CSCI160 Computer Science I
Lab Assignment: Mad Libs and various stuff

Objectives:

- Closely follow instructions
- Create, save, locate and execute Python scripts **from the command line**
- Have your program gather input from the user
- Convert input data to appropriate types (if necessary)
- Use string concatenation
- Convert mathematical expressions to syntactically correct Python code
- Demonstrate that you understand the various division operators
- Format output

Task 1: MadLibs

1. Familiarize yourself with the concept of a MadLib. If you have done MadLibs before you may skip this step

    - Go to the following URL: http://www.eduplace.com/tales/content/wwt_008.html
    - Fill in each of the blanks with the type of word required. If you don't know what one of the words means, use the internet or a dictionary to look them up.
    - Click "See Your Wacky Web Tale"
    - Notice how the words you entered in the web form have been inserted (concatenated) into the story

2. Build a sample MadLib using the poem “Roses Are Red” as a guide

    - Open your text editor of choice and create a new source file
    - Name the file **<your last name>_Roses.py** Please do not actually name your file “your last name” . . . instead, actually type your own personal last name. Save the file
    - **Be sure you know exactly where you saved the file. I will not know this!**
    - As comments include your name, class, date and brief program description
    - Below your authorship comments include an additional comment that says “Begin gathering user input”
    - Add the following lines of code

    ```
    adjective1 = input("Tell me an adjective, and click enter: ")
    noun1      = input("Tell me a noun (plural), and click enter: ")
    noun2      = input("Tell me another noun, and click enter: ")
    adjective2 = input("Tell me an another adjective, and click enter: ")
    ```

    - Following that add a comment that says “Process input and display MadLib”
    - Add the following lines of code. Notice the use of the + operator. In a string context this operator will perform concatenation instead of addition.

    ```
    print ("Roses are " + adjective1)
    print (noun1 + " are blue")
    print (noun2 + " is " + adjective2)
    print ("And so are you!")
    ```

- Save your script, open a command prompt or terminal depending on your OS
- Navigate to the directory that contains your script file
- Execute the script by typing: **python <your last name>_Roses.py** Please remember that you have to substitute **<your last name> WITH YOUR ACTUAL LAST NAME . . .**

3. Design your own MadLib . . . BE CREATIVE . . . but also be appropriate ;)

   - You will probably benefit from planning your story on paper first. This will assist you in determining the variables you will need.
   - Here is a helpful article on techniques for MadLib creation: https://hobbylark.com/party-games/How-to-Make-Your-Own-Mad-Libs
   - Feel free to adapt song lyrics, famous poems, historical speeches or any other text you see fit.
   - There are 1000s of examples of MadLibs on the InterWeb. Use these for guidance but create your own narrative.

4. **Requirements:** Your story must contain at least 10 sentences

Task 2: Projectile Motion, 100 years old and Digit Sum

1. **Problem Statement:** Suppose a ball is thrown straight up in the air with an initial velocity of 50 feet per second and an initial height of 5 feet. How high will the ball be after 3 seconds?

   Note: The height after **t** seconds is given by the expression: **$-16t^2 + vt + h$** where
   a. **v** => is the initial velocity
   b. **h** => is the initial height.

2. Create a python script named **<your last name>_Velocity.py** You will be adding the following three problems to this script file. They should run sequentially without interruption.

3. Implement the calculation above in Python code. The answer to this problem using the above givens is 11. Use this to double check your work

4. When you have your algorithm working with the given values modify your code to ask the user for input of:

   a. initial velocity
   b. initial height.

5. Do not forget that the **input()** function returns string data as a result. If you want to use this data in a mathematical expression you must convert it. The relevant conversion functions are:

   a. int( )
   b. float( )
   c. str( )
   d. these will all return a new value that must be stored in a variable.
   e. Examples:
      i. **x = int(x)** # convert variable x to an integer
      ii. **y = float(y)** # convert variable y to a float
      iii. **z = str(z)** # convert variable z to a string

6. Write statements to prompt the user to enter their name and their age. Print out a message addressed to them that tells them the year that they will turn 100 years old.

7. Ask the user to enter a 4-digit number (no need to verify, just trust in your fellow human beings to enter correct data). Using your secret **math** powers, separate each digit and print; also display the sum of the digits.

   Your code should work for any 4-digit number and your display should match the following

   Example:

   ```
   Enter a four-digit number: 1234

   1
   2
   3
   4

   1 + 2 + 3 + 4 = 10
   ```

8. TEST, TEST, TEST . . . and then TEST again. Verify that your program functions properly.

Submit your MadLib and Projectile Motion scripts in to the assignment area in Blackboard

Your code must be runnable AS IS!! In order to receive full credit. Partial credit will be awarded if not, so don’t delete any code that doesn’t work; comment it out. If I have to modify your script file in any way to make it functional, points will be taken off.