STEAM CLOWN™ PRODUCTIONS

TOP DOWN - BREAKING A PROBLEM DOWN

Putting a few Python structures together

Last Updated: Tuesday, February 12, 2019

STEAM CLOWN™ & Squeaky Hinge PRODUCTIONS

Objective, Overview & Introduction

- This lesson and Lab is to bring together the basic Python constructs, including:
  - Conditional IF statements
  - For loops
  - While Loops
- Using Top Down design flow to break a coding problem down

WHAT YOU WILL KNOW…

- Prior Knowledge
  - How to open and run Python on a Raspberry Pi or other device
  - Familiarity with Python constructs like if, elif, else, while, for loops
  - Debugging skills to break down a python coding challenge

- What You Will Know & Be Able To Do
  - Use your Debugging skill to construct a top down flowchart to describe the python coding challenge
  - Impalement Python code to solve the coding challenge
  - Describe to classmates how you solved the coding challenge

STEAM CLOWN™ & Squeaky Hinge PRODUCTIONS

STEAM CLOWN™ PRODUCTIONS

Attribution-NonCommercial-ShareAlike 4.0 International (CC BY-NC-SA 4.0)

See Appendix A,B,C, for Licensing & Attribution information

These slides are an adaption, to better target my SVCTE High School Mechatronics Engineering class, primarily from Dr. Charles R. Severance's Python for Everybody class https://www.py4e.com/ … but from other sources as well. See Appendix A

CC BY-NC-SA 4.0
https://creativecommons.org/licenses/by-nc-sa/4.0/
https://creativecommons.org/licenses/by-nc-sa/4.0/legalcode

GNU Public License
Any included Programming Code Is licensed under the GNU General Public License v3.0

EUPL (European Union Public Licence) Code and Content is also licensed under the EUPL 1.2 or later

STEAM CLOWN™ & Squeaky Hinge PRODUCTIONS

How will you be measured

- Individual Students will submit working code
- Students teams will present diagram of Top Down design flow chart, and this will be graded
- Students teams will present orally how they solved the coding challenge, and depth of understanding will be graded
- Success will be determined by how well your code runs as checked by the instructor after you have turned in your Lastname-Firstname-ProgramName.py text files

STEAM Clown™ & Squeaky Hinge Productions

New Words or Concepts...

- Top Down
- Sudo Code

Where can I run my Python code?

- The main way we will implement Python code will be by running it on a Raspberry Pi, using the Linux command terminal shell, or the Idle3 Python interpreter
- If you don't have a Raspberry Pi, or if you don't have Python installed, there are a few Python interpreters online. This lets you try code with out having to install Python on your own PC or physically have a Raspberry Pi or other hardware. Here are a few. If you find a better one, please let me know
  - Python 3 On-Line Python Interpreter - Tutorials Point
    - Python 2.7 On-Line Python Interpreter - Tutorials Point
  - Python Interpreter - Online GDB
  - Python Shell - Python.org

STEAM Clown™ & Squeaky Hinge Productions

I GOT THIS... CAN I JUMP AHEAD?

- Jump Ahead and do the labs, save them and turn them in (show me and turn in later)
- Still need something to do? Try writing your own program or try this Extra Credit (see below) (show me and turn in later)

TOP DOWN

- Learning to program means learning how to solve problems using code
- Conceptually it is not very difficult to write a program that solves a problem that you can solve yourself
- The skill you need to acquire is thinking very precisely about how you solve the problem and breaking it down into steps that are so simple that a computer can execute them

Source for next few slides - https://adriann.github.io/programming_problems.html

Small Chunks Make Small Errors

- To make good progress in your programming task, you need to test your work as early and as thoroughly as possible.
- Everybody makes mistakes while programming and finding mistakes in programs consumes a very large part of a programmer's work-day.
- Finding a problem in a small and easy piece of code is much simpler than trying to spot it in a large program.
- Test each sub task you identified during your task-breakdown by itself.

STEAM Clown™ & Squeaky Hinge Productions

Putting small code together...

- Only after you're confident that each part works as you expect you can attempt to plug them together.
- Make sure you test the complete program as well, errors can creep in in the way the different parts interact.
- You should try to automate your tests. The easier it is to test your program, the freer you are in experimenting with changes.

