



STEAM CLOWN™ PRODUCTIONS

MY FIRST PYTHON PROGRAM

Last Updated: Tuesday, January 22, 2019



OBJECTIVE, OVERVIEW & INTRODUCTION

- Now that we have learned about the Python Shell, you will now put it all together and write a python program in a Python Editor, which you can then execute later.
- Using a Python Editor, you will implement a number of conditional statements, including:
 - if, else, elif, while, for conditional structures
- You will have an opportunity to show your coding skill by turning in a number of Python labs. You will be measured on how well you implement these labs

WHAT YOU WILL KNOW...

- Prior Knowledge & Certifications
 - You should have an understanding of variable assignment, math, and string concatenation, and other basic Python language structures
- What You Will Know & Be Able To Do
 - You will be able to create, edit and save a Python program
 - You will be able to execute it from the command line as well as from the IDE
 - You will be able to describe and implement basic conditional Python structures like assigning variables, doing math, understanding the order of operations, and getting input data from the user using input



STEAM CLOWN™ PRODUCTIONS



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](#)

EURL (European Union Public Licence) Code and Content is also licensed under the [EURL 1.2 or later](#)



HOW WILL YOU BE MEASURED

- You will be measured based on your ability to code the labs listed in this presentation. You will need to create programming code that produces the expected output
- 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

NEW WORDS OR CONCEPTS...

- Operators (Math)
- Operator precedence
- Integer Division
- Conversion between types
- User input

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, you can execute your code on-line using a Python interpreter
 - [Python 3 On-Line Interpreter](#) - Tutorials Point
 - [Python Shell](#) – Python.org



I GOT THIS... CAN I JUMP AHEAD?

- Jump Ahead and do Lab #1, Lab #2, save them. (show me and turn in later)
- If you are still ready to move on, then also do Lab #3 and Lab #4 (show me and turn in later)
- Still need something to do? Try this Extra Credit Prime Number lab (show me and turn in later)



STEAM CLOWN™
& **Squeaky Hinge**
PRODUCTIONS

© Copyright 2018 STEAM Clown™

PYTHON ON THE COMMAND LINE

- Open a terminal

```
$ pwd
$ ls
$ cd myPython
$ pwd
$ ls
```
- We are now in your python code directory

```
pi@rasp
File Edit Tabs Help
pi@raspberrypi:~ $ pwd
/home/pi
pi@raspberrypi:~ $ ls
ClassLab Documents MagPi myPython
Desktop Downloads Music Pictures
pi@raspberrypi:~ $ cd myPython
pi@raspberrypi:~/myPython $ pwd
/home/pi/myPython
pi@raspberrypi:~/myPython $ ls
pi@raspberrypi:~/myPython $
```

CREATING (CAT) A PYTHON.PY FILE

- Use the **cat** command to create a file

```
~/myPython $ cat > firstPython.py
```

```
~/myPython $ ls
```

- We have now created a Python text file

```
pi@raspberrypi: ~/myPython
File Edit Tabs Help
pi@raspberrypi:~/myPython $ pwd
/home/pi/myPython
pi@raspberrypi:~/myPython $ cat > firstPython.py
# this is a comment, and this is my first Python Program
# This is another comment
print("Hello World")

^C
pi@raspberrypi:~/myPython $ ls
firstPython.py
pi@raspberrypi:~/myPython $
```



ECHOING (CAT) A PYTHON.PY FILE

- Use the **cat** command to echo a file (No **>**)

```
~/myPython $ cat firstPython.py
```

```
~/myPython $ ls
```

- We can “echo” a Python text file

```
pi@raspberrypi: ~/myPython
File Edit Tabs Help
pi@raspberrypi:~/myPython $ ls
firstPython.py
pi@raspberrypi:~/myPython $ cat firstPython.py
# this is a comment, and this is my first Python Program
# This is another comment
print("Hello World")

pi@raspberrypi:~/myPython $
```



RUNNING A PYTHON.PY FILE

- You need to run the python3 executable

```
~/myPython $ which python3
```

```
/usr/bin/python3
```

```
~/myPython $ python3 -V
```

```
Python 3.5.3
```

```
~/myPython $ python3 firstPython.py
```

```
Hello World
```

```
~/myPython $
```

