Programming Fundamentals – Spring 2013

(BS-SE-F12 Morning & Afternoon)

Programming Project

Submission Deadline: **Monday, 24th June, 2013 (till 6:00 PM)**

Submission Folders:
\\printsrv\Teacher Data\Ahmad Ghazali\PF - S13\Proj Morning
\\printsrv\Teacher Data\Ahmad Ghazali\PF - S13\Proj Afternoon

Instructions

- **This is an individual project. You are NOT allowed to work/submit in form of group. Absolutely NO collaboration is allowed. Any traces of plagiarism/cheating would result in an "F" grade in this course.**
- Do **NOT** copy even a single line of code from any other person or book or Internet or any other source.
- This project needs to be submitted in **soft form**. See **Submission Procedure** at the end.
- Late submissions will NOT be accepted, in any case.
- Clearly mention your **Name**, **Roll Number** and **Section** in comments at the top of the CPP file.

In this programming project, you are required to implement a Student Record Management System with some very basic features.

You are given an input file (**`students.txt`**) which contains the GPAs of several students. A sample input file is shown below:

```
6
BITF12M706 Umar Akmal
5
3.7 4 F 2 2.3
BSEF12A402 Muhammad Hafeez
3
3.3 3 1.7
BCSF12M200 Shahid Khan Afridi
4
F 2.3 3.7 4
BITF12A238 Misbah ul Haq
6
3.3 3.2 3 4 F 3.7
BCSF12A654 Junaid Khan
4
3 4 2.3 3.7
BSEF12M345 Saeed Ajmal
5
3.7 F 2.7 4 3.5
```

The first line of input file contains a positive integer *N* indicating the number of student records present in the file. The record of *each* student is stored on three lines in the input file.

- First line contains the roll number of the student followed by his/her name. There will be single space between the roll number and name of a student. (*Note:* Name of a student can contain spaces within it. You can assume that the roll number will have *exactly 10 characters* and the name of a student can have *a maximum of 40 characters*).

- The second line contains a positive integer *C* indicating the number of courses that student has studied in the current semester.

- The third line contains GPAs of that student in *C* courses. GPA of a course can be a floating point value (up to at most 1 place after the decimal point), or it can be the letter **`F`** (for failure). After each GPA there will be a single space. Last GPA on a line will be followed by the newline character.

You are required to write a C++ program which reads the input file (**`students.txt`**) and stores all of its information in variables of appropriate types. You will need to define and use the following structure (**`Student`**) in your program:

```
const int ROLL_NO_LENGTH = 11;
const int NAME_LENGTH = 41;

struct Student
{
      char rollNo[ROLL_NO_LENGTH];      // Roll No. of the student
      char name[NAME_LENGTH];           // Name of the student

      int numCourses;
                  // Number of courses taken by the student in current semester

      double* gpa;
                  // Pointer to a dynamically allocated array containing GPAs of all courses taken
                  // by the student in the current semester, as per numCourses

      double sgpa;
                  // To store the calculated Semester GPA of the student
};
```

You will need to dynamically allocate an array of **`Student`**s to store information about all the students present in the file. In each **`Student`** record, you will have to dynamically allocate an array of double values (which will be pointed to by the pointer **`gpa`**), based upon the number of courses taken by a student. The GPAs of courses taken by each student should be stored in this array. You should store a GPA of 0 for each course in which a student got an **`F`** grade.

After reading and storing all the data from the input file, your program should calculate and store SGPA for each student. SGPA will be calculated by taking the average of GPAs of all courses that a student has taken in a semester. For example, if a student has taken **6** courses in a semester and obtained the following GPAs:

```
3.3 F 2.3 4 2.7 F
```

His/her SGPA will be calculated as follows:

$$\frac{3.3+0+2.3+4+2.7+0}{6}=\frac{12.3}{6}=2.05$$

Note: We will assume that all courses have the same number of credit hours.

