## CSC 401 ASSIGNMENT THREE

Due Date: Tuesday, Aug. 11th by 11:58 PM

The purpose of this assignment is to assess your understanding of

- Nested Loops
- While Loops
- Two-Dimensional Lists
- Dictionary
- Tuple

## SUBMISSION

- Include your full name as a comment on the first line of your Python program file.
- Include the problem number as a comment before each user defined function.
- **Code all the problems in one Python file (.py) labeled as YourName_HW3.py**
- Upload one file to Submissions folder.

## PROBLEMS

Note: You may not use Python statements, functions, data type, etc. that were not discussed in the reading assignment or the lecture notes/video for this week or previous weeks. This is a class for students who have not programmed before and I expect everyone to code on the same level. If you have a better way of writing the code, then upload two versions: one that codes according to the specifications and the other that demonstrates advanced programming techniques.

In this assignment the solution to each problem should be coded as a **user-defined function**. **All of the functions should be in one file.**

## PROBLEM 1 (10 POINTS)
## NESTED FOR LOOPS, ACCUMULATOR

In the lectures, we calculated the sum of the values in each sublist of a multi-dimensional list. Now you are to write a function sumColumns(lst) that takes a multi-dimensional list of any size as a parameter, and returns the sum of each column in an one-dimensional list. The number of sublists and the length of each sublist must be the same, i.e. 4 rows x 4 columns. Hint (use indexes and range)

Sample:

```
>>> sumColumns([[14,82,73],[46,26,7],[26,95,21]])
[86, 203, 101]
```

## PROBLEM 2 (10 POINTS)
## WHILE LOOP, ACCUMULATORS

Write a function numLetters() that keeps prompting the user for words until they hit return/enter key. The function should then return (not print) the percent of 3-letter words that were entered. So, if a user entered ten words and 4 of the words had 3 letters, then the percent of 3-letter words is 40

## PROBLEM 3 (10 POINTS)
## WHILE LOOP, MATH LIBRARY

Write a function getNumbers(n) that accepts a positive integer, n, and creates a sequence of numbers as follows:

- If n is an even number, then n is replaced with the floor of n**.5
- If n is an odd number, then n is replaced with the floor of n**1.5
- floor() rounds down a number and is available as a function in the math library
- continues calculating until after n is 1
- collects all the values of n in a list and return (not print) the list

Sample:

```
>>> getNumbers(2)
[2, 1]
>>> getNumbers(10)
[10, 3, 5, 11, 36, 6, 2, 1]
>>> getNumbers(36)
[36, 6, 2, 1]
>>> getNumbers(37)
[37, 225, 3375, 196069, 86818724, 9317, 899319, 852846071, 24906114455136, 4990602, 2233, 105519,
34276462, 5854, 76, 8, 2, 1]
>>>
```

## PROBLEM 4 (15 PTS)
## DICTIONARY, WHILE, TUPLE

Write a Python function member**()** that creates a dictionary for each member. The function allows the user to enter a 4-digit member ID for each member. The program will keep prompting the user for a first name and last name. If the member (key is the tuple first name and last name) does not have a member ID on record (i.e. in the dictionary), the program will then ask for the member ID, and store that information in the dictionary. If the member already has a member ID, the program will display it, and ask for confirmation whether a new member ID should be assigned (and, if so allows the new member ID to be entered). When the user hits the return (or enter) key, the program prints a report listing all members by last name, first name, with their member ID. Note: you may assume that the member ID input will always be 4 digits i.e. no need to validate.)

Sample:

```
>>> member()
First name: Ricky
Last name: Ricardo
Member ID: 1111
First name: Lucy
Last name: Ricardo
Member ID: 1111
First name: Ethel
Last name: Mertz
Member ID: 3333
First name: Lucy
Last name: Ricardo
Lucy Ricardo has id 1111 Update? y
Member ID: 2222
First name: Fred
Last name: Thursday
Member ID: 4444
First name: Ricky
Last name: Ricardo
Ricky Ricardo has id 1111 Update? n
First name: Endeavor
Last name: Morse
Member ID: 5555
First name:

Contents of Dictionary:
Mertz, Ethel has id 3333
Morse, Endeavor has id 5555
Ricardo, Lucy has id 2222
Ricardo, Ricky has id 1111
Thursday, Fred has id 4444
```

IF YOU HAVE ANY QUESTIONS REGARDING THIS ASSIGNMENT, PLEASE POST THEM TO THE ASSIGNMENT THREE DISCUSSION FORUM.