STEAM Clown™ & Squeaky Hinge Productions

Top Down – Break Task Into Small Steps

- Solve a few instances of a problem by hand and think about what you did to find the solution
- If the task is finding the smallest number in a list, look at some short lists yourself
- A reasonable method would be to
  - Look at the first element of the list, write it down…
  - Then look at the next element and compare which is smaller
  - Cross out the larger one, save the smaller one
  - Repeat this process until you have looked at every element in the list

STEAM Clown™ & Squeaky Hinge Productions

FIND THE SMALLEST NUMBER IN A LIST

- Look at the first element of the list, write it down…
- Then look at the next element and compare which is smaller
- Cross out the larger one, save the smaller one
- Repeat this process until you have looked at every element in the list

~~[23, 9, 35, 99, 8, 7, 66, 2, 25, 4]~~

Save 23

(it's the smallest we have seen)

Smallest = 2

STEAM CLOWN™ & Squeaky Hinge PRODUCTIONS

Top Down - Teach the Computer

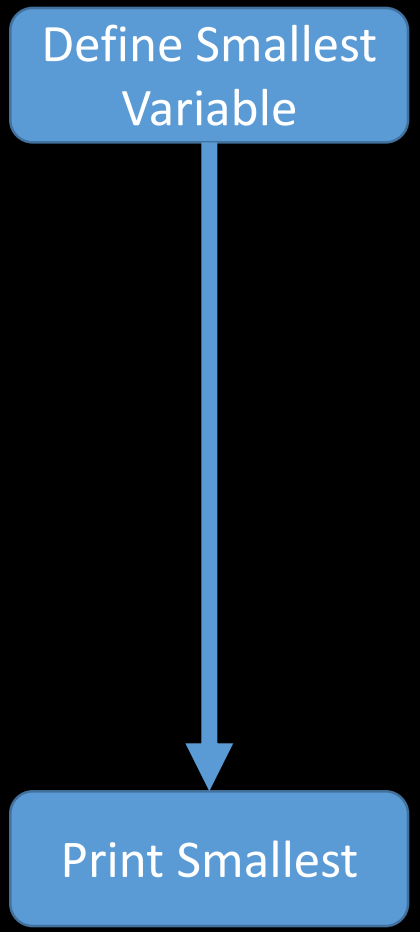
1. How to find the smallest element,
2. How to write it down,
3. How to cross it out, and wrap this in a loop.
4. Then continue this task breakdown process until you're confident you know how to write the necessary program.

STEAM Clown™ & Squeaky Hinge Productions

Break the Problem Down - Sudo Code

[23, 9, 35, 99, 8, 7, 66, 2, 25, 4]

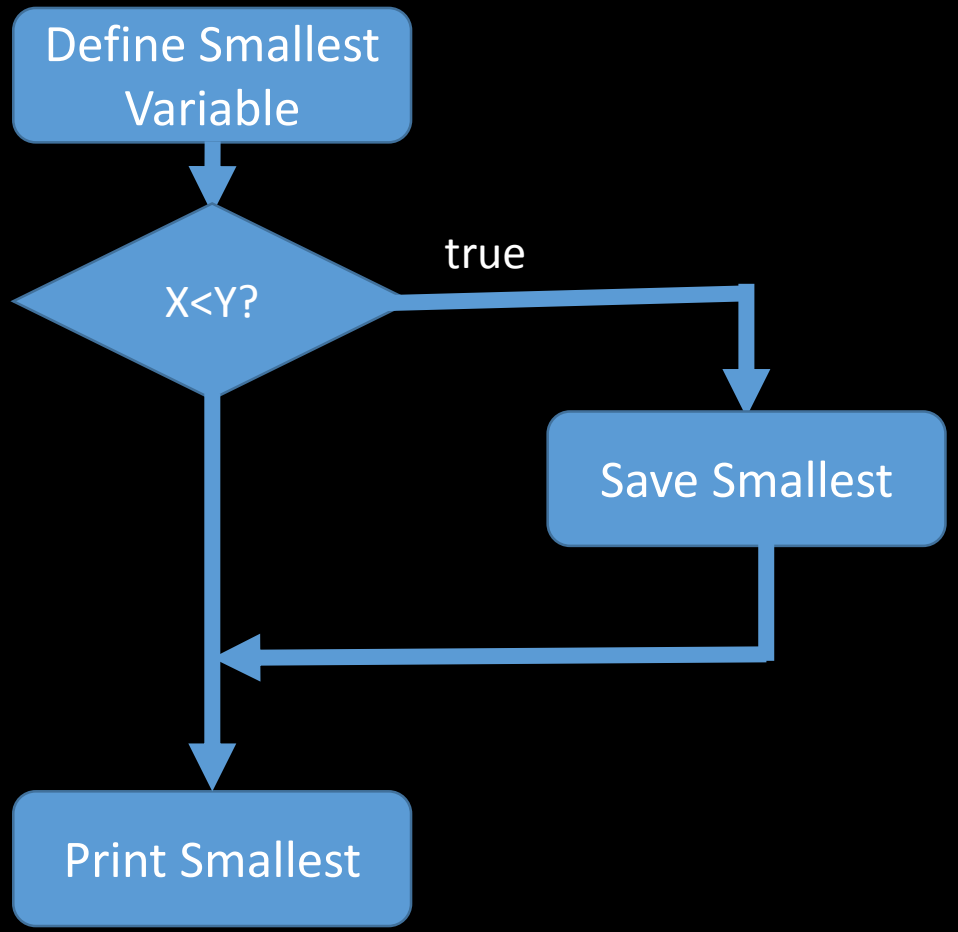
- What are we trying to doing?
  - Variable to hold "smallest"
  - Print smallest



