BIOS 511 Lab 5
Basic Use of Formats and Informats

Please read the following instructions carefully before beginning this lab. You will need to start by downloading the DM dataset from the Lab-03 assignment on the Sakai site. This dataset will be the basis of all activites for this lab. Refer to Lab-03 for documentation on this dataset.

Instructions:

- All tasks should be completed in a single SAS program named lab-05-PID.sas where PID is your student PID number. Please make sure to include an appropriate header in your SAS program.

- All the output that your SAS program produces in this lab should be delivered to a *single HTML file* named lab-05-PID-output.HTML.

- In your code, before starting a new task, include a block comment with the task number

```
/*****************************************************************
                        SAS Code for Task # X
 *****************************************************************/
```

- Recommendation: While initially writing the SAS program, do not concern yourself with creating a permanent output file. Simply view results in the results window to verity that your program has created the desired output. Once your program is essentially complete, add the appropriate ODS statements to create the permanent HTML file based on the requirements above.

- You will upload the SAS program, SAS log, and HTML output file to document completion of the lab.

BIOS 511 Lab 5
Basic Use of Formats and Informats

Task 1: For this task, you will create a format and use the format to perform several simple analyses.

Step 1: Write a PROC FORMAT step that creates a format called AGECAT that maps the values of the AGE variable to the formatted values "1: <50", "2: 50 to <65", "3: >=65", or "4: Missing". When writing the VALUE statement that is used to create the AGECAT format, use the LOW and HIGH keywords to define the ranges.

No template is provided for this exercise. A solution that does not use the LOW and HIGH key words will not be given full credit. You may choose to use the OTHER keyword at your discretion.

Step 2: Write a DATA step that creates a dataset named DM1 in the work library that uses the ECHO DM dataset as input.

For this DATA step, do the following:

Step 2.1: Use a FORMAT statement to permanently attach the AGECAT format to the AGE variable.

Step 2.2: Using the PUT function and user-created AGECAT format, create a new variable named AGECATEGORY that has the values "1: <50", "2: 50 to <65", "3: >=65", or "4: Missing".

Note that everything you can do with the PUT function you can do with IF/THEN/ELSE statements. This is simply another technique.

Step 2.3: Use a LABEL statement to assign the label "Age Category" to the AGECATEGORY variable.

No template for this DATA step is provided. Your solution should abide by the convention that LABEL and FORMAT statements typically go at the top of a DATA step just beneath the SET statement.

Before moving on, write a simple PROC CONTENTS step that prints the metadata for the variables in the DM1 dataset. Observed the length of the AGECATEGORY variable. When a variable is defined using the PUT function, SAS will set its length using the longest formatted value. Thus, when a variable us created using the PUT function, it is not critical to define its length with a LENGTH statement. Note that if you used a LENGTH statement (good for you) you will need to comment out that statement momentarily to see this behavior.

Before moving on to step 3, comment out the PROC CONTENTS and uncomment the LENGTH statement (if you had one).

Step 3: Write a simple PROC FREQ step that uses the DM1 dataset as input and performs a one-way frequency analysis on the AGE variable. Temporarily assign AGE the label "Age Category" in the PROC FREQ step. Do not use any options on the TABLE statement. No template is provided for this exercise.

For this analysis, PROC FREQ will use the formatted values of the AGE variable in the analysis but will exclude the values with a missing value for AGE even though the formatted values do not appear missing.

Step 4: Write a simple PROC FREQ step that uses the DM1 dataset as input and performs a one-way frequency analysis on the AGECATEGORY variable. Note that if the AGECATEGORY variable is used in the analysis, missing is treated as a category (because the actual value of the AGECATEGORY variable is non-missing).

BIOS 511 Lab 5
Basic Use of Formats and Informats

Task 2: For this task, you will learn to use the INPUT function and SAS *informats*. Whereas a *format* is a map of character or numeric values to character values, an informat is a map of character values to numeric values.

Step 1: Using the PROC FORMAT step, one can create *informats* in much the same way as they can create formats. For example, the following PROC FORMAT step creates an informat that maps the values "M" and "F" to numeric quantities 1 and 2, respectively.

```
proc format;
 invalue sexn
  "M" = 1
  "F" = 2
  OTHER = .;
run;
```

Note that informats have names just like formats do (e.g., SEXN) but they are created with an INVALUE statement instead of a VALUE statement. Just as the PUT function can be used to create a new character variable based on formatted values based on a given format, the INPUT function can be used to create a new numeric variable based on a given informat.

Step 2: Write a DATA step that creates the DM2 dataset in the work library using the DM1 dataset as input. Add the following assignment statement to that DATA step.

```
sexnum = input(sex,sexn.);
```

Inspect the DM2 dataset before moving on to see the values of the SEXNUM variable.