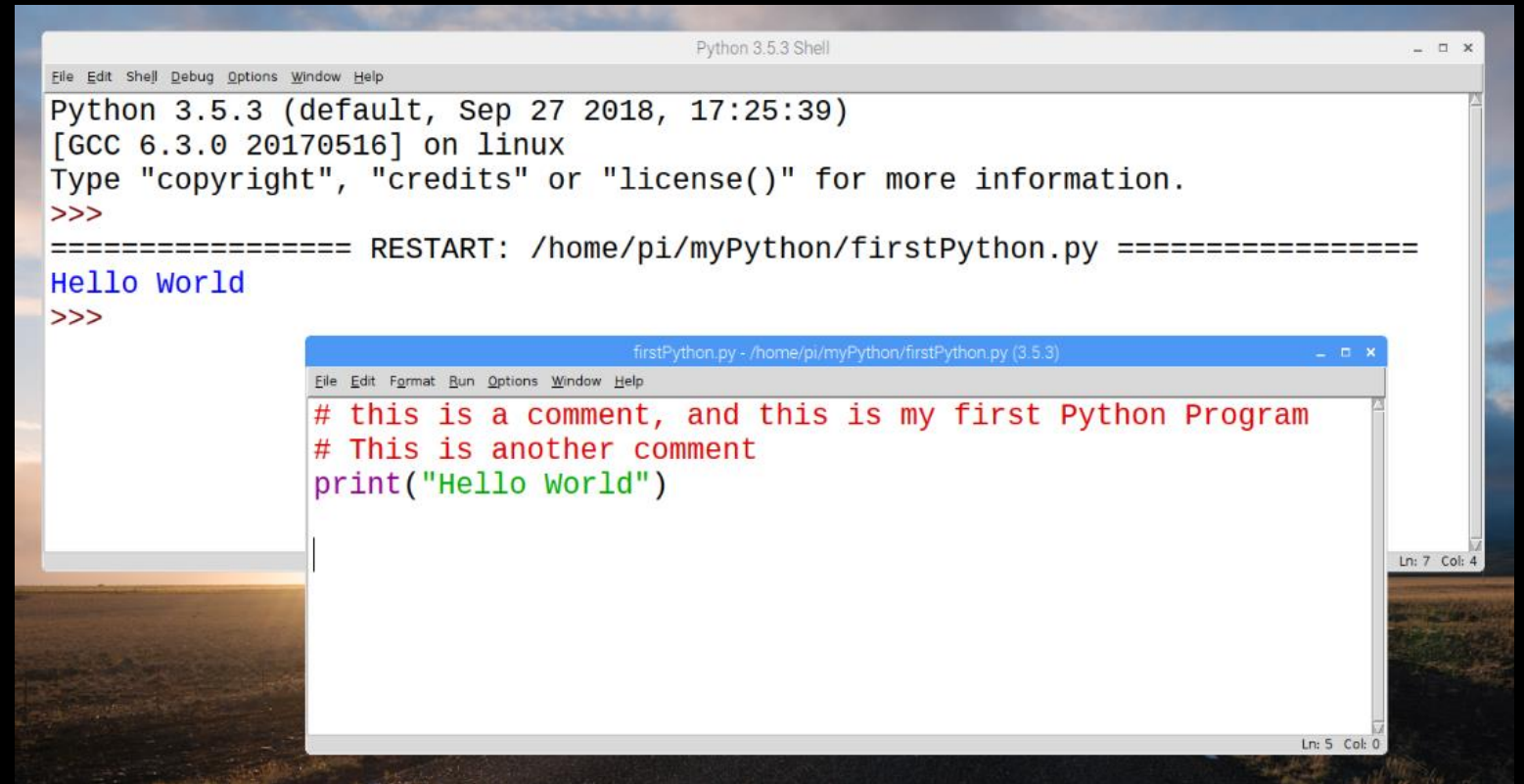


STEAM CLOWN™
& **Squeaky Hinge**
PRODUCTIONS

© Copyright 2018 STEAM Clown™

OPEN PYTHON FILE IN IDLE3

- Open Idle3
- File → Open → myPython → firstPython.py
- Run Module



The image shows two overlapping windows from the Python 3.5.3 Shell. The background window is the 'Python 3.5.3 Shell' window, which displays the following text:

```
Python 3.5.3 (default, Sep 27 2018, 17:25:39)
[GCC 6.3.0 20170516] on linux
Type "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: /home/pi/myPython/firstPython.py =====
Hello World
>>>
```

