Python

Python is a high level programming language which has become very popular in recent years. High level languages create an abstraction between the programmer and the computer. This means the language itself takes care of the small details, such as when and how memory is managed. Python code can be written to scripts that are then able to be used as needed. Scripts are simply programs that are written to text files then fed to an interpreter which figures out how to run them on the present hardware.

A command shell is also provided by python which allows the programer to interact with the interpreter, providing immediate feedback on individual commands. You have seen the use of shells before; `bash` is the shell you have been using in your terminal to issue commands. We will be using a program called `ipython` as our command shell because it has some added functionality which makes programming in python easier.

To start the python shell, open a terminal and type `ipython3` and you will see a new prompt appear. You may now issue commands to the inerpteter. Follow along with the below example and feel free try some of your own commands (the python shell is a great tool for double checking syntax when writing scripts). Note do not type the text following the pound sign (#) in the example below.

```
denver:~$ ipython3
Python 3.4.1 (default, Nov 3 2014, 14:38:10)
Type "copyright", "credits" or "license" for more information.
(more text)
In [1]:  # This is the python prompt, press enter to get another
In [2]:  print("Hello World") # Press enter to issue the command
Hello World
In [3]:  3 + 7
10
In [4]:  name = "myName" # Put your own name between the quotes
In [5]:  print(name) # Name is a variable that holds the string you gave
myName
In [6]:  name = name + " lastName" # We can add strings to each other
In [7]:  print(name)
myName lastName
In [8]:
```

Enter the following expressions and see how python evaluates them.

- `91 - 47`
- `9 * 7`
- `3 ** 4`
- `age = input("Enter your age:  ")`
  This should immediately prompt you to enter text. Enter a number and hit enter.
- `print(age * 2)`

- `age = int(age)`
- `print(age * 2)`
- `9 / 2`
- `int(9 / 2)`

Think of more commands to enter and see what happens. Don't worry about entering bad commands, `ipython` will just tell you there was an error and ask for a new command! When you are finished type `quit()` to exit `ipython`.

First Python Program

Figure 1 is an example python program that demonstrates a few important concepts.

```
#!/usr/bin/env python3.4

def greet():
  print("Hello, world!")
  name = input("Enter your name: ")
  if name == "Debbie":
    print("Hello, Debbie! I am glad you are in CS192")
  else:
    print("Hello ", name)

if __name__ == "__main__":
  greet()
```

Figure 1

We need to write this script to a text file so that we can ask python to run it. Enter the following commands into a terminal.

```
denver:~$ mkdir python_fun
denver:~$ cd python_fun
denver:~/python_fun$ touch greet.py
denver:~/python_fun$ chmod 700 greet.py
denver:~/python_fun$ gedit greet.py &
```

Now type the code from Figure 1 into `gedit` keeping note that the indentation in the code is manditory! An indentation level of two was used in the example; however, as long as you are consistent any level is fine (other common levels include four and eight). Using the tab button for your indentation tells your editor to move forward one level and will help you stay consistent.

Once you have finished writing the script we can run it! From you terminal type the following (make sure you are in your terminal and not ipython!).

```
denver:~/python_fun$ ./greet.py
Hello, world!
Enter your name:  my_name
Hello my_name
denver:~/python_fun$
```

We can also run our programs from inside of ipython, which is sometimes more convenient. Run the following commands from the terminal. Note that the introduction message will no longer be displayed in this document.

```
denver:~/python_fun$ ipython3
In [1]:  run greet.py
Hello, world!
Enter your name:  my_name
Hello my_name
In [2]:
```

From here on scripts will be run from inside of ipython in this document. If you wish to issue terminal commands make sure that you open a new terminal window, or quit ipython first.

Instructions

Now you will create your own function which will be very similar to the `greet` function already present. This new function will be created from scratch and will be placed in the `greet.py` file. You will also add a while loop to your program (you can find the syntax for a while loop by searching it on google)!

As you are writing new code to `greet.py` in `gedit` make sure to save often, go to your `ipython` session and run your program. Saving often and trying out little pieces at a time will make writing code easier.

Complete the following:

1. Add a new function named `ask_age`
2. Ask the user for their age and store it in a variable
3. If the age entered is nineteen print "Your age is prime!"
4. Call this function right below where `greet` is called
5. Search online for a python "while loop" to see the syntax for creating one
6. Add a `while` loop to your program to repeat some part of it
7. Make sure your program works correctly
8. Quit ipython by typing `quit()` then go to the assignment page for submission instructions