CS 1428
Fall 2018
Gentry Atkinson
Lab 11

Introduction:

You have previously been introduced to BASH and shell scripting in Lab 5. There are many other scripting languages and each one is meant to serve some purpose. BASH was very useful for automating tasks and administrating Linux systems. Another popular scripting language is Python, which was first introduced in 1991 and heavily emphasizes readability and dynamic data typing.

Python executes more slowly than a compiled C++ program but for small tasks it can be much easier to write a small Python script than it is to write and compile a C++ program. In addition to ease of use, Python also makes it much easier to incorporate external libraries than C++ does. Because of these two facts Python has become very popular for data processing and analysis. It is also very useful for algorithm prototyping, machine learning, and deep learning.

This purpose of today's lab is to introduce you to the Python scripting language.

Directions:

1- Open Notepad++ (or any text editor) and create a new file. Name it yourLastName_lab11.py. If you are using Notepad++, select Language->P->Python to give you the correct syntax highlighting.

2- Copy the following as the first line of the file:

```
#! python
```

Remember that your shell script from Lab 5 had a similar opening line.

3- Python does not require you to declare a variable or define its data type before using it. You are also not required to end every line with a semicolon and should not. Experiment with this by copying the following line:

```
phrase = "This is lab 11."
print(phrase)
```

4- Type **cmd** into program search bar at the bottom left of your screen to bring up a command prompt (or however your most comfortable bringing up a command prompt). Use the **cd** command to change to the directory where you've saved your program. Type the following command to run your script:

> **python myLastName_lab11.py**

This should show you the output of what you've written so far. Notice that there's no compiling step. Python interprets your program one line at a time rather than converting all of the code to machine code first.

5- Creating array-like structures in python is similar to C++ but much simpler. Copy the following code to create a list of values:

> **myList = [1, 2, 3, 'a', 'b', 'c', 1.1, 2.2, 3.3]**

Notice that this one list can hold several data types.

6- Python makes iterating through lists much easier than what you're used to in C++ (although in all fairness, most scripting languages do). Copy the following code to print each value in your list:

> **for value in myList:**
> **print(value)**

The indentation is very important! Python uses indentation rather than curly brackets to separate blocks. Also notice that you can swap any word you want for 'value'. It's a placeholder for the values that you're extracting from the list. Finally, notice that these values are not necessarily stored in order.

7- Conditions in Python are accomplished **if** statements just like in C++. The syntax is only slightly different. Modify your for loop so that it looks like this:

> **for value in myList:**
> **if (isinstance(value, int)):**
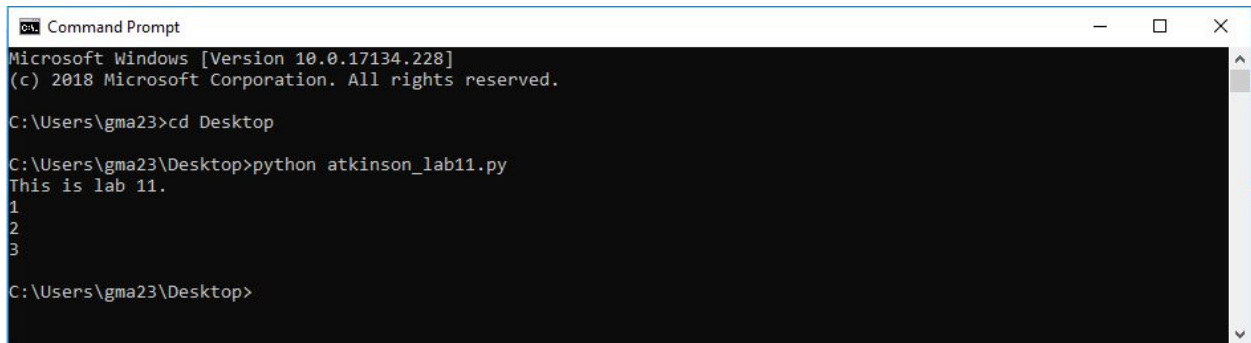> **print(value)**

Once again indentation is very important. Now the loop only prints the integer values.

8- Comments in Python begin with a '#' rather than a '//'. Add a comment to each line of your **for** loop explaining what action is being performed.

9- If you're interested in seeing some more of what you can do with Python you can try some of the following tutorials:
- https://www.w3schools.com/python/
- https://www.codecademy.com/learn/learn-python
- https://github.com/zhiwehu/Python-programming-exercises/

10- Run your program from the command line. Correct any errors. Your output should look something like this:

11- Submit your .py file through TRACS. You can leave when you're done.