INTRODUCTION TO PYTHON EXERCISES

Here are some practice exercises for you to try out in Python. Though these aren't related to statistical learning, it's important for you to get a feel for the Python language and be comfortable with it. This is especially true for those of you who have no prior programming experience. If you need any help for any of the problems, email me at kctsiolis@gmail.com.

(1) Write a script that generates a random integer between 0 and 100 (inclusive) and gives the user 7 tries to guess it. For every incorrect guess, tell the user whether the secret number is higher or lower. If the user fails to guess it after 7 tries, tell them what the number was. Note that to generate the number, you will need to import random and use the function random.randint. Look up the function online for complete details. Be sure to notify the user when they enter an invalid guess (a number less than 0 or greater than 100). (*Optional additional exercise: For those of you who are familiar with try/catch in Java, you can use something similar in Python to check for other errors like when the user enters a float or text as input. Fun aside: When you play this game, what is a strategy that ensures that you win every time?*)

(2) Say that you are a cashier at a sports store that sells only 5 items (we'll just say that this store is very exclusive).

1. World Cup Soccer Ball Replica - $29.95
2. Soccer Cleats - $74.45
3. Shin Pads - $19.75
4. Sports Bag - $23.55
5. Signed Cristiano Ronaldo Framed Picture - $225.00

For a limited time, the store is paying the taxes, so the client's total is only based off of the amounts above. Your job is to ask the user which item(s) they want and how many of each. Be sure to return an error if the user does not enter sensible numbers (you can't order -1 items). After that, report a total and prompt the user to pay in cash. Based on how much the user pays, calculate how much change should be given AND the number of hundreds, fifties, twenties, tens, fives, toonies, loonies, quarters, dimes, and nickels that this change comes in the form of. (***Updated Hint:*** *Use the integer division we were discussing. Modular division might also be helpful, but be careful with it and there are ways of doing without it. I apologize if the initial hint I gave on Thursday threw anyone off about this. Also, Python often makes precision errors in its calculations, so I suggest looking up a function that allows you to round to two decimal places.*)

(3) The Taylor series for $\cos(x)$ is shown below.

$$\cos(x) = \sum_{n=0}^{\infty} (-1)^n \frac{x^{2n}}{(2n)!} \qquad \text{for all } x$$

Define a function for the Taylor approximation of $\cos(x)$ that takes x and k (the number of terms in the sum) as its arguments. Allow k to be decided by user input. Plot a graph of the Taylor approximation of $\cos(x)$ and compare this with the graph of $\cos(x)$ generated with the help of NumPy.