Tutorial MATH 1MP3 – January 26, 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) Write a python function called `number_to_day` that takes in a number between 1 and 7 and returns the corresponding day of the week as a string.
   ```
   number_to_day(1) # should return "Sunday"
   ```
2) Write a python function called `my_max` that takes a list of numbers as an argument and returns the maximum of the list. Test it by running the following lines:
   ```
   my_max([1,2,3]) # should return 3
   my_max([1,0,100,-2]) # should return 100
   my_max([-1,-2,-3]) # should return -1
   ```
3) Write a python function called `case_coun`t that accepts a string and returns a list of two numbers, where the first number in the list is the number of lowercase letters in the string and the second number in the list is the number of uppercase letters.
   ```
   case_count("The Quick Brown Fox") # should return [12,4]
   ```
4) Write a function called `make_unique` that takes a list and returns a new list with only the unique elements.
   ```
   make_unique([1,2,3,3,4,4,6,5,5,5,7,5]) # Should return
   [1,2,3,4,6,5,7]
   ```
5) Write a python function called `is_prime(n)` that takes a number n as a parameter and check to see if the number is prime or not. Should return True or False depending on if the number is prime.
   ```
   is_prime(23) # True
   is_prime(161) # False
   is_prime(211) # True
   ```
   Hint: Have a for loop that checks if `i` is a factor of `n`.
6) Write a python function `only_even` that returns only the even numbers from a given list.
   ```
   only_even([1,2,3,4,-1,-2,9,27,32,2]) # Should return [2,4,-
   2,32,2]
   ```
7) Write a python function `make_primes(n)` that uses your `is_prime` function from 4) to return a list of all the primes less than some number `n`.
   ```
   make_primes(10) # [2,3,5,7]
   make_primes(50) # [2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37,
   41, 43, 47]
   ```
8) Using **recursion**, write a function called `fibonacci(n)` that calculates the `n`th Fibonacci number (recall: From last weeks lab, the fibonacci numbers are acquired by taking 1 and 1 as the first and second fibonacci number, the third fibonacci number is the sum of the first and second (2=1+1), the fourth fibonacci number is the sum of the second and third (3=2+1), etc.)