You are required to use functions to decompose this program into smaller, easy-to-understand steps. You should write separate functions to perform each of these tasks:

- A function `readAllRecords` to open the input file and read the records of all students. This function will call the following function in a loop:
    - The function `readOneRecord` to read the record of one student from the file

- A function `computeAllSGPAs` to calculate the SGPA for all students. This function will call the following function in a loop to accomplish its task:
    - The function `computeOneSGPA` to calculate the SGPA for one student

After reading the data from the input file and calculating the SGPA for each student, your program should display a menu to the user, from which user can select any of the following options:

1. Display a list of all students. Roll number, Name, Number of courses, and SGPA of each student should be displayed. Use appropriate format manipulators to format the output in a neat tabular way.
2. Display the student(s) who have got maximum SGPA in the semester.
3. Display the student(s) who have got minimum SGPA in the semester.
4. Searching the list of students by roll number. The user might specify complete roll number or a part of the roll number. Your program should display a list of all students that match the search string given by the user. For example, if the user enters the search string **BSE** then a list of student of BSE degree program should be displayed. If the user enters **BSEF12** then all students of BSEF12 should be displayed. The user may give input in CAPITAL, small, or mIxEd case.
5. Update/modify the record of a particular student. The user will specify the complete roll number of the student whose record he/she wants to update. After searching the list of students, details of that student should be displayed. Then, another menu should be displayed which allows the user to carry out following two types of modifications:
    i) Update the Name of the student

    The user should be asked for the new/updated name. After that this new name should be stored in the record of that student.

ii) ***Add a course (i.e. its GPA) for the student***

The user should be asked for the GPA of the new course. The user might enter a double value or the character **`F`** (for Fail). Perform input validation so that the entered value is between 0 and 4.

After taking input, the variable **`numCourses`** should be updated and the array **`gpa`** will need to be re-allocated. After doing so, the **`sgpa`** of the student will be re-calculated as well.

6. Quit the program. This option will allow the user to quit the program. Note that before terminating the program, all data (from the array of Students) should be written back to the file (**`students.txt`**), reflecting any changes made during the execution of the program.

You should implement a separate function for performing the tasks related to each of the above menu items.

A portion of the sample run of your program might look like this:

```
---- Welcome to Student Record Management System ----

Reading data from "students.txt"...

File opened successfully and records of 6 students have been read.

Menu
1. Display list of all students
2. Display student record(s) having Maximum SGPA
3. Display student record(s) having Minimum SGPA
4. Search by Roll Number
5. Update a Student record
6. Quit

Enter your choice: 1

Roll No.        Name                Courses    SGPA
BITF12M706      Umar Akmal          5          2.40
BSEF12A402      Muhammad Hafeez     3          2.67
...
```

Note: List will contain the records of all the students; the above list is just a sample.

Submission Procedure

You are required to submit this project in soft copy format (in the folder specified for your section). Put the **.CPP file** (**e.g. project.cpp**) and the **input file** (**students.txt**) in a folder (do NOT include any other files in your submission). The name of the folder should be your complete roll number (like BSEF12M234), then **compress** that folder, and copy the **.RAR** file in the specified submission folder for your section.

Please note:

- Do NOT use any **global variables** in your programs. However, you can use global named constants.
- These *good programming practices* will also have their (significant) weightage in the marking of this project:
  - Comment your code intelligently. **Uncommented code *will* not be given any credit.**
  - Indent your code properly.
  - Use meaningful variable and function names.
  - Use meaningful prompt lines/labels for input/output.
- There should NOT be any dangling pointers, memory leaks, or any other type of run-time error in your program.
- Moreover, if your submitted program gives an error at the time of compilation, you will get a ZERO in this project.

☺ GOOD LUCK! ☺

Remember: *Honesty always gives fruit (no matter how frightening is the consequence); and Dishonesty is always harmful (no matter how helping it may seem in a certain situation)!*