STAT 506: Final Project

You are to obtain a data set (or multiple *related* data sets) from real life and demonstrate many of the concepts we have learned throughout the course. *Note*: you must find a real-life data set for use with this project. It does not need to be relevant to your research areas -- just try to find something somewhat interesting to you that is conducive to demonstrating many of the skills you have learned this semester. And don't just use the data we've been using in class.

Below you will find a list of many of the grouped topics we have covered. You are to **choose 10 topics** from the 35 listed below and demonstrate that you have mastered the skills by writing code for each and every item listed under that topic (*e.g.* for topic #2, you have to demonstrate all 3 items listed as parts a-c). There are plenty of topics to choose from -- if a topic doesn't make sense for your data, then pick a different topic.

- At least **4** topics must be from **PG1**
- At least **4** topics must be from **PG2**

On the first page of your project, please provide a **very brief description of your data** and from where it was obtained. Also list just the **numbers of the 10 topics** you chose and **the order in which you did them**. For example,

1, 2, 6ab, 21, 5, 7, 22, 6c, 25, 8, 28

Then for each topic, write code to demonstrate your mastery of each item listed for that topic. You cannot double up on requirements by using the same statement for two different topics (*e.g.*, the PROC PRINT used for #3 should not also count for #23c and #27e). You **MUST use comments** in your code to show me which item is being fulfilled. For example, if I were working on #3, I would use comments after my code like this:

```
proc print data=project.data(obs=10);   /*  3a, 3c   */
  var Stuff;                            /*  3b  */
  etc...
```

You should submit to Brightspace by the deadline:

1. A single .pdf file that is fairly readable and easy to follow:
    - On the first page, include a brief description of your project & the list of your chosen topic numbers
    - Then for each topic, include <u>both</u> your code and any relevant output
        - if your output is a PROC PRINT (for example) of a large table, I don't need to see the entire thing; just enough to show that your code worked and produced the desired result
        - if your output is a SAS table or .csv file or something, just include a screenshot of that part of the log to show it worked; you don't need to upload the output file itself
2. Your reproducible code as a .sas file
3. Any data sets used in the project (or include a link to where they could be easily downloaded)

********** Topics from **PG1** **********

Accessing Data (PG1.02)

1) Importing data:
   a) PROC IMPORT step
   b) PROC CONTENTS for the imported table, using the VARNUM option

2) Access SAS tables or Excel workbooks in a library:
   a) LIBNAME statement
   b) PROC CONTENTS for the library (*we didn't cover this in class, but give it a try: you'll need to use the _ALL_ keyword and NODS option*)
   c) PROC CONTENTS for a single table in the library

Exploring and Validating Data (PG1.03)

3) Print SAS table(s) using:
   a) PROC PRINT
   b) VAR
   c) OBS= option
   d) Use either IN or LIKE in a WHERE statement
   e) Apply at least two different formats

4) Clean up a table that has duplicates by using:
   a) PROC FREQ to identify duplicate rows or column values
   b) PROC SORT to remove unwanted duplicate rows

Preparing Data (PG1.04)

5) Use a DATA step to create a new table
   a) SET statement
   b) WHERE statement
   c) DROP or KEEP statement
   d) FORMAT statement
   e) LENGTH statement
   f) Create two columns using assignment statements; at least one should be in an IF conditional

6) Demonstrate 3 separate IF-THEN statements
   a) IF THEN statement
   b) IF THEN; ELSE IF THEN statement
   c) IF THEN DO; END; statement

Analyzing and Reporting on Data (PG1.05)

7) Use titles and footnotes:
   a) Add a title using a TITLE statement
   b) Add a subtitle using a TITLE2 statement
   c) Add a FOOTNOTE
   d) Use macro variables in at least two of the above
   e) Display labels in a PROC PRINT using a LABEL statement
   f) After displaying the above titles and footnote in the PROC PRINT, clear the titles and footnote

8) Create frequency reports:
   a) Save a one-way frequency report as a SAS table, using the ORDER option
   b) Create a two-way frequency report for two columns. Use formats if helpful. Suppress some of the statistics that would be displayed by default

9) Create a summary statistics report using PROC MEANS:
   a) Produce some non-default summary statistics
   b) Use the VAR statement
   c) Use a CLASS and a WAYS statement
   d) Output the results to a new table

Exporting Results (PG1.06)

10) Export SAS tables
   a) Use PROC EXPORT to save a SAS table as a CSV file
   b) Use PROC EXPORT to save a SAS table as some other file type (not CSV)
   c) Use a LIBNAME engine to save a SAS table in an Excel workbook

11) Export SAS output: For each PROC below use a different type of ODS
   a) PROC FREQ with some options of your choice
   b) PROC MEANS with some options of your choice
   c) PROC UNIVARIATE with some options of your choice

Using SQL in SAS (PG1.07)

12) Use queries (at least 2) in PROC SQL to
   a) Create a new table containing only some of the columns from another table
   b) Display a subset of a table based on some WHERE clause
   c) Display a table sorted by two of its columns

13) Use queries in PROC SQL to
   a) Display the result from merging at least three tables
   b) Use aliases

********** Topics from **PG2** **********

Controlling DATA Step Processing (PG2.01)

