ECE101: Introduction to Computer Systems
Homework #2 - 100 Total Points

A. Purpose

This assignment is to test each students understanding of basic python programming using conditional statements and control flow, as well as students understanding of binary data representations.

Please read the following lines carefully before you start on the exercises. This exercise has two components, the first component is python coding, and the second component is handwritten.

For the python portion of this exercise, we have included a directory called
StudentID_LastName_FirstName_HW2.zip. Inside this directory, there are a total of six files.

The file ECE101_HW2_Unit_Test.cpython-37.pyc, is to be used to test your code, just as we have discussed in lab sections, and the file Report.txt is to be used for writing your report for this exercise.
NOTE: YOU HAVE TO USE PYTHON3
After completing all of the problems, just run:

$python3 ECE101_HW2_Unit_Test.cpython-37.pyc

If it doesn't work again (version related issue), you can skip this test. This is just to automate the process of testing your code.

Finally, there are four other files with names as follows Prob0.py, Prob1.py, Prob2.py, Prob3.py. Your code for the python component of the exercies must be written in these files.

Important note:
The first line of text in each of these files will begin with a statement similar to this, **def ProbX**. Also, the second line of text in each of these files will begin with the statement #### **BEGIN: Your code goes after this line.**

You are to delete the third line of text which is **pass**, and start to write your code where the text **pass** used to be. Do not delete anything else from inside the ProbX.py files.

B. Exercises

0. **Converting Numbers To Strings** (Perform this exercise in the **Prob0.py** file.)

   Assume that you are given a variable called **num**, which exists. You can also assume it is always an integer value greater than or equal to zero (0).

   You are to convert each digit of the value in the variable **num** to a string, and display that string. Note that each converted digit corresponds to the English word for the value of that digit.
   In your code, you must define at least one variable called **temp**. This variable will contain the converted string.

Example: assuming the variable **num = 428**, then your job is to write a piece of code that would store the string ***Four Two Eight*** in the variable **temp**, and also display it on the terminal. **REMEMBER TO END THE PRINT OUTPUT with ' '. Eg. print("XYZ", end = ' ') for the automated test to work.**

Hint: the python built-in functions **str()** and **int()** will be useful.

NOTE the following:

- **The strings must be seperated by ONLY one space character.**
  In the example shown above, there is only one space between **Four** and **Two**, and only one space between **Two** and **Eight**.
- **The first character of each string must be upper case.**
  In the above shown example, the displayed strings are **Four Two Eight**, but not **four two eight**.
- **The strings must be the coresponding names of the digits, not the decimal value they represent.**
  In the example above, the displayed string is not **4 2 8**, but rather **Four Two Eight**.
- **The strings must not have any other special formatting**.
  In the example shown above, the displayed string is not "The value you entered is Four Two Eight", or some other special string format.

1. **Displaying Right-Sided Triangles** (Perform this exercise in the **Prob1.py** file)

Assume that you are given a variable called **height**, which exists. You can also assume it is always an integer value greater than zero.

You are to display a right-sided right-angled triangle (this means the 90 degree angle is on the bottom right side as shown in figure 1). The base and height of the triangle are equal to the value of the variable **height**. The displayed format you are allowed to use is the star (*) symbol.

Example: Assuming the variable **height = 10**, then your your is to display the following triangle;

Figure 1: Right-sided right-angled triangle

Hint: the python built-in function **str()** and **int()** be useful.

NOTE the following:

- There are no spaces between subsequent stars
- The number of stars representing the base are equal to the number of stars representing the height
- The 90 degree angle is at the bottom right, not the bottom left

2. **Displaying Values in Rows and Columns** (Perform this exercise in the **Prob2.py** file.)

   Assume that you are given a variables **rows** and **cols**, which exists. You can also assume it is always an integer value greater than zero.

   You are to display a table of integers according to the following format;

   (a) The number of integer values for each row of your table must be equal to the value of the variable **cols**. This means if **cols** = **10**, then each row must have EXACTLY 10 integers.

   (b) Each integer value in the row must be followed by EXACTLY one space.

   (c) The number of integer values for each column of your table must be equal to the value of the variable **rows**. This means if **rows** = **5**, then each column must be EXACTLY 5 integers.

   (d) The first integer value for each row MUST ALWAYS be equal to the row number. What this means is that if the current row is three, then the first value in that row must be a 3.

   (e) The integer values in each row must ALWAYS be a multiple of the current row number. What this means is that if the current row is two (in this case the row number is 2), then the values in that row (row 2) must look like 2 4 6 8 ... etc with exactly one space between each value.

   Example: Assuming the variables **rows** = **5** and **cols** = **20**, then your job is to up with a piece of code that displays the following table of numbers;

   1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
   2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40
   3 6 9 12 15 18 21 24 27 30 33 36 39 42 45 48 51 54 57 60
   4 8 12 16 20 24 28 32 36 40 44 48 52 56 60 64 68 72 76 80
   5 10 15 20 25 30 35 40 45 50 55 60 65 70 75 80 85 90 95 100

   NOTE the following:

   - There are exaclt 10 integer values in each row since input parameter two is ten
   - Each integer value in each row is followed by exactly one space
   - There are exactly 5 integer values in each column, since input parameter one is 5
   - The first integer value for each input parameter is always equal to the row number
   - All the integer values for each row is a multiple of the row number

3. **Empirical Testing of the Collatz Conjecture** (Perform this exercise in the **Prob3.py** file.)

   The Collatz Conjecture (or $3n + 1$ conjecture) says that if you repeat the process of TOPTO of any natural number, it always comes back to 1. HOPTO means have-or-triple-plus-one. The process simply interprets as follows;

   - Take any natural number
   - If that number is even, then Half it
   - If it's odd, then triple it and add 1

   Assume that you are give a variable called **num**, which exists. You can also assume it is always an integer value greater than zero.

   Write an algorithm that repeats the HOPTO process on the value of the variable **num** and see if your code satisfies the Collatz Conjecture. Your algorithm must display the new value at each iteration without any other special formatting. Note that the first value printed must be the original value of variable **num**, and the last value printed must be a 1 to prove that your algorithm satisfies the

Collatz Conjecture.

Example: Assuming the variable **num = 5**, then your job is to come up with a piece of code that displays the following;
5
16
8
4
2
1

C. Submission

Save your code into the folder using the recommended folder naming criteria (StudentID_LastName_FirstName_HWXX).

You are also welcome to take a snapshot of your handwritten version of the homework and add to the folder you created above, otherwise, please hand in your handwritten homework during lectures.

Zip up the folder and submit it under the appropriate assignment on blackboard.