420-LCU-05 Programming in Python - Assignment 2
Due Friday March 11, 2022 at 11:59 p.m.

Identification section: This section must be either in a comment, with a '#' preceding each line, or enclosed within triple quotes ('''). The grader and I need this section for the accurate processing of your assignment. Assignments missing this may lose up to 5% of the mark.

```
# Your Name and ID
# 420-LCU Computer Programming, Section #
# S. Hilal, instructor
# Assignment 2
```

Submission: Submit your assignment in 1 Python file, with the extension .py. Please do not create a ZIP file. Be sure to respect other instructions specified in the assignment. An important part of each assignment is to correctly follow the instructions closely. **Late assignments** are accepted up to 1 week from deadline. But late penalty will be applied.

Learning Objectives:

- Practice using lists, strings, loops and Multidimensional (nested) lists.
- Practice with built-in functions and methods for strings and lists.

A Simple Class Calculator

For this assignment, you will develop a small application that will enable a teacher to **enter**, **analyze** and **report** on the grades of the students in the class. The teacher can enter students' grades in different components of the course. The program can calculate a student's **overall grade**, **letter grade**, in addition to some overall class statistics.

Description of Data:

- A student is identified by full name, a 5-digit ID number in addition to a 2-letter code indicating their program. For each student, there are 5 grades based on 2 tests, 2 assignments and 1 project.
- Each test is marked on 25 and is worth 25% of the total grade. Each assignment is marked on 10 and is worth 10% of the total grade. The final project is marked on 30 and is worth 30% of the total grade. Complete total of 100 points and 100%.
- Each grade is entered as an integer or float value.
- Your program should be able to process the information for any number of students but you can limit your tests to 8 students.

Description of Program:

Data Table: Your program defines a 2-dimensional list named **students** (this will be a list of lists or a matrix). The list **students** will hold the data for all students. Each list element (row in the matrix) holds the data for one student. Note that the list is initialized as an empty list at the very beginning of your program and will be filled in as the program progresses.

Students = [] # List of students is initialized

Data Entry: Your program asks the user to enter the information for each student (student record). The program verifies and stores the information in **students**.

Important: When a student record is entered, your program should calculate the total grade and letter grade and store them (append them) with all the entered information on a list named **student_list**. **student_list** would then added to the master list **students**.

Here is an example of how the list **students** should look like after entering the information for 3 students:

```
students=[["Lea Smith",12345,"HH",20.0,24.5,10.0,9.5,28.0,92.0,'A'],
          ["Bob Green",23456,"HP",22.5,24.5,10.0,8.0,21.0,86.0,'B'],
          ["Ben Johns",34567,"B2",24.5,20.0,9.0,10.0,27.0,90.5,'A']]
```

The data for each student is stored in the following order: student's name (string), ID (an integer), program (2-character string: HH, HP or B2), 2 test grades (on 25 each) , 2 assignment grades (on 10 each), and 1 project grade (on 30). **All grades must be stored as floats**.

The program menu **Display exactly as shown here in your code.**

Welcome to the Simple Class Calculator. Here's the list of available options:

1- Enter a student record (Name, ID, program, and 5 grades separate by commas, no spaces).
2- Display the full class data sorted in ***alphabetical order*** based on ***name***.
3- Calculate and Display the class descriptive statistics.
4- Display the record of a given student
5- Display the list of student names below class average.
6- Exit

Select an option by entering its number or 6 to exit:

Running Your Program:

Your program starts by presenting the user with the above **menu** of numbered options. A user selects an option by entering the corresponding number. When the program completes processing an option, **the full menu is displayed again** and the user is prompted to make another selection. This continues until the user selects the option **to exit**.
Important: Your program must ensure that the user enters a valid option and that a requested option can actually be processed or give an appropriate message. E.g. the program cannot process option 3 if the list **students** is empty and should display an error message.

Include other validation tests that you see necessary and add comments and appropriate error messages.

