CS 700: Assignment 1

Due Date: *Oct 02 (Su) 11:59 PM* **Overdue submissions:** 25% penalty per day

Section 1

Write a program that reads the following from each data line of an input file: a student's name, the number of exam scores, followed by all the exam scores.

For example, a line of the input file may look like the following:

SPWKLZWHCIE GAPCEXAS 11 100 88 95 94 98 100 90 71 100 92 98

where,

SPWKLZWHCIE GAPCEXAS is the student's name, 11 is the number of exam scores, and
100 88 95 94 98 100 90 71 100 92 98 are the exam scores.

Write necessary functions and put them in a module to process this data such that the program

1. [10%] Reads each data line from a text file (a *sample* input file is included).
2. [20%] Outputs each student's name, average exam score, and letter grade (use the following table) in a **well-formatted** fashion to a console display.

Letter Grade	Percentage
A	95%–100 %
A−	91.67%–95 %
B+	88.3%–91.67 %
B	85%–88.3 %
B−	81.67%–85 %
C+	78.33%–81.67 %
C	75%–78.33 %
C−	71.67%–75 %
D+	68.33%–71.67 %
D	65%–68.33 %
D−	61.67%–65 %
F	0%–61.67 %

3. [5%] Writes all information including each student's data (name, the number of exam scores, followed by all the exam scores), average exam score, and letter grade stored in a **well-formatted** fashion to an output file.

Section 2

Now, suppose the input file is modified so that all students have the same number of exam scores. The number of students and number of scores are data items; the first line of the input file contains only the number of students and number of scores. For example, the first line of an input file may contain 25 10, where 25 indicates number of students and 10 is the number of scores for all students (a *sample* input file is included).

Create another version of the above program so that it:

4. [25%] Stores the exam scores in a two-dimensional array. The row subscript should be the student number and the column subscript should be the exam number.
5. [20%] Calculates the average score for each exam.
6. [15%] Uses the average score for each exam to calculate and display the student's grade for that exam. If the student's score is within + or – 5 points of the average, it gives a grade of C. If the grade is more than 5 points above (below) the average, it gives a grade of B (D). If the grade is more than 15 points above (below) the average, it gives a grade of A (F).
7. [5%] Writes each student's name to the output file followed by the student's score and grade on each exam in a well-formatted fashion.

Marking Scheme:

- **Working program [50%]** A working program which satisfies all of the requirements automatically receives 50% of the total assignment mark. Each element of non-compliance will be penalized with respect to its severity.
- **Program Structure [25%]** A program which follows principles of Object-Oriented Design and structured programming rules (procedural, modular, uses parameters) to perfection automatically receives 25% of the total assignment mark. Marks are deducted depending on severity and number of occurrences of non-compliant elements.
- **Program Documentation [15%]**
    - **Internal documentation [10%]:** Documentation should be complete and in a standard format. Every non trivial part of the code should have a *clear* comment that explains it. In addition, every method or function, including the main program should have an explicative comment header. This header includes: module name, author, date of creation and purpose. A description of parameters and method output is mandatory. Marks are deducted according to the absence of these elements.
    - **External documentation [5%]:** HTML documentations generated by a program such as Doxygen should be provided.
- **Program Style [5%]** Style refers to Occam's razor principle. Code that is needlessly tricky, obscure, or difficult to read will be marked accordingly. Program text indentation is also an element of style and must be present. Significant constant, variable and structure names must be used. Marks are deducted on the basis of the frequency of these errors.
- **Version Control System [5%]** Track of changes that made to a program should be kept in a **private** Github repository using an IDE integrated version control system**.** The teaching assistant must be invited as the **only** collaborator to have an access to the repository. A link to the repository and the corresponding screenshots should be included in the submission.
- Test each program with at least two different sets of input values.
- All files to be submitted should be placed in a single directory and zipped together into a single file for uploading to **UR Courses**.

- Your submission for each programming question includes: (a) the source code files (.CPP and .h files); (b) a Readme.txt file containing a brief explanation of each file along with a link to the Github repository of the assignment; and (c) screen shots of testing runs and the Github repository.