**Assignment Three Grading Rubric**

Learning outcomes:

- Create user-defined functions
- While loop conditionals
- Initialize a dictionary
- Add and update data to a dictionary
- Sort the contents of a dictionary
- Use a tuple as a dictionary key
- Access data in a dictionary
- Use return statements to exit a function

| Problem Numbers | Proficient 10 – 19 | Nearing Proficiency 8 - 7 | Needs Improvement 6 – 0 |
|---|---|---|---|

| | | | |
|---|---|---|---|
| | Shows a comprehensive understanding of user-defined functions, the dictionary data type, iteration control structures and computational thinking | Shows an adequate understanding of user-defined functions, the dictionary data type, iteration control structures and computational thinking | Shows a minimal or no understanding of user-defined functions, the dictionary data type, iteration control structures and computational thinking |
| One | Correctly defines<br>sumColumns() to accept a two-dimensional list as an argument and returns the results as a one-dimensional list as shown in the sample case.<br>Uses nested for loops<br>All calculations are correct | Adequately defines sumColumns() to accept a one-dimensional list as an arguments and returns the results as a one-dimensional list as shown in the sample case<br>Uses nested for loops<br>Some calculations are incorrect | sumColumns() has minimal or no correct code.<br>sumColumns() does not accept any arguments and returns no results or incorrect results<br>Do not use nested for loops<br>Results are printed<br>Most calculations are incorrect.<br>Output differs substantially from that shown in the sample case. |
| Two | Correctly defined numLetters()<br>numLetters() does not accept any arguments<br>User is prompted to enter words<br>Loop is ended on enter/return<br>Calculation is correct and returned<br>Output is displayed as described | Adequately defined numLetters()<br>numLetters() accepts an argument<br>User is prompted to enter words<br>Loop is ended with a response rather than the enter/return<br>Calculation is correct but not returned<br>Output is partially displayed as described | numLetters() has minimal or no correct code<br>numLetters() accepts an argument<br>User is not prompted to enter words<br>Infinite loop<br>Calculation are incorrect<br>No calculation is returned<br>No output or output is incorrect |
| Three | Correctly defined getNumbers(n) to accept a positive integer<br>While loop is ended when n | Adequately defined getNumbers(n) to accept a positive integer<br>While loop is ended | getNumbers() has minimal or no correct code<br>getNumbers() does not accept an argument |

| | | | |
|---|---|---|---|
| | is greater than 1<br>Correctly identifies even and odd number<br>Calculation is correct<br>Uses floor() from the math library<br>Creates a list of all the resulting numbers<br>Returns the list | prematurely<br>Incorrectly identifies even or odd numbers<br>One of the calculations is incorrect<br>Uses floor() function<br>Stored in a container other than a list<br>Returns collected numbers | while loop is ended incorrectly or does not use a while loop<br>Even and odd numbers are not identified<br>Calculations are incorrect<br>Uses built-in function to convert results of the calculation<br>Prints results |
| Problem Number | Proficient<br>15 – 13 | Nearing Proficiency<br>12 – 10 | Needs Improvement<br>9 - 0 |
| Four | Correctly defines the user-defined function member()<br>Correctly initializes the dictionary<br>Correctly adds data to the dictionary<br>Correctly updates the dictionary<br>Correctly uses a tuple for the dictionary key<br>Correctly use the member ID as the value for each key<br>Prompts the user for first name, last name and ID<br>Correctly updates the ID when requested<br>Prints the dictionary as formatted in the example | Adequately defines the user-defined function member()<br>Incorrectly initializes dictionary outside of function<br>Correctly adds some data to the dictionary<br>Correctly uses a tuple for the key, but incorrectly uses a llst for the value for each key<br>Prompts the user for some of the requested data<br>ID not correctly updated<br>Prints the dictionary but not in the format used in the example | Minimally defines or no definition of user-defined function member() as described<br>Dictionary is not initialized<br>Minimal or no data added to the dictionary<br>Dictionary key is not a tuple<br>Dictionary key data is incorrect<br>Dictionary value is missing or incorrect<br>User is not prompted to update.<br>Update is incorrect or incomplete<br>Dictionary is not printed |
| For all problems | User-defined functions do not use any data types, statement, methods or operators that have not been presented in week 3 or prior week's lectures or reading | User-defined functions use a data type, statement, method or operator that have not been presented in week 3 or prior week's lectures or reading | User-defined functions use more than one data type, statement, method or operator that have not been presented in week 3 or prior week's lectures or reading |

| | | | |
|---|---|---|---|
| | assignments.<br><br>Complete thorough testing<br>User- defined functions have no syntax errors<br>User-defined functions execute with no run time errors<br>All problem specifications are correctly coded | assignments.<br>Only tested with given data or partially tested<br>User-defined functions have one or two syntax errors.<br>User-defined functions do not execute because of a run time error<br>Some deviations from specifications | assignments.<br><br>Minimal or no testing<br><br>User-defined functions have more than two syntax errors<br>User-defined functions do not execute because of run-time error<br>Hardly follows the specifications |