Introduction to Programming
Variables, Assignment, Expressions, Logical Expressions

Hye-Chung Kum
Population Informatics Research Group
http://pinformatics.org/

License:
Data Science in the Health Domain by Hye-Chung Kum is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License

Course URL:
http://pinformatics.org/phpm672

How was assignment 1?
Assignment Plan

- 1: Type what I gave you and run
- 2: Write your own relatively simple
- 3: Write your first real program (reusable elegant code)
- 4: Combining Tables (mini tables)
- 5: Combining Tables (normal tables)
- 6: Macros
- Final project

What you learned so far...

- Assignment 1
  - Setup work environment
  - Use the SAS software
  - SAS programming basics
    - data step & proc step
    - libname
    - Writing code & Reading logs
- Assignment 2 (over the next 4 lectures)
  - Understand variables (names, types, labels)
  - To write conditional logic codes
  - Subset columns (variables) from a table
  - Subset rows (observations) from a table
  - Recode, rename variables and calculate new variables
  - Label variables and values
Basics of Programming: SAS

- **data** step
  - Row at a time
- **proc** step
  - Full table
- **libname**: directory location (folder)
  - No libname: temporary data
- **run;** (missing last results)
- **;** (I am done. Can be more than one line)
- **log & lst (html)**: computer communicating back with you what happened
  - Learn to READ the log
- **comments**
  - /* comments */
  - * line comments
  - Length limit 256. If you are using it for long lines pay attention to log for messages.

Some More Basics

- **Vocabulary**
  - Directory = folder
  - Observations = rows = obs
  - Variables = columns = var(s)
SAS online manual

- [https://documentation.sas.com/?cdcId=pgmsascdc&cdcVersion=9.4_3.5&docsetId=pgmsashome&docsetTarget=home.htm&locale=en](https://documentation.sas.com/?cdcId=pgmsascdc&cdcVersion=9.4_3.5&docsetId=pgmsashome&docsetTarget=home.htm&locale=en)

- Google to get help

- Stackoverflow

Talking to a Computer
Example mini-computer

- **CPU (Processor):** Type of instructions it can run
  - Example CPU: Intel(R) Core(TM) i7 CPU Q720 @ 1.6 GHz
- **RAM:** memory
  - 16 GB / 8GB / 4GB / 2GB
- **Hard drive:** permanent memory for storage

Example mini-computer

- **CPU (Processor):** Instruction set (2 bit)
  - 00: Save to
  - 01: Retrieve from
  - 10: Add
  - 11: Subtract

- $5 \times 3 = ?$
Example mini-computer

CPU (Processor)
- Instruction set (2 bit)
  - 00: Save to
  - 01: Retrieve from
  - 10: Add
  - 11: Subtract

RAM
- 0001000
- 01110101
- 10010001
- ...

$5 \times 3 = ?$
- Add 5
- Add 5
- Add 5

Example mini-computer

CPU (Processor)
- Instruction set (2 bit)
  - 00: Save to
  - 01: Retrieve from
  - 10: Add
  - 11: Subtract

RAM
- 0001000
- 01110101
- 10010001
- ...

$5 \times 3 = ?$
- Add 5
- Add 5
- Add 5

<table>
<thead>
<tr>
<th>Address</th>
<th>Instruction</th>
<th>Operand</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>10</td>
<td>0101</td>
</tr>
<tr>
<td>01</td>
<td>10</td>
<td>0101</td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td>0101</td>
</tr>
</tbody>
</table>
Example mini-computer

CPU (Processor)
- Instruction set (2 bit)
  - 00: Save to
  - 01: Retrieve from
  - 10: Add
  - 11: Subtract

Address Instruction Operand
00 10 0101
01 10 0101
10 10 0101

RAM
00001000
00010101
01010001
...

Binary Numbers
1001 0001 0101
8 4 2 1
8*1 + 1*1 = 9
l*1 = 1
4*1 + 1*1 = 5

5 * 3 = ?
- Add 5
- Add 5
- Add 5

Address Instruction Operand
00 10 0101
01 10 0101
10 10 0101

Example mini-computer

CPU (Processor)
- Instruction set (2 bit)
  - 00: Save to
  - 01: Retrieve from
  - 10: Add
  - 11: Subtract

Address Instruction Operand
00 10 0101
01 10 0101
10 10 0101

RAM
00100101
01100101
10100101
...

5 * 3 = ?
- Add 5
- Add 5
- Add 5
### Example mini-computer

#### CPU (Processor)
- Instruction set (2 bit)
  - 00: Save to
  - 01: Retrieve from
  - 10: Add
  - 11: Subtract

#### RAM
<table>
<thead>
<tr>
<th>Address Instruction</th>
<th>Operand</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>ADD</td>
<td>10</td>
</tr>
<tr>
<td>01</td>
<td>ADD</td>
<td>10</td>
</tr>
<tr>
<td>10</td>
<td>ADD</td>
<td>10</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Higher level language
- **Keyword**
- **Vocabulary of language**
  - SAS: proc/data/print

### Variables, Types, Assignments, Expressions
What is a **Variable**?

- A user defined name to represent a piece of memory for storing evaluated value(s). A variable consists of 5 items
  - **Name:**
    - Meaningful human readable name
    - How the user refers to variable
  - **Data Type:**
    - How to interpret variable for data representation
  - **Size:**
    - How much storage memory is needed to store data value
    - Can be inferred from data type
  - **Value:**
    - Actual value associated with variable stored in memory
  - **Storage location:**
    - Usually hidden from user by the interpreter or compiler
    - How the computer refers to a variable

**For Our Purposes: Columns**
- Many variables. A columns of variables

<table>
<thead>
<tr>
<th>Name</th>
<th>Data Type</th>
<th>Size</th>
<th>Memory Location (hidden from user)</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Radius</td>
<td>float32</td>
<td>4 bytes</td>
<td>0x1800F040</td>
<td>3.23</td>
</tr>
<tr>
<td>currKey</td>
<td>char</td>
<td>1 byte</td>
<td>0x1800F049</td>
<td>‘k’</td>
</tr>
<tr>
<td>firstName</td>
<td>string</td>
<td>6 bytes</td>
<td>0x1800B0E0</td>
<td>“morgan”</td>
</tr>
<tr>
<td>width</td>
<td>int32</td>
<td>4 bytes</td>
<td>0x1800CCE8</td>
<td>800</td>
</tr>
<tr>
<td>type</td>
<td>int8</td>
<td>1 byte</td>
<td>0x1800CCE7</td>
<td>27</td>
</tr>
</tbody>
</table>

- var label;
- value label (interpretation)
- SAS: proc contents