the income variables analysed above expressed in net or gross terms (i.e. before or after deduction of income taxes and mandatory social contributions)?

   ____

   ____

   ____

   ____

Guidelines

- For this comparison, proceed according to the following steps:
  - use a subroutine to prepare the data and calculate the estimates;
  - select the age-group;
  - call the subroutine for each of the individual country data sets.
- Writing a subroutine:

```
define dofiles ().          (dofiles is just a name, but you may choose another)
<add all the commands needed, each with a dot at the end >.
!enddefine .
```

- Please be careful when closing a subroutine: if you fail to close it properly your job will block the batch machine on our system.

  REMINDER: **a) do not forget the exclamation mark !**
  b) no space between the exclamation mark and the command enddefine

  c) no misspelling of the command like endefine , eddeffine, etc.

  If you accidently misspell enddefine, the command is not recognized.

- The closing brackets at the end of the define command are needed, and can optionally contain further parameters. For our exercises no use of parameters is being made, therefore no space between the brackets. In case you are interested in using the full functionality of macro's we refer to the SPSS manuals.
- Be careful with selecting records: selecting on age-group is harmless, but pay attention when using variables that might be empty throughout the entire dataset; like gross wage or net wage in this example. When you use the select if command, SPSS will delete all records that do not fulfil the condition. Selecting on a variable that is empty will create an empty file, and will therefore result in an SPSS ERROR ! To circumvent this, do not select, but in stead create new labour earnings variables that copy the value from pnwage, pgwage or pself only if positive.
- Calling a subroutine by its name :

```
dofiles.
```

- In the end, your code should look something like:

```
define dofiles () .
<apply the weights>
<select on age-group>
<create the new labour earnings variables>
```

```
<find the median of pgwage for adults with positive earnings>
<find the median of pnwage for adults with positive earnings>
 <find the median of pself for adults with positive self-employment
    earnings>
!enddefine .


get file = uk99p
     /keep = <list of variables to use>.
dofiles .


get file = us00p
     /keep = <list of variables to use>.
dofiles .
```

- Do not include too many countries in one and the same job, since the output will grow accordingly. Long listings do end up in our review queue, and have to be examined manually before being released. Please try to avoid long listing since this is a burden for the LIS team.

Program

```
title "** BASICS II - Exercise 3 **" .

title "** Part 1 **" .
define dofiles () .
weight by pweight .
select if page ge 16.
if pgwage gt 0 posgwage = pgwage.
if pnwage gt 0 posnwage = pnwage.
if pself gt 0 posself = pself.
frequencies variables = posgwage posnwage posself
      / statistics = median
      / format = notable .
!enddefine .
get file = uk99p
     / keep = pweight page pgwage pnwage pself  .
dofiles .
get file = us00p
     / keep = pweight page pgwage pnwage pself  .
dofiles .
get file = mx00p
     / keep = pweight page pgwage pnwage pself  .
dofiles .
title "** Part 2 **" .
get file = br06p
     / keep = pweight page pgwage pnwage pself  .
dofiles .

get file = co04p
     / keep = pweight page pgwage pnwage pself  .
dofiles .

get file = gt06p
     / keep = pweight page pgwage pnwage pself  .
dofiles .

get file = pe04p
     / keep = pweight page pgwage pnwage pself  .
dofiles .

get file = uy04p
     / keep = pweight page pgwage pnwage pself  .
dofiles .
```

Results

	Gross wage earnings	Net wage earnings	Self-employment earnings
UK99	*13,513*	*10,628*	*10,400*
US00	*25,000*	*NA*	*12,500*
MX00	*NA*	*27,400*	*14,400*
BR06	*5,865*	*NA*	*6,000*
CO04	*4,320,000*	*NA*	*2,400,000*
GT06	*12,000*	*NA*	*6,000*
PE04	*NA*	*5,986*	*2,980*
UY04	*NA*	*54,042*	*42,577*

Answers to questions 1-2:

1. Country with the highest median wages from employment

   Because incomes are expressed in national currencies, it is not possible to compare values directly.

2. *Whereas it is obvious from the results above that MX00 contains only net incomes and US00 only gross incomes (as only one of the two wage variables is filled), for the UK99 dataset it is necessary to look at the documentation on-line to see that all other income variables are reported gross of taxes and contributions (with, in addition, the net wage variable).*

Comments

- Please note that all the results are given in nominal terms and in national currency. In order to make a direct comparison, convert these values to a common currency with the use of exchange rates (or PPPs) and deflators. LIS leaves it up to the researcher to choose the exchange rates and/or deflators that are best suited to his/her purpose.
- LIS detailed income variables are ideally filled with gross values (before taxes and mandatory social contributions are deducted), so that their overall sum (reported in summary income variable ***gi***) is equal to total gross income, from which taxes and contributions are subtracted to get to the final net disposable income figure (***dpi***). In some instances though, the original datasets only report net incomes: whereas total gross income is then not available (and the summary income variable ***gi*** is thus left empty), total final net disposable income figure is obtainable by aggregating all the net incomes (and this is exactly what ***dpi*** includes). In those cases, each LIS detailed income variable will contain net values instead of gross; only for wages there are two separate variables for those two amounts. For all other

variables you need to take care when comparing across datasets that you are not comparing two different concepts of income.