Break the Problem Down - Sudo Code

[23, 9, 35, 99, 8, 7, 66, 2, 25, 4]

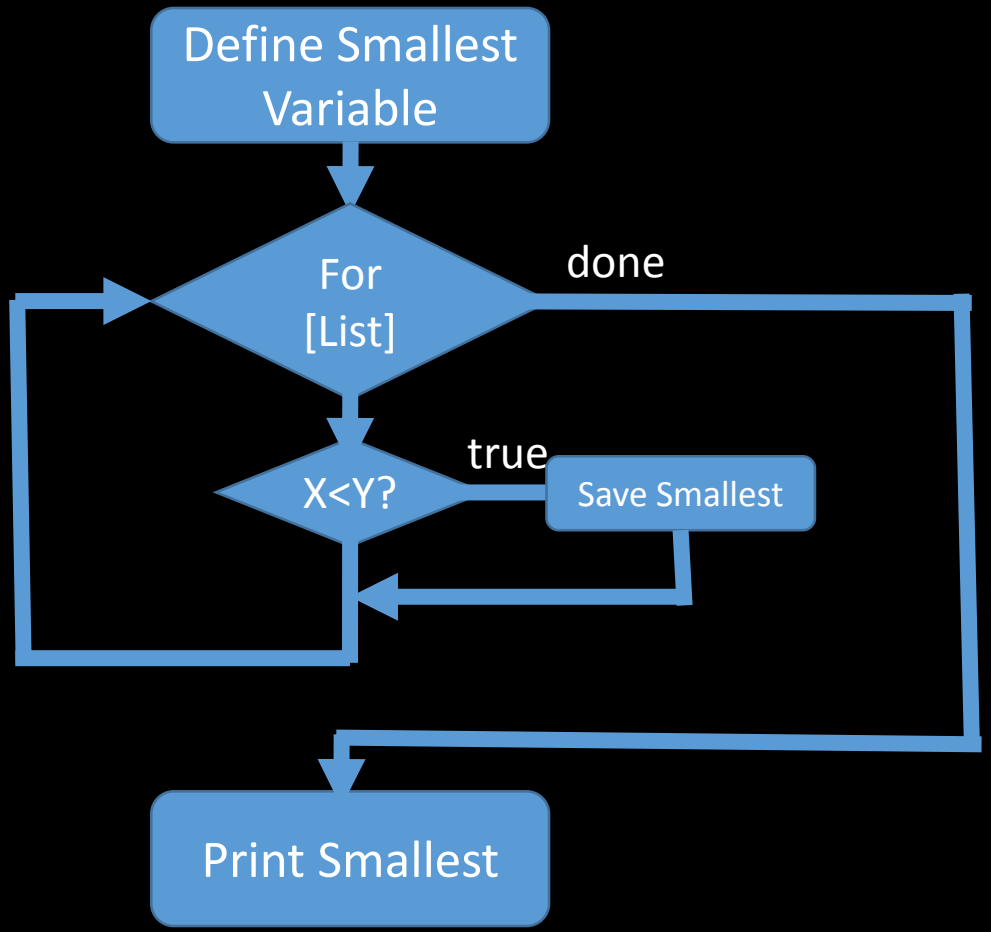
- What are we trying to doing?
  - Variable to hold "smallest"
  - Test x < y
  - Save & print smallest