14) Demonstrate the following
    a) SELECT group, including
        (1) OTHERWISE
        (2) Explicit output to multiple tables
    b) DROP= and/or KEEP= option
    c) At least two informative PUTLOG statements

Summarizing Data (PG2.02)

15) Demonstrate the following
    a) Use of RETAIN
    b) Sum Statement
    c) Use of FIRST. and LAST. in IF conditionals
    d) PROC SORT

Manipulating Data with Functions (PG2.03)

16) Demonstrate the use of <u>two</u> different CALL routines
    a) One may be one of the routines discussed in the slides (SORTN, MISSING), but doesn't have to be
    b) One must be a different routine than the ones discussed in the slides

17) Use the following functions
    a) TODAY
    b) MDY
    c) Usage of two of (YEAR, QTR, MONTH, DAY, WEEKDAY)
    d) INTCK
    e) INTNX

18) Use the following functions
    a) SUBSTR
    b) LENGTH
    c) SCAN
    d) PROPCASE
    e) One of (RIGHT, LEFT, UPCASE, LOWCASE, CHAR)
    f) CATX (or CAT, CATS, CATT)

19) Use the following functions
    a) FIND
    b) Use SUBSTR on the left side of an assignment statement (SUBSTR()=value)
    c) TRANWRD
    d) COMPRESS
    e) One of (TRIM, STRIP, COMPBL)

20) Use at least 8 different functions including at least one from each part
    a) ROUND, CEIL, FLOOR, INT
    b) LARGEST, SUM, MEAN, MIN, MAX
    c) N, NMISS, CMISS

21) Converting Column Type
    a) Character to Numeric (INPUT function)
    b) Numeric to Character (PUT function)
    c) Demonstrate automatic conversion (*need to show the log here*)
    d) PROC CONTENTS showing the original/new column types

Creating Custom Formats (PG2.04)

22) Create and use at least 2 user-defined formats using VALUE statements:
    a) One must be for a character column
    b) One must be for a numeric column
    c) Save these as permanent formats
    d) PROC FREQ applying the formats you created

23) Create and use at least 1 user-defined format using CNTLIN:
    a) Use a DATA step to prepare the table
    b) Use PROC FORMAT to read in the table as format(s)
    c) PROC PRINT applying the format(s) you created

Combining Tables (PG2.05)

24) Types of combining:
    a) Concatenate two or more tables using RENAME= option
    b) Simple merge (ONE to ONE)

25) Merge two or more tables—at least <u>two</u> different merges must be done
    a) One must have non-matches
        (1) Use IN= options with a IF conditional statement
        (2) Direct output to 2 different tables (matches and non-matches)
    b) One must use the options below
        (1) RENAME= option
        (2) DROP=/KEEP= options

Processing Repetitive Code (PG2.06)

26) DO Loop Processing—at least <u>two</u> loops must be made
    a) Using an index column
    b) BY option
    c) WHILE or UNTIL
    d) Explicit output in loop

Restructuring Tables (PG2.07)

27) Rotating a narrow table to a wide table
    a) Appropriate IF statements
    b) BY statement
    c) Conditional output
    d) DROP old columns (or KEEP only the important ones)
    e) PROC PRINT

28) PROC TRANSPOSE
    a) BY
    b) OUT= option
    c) NAME= option
    d) PREFIX= option
    e) PROC PRINT showing the transposed table

Arrays (PG2.extra) *<NOTE: Arrays aren't technically part of PG2, but we'll talk about them anyway.>*

29) SAS Arrays: at least <u>two</u> arrays must be made:
    a) One must be a "Table lookup" array: Create new columns from array statement using
        (1) initial values
        (2) _temporary_
    b) One must be used in either
        (1) A DO loop, using DIM(array_name)
        (2) Another function

********** Topics from **MC1** **********

<NOTE: We won't get through all of this in lecture, but I'll leave these as options in case you want an excuse to dive in deeper with macros.>

Macro Variables (MC1.02)

30) Demonstrate the following:
    a) Usage of macro variable name delimiter "."
    b) At least two of (%UPCASE, %SUBSTR, %SCAN, %INDEX)
    c) At least two of (%EVAL, %SYSFUNC, %STR, %NRSTR)

Macro Definitions (MC1.03)

31) Write and call at least <u>two</u> macro programs (different from those in another topic) using:
    a) %MACRO
    b) %MEND
    c) OPTIONS MPRINT;
    d) OPTIONS MCOMPILENOTE=ALL

32) Write and call at least <u>two</u> macro programs (different from those in another topic):
    a) One must use positional parameters
    b) One must use keywords parameters with default values

DATA Step and SQL Interfaces (MC1.04)

33) Demonstrate macro use during SAS execution:
    a) At least two uses of the CALL SYMPUTX routine
    b) At least two uses of the SYMGET function

Macro Programs (MC1.05)

34) Macro level conditional processing:
    a) %IF %THEN %ELSE
    b) OPTIONS MLOGIC;
    c) %IF %THEN %DO; %END;

35) Macro level %DO loops:
    a) Write a macro level %DO loop
    b) Use indirect references using &&
    c) %put _USER_; (show log here)