Description of the Different Options (above):

<u>For each option,</u> the program output is shown in bold. User input is shown in red

Option 1: The program asks the user to enter a student record.

Here are 7 examples for option 1. Important Note: After each example, the main menu is displayed and the user selects option 1 again to enter another student record.

1- **Enter Student Record (Name, ID, program, and 5 grades separate by commas, no spaces):** Lea Smith, 12345, HH,20,24.5,10,9.5,28
Record Accepted.
2- **Enter Student Record (Name, ID, program, and 5 grades separate by commas, no spaces):** Bob Green,23456,HP,22.5,24.5,10,8,21
Record Accepted.
3- **Enter Student Record (Name, ID, program, and 5 grades separate by commas, no spaces):** Ben Johns,34567,B2,24.5,20,9
Record Incomplete, Record rejected.
4- **Enter Student Record (Name, ID, program, and 5 grades separate by commas, no spaces):** Ben Johns,34567,B2,24.5,20,9,10,27
Record Accepted.
5- **Enter Student Record (Name, ID, program, and 5 grades separate by commas, no spaces):** Ben Johns,345678,B2,25,22.5,10,8.5,28
ID must have 5 digits. Record rejected.
6- **Enter Student Record (Name, ID, program, and 5 grades separate by commas, no spaces):** Bob Green,22222,B2,25,25,10,9,27
Record Accepted.
7- **Enter Student Record (Name, ID, program, and 5 grades separate by commas, no spaces):** Zoe Clark,22222,B2,22,19,10,9,26
Duplicate ID number. Record rejected.

Here are some additional requirements and hints:

- A student record entered using **input**, is string: e.g. "Lea Smith, 12345, HH,20,24.5,10,9.5,28" **.** Use split(",") and create the list **student_list** to hold the data for 1 student.
- There may be duplicate names but IDs are unique.
- The program rejects an incomplete record (missing any of the 8 data items). Just check count of items entered.
- When a valid record is entered, calculate total and letter grades and store on the **student_list**
- The complete **student_list** is then added to the list **students**.

<u>Option 2</u>: Display the full list in alphabetical order. Include all information entered in addition to total and letter grades.

<u>Option 3</u>: **Class descriptive statistics are:**

- Number of students entered

- Class average based on total grade
- Average grade for each test, each assignment and the project.
- Letter Grade distribution (how many of each letter grade)

Your program prints each item as above (on a separate line) with the text (as given) and value on the same line.

Option 4: The program asks the user to enter the ID of a student. The program prints the full record of the requested student.

Enter the ID of a student: 12345

The program prints:

Name: Lea Smith ID: 12345

Test Grades: 20.0, 24.5 Assignment Grades: 10.0, 9.5 Project Grade: 92.0

Total Grade: 92 Letter Grade: A

Option 5: The program displays the list of students with total grade below the class average in the following format, separate items by tab and print each student on a separate line:

Name: John Doe ID:33333 Grade:66 Letter-Grade: C

Table of the letter grades that correspond to the total score:

Total Grade	Letter Grade
87 or above	A
From 75 to 86 inclusive	B
From 65 to 74 inclusive	C
Below 65	F

Writing your program:

1- You can use any of the list/strings built-in functions and methods that we have seen in class.
2- The program will display the main menu following the completion of each option and until the user selects option 5.

Testing Your Program:

The program does not store any data between runs. It is a good idea that you create a few sets of data that you can use to test your program and always use the same data. You can store your data in a comment at the beginning of your program such that you can copy and paste the data for input rather than typing each time. The more records you use to test the better. The records I used for my tests are provided below for you. Feel free to add more.

#Lea Smith, 12345, HH,20,24.5,10,9.5,28
#Bob Green,23456,HP,22.5,24.5,10,8,21
#Ben Johns,34567,B2,24.5,20,9
#Ben Johns,34567,B2,24.5,20,9,10,27
#Zoe Clark,22222,B2,22,19,10,9,26