Assignment 2

Submission format:

* Submit your code as a .sas file. The filename MUST have the following format:

LastnameFirstname\_Username\_AsssignmentNumber.sas.

For example, my submission will be: PhamMinh\_mtpsma\_Assignment1.sas

* Include your problem #’s in your solution (as a comment).
* The output of the code is submitted in a Word document. The filename MUST have the same format ( but with .doc extension). In the Word document, you should have problem #’s too.
* The code should be written so that, except for file-directory locations, the code can be directly run on any computer with SAS or SAS University ed.
* Any text that is not code ( such as responses to a question) will be included in the .sas file as comment lines. (IMPORTANT). The word document is strictly for SAS output. Everything else should be in the .sas file.
* You must submit the HW to the correct myCourses dropbox.
* Duedate: 5pm Monday, Sep 23rd, 2019.
* Make sure you have both files in the submission.

Some preliminary materials:

To understand some materials from class this week—the length of character variables, and how the RETAIN statement works—it will be useful to remember that the DATA step really takes place in two phases:

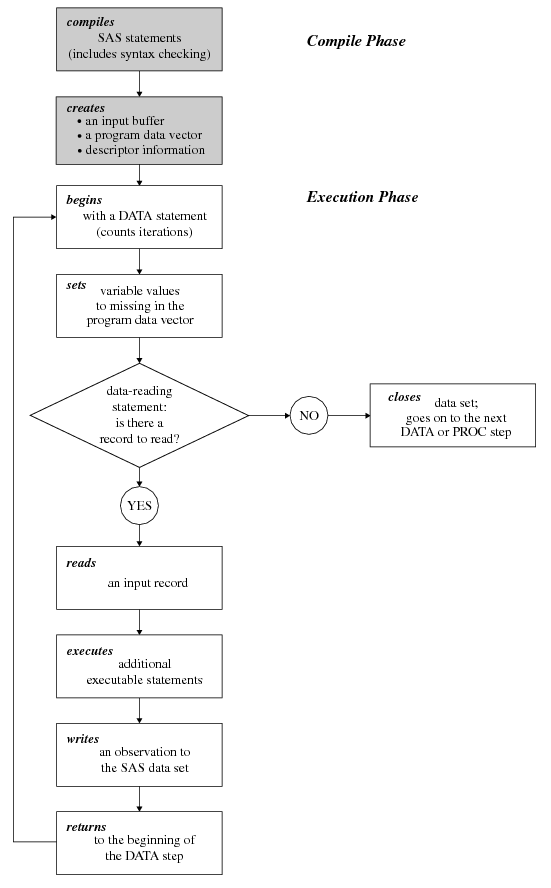
1. Compilation, or compiling, phase
2. Execution, or executing, phase

The idea of compiling and execution phases will be familiar to those of you have worked with languages that need to be compiled, such as C++. The idea is that compiling is like *preparing* the program to run, and executing is actually *running* the program.

To help “see” the compiling/execution phases, read   
*611-DataStepCompileAndExecute.pdf* *.* This will also be used as the basis for a homework question. For more details—and this is a very important aspect of learning how SAS “thinks”—see <http://support.sas.com/documentation/cdl/en/lrcon/65287/HTML/default/viewer.htm#p08a4x7h9mkwqvn16jg3xqwfxful.htm>. And then use the Next Page feature for more information. Here is the first page of this link.

**Flow of Action**

When you submit a DATA step for execution, it is first compiled and then executed. The following figure shows the flow of action for a typical SAS DATA step. Flow of Action in the DATA Step



1. (2pts) This problem is associated with *DataStepCompileAndExecute.pdf* . Read *dfwlax\_space.txt* into the SAS data set named dfwlax. You can use the following code if you wish:

**%let dirdata=*fill this in...;***

**data** dfwlax;

infile "&dirdata.dfwlax\_space.txt";

length Flight Dest $**3**;

informat Date mmddyy8.;

format Date mmddyy10.;

input Date Flight FirstClass Economy Dest;

**run**;

Now execute the following code. Notes:

* 1. The PUT statement writes variables to the SAS log (by default, at least)
  2. You should be able to figure out what the RETAIN statement and the   
     “i + 1;” statements are doing.