Break the Problem Down - Sudo Code

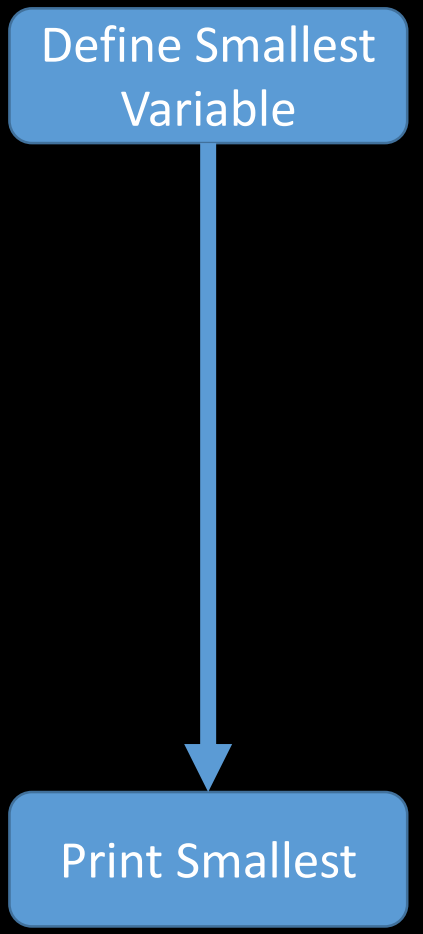
[23, 9, 35, 99, 8, 7, 66, 2, 25, 4]

- Now what are we trying to doing?
  - Loop through the list
    - Test x < y
    - Save Smallest
  - Print Smallest at the end



Code - Let's Write It...

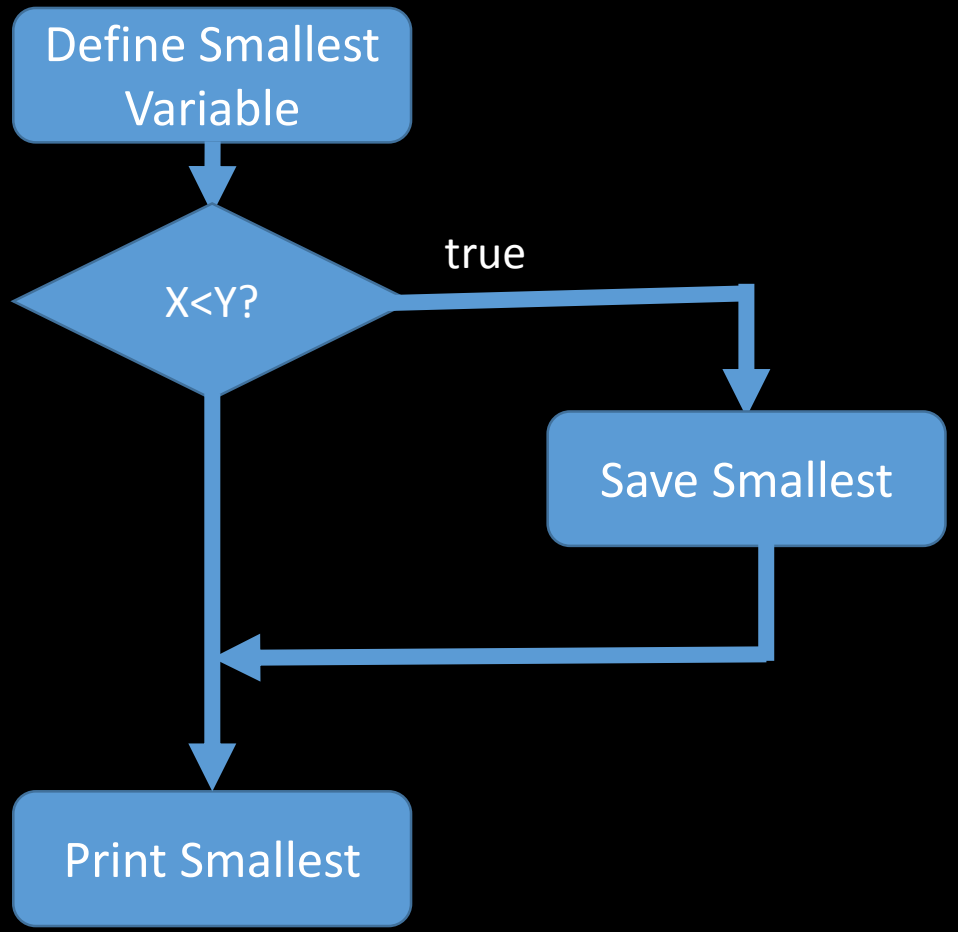
```
s = 1000
print('The Smallest Number is ', s)
```



CODE - TEST S>N?

