**Python**

**Exercises**

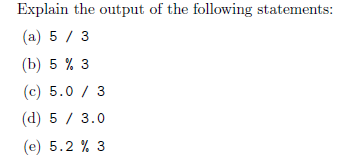
The code must be submitted under your name in GitHub in a repository called Python. Work individually.

Each file will have the name: exerciseX.py where X is the exercise number. You will have 13 files at most.

Do not commit code that does not compile. The code that you commit should have been tested. -10 points for code that does not compile on the top of your grade.

You will provide a hardcopy with your code to Dr. Scharff on 12/6.

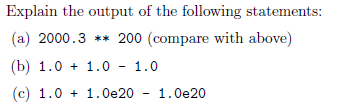
**Exercise 1**



Explanations:

1. The answer is an integer with everything after the decimal dropped off. In order to keep the decimal, one of the values has to be a float.
2. The answer is 2 because it is using the modulus function. This returns anything that remains that is not a multiple of 3.
3. This returns 1.66 repeating because it is returning a float as an answer because one of the values had a decimal in it. This means that a float type number will be returned.
4. The same exact thing happens here because one of the values has a decimal in it.
5. 2.2 is returned because that is what was left over after 3 was taken out of 5.2 and 2.2 is too small for another 3 to be taken out of. It is modulus with floats.

**Exercise 2**



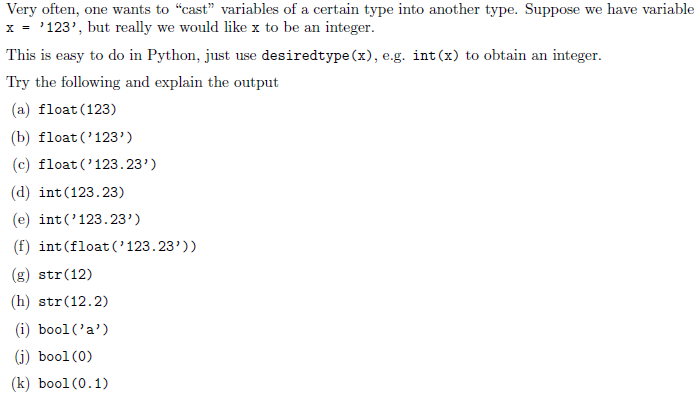
Explanations:

a) After using two different compilers, I get the same ‘Numerical Result out of Range’

b) The result is a float with order of operations occurring from left to right. The addition occurs before the subtraction because they are both the same precedence.

c) This is a special case because the 1.0e20 is such a large number that adding 1 to it is not something that can be represented with decimals. This means that the number does not actually get added to the memory address. After doing some tests, the smallest number that can be added to the value is 10000. This will result in 1.0000000000000002e+20. This is the furthest that python will go in storing a float. Anything smaller will not be recorded. So what happens in this example is that 1.0 gets added to 1.0e20, but the value remains 1.0e20 since adding a one cannot be represented. After this, 1.0e20 is reduced by 1.0e20 which causes the value to go to 0.0.

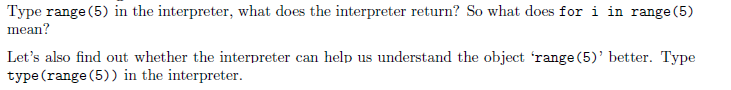
**Exercise 3**



Explanations:

1. 123 is returned as 123.0, which is a float that has a decimal.
2. 123 is inputted as a String and then returned as a float.
3. 123.23 is inputted as a String and then returned as a float.
4. 123.23 is inputted as a float and then converted to an int which means that decimal is dropped off.
5. 123.23 is inputted as a String, but is not able to be converted as an int because it represents a float.
6. 123.23 is inputted as a String, then converted to a float, and then finally converted into an int.
7. Int 12 is inputted and a String is the result.
8. float 12.1 is inputted and a String is the result.
9. String a is the input and a bool of True is the result.
10. Int 0 is the input and a bool of False is the result.
11. Float 0.1 is the input and bool of True is the result.

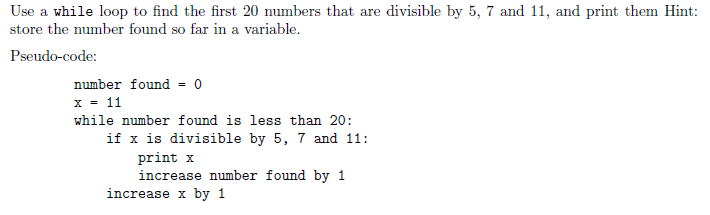
**Exercise 4**

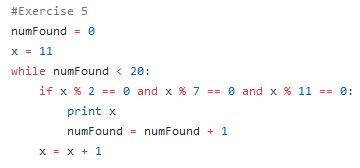
**C:\Users\js17500n\Desktop\ex4.JPG**

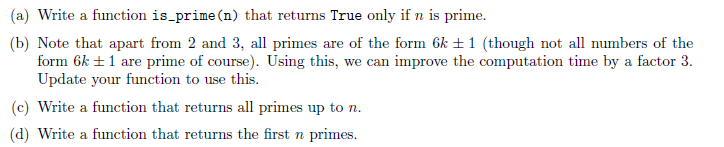
Explanations:

