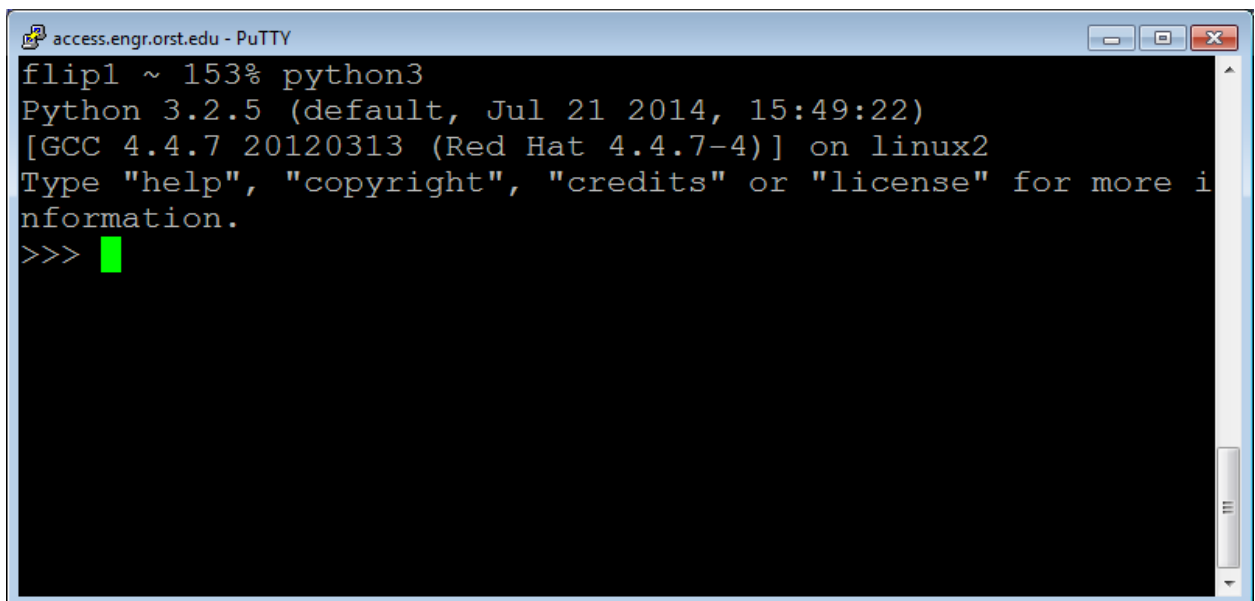


LAB #3

Programming w/ Python

(2 pts) Continue Programming in Python

To begin, we are going to execute a few python statements to understand the semantics and syntax before writing a full program. Let's enter the following examples for you to see what python syntax is. First, Python statements do not need to be ended by a semicolon, which is ignored if supplied. However, it might be a good habit to get into because other languages require it, and it will be required in CS 161!☺ Let's start the Python interpreter by just typing **python3** at the command prompt.



```
access.engr.orst.edu - PuTTY
flip1 ~ 153% python3
Python 3.2.5 (default, Jul 21 2014, 15:49:22)
[GCC 4.4.7 20120313 (Red Hat 4.4.7-4)] on linux2
Type "help", "copyright", "credits" or "license" for more i
nformation.
>>> █
```

Assign information to variables that store the information in memory to retrieve for later use. You can assign strings of text, whole numbers, and real numbers.

```
real_num= 19.4;
whole_num = 250;
message = "Hello Everyone!";
```

Now, print the values stored in these variables by using the variable name.

```
print(real_num);
print(whole_num);
print(message);
```

There are expressions that you can use to **change the values in the variables**. Let's change the values in the variables we just created.

```
real_num = real_num + 10.6;
whole_num = whole_num + 10;
message = message + "Python!";
```

Now, **print the values in these variables again** by using the variable name.

```
print(real_num);  
print(whole_num);  
print(message);
```

What if we want to **concatenate variables/information together**? We have to convert the numbers to strings to put them together with strings in a print statement. This is because the + symbol can also be used between numbers to mean addition. It depends on the context you are using the + symbol. We can temporarily change the type of a variable using typecasting. In this case we need the numbers to act as strings of text. We use a function called str() to do this.

```
print(str(real_num)+" "+str(whole_num)+" "+message+" pretty cool!");  
print(real_num+whole_num);
```

Next, we need to be able to **get information from the user**. Since we are reading something from the user, we need to be able to store it somewhere to be able to retrieve it later. Problem is that ALL the input comes in as strings of text, not a number. We can prevent this with a typecast to go from a string of text to a different kind of data, such as a real or whole number. In CS, we call these floating point and integer numbers, respectively. Therefore, we have to use the functions float() and int() to change the string to a real number or whole number.

```
real_num = input("Enter a different real number: ");
```

```
print(real_num);  
print(real_num+message);
```

```
real_num = float(input("Enter another number: "));
```

```
print(real_num);  
print(str(real_num)+message);
```

To exit, just type **exit()**;

Let's put it all together in an example program*. Open a new file by typing:

```
vim lab3.py
```

```
num = int(input("Enter a different num: "));  
print("Your number is: " + str(num));
```

```
num = num**10;  
print("Your new number to the 10th exponent: " + str(num));
```

Now run your program by typing **python3 lab3.py**.

***Remember:** Here is a Linux and vim cheat sheet to help you reference some of these commands quickly. <http://classes.engr.oregonstate.edu/eecs/fall2016/cs160-001/labs/CheatSheet.pdf>

At this time you can pair with someone for design or work alone!!!

(5 pts) Design a Program on Paper

Now, **write the steps needed to solve the following problem** on a piece of paper: The user provides the amount of coupons won as input and a program outputs how many candy bars and gumballs you can get. You can redeem 10 coupons for a candy bar or 3 coupons for a gumball, and you prefer candy bars to gumballs. Output how many candy bars and gumballs you get, when you spend all of your coupons on candy bars first and any remaining coupons on gumballs. **You are only allow to use what we have covered in class!**

When designing, remember to specify three things:

1. **How do you understand the problem?** What are the inputs, outputs, and expectations for the problem?
2. **Write the steps needed to solve the problem?** What are the logical steps you need to take to get from the inputs to the expected outputs?
3. **What do you do to test your program/solution to make sure it is correct?** Write the input given to your program to test to see if it is correct, and the output from the given input. What are the BAD values that might be entered by the user? What would you hope to happen with these bad and good values? Here is an **example of a few test cases** for this program:

Testing Input:	Expected Output:	Actual Output:
0	0	0
1	0	0
...		

(3 pts) Now, trade designs with someone/a group to see how well they can follow your logic!!!

Write the program in Python and Test someone else's design...

Now, let's write the python syntax for the solution designed above. Open a new file by typing: **vim candy.py**

Was the design and test cases adequate enough for you to follow?

- What was good about the design?
- What could use improvement?
- How thorough are the test cases?
- What are the actual outputs from the BAD test cases?
- Do the actual outputs match the expected outputs?