[23, 9, 35, 99, 8, 7, 66, 2, 25, 4]

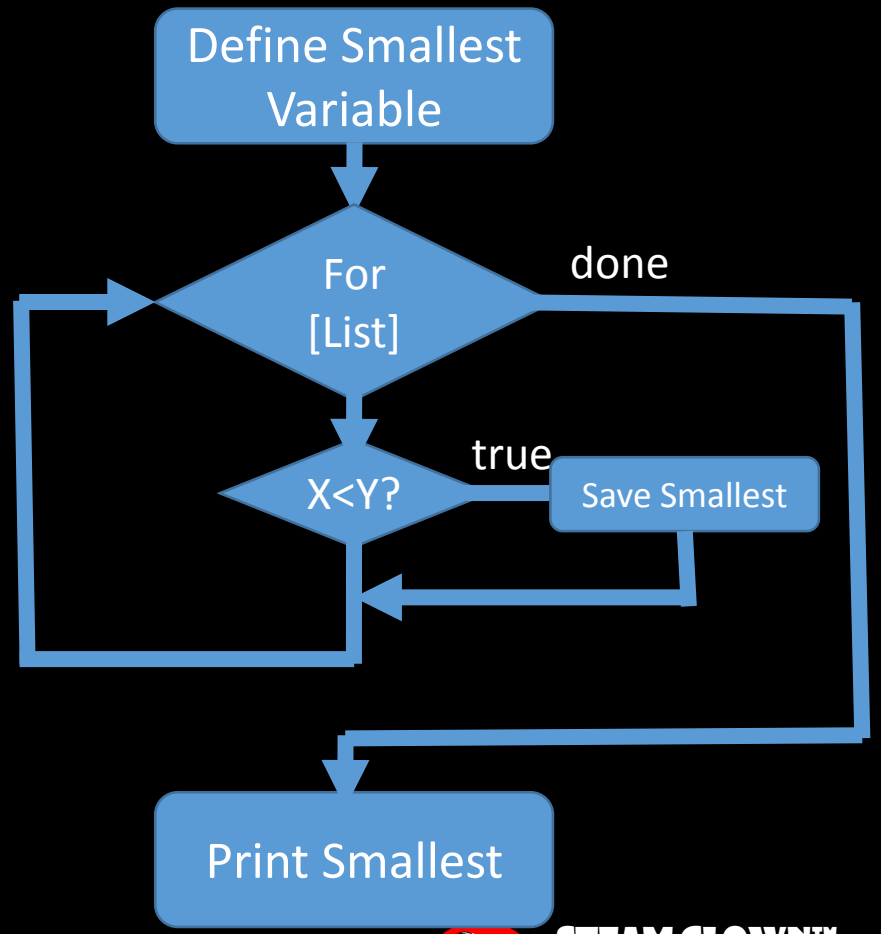
```
s = 1000
n = 23
print(s, n)
if s > n:
    s = n
    print(s)
print('The Smallest Number is ', s)
```



CODE - LOOP OUR LIST

[23, 9, 35, 99, 8, 7, 66, 2, 25, 4]

```
s = 1000
x = [23, 9, 35, 99, 8, 7, 66, 2, 25, 4]
for n in x:
    if s > n:
        s = n
        print(s)
print('The Smallest Number is ', s)
```



STEAM CLOWN™ & Squeaky Hinge PRODUCTIONS

Express your thoughts as code?

- *"How"* you express your thoughts as code
- In the same way that you can express the same argument in different ways in a normal English essay, you can express the same problem-solving method in different ways in code
- Try for brevity. The lines that you don't write are the lines where you can be sure that the don't have bugs
- Don't be afraid to Google for idiomatic ways of doing the things you'd like to do (after you tried doing them yourself!)

STEAM CLOWN™ & Squeaky Hinge PRODUCTIONS

Best code? Comments?

- Remember that you don't write the program for the computer, you write it for other humans (maybe a future you!)
- Choose names that explain things, add comments where these names don't suffice
- Never comment on *what* the code is doing, only write comments that explain *why*

STEAM CLOWN™ & Squeaky Hinge PRODUCTIONS

Picking Variable Names?

