Programming HW #5

Specifications:

- You are allowed at most five `DATA` steps and seven `PROC` steps to complete this assignment.
- Build the necessary references. Your Data folder is InputDS for SAS data sets and RawData for raw files, your Results folder is Results, and your personal storage location is HW5. As you'll see below, you will, separately, need your HW4 library again.
- For this assignment, the data we need is split across five files: LeadProjects, O3Projects, COProjects, SO2Projects, and TSPProjects. Each file contains real data about job sites where a particular pollutant had to be cleaned up. E.g., O3Projects contains data about jobs cleaning up ozone pollution.
  - All text files are in your RawData folder
  - The data sets are *nearly* identical in their structure, but not in their contents
    - Variables are not the same, but common variables are in the same order
    - Values of the variables are, as expected, different
    - Delimiters are the same
    - If a variable requires data cleaning, it likely needs cleaning in every data set that contains that variable.
  - Recall, you worked with the LeadProjects data set in a previous assignment.
    - Do not read in that file again! Instead, use the data set we created in the previous assignment. For fairness in grading – you'll be using my version from the Results library.
- Similar to using the Lead data set again, you'll need your custom format from HW4 again.
  - Recall we named in MyQtr in HW4. Normally you would use your own, but, again for fairness in grading you'll be using mine. I've placed it in the Results library as well.
  - Again, you do not have the budget to recreate it – the idea is to practice using items that already exist on your computer (or on your company's server!) so that you are not redoing the same work repeatedly.
  - Yes, there is already a format called MyQtr in the InputDS library, but don't be tempted to use it! Instead, you need to ensure your code is set up to use the specific one designed for this project and not one you see with the same name. (Note: SAS is going to use the first format it finds with the name MyQtr!)
- You need a second custom format for this assignment. I've placed the second format, $PolMap, in the InputDS library for you. [You do not have the budget to recreate it either!]
- Before validation:
  - Combine all five pollutant data sets into a single data set. (Despite the efficiency of interleaving, pay close attention to that budget!)
  - Label the *data set* `Cleaned and Combined EPA Projects Data` to match my *data set* label.
  - Perform any necessary data cleaning.
    - Typically, deriving a variable in multiple places goes against GPP since it requires more work to maintain and increases likelihood of mistakes. Hence, in practice, all data cleaning should be placed in a single location unless it is not possible/practical to so.
    - **Because of this, your DATA steps that read in the data should be exactly four statements long: DATA, INFILE, INPUT, RUN. *That's it!***
- Validation: My data set is named HW5DugginsProjects and is in the Results library; you should name this data set using just your last name and the word Projects. (For example, Tony Stark would save his as HW5StarkProjects.)
  - Compare both the descriptor and content portions of my data set to yours.
  - Use a threshold of 1E-9 for numeric comparisons.
  - Use a macro variable (with whatever name you choose) to control the redundant settings during validation)
  - To ensure your validation lines up as intended, be sure to plan ahead (as always) but also be sure to get rid of the variable called Member from your descriptor portion - our last names aren't the same, so our data set names won't be equal either!

- For our analytical/reporting component this week, our goal is to make one set of graphs (one for each Pollutant) that shows the first and third quartiles of total job cost. And, for extra practice, we are going to do this two ways – first, the brute force method that uses only bar charts to imitate what a more naive programmer might produce (and, not incidentally, to practice a lot of our graphical skills). Then, second, using graphical tools from your reading to make the graph properly. My graphs are named : HW5DugginsBadPlot through HW5DugginsBadPlot4 and HW5DugginsGoodPlot through HW5DugginsGoodPlot4. Below are some specific instructions that you cannot infer from the graph. For each *new* customization you are asked to include, I've given you the necessary code as well.
  - Make sure your statistics calculations are efficient! For reference, my stats data set has 6 columns and 120 records.
  - All graphs are 300 DPI, six inches wide, and should appear in a PDF report (mine is called HW5 Duggins Projects Graphs) and as individual PNG files.
  - All graphs should be saved in your HW5 directory.
  - Name the graphs the same way I did. (SAS automatically counts for you, you just need to provide the name!)
  - To prevent the automatic page breaks in the PDF destination, use the STARTPAGE= option in your ODS PDF statement. Using the value NEVER for this option will keep SAS from inserting a page break until the page is full. This should let you get two graphs per page (assuming the VTL doesn't do something weird here.)
  - In the brute force version of the graphs:
    - Each graph shows the bar chart for both the 25th and 75th percentiles, with the 25th percentile in front since it cannot be bigger than the 75th percentile. We can use a single-color bar graph in front to block out the part of the group-colored bars we don't want to see.
    - To include two compatible plots in the same graph, just include both plotting statements in the same SGPLOT step. Thus, to get two vertical bar charts, you need two VBAR statements.
    - Unfortunately, when you add two graphs they both show up in the legend. Use some of the skills from your reading to select a specific plot for the legend.
    - Each graph shows side-by-side bars based on dates and uses Region as the charting variable. Separate graphs are generated for each unique value of PolCode. To do this, you need to remember how to get a PROC step to loop through and carry out the same operation (e.g., making a graph) on different subsets of data. (This is not new - we've been doing it since way back with PROC PRINT!)
    - As with the last assignment, choose your own colors for the bars. I used the same ones again, but you could use different ones if you like. [When specifying colors, just like anything else, order matters – put the colors for the first graph first, second graph second, etc.]
    - To simplify things with the bar charts, use the NOOUTLINE option on them so you don't have to worry about the borders of the boxes.
    - The data labels are the number of observations in each group and only appear on the Q3 graph. They are in 7pt font.
    - I forced the groups to appear in a specific order by using the GROUPORDER= option. My groups are arranged so that earlier dates come first.
  - For the better graph:
    - For the grid lines, make them thicker than normal and select your own shade of gray.
    - Once you figure out the right graph to use, you can use the HIGHLABEL= option to apply your data labels. (Then, you can use LABELATTRS to adjust attributes like font size or color.)
    - Similar to your bar charts, you can use LINEATTRS to set the color of the lines around the boxes.
    - I used only the bottom 87.5% of the vertical axis for plotting.,The rest is used for the legend.
  - For both sets of graphs, do the following to get the titles and footnotes set up properly.
    - My third title is 8 pt font
    - None of the axes in any graph has a label - that text at the bottom is a FOOTNOTE
    - Prevent SAS from automatically adding the PolCode grouping (e.g. PolCode=5) by using the global option NOBYLINE
    - To get SAS to automatically place each value of the pollutant in the second title, use the keyword #BYVAL1 in your title. [Once you get this working, think about why a macro variable would *not* be appropriate here!] The code I used is below.

      ```
      title2 ‘By Region and Controlling for Pollutant = #byval1’;
      ```

** As with previous assignments, but especially with graphs, there may be differences that occur because you are operating on the VTL when producing output outside of SAS. Before spending countless hours agonizing over what could have gone wrong, please just ask if the discrepancy you're seeing is something we can (and should) control at this point in the class! (Keep in mind the graphs, and the assignment as a whole, can be done with only code presented as part of this course – just like all your assignments!)

** There is a rubric item regarding having a note indicating that operations were carried out on missing values. We now know enough to avoid this note. **For this and all future assignments, ensure that note does not appear by explicitly coding to avoid it.**

** I know the directions seem longer this week, but I wanted to be pretty explicit about the extra items in the plots. I know plots can feel a little overwhelming because there is **so** much we can change with them. Don't hesitate to ask for guidance on the graphs (or anything else)!