Tutorial MATH 1MP3 – February 2, 2017

The following questions are to be done in groups of two or three.

- Open PyCharm and create a new project (File -> New Project, make sure the drop down box in "interpreter" points to your Anaconda installation!)
- In this project create a new python file (File -> New, then click "Python File" in the menu that appears)
- Write the code for the following questions in that python file. Just have each question answer follow the previous one in the code. (I recommend doing each question one at a time, though)

For the following questions, write a python program (in PyCharm or otherwise) that prints the answer.

For all the following questions your answers should be python functions that work on any input, not just the example inputs given!

Don't forget docstrings!!

1)
   a) Write a python function called `read_file` that reads in the file lipsum.txt (Found at http://beastman.ca/math1mp/lipsum.txt) and prints out all the text in it.
   b) Modify the function so it prints out all the text line-by-line
   c) Modify the function so returns a list where every element in the list is a line from the text
2) Write a function called `multiples` that takes two arguments, a number `m` and a number `n`. The function should return a list that starts with `m` and returns all multiples of `m` less than or equal to `n`.

```
multiples(3, 15) == [3, 6, 9, 12, 15]
multiples(2, 9) == [2,4,6,8]
```

3) This question works through floating point numbers. "Floating point" is computer speak for decimal. Real decimal numbers are infinite (pi, e, 0.33333....., etc.) but computers are finite. Because of this problem of representing an infinite thing as a finite thing, we run into all kinds of strange behaviour. The following code demonstrates that!
   a) Consider the following code (don't run it yet, just think through it!)

```
sum = 0
for i in range(10):
    sum += 0.1
    if sum == 0.3:
        break
print(sum)
```

   What should the result of this code be? Logically, you'd think it would print 0.3 at the end. Go ahead and run it.

   b) Another one:

```
number = 9.3
while number % 0.5 != 0:
    number -= 0.1
print(number)
```

   What should the result be? What is it actually?
   c) Go back to the code in part a).
      1) Make it so that it prints out the value of "sum" at each line. Is the problem obvious now?

2) Whenever you try to compare two floating point numbers `x` and `y` with `x == y`, things are going to work poorly. Instead, you should write `abs(x-y)<tol` where tol is some tolerance value, usually small (`1e-6`, for instance).
3) Rewrite part a) so that it checks if sum and 0.3 are with 1e-6 of one another, instead of checking if they're equal.
4) Re run the program.

4) Write a function that implements the quadratic formula. i.e. call the function `quadratic(a,b,c)` where a, b, and c are the coefficients of the quadratic $ax^2 + bx + c$. Have the function return a list of the two roots. Where the first root is the smaller of the two. If the quadratic has no real roots, then return `None`. If the quadratic has one real root, return a list with a single element in it.

```
print(quadratic(1,2,3)) # Should be None
print(quadratic(1,0,-2)) # Should be [-1.41421356, 1.41421356]
print(quadratic(1,0,0)) # Should be [0]
print(quadratic(-17,5/3,math.pi)) # should be [-0.3836, 0.4816]
```

5) Write a function called `all_even_digits(min, max)` that returns all the numbers between min and max (inclusive) with all even digits. i.e. the numer 1464 has all even digits but the number 1234 does not (since 3 isn't even).

```
all_even(200, 300)
```

Should return

```
['200', '202', '204', '206', '208', '220', '222', '224', '226',
'228', '240', '242', '244', '246', '248', '260', '262', '264',
'266', '268', '280', '282', '284', '286', '288']
```