Assignment 1

- No data
- Yes readme.txt
What we are going to learn

- Big Picture
- How to avoid code confusion and associated programming errors.
- Common pitfalls.
- Programming Style Guidelines.
- Basic ideas behind good programming methodologies and good programming etiquette.
Why are your programming habits SO IMPORTANT?

- We’ll talk about this over and over, so this is just a first assault!
- Programming done poorly is almost worthless:
  - You won’t be able to understand what you programmed just last week,
  - Others won’t be able to understand what you tried to accomplish,
  - And neither you nor anyone else can FIX your bad code. So ......
  - The time to develop good habits is NOW!
Outlining and Sentence Diagrams

- Remember when your English teacher...(Here it comes- this is one of those “when I was younger lectures...!”)

- So, here are my notes for what I want to tell you:
  - Planning is important...to?
    - You and the people you interact with!
  - Planning saves time...why?
    - Outcomes trump effort
  - Planning is not easy...why?
    - Requires crystal clear thinking (computers only know 0/1)
    - Requires re-thinking
    - Sometimes requires throwing stuff away!
  - Planning can be irritating
    - Not making progress!
What does this (planning) mean vis a vis programming?

• Think “top-down”
  ◦ Design the program before you code.
  ◦ Break the problem down into small steps
  1. State the problem clearly.
  2. Define the inputs and outputs
  3. Describe the algorithm:
    • Psuedocode, flow charts, or even comments!
  4. Translate the steps to SAS code
  5. TEST EACH STEP on a small version
Cool web sites

- Many
  - Wikipedia
  - Google or Bing:
    - Programming practices
    - Software design
    - Software design tutorials
- Also see: “Design Patterns: Elements of Reusable Object-Oriented Software” by Gamma, Helm, Johnson, and Vlissides.
- If you are really interested, I recommend taking programming 101 in computer science
  - undergraduate class
  - many online courses (coursera)
  - Sometimes class for non-majors
Software Design Techniques

Software Design Patterns for Information Visualization
Jeffrey Heer, Maneesh Agrawala
PDF (2.0M)
Now, let’s discuss some of the more mundane, but essential, aspects of good programming.
Pitfalls, variable names

- Properly formed variable names are
  - Easy to read and meaningful
    - *xyr* versus *dblXYCoordinateRatio*
  - Adhere to a naming convention
    - For example ‘int’ prefix for integer variables.
  - Begin with a letter and can contain letters, numbers and ‘_’. In particular, no spaces are allowed.
Pitfalls, operator precedence rules

- Make sure you know what has precedence

```
>> 1/2+3       >> 1/(2+3)
= 3.5000      = 0.2000
>> 2^2^3
= 64
>> (2^2)^3
= 64
>> 2^(2^3)
= 256
```

- When in doubt, use parentheses.
- TEST it out
Pitfalls, ordering of arguments

- Functions (see later lectures) may have many input arguments. Their order will matter, just as

\[ \frac{a}{b} \neq \frac{b}{a} \]
Programming Etiquette
Readable Programs

- **Whitespace**
  - Grouping
  - Indentation
    - to show control flow

- **Documentation**
  - Naming
  - Comments

- **Modular Code**
  - Break large blocks into smaller pieces
  - Use sub-routines or functions (more later)

Write programs for people first, computers second. -- Steve McConnell

Will you be able to read and understand your own code six months from now?
Whitespace

Use **indentation** to show logical structure

Which script is more readable?

```
x = 3; if x < 3 then y = 3; else y = 5;
```

or

```
x = 3;
if x < 3 then y = 3;
else y = 5;
```
Use meaningful names

Which is more readable?

```plaintext
xx = yyy( x );
xxx = PinkFlamingo( xx );
x4 = max(find(xxx)~=0);
floyd = x4.balance;
```

or

```plaintext
currID = CustomerID( custName );
currAccounts = BankAcct( currID );
mainAcct = max(find(currAccounts)~=0);
currBalance = mainAcct.balance;
```
Documentation

use comments to clarify meaning

- The first comment at the beginning of the script or function should describe what the script or function does.
- Approximately one comment per group of commands is about right.
- Avoid comments which just repeat what the associated code does.
- Use comments to document tricky code
- Use comments to give credits
- Did you see what google did on the csv file?
Google Flu

http://www.google.org/flutrends/us/data.txt

Google Flu Trends - United States
Copyright 2015 Google Inc.

Exported data may be used for any purpose, subject to the Google Terms of Service (http://www.google.com/accounts/TOS?hl=en_US). If you choose to use the data, please attribute it to Google as follows: "Data Source: Google Flu Trends (http://www.google.org/flutrends)".

Each week begins on the Sunday (Pacific Time) indicated for the row. Data for the current week will be updated each day until Saturday (Pacific Time). Note: To open these files in a spreadsheet application, we recommend you save each text file as a CSV spreadsheet. For more information, please visit http://www.google.org/flutrends

CSV FILE
Basics of Programming: SAS

- data step
  - Row at a time
- proc step
  - Full table
- Libname: directory location (folder)
- 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
- (no longer available)
  - http://support.sas.com/onlinedoc/913/docMainpage.jsp