The foreground window is the 'firstPython.py - /home/pi/myPython/firstPython.py (3.5.3)' editor window, which displays the following code:

```
# this is a comment, and this is my first Python Program
# This is another comment
print("Hello World")
```

NUMERIC EXPRESSIONS

- Because of the lack of mathematical symbols on computer keyboards - we use “computer-speak” to express the classic math operations
- Asterisk is multiplication
- Exponentiation (raise to a power) looks different than in math

Operator	Operation
+	Addition
-	Subtraction
*	Multiplication
/	Division
**	Power
%	Remainder

MATH

- $+$, $-$, $*$, $/$, $**$ (power), $\%$ (modulo)
- $a=1+1$, where $a=2$
- $a=3-2$, where $a=1$
- $a=3*2$, where $a=6$
- $a=4/3$, where $a=1.33333$
- $a=4//3$, where $a=1$ ← Integer Math, so no remainder
- $a=4\%3$, where $a=1$ ← this is the remainder, $4/3 = 1R1$
- $a=3**4$, where $a=3*3*3*3=81$



ORDER OF EVALUATION

- When we string operators together - Python must know which one to do first
- This is called “operator precedence”
- Which operator “takes precedence” over the others?

x = 1 + 2 * 3 - 4 / 5 ** 6

OPERATOR PRECEDENCE RULES

Highest precedence rule to lowest precedence rule:

- Parentheses are always respected
- Exponentiation (raise to a power)
- Multiplication, Division, and Remainder
- Addition and Subtraction
- Left to right

Parenthesis
Power
Multiplication
Addition
Left to Right



STEAM CLOWN™
& **Squeaky Hinge**
PRODUCTIONS

© Copyright 2018 STEAM Clown™

```
>>> x = 1 + 2 ** 3 / 4 * 5
```

```
>>> print(x)
```

```
11.0
```

```
>>>
```

Parenthesis
Power
Multiplication
Addition
Left to Right



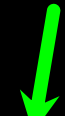
1 + 2 ** 3 / 4 * 5



1 + 8 / 4 * 5



1 + 2 * 5



1 + 10



11



STEAM CLOWN™
& **Squeaky Hinge**
PRODUCTIONS

© Copyright 2018 STEAM Clown™

OPERATOR PRECEDENCE

Parenthesis
Power
Multiplication
Addition
Left to Right



- Remember the rules top to bottom
- When writing code - use parentheses
- When writing code - keep mathematical expressions simple enough that they are easy to understand
- Break long series of mathematical operations up to make them more clear



STEAM CLOWN™
& **Squeaky Hinge**
PRODUCTIONS

© Copyright 2018 STEAM Clown™

LAB #1 - MAKE SOME EDITS...

- Create a **variable1** = 7
- Create another **variable2** = 5
- Create another **variable3** and assign it **variable1** plus **variable2**
- Print **variable3**
- Run it... raise your hand and let me check your results



LAB #2 - DO MORE MATH...

- Print the result of **variable1** times **variable2**
- Print the result of **variable1** divided by **variable2**
 - Is it right?
 - What is the type of the result?
 - Now force it to do integer division
- Assign **variable3** the result of **variable1** modulo **variable2**
- Print **variable3**
- Run it... raise your hand and let me check your results



USER INPUT

- We can instruct Python to pause and read data from the user using the `input()` function
- The `input()` function returns a string

```
nam = input('Who are you? ')\nprint('Welcome', nam)
```

Who are you? **Chuck**
Welcome Chuck



STEAM CLOWN™
& **Squeaky Hinge**
PRODUCTIONS

© Copyright 2018 STEAM Clown™

USER INPUT TYPE

```
nam = input('Who are you? ')
print('Welcome', nam)
```

Who are you? **Chuck**

Welcome Chuck

What “type” is **nam**?

