PHPM 672/677 Lab #3: Arrays & Loops
Due date: Submit by 11:59pm Monday 2/17

Overview

Most assignments will have a companion lab to help you learn the task and should cover similar content. You will be given time in class to work on the lab and ask me questions as you work through them. You should seek help from all sources (me, peers, internet) in doing this lab. **You will get minimally commented code that completes the lab.** Your first goal is to read & understand the source code that was provided. If you are confused, READ the chapters from the book (see assignment 2). Then, you should try to replicate the code yourself, if possible without looking at the given code. You do not need to document sources of help as this is a learning experience, rather than a graded homework. However, it is your responsibility to complete the lab outside of class, if you were not able to complete them during the allotted time in class. There will not always be sufficient time in class to complete, but it is important that you complete them in order to learn the required contents.

Labs are part of the companion assignment. Labs will not be graded, but if you do not submit them 2 points will be taken off the companion assignment grade.

Submission. Labs are due on the same day as the companion assignments. HOWEVER, for the labs to be useful in helping you do the companion assignments, you should complete the labs in the first week, so that you have sufficient time to work on the assignment the second week. For, labs that are submitted within one week, I will do my best to give you feedback within a few days so that it will help you do the assignment. You have the option to submit one of the following.
THERE ARE THREE OPTIONS for lab 3. Pick one you are most comfortable doing.

OPTION 1: Read Code - Detailed Commented Code

For those of you who are still unsure about reading SAS code submit two detailed commented code

- Download lab3_for.sas & lab3_while.sas
- Submit FULLY commented code (i.e. line by line translation into English) for lab3_for.sas
- Submit commented code for lab3_while.sas, ONLY for the sections that are different from lab3_for.sas. You DO NOT need to repeat comments for identical code.
- Submit the log & results from running the code provided (6 files. lab3_for.sas/log/lst & lab3_while.sas/log/lst). You should be reading the log & results to understand what the program does and add comments.

OPTION 2: Rewrite SAS code you read

Recommended for most (i.e., pick this if you are not sure).

For those of you who are comfortable reading but still struggling to write SAS code

- Download lab3_for.sas & lab3_while.sas
- Read the code line by line and understand what I did.
- Download lab2.sas and try to write the same code you read in lab3_for.sas (see P1 below)
- Repeat. That is start with lab2.sas and try to write same code for lab3_while.sas (see P1)
- Now extend your _for program to add more binary variables (P1.6)
- Submit code, log, results (html) from three programs (12 files. lab3_for.sas/log/lst & lab3_while.sas/log/lst & lab3_for2.sas/log/lst & lab3_while2.sas/log/lst)

OPTION 3: Write new SAS code

For those of you who are comfortable reading & writing SAS code

- Submit code, log, results (html) from the optional exercise (P2) below (3 files. lab3.sas/log/lst)

Objective

In this assignment, you will learn to write elegant code. By the end of this assignment, you should be able to

- use `for` loops (iterative loops)
- use `while` loops (conditional loops)
- use one dimensional `arrays`

Setting Up

1. You will be working in the same working directory (i.e. folder on your computer) called lab2. This should have your SAS files from lab2.
2. Copy the SAS program you wrote for lab2 into a new SAS file in the directory lab2/ with appropriate names (see below P1.2 and P1.3)
3. You will edit this program.
4. If you did not have lab2 working, it is ok for you to get a copy of all the required files from a peer to do lab3.

Getting Data: We will be using the same data as we used in lab 2.

P1. Coding with Arrays and Loops

The more different syntax you try to use, the more you will learn. There are many different ways to do one task. You have to try different ways to understand the difference between methods, find out which works best for you as well as understand what might work best in a given situation.

In lab2 you wrote code to do the following:
Code all binary variables to be 1 for YES, 0 for NO, and missing for all other values. For analysis, this is the best way to code all binary variables.

1. **Edit the sections of your code that did the above in lab2 as follows.**
    - Try to read the lab 2 *source code* (this typically refers to your .sas file. The code a programmer wrote to have the computer execute) provided as well as slides from class. If you can, try to write these programs without looking at the slides. Your goal is to write elegant code, with less repeated text so that if you find an error, something changes in the algorithm, or you want to extend your program, you have less places to fix/change.
2. **[Using `for` loops]**
    - Use arrays and `for` loops (in SAS these are just iterative `do` loops) to rewrite your code to accomplish the same tasks. Remember to keep the same code for generating the descriptives at the end. These programs should be saved out with postfix "_for" with corresponding logs (i.e. lab3_for.sas).
3. **[Using `While` loops]**
    - Use arrays and `while` loops to rewrite your code to accomplish the same tasks. Remember to keep the same code for generating the descriptives at the end. These programs should be saved out with postfix "_while" with corresponding logs (i.e. lab3_while.sas).
4. **[Testing]**
    - Now you should have 3 programs (lab2.sas, lab3_for.sas, lab3_while.sas) that do identical things using different code. That means the output should also be identical. Parts of the log will not be identical, because this reflects the code being run. But the results of the descriptive analysis on the variables constructed using arrays and loops should be identical as the results from lab2. Compare the output by either using Word/review/compare or diff (small program that will compare text files.) Confirm that the RESULTS are identical.

5. **[Debug]**
    - If anything is not identical, debug your code and fix. Retest the new code, and repeat until you pass the test (i.e. results are identical).
6. **[Extend your _for & while program]**
    - Now extend your two programs (both _for & _while) to add more binary variables from the original nsduh dataset (`snfever, chewever, pipever, crkever`). Remember to keep the same code for generating the descriptives at the end.
    - You should simply "update" your program (i.e. save out to the same program). These programs should be saved out with a postfix (i.e. lab3_for2.sas) with corresponding logs.
        - Compare the results from this output to those from _for & _while. You should check that the outputs for the original variables are identical to what you had before. It is not unusual for programmers to introduce bugs to old code that used to work in the process of extending it.
    - So it is important to test the code to confirm the things that used to work is still working after you have extended your program. This should be your final version, which you might need to come back to in future labs.
7. **[Optional]**
    - Look at what was done for P3.1 in lab2.sas for creating a variable called subst when any of the four ever variables was a yes. Now create a new variable for substance abuse to use all the substance abuse variables (including the four new ones you added above) called `subst_ext`. But this time you must use arrays and loops, rather than write out a long conditional statement.

P2. [Optional small exercise to practice more coding]

This does not need to be submitted. If you want more practice writing code try problems at https://pinformatics.tamhsc.edu/phpm672/lab3op3.html (under lab/). The Grades Data is under the data tab (phpm672/data/L3 op2: Grades)

There are also many other practice problems you can try on the Internet.