Just as is the case for formats, SAS has a wide array of built-in informats that are available. The informats for reading dates are particularly useful. The *yymmdd10* informat can be used to read character variable dates in the form "YYYY-MM-DD". Thus, we can use the INPUT function and the *yymmdd10* informat to create numeric date variables. This strategy could have been used in the prior lab too.

Step 3: *Add additional code to the DATA step that creates the DM2 dataset* to create three additional variables TRTSTDTN, TRTENDTN, and TRTDUR.

Step 3.1: The values of the numeric variables TRTSTDTN and TRTENDTN should be equal to the corresponding dates stored in the character variables RFXSTDTC and RFXENDTC. To create these variables you will use the INPUT function and the *yymmdd10* informat. The first of the two assignment statements is provided for your below. You must write the second.

```
trtstdtn = input(rfxstdtc,yymmdd10.);
```

Step 3.2: The values of the numeric TRTDUR variable should be the number of weeks a subject was treated. It should be derived based on the values of TRTSTDTN (treatment start date) and TRTENDTN (treatment end date) and so the assignment statement that creates TRTDUR *must* come after the assignment statements that create TRTSTDTN and TRTENDTN in the DATA step.

It is critical that you understand why the assignment statement that creates the TRTDUR variable must come after the assignment statements that creates the TRTSTDTN and TRTENDTN variables. Make sure you understand why. Explain it to your neighbor or call me or one of the TA's over and explain it to us. If you cannot do that, you don't understand!

BIOS 511 Lab 5
Basic Use of Formats and Informats

When one computes the difference between two dates, it is common to add 1 to the resulting difference. To see why, consider a subjects whose first and last date of treatment was the same day. That subject was treated for 1 day but the difference in dates is zero. Use the +1 convention when taking the difference of two dates so that a person who started and ended treatment on the same date has a treatment duration of 1 day. To create TRTDUR as the number of weeks, make sure to divide by 7.

Step 3.3: Add a FORMAT statement to the DATA step assigning both the TRTSTDTN and TRTENDTN variables the *date9* format. Use a single FORMAT statement for this. Note that the FORMAT statement does not change the data type for these two variables. They are both numeric and so differences between them are meaningful and well-defined even if the format causes the dates to display as combinations of numbers and letters (i.e., 01MAR2018) which do not appear numeric.

Step 4: Use PROC MEANS to analyze the variable TRTDUR, producing the following statistics in an output dataset: N, mean, standard deviation, min, and max.

- The AGECATEGORY variable should be included in the class statement, as should the ARMCD variable.
- Rather than taking the default output from PROC MEANS, for this task you will write the analysis results to a SAS dataset using an OUTPUT statement.
- Use the NOPRINT to suppress the printed output from PROC MEANS.
- The results dataset that you create with the OUTPUT statement should be named WORK.TRTDUR_SUMMARY.
- You should instruct PROC MEANS to perform the analysis of the TRTDUR variable by treatment group as well as by age category *but not by both.* To achieve this, use a WAYS statement (see the SAS documentation for an example that should help guide you or an example from the lecture programs).
- The variables storing the summary statistics can be named anything of your choosing (or you can use the AUTONAME option).
- If this step is done correctly, the WORK.TRTDUR_SUMMARY dataset should contain the following data (your variable names may differ from mine):

ageCategory	ARMCD	_TYPE_	_FREQ_	n	mean	std	min	max
	ECHOMAX	1	282	282	52.2649	1.41332	48.2857	56.7143
	PLACEBO	1	320	320	52.2080	1.34072	48.2857	55.7143
1: <50		2	27	27	52.3598	1.38639	50.0000	54.7143
2: 50 to <65		2	281	281	52.1886	1.43380	48.2857	56.7143
3: >=65		2	290	290	52.2502	1.31464	48.2857	55.5714
4: Missing		2	4	4	53.5000	0.93678	52.2857	54.4286

Step 5: Write two PROC PRINT steps to print out the summary statistics in the WORK.TRTDUR_SUMMARY dataset. In the first PROC PRINT step, print out the analyses by treatment group and in the second step print out the analyses by age category.
For both PROC PRINT steps do the following:

BIOS 511 Lab 5
Basic Use of Formats and Informats

- Suppress default observation numbering using the appropriate PROC statement option.

- Only print the classification variable relevant to the summary being printed (e.g., ARMCD or AGECATEGORY) and the summary statistics.

- Use a FORMAT statement to apply W.D numeric formats to the mean, min, and max (6.2) and standard deviation (7.3) to make the display more visually appealing.

- Use a LABEL statement to give the summary statistics useful labels. For practice, your labels should match the solution as much as possible (this will require that you use the SPLIT option on the PROC PRINT step).

- Use a WHERE statement to control which observations are printed in which PROC PRINT step.