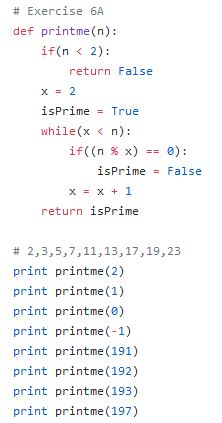
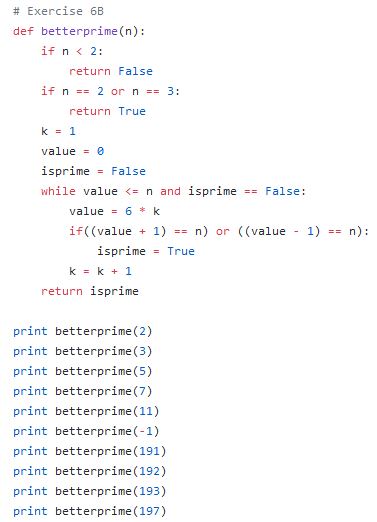
1. The interpreter returns a range of 0 through 5, so “for i in range(5)” means to iterate from 0 to 4.
2. The interpreter returns a class called range.

**Exercise 5**

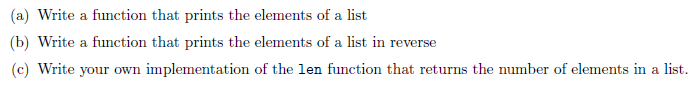


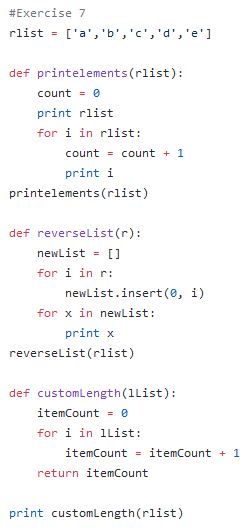
****

**Exercise 6**

****

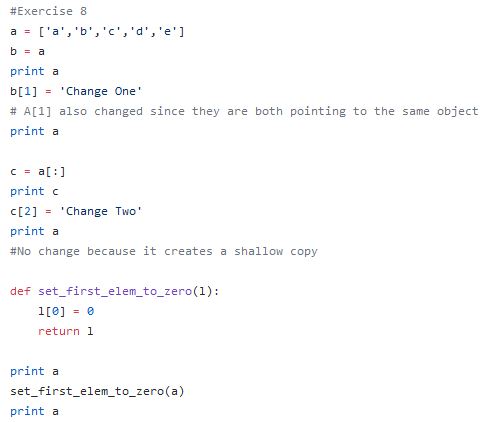
**Exercise 7**



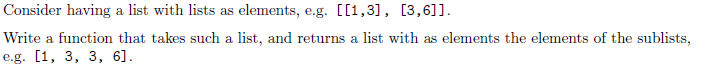
****

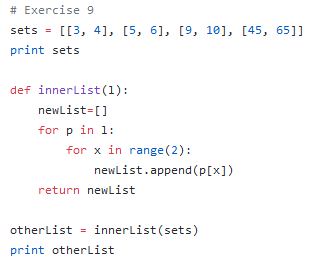
**Exercise 8**

  
#Original list is changed because it is passed in and then returned. A new copy is never made.

****

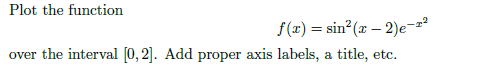
**Exercise 9**

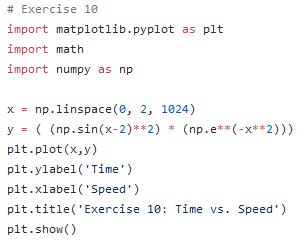


****

**Exercise 10**

Use mathplotlib

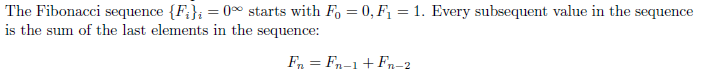


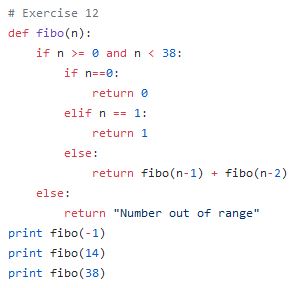
****

**Exercise 11**



**Exercise 12**

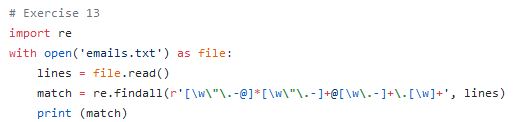


****

**Exercise 13**

Write a Python program that extracts the email addresses of a file. An email file emails.txt is provided to test your program.

<http://rubular.com/> is a site that can be useful to get familiar with regular expressions.

****

**References**

Stanford courses on Python <https://web.stanford.edu/~schmit/cme193/exercises.html>