Code:

**data** onboard;

retain i **0**;

put \_n\_=;

i+**1**; put \_all\_;

set dfwlax;

i+**1**; put \_all\_;

Total=FirstClass+Economy;

i+**1**; put \_all\_;

**run**;

Explain how the last three PUT statements correspond to the three steps shown in *DataStepCompileAndExecute.pdf*. Also, there are only 10 records in dfwlax, so why did the value of \_N\_ get to 11? Why did the value of i get to 31?

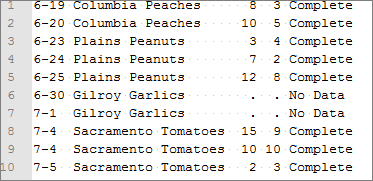
1. (2 points) Start from reading the mydata\_Plus.csv file, and convert all of q1, q2, q3, and q4 values as follows:

* “.” (for missing values) to 9
* 4’s to 20
* 5’s to “.”

Then submit your code.

1. (2pts) Let’s reconsider the Games.dat data, but now modified to Games­\_Plus.dat:

The last column indicates whether the information on the game is completed or that there is no data yet available on that game.



Suppose you want to create a SAS data set such that:

1. You want to accumulate the maximum number of runs *that you know about*. In this case, for example, record 6 should list MaxRuns=8.
2. You only want to accumulated the total number of runs to date until you don’t have information—when this happens you want to set RunsToDate to a missing value. In this case, for example, record 6 should list RunsToDate=.;

Write SAS code to handle this situation. Also note that you will need to figure out how to read in “No Data” correctly.

1. (3 points) A data set of monthly Rochester snowfall was created about 10 years ago. That data appear in each of these files:

RochesterSnowfall.xls 🡨 Excel Spreadsheet

RochesterSnowfallTab.txt 🡨 tab delimited

RochesterSnowfallSpace.txt 🡨 space delimited

RochesterSnowfall.csv 🡨 comma delimited

Select the csv file for the assignment, but feel free to try the others.

I encourage you to look at each of these first three files to see how they differ—they are all the same data, but have different format characteristics. Also, you will notice that that there is a “T” in some of the months, which stands for “Trace”. (A trace of snow is a very small amount, too small to be recorded as a number.)

Write a SAS program that reads in the data for snowfall *from the 1884-85 season until the 2001-02 season* (the 2002-03 season is incomplete so please eliminate it from consideration). Use PROC PRINT to print out a listing of the data with an appropriate title, and put season as the first column of o/p—do not show line numbers.

Hints and notes:

* 1. There are extraneous lines before, and after, the relevant data. There are two options to the INFILE statement you will probably find useful:

FIRSTOBS = record-number

OBS = record-number

* However, please note that if you did not know how many lines to skip at the beginning, and how many to skip at the end—or you had hundreds of such files with different amounts to skip in each one---then you would need to use a more clever way to read in the data.
  1. If you read the data as numeric (probably best since you might want to do math on it if you decide to calculate summary statistics later), then the T’s will be converted to missing values and create a warning in the SAS LOG.
* However, please note that this is not ideal, because in the final data set you can distinguish between a real missing value (“.”), a trace (“T”), or another value, which may be an error in the data file (e.g., “broken gauge”)
  1. Each attempt to read T as a numeric variable will produce an error in the SAS log. By default, only 40 errors are allowed before SAS stops reporting the errors. To allow for more errors, use a global statement such as  
      options errors=1000;
  2. Because T really means about 0 snow, please *convert all the missing values to 0’s.*
  3. To “put season as the first column of o/p—do not show line numbers,” you will need to learn a bit more about PROC PRINT.

Clean up the data in other ways if needed—using code, of course. Then submit your code and a sample (≤1/2 page) of the output.