<class 'str'>

```
age = input('what is your lucky number? ')
print('Lucky Number is ', age)
```

What “type” is **age**?

<class 'str'>



STEAM CLOWN™
& **Squeaky Hinge**
PRODUCTIONS

© Copyright 2018 STEAM Clown™

HOW DO WE CHANGE A DATA TYPE?

```
age = int(input('what is your lucky number? '))  
print('Lucky Number is ', age)
```

Now what “type” is `age`?
<class 'int'>

- If we want to read a number from the user, we must convert it from a string to a number using a type conversion function

```
inp = input('Europe floor?')  
usf = int(inp) + 1  
print('US floor', usf)
```

- Later we will deal with bad input data

Europe floor? 0
US floor 1



STEAM CLOWN™
& **Squeaky Hinge**
PRODUCTIONS

© Copyright 2018 STEAM Clown™

LAB #3 - GETTING DATA FROM THE USER

- GREETING

- Write a program to prompt the user for their name, and then prints a greeting.

What is your name: Jim

Output this text

Hey, Jim, How's it going?



STEAM CLOWN™
& **Squeaky Hinge**
PRODUCTIONS

© Copyright 2018 STEAM Clown™

LAB #4 - GETTING DATA FROM THE USER

- WEEKLY PAY CHECK

- Write a program to prompt the user for their name, hours and rate per hour to compute gross pay.

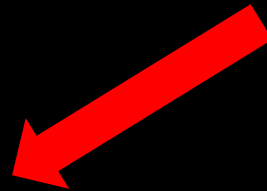
What is your name: Jim

Enter Hours: 35

Enter Rate: 2.75

Jim, you made:
\$96.25 this week.

Output this text



STEAM CLOWN™
& **Squeaky Hinge**
PRODUCTIONS

© Copyright 2018 STEAM Clown™

SUMMARY

- Operators (Math)
- Operator precedence
- Integer Division
- Conversion between types
- User input



STEAM CLOWN™ PRODUCTIONS

REFERENCE SLIDES



STEAM CLOWN™
& **Squeaky Hinge**
PRODUCTIONS

© Copyright 2018 STEAM Clown™



STEAM CLOWN™ PRODUCTIONS

APPENDIX



STEAM CLOWN™
& **Squeaky Hinge**
PRODUCTIONS

© Copyright 2018 STEAM Clown™



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



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



Under the following terms:



Attribution — You must give [appropriate credit](#), provide a link to the license, and [indicate if changes were made](#). You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.



NonCommercial — You may not use the material for [commercial purposes](#).



ShareAlike — If you remix, transform, or build upon the material, you must distribute your contributions under the [same license](#) as the original.

No additional restrictions — You may not apply legal terms or [technological measures](#) that legally restrict others from doing anything the license permits.

Please maintain this slide with any modifications you make

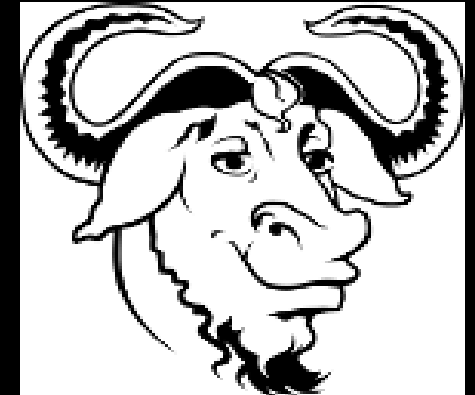


STEAM CLOWN™
& **Squeaky Hinge**
PRODUCTIONS

© Copyright 2018 STEAM Clown™

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



Please maintain this slide with any modifications you make



STEAM CLOWN™
& **Squeaky Hinge**
PRODUCTIONS

© Copyright 2018 STEAM Clown™

APPENDIX C: PRIMARY SOURCES & ATTRIBUTION FOR MATERIAL USED

Please maintain this slide with any modifications you make



STEAM CLOWN™
& **Squeaky Hinge**
PRODUCTIONS

© Copyright 2018 STEAM Clown™



STEAM CLOWN™
& **Squeaky Hinge**
PRODUCTIONS

© Copyright 2018 STEAM Clown™