- Better naming and a better task breakdown make the comments obsolete.
- Revise your code just as you would revise an essay. Sketch, write, delete, reformulate, ask others what they think.
- Repeat until only the crispest possible expression of your idea remains.
- Revisit code you've written a while ago to see whether you can improve it with things you've learned since.

STEAM Clown™ & Squeaky Hinge Productions

LAB #1 – SUDO CODE, COMMENTS, VARIABLES

- Add Sudo Code, Comments, better Variable names and more Print statements to tell me what is going on
- Turn in Sudo Code comment AND Python code in a program named lastName-FirstName-Smallest.py

```
s = 1000
x = [23, 9, 35, 99, 8, 7, 66, 2, 25, 4]
for n in x:
    if s > n:
        s = n
        print(s)
print('The Smallest Number is ', s)
```

STEAM CLOWN™ & Squeaky Hinge PRODUCTIONS

Lab #2 – Write Sudo Code, Draw Flow Chart, Write Code

- Write a program that prints a multiplication table for numbers up to 12
  - First write a sudo code (English step by step) as comments at the top of a program file
  - Draw Flow Chart in your note book
  - Write Code, Test it, Run it
- Turn in Sudo Code comment AND Python code in a program named lastName-FirstName-Mult.py

STEAM Clown™ & Squeaky Hinge Productions

Extra Credit- Count Up/Down

- Challenge 1: - Up / Down Counter: Write a standalone Python program that will count from 0 to 25 and then back down to 0. The output should print to the console the following:

```
Starting to count up
0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25
Starting to count down
25,24,23,22,21,20,19,18,17,16,15,14,13,12,11,10,9,8,7,6,5,4,3,2,1,0
Done
```

- Before you run off to Google to find code that is already built to do this, please take some time to solve this on your own.

Hint: - You will need a variable to hold the countValue. I expect this program to use a while loop, and a few If /else statements to test what the current value to the counterValue is. You will need to test the counterValue to see when it should be counting up and when it should be counting down.

STEAM CLOWN™ & Squeaky Hinge PRODUCTIONS

REFERENCE SLIDES

STEAM CLOWN™
& Squeaky Hinge
PRODUCTIONS

APPENDIX

STEAM CLOWN™ PRODUCTIONS

CAN I GET A COPY OF THESE SLIDES? YES, PROBABLY…

Most presentation lecture slides can be found indexed on www.steamclown.org and maybe blogged about here on Jim The STEAM Clown's Blog, and on STEAM Clown's Mechatronics Engineering Google site, where you can search for the presentation title. While you are there, sign up for email updates

STEAM CLOWN™ & Squeaky Hinge PRODUCTIONS

Appendix A: License & Attribution

- This interpretation is primarily the Intellectual Property of Jim Burnham, Top STEAM Clown, at STEAMClown.org
- This presentation and content is distributed under the Creative Commons License CC-BY-NC-SA 4.0
- My best attempt to properly attribute, or reference any other sources or work I have used are listed in Appendix C

Please maintain this slide with any modifications you make

STEAM CLOWN™ & Squeaky Hinge PRODUCTIONS

Appendix B: Code License & Attribution

- This interpretation is primarily the Intellectual Property of Jim Burnham, Top STEAM Clown, at STEAMClown.org
- The programming code found in this presentation or linked to on my Github site is distributed under the:
  - GNU General Public License v3.0
  - European Union Public Licence EUPL 1.2 or later
- My best attempt to properly attribute, or reference any other sources or work I have used are listed in Appendix C

Please maintain this slide with any modifications you make

Appendix C: Primary Sources & Attribution for Material Used

- Charles R. Severance slides can be found on the https://www.py4e.com/ site are Copyright 2010 - Charles R. Severance (www.dr-chuck.com) of the University of Michigan School of Information and made available under a Creative Commons Attribution 4.0 License. Please maintain this last slide in all copies of the document to comply with the attribution requirements of the license. If you make a change, feel free to add your name and organization to the list of contributors on this page as you republish the materials.
  - Initial Development: Charles Severance, University of Michigan School of Information
  - Modifications and Adaptions by Jim Burnham, Top Clown @ www.steamclown.org
- Adrian's Homepage - https://adriann.github.io/ & https://adriann.github.io/programming_problems.html

Please maintain this slide with any modifications you make

STEAM CLOWN™
& Squeaky Hinge
PRODUCTIONS