# STAT 5110/6110: SAS Programming and Applications

## 1-A. The First SAS Program

Peng Zeng

Department of Mathematics and Statistics

Auburn University

# What is SAS?

SAS (previously, Statistical Analysis System) is an integrated system for performing tasks such as

- data entry, retrieval, and management
- report writing and graphics
- statistical and mathematical analysis
- business planning, forecasting, and decision support
- operations research and project management
- quality improvement
- applications development

As a summary:

data access, data management, data analysis, data presentation.

# Access to SAS

SAS Studio via SAS OnDemand for Academics

- free to access
- free online storage space (5GB)
- web-based integrated development environment

`https://www.sas.com/en_us/software/on-demand-for-academics.html`

About SAS Studio:

- CODE tab: type your SAS programs here
- LOG tab: display notes or error messages
- RESULTS tab: display output such as figures, tables, etc.
- Left panel: files and folders, libraries
- Menus, toolbar

# SAS Data Sets

A SAS data set contains the information we want to analyze.

| Name | Gender | Exam 1 | Exam 2 | Exam 3 |
|---|---|---|---|---|
| aa | M | 43 | 70 | 81 |
| bb | F | 67 | 93 | 95 |
| cc | F | 83 | 90 | 50 |
| dd | M | 57 | 76 | 92 |
| ee | F | 74 | 48 | 67 |
| ff | M | 52 | 82 | 92 |

- Rows correspond to observations (cases, subjects, etc).
- Columns correspond to variables (attributes, covariates, etc).

A typical SAS program first creates a SAS data set and then works on it to compute statistics, create reports, draw graphs, or build models.

# The First SAS Program

```
data grades;      /* create a SAS dataset with name "grades" */
   input name $ sex $ exam1 exam2 exam3;
datalines;            /* "input" lists names of variables */
aa   M   43   70   81
bb   F   67   93   95    /* contents of the dataset  */
cc   F   83   90   50    /* one observation per line */
dd   M   57   76   92    /* each column indicates one variable */
ee   F   74   48   67
ff   M   52   82   92
;

proc print data = grades;  /* show contents of the dataset */
run;

proc means data = grades;  /* compute descriptive statistics */
   var exam1 exam2 exam3;
run;
```

# SAS Programs, Statements, Steps

A SAS program is a sequence of statements executed in order. A statement gives information or instructions to SAS and must be appropriately placed in the program.

| | |
|---|---|
| SAS programs | articles |
| statements | sentences |
| data steps or proc steps | paragraphs |

Statements are usually organized in steps.

- DATA steps: read/write data from/to external files, create a data set, defining variables, create subsets, merge data sets, etc.
- PROC steps: analyze SAS data sets to produce statistics, tables, plots, etc.

# SAS Steps

SAS steps begin with either of the following:

- DATA statement
- PROC statement

SAS detects the end of a step when it encounters one of the following:

- a RUN statement (for most steps)
- a QUIT statement (for some procedures)
- the beginning of another step (either DATA or PROC)

# Syntax Rules for SAS Statements

Each SAS step consists of some SAS statements.

- Each SAS statement usually begins with a SAS keyword
- One SAS statement may contain one or more options
- Each SAS statement ends in a semicolon (;)

SAS statements are free-format.

- SAS statements can be in upper- or lower-cases.
- One or more blanks can be used to separate words.
- They can begin and end in any column.
- A single statement can span multiple lines.
- Several statements can be on the same line.

SAS does not distinguish upper-/lower-cases in general. However, texts that are enclosed in quotation marks are usually case sensitive.

# Comments

To make your programs more readable and easier to understand, you can insert comments into your programs.

- Start with $*$ and ends with a semicolon (;).
- Type your comments between $/*$ and $*/$.

SAS will ignore comments when executing programs.

Writing comments is a good habit, particularly when you are working on a large project.

# Names of SAS Data Sets and Variables

SAS names (for data set or variable) have these characteristics:

- start with a letter or the underscore character (_)
- subsequent characters can be letters, underscores, or numbers.
- no blanks or special characters (such as commas, semicolons)
- no more than 32 characters in length
- can be uppercase, lowercase, or mixed-case.

Example of valid names:

- data5mon
- FiveMonthsData
- _5_month_data_

Example of invalid names:

- 5monthsdata
- data#5
- five months data

# Submitting SAS Programs

## SAVE YOUR PROGRAM FREQUENTLY !!!

- Submit your program to run by clicking the icon in the toolbar.
- Submit the whole program or the selected steps.
- Save your program every time before you submit it to run.
- A run statement tells SAS to process the codes before it.
- If there is no run statement, SAS will not process anything when you submit your program.

# In-Class Exercise

- Login to SAS OnDemand for Academics, launch SAS Studio, and get familiar with the environment.
- Work on the first SAS program (01-grades.sas).
  - Type the code (or copy and paste) in SAS Studio. Pay attention to the different colors of the codes in the CODE tab.
  - Run the code and read the log messages in the LOG tab.
    - After successfully executing the data step, the log window contains something like "The data set WORK.GRADES has 6 observations and 5 variables."
    - After successfully executing a procedure step, the log window contains something like "There were 6 observations read from the data set WORK.GRADES."
  - Read the output in the RESULTS tab.
  - Explore the information in the OUTPUT DATA tab.

# In-Class Exercise: Continued

- Modify the SAS codes as follows. Compare the output with the original output.
  - Remove the dollar sign ($) in the input statement.
  - Write the semi-colon in the end of the last data line instead of a new line in datalines as follows.

```
   ff  M  52  82  92;
```

  - The statement run; occupies a single line. Write this statement in the end of the previous line.

```
   var exam1 exam2 exam3; run;
```