Information and Computer Science Department

Introduction to programming in Python & C

Semester 201

Lab project

Develop a registration system for students. Each student has

- Name
- ID
- Major
- List of registered courses, each course has
    - Code (e.g. MATH101)
    - Name (e.g. Calculus I)
    - Credit hours
    - Letter grade

Your program should load the data once from the file "***students_info.txt***" into lists and dictionaries as you need. Check File Structure for more info.

After loading the data, your program should display the following menu:

```
1. Display registered courses for a student and his GPA.
2. Display students in a course.
3. Display students with GPA more than or equal to an entered GPA value.
4. Add a course to a student.
5. Drop a course from a student.
6. Add a student info.
7. Delete a student info.
8. Save data to the file.
9. Quit.
```

Your program must loop as long as **option 9** has not been selected. It must display an appropriate error message if an invalid choice is entered.

Each of the options must be implemented in separate function.

Option 1: Display registered courses for a student and his GPA.

This option asks the user for a student ID then search the loaded lists/dictionaries for that student and shows his finished courses and GPA

- To calculate the GPA check the Course Points and GPA Calculations section
- If the entered ID is invalid your program should show an appropriate error message
- After your program shows the output it must wait until the user press “Enter” to show the main menu

Option 2: Display students in a course.

This option asks the user for a course code then it search the loaded lists/dictionaries for student who took this course.

- If your program does not find any student for that course, it should display appropriate error message
- After your program shows the output it must wait until the user press “Enter” to show the main menu

Option 3: Display students with GPA more than or equal to an entered GPA value.

This option asks the user a GPA then it go over the list and displays those whom GPA is higher than the entered GPA

- If the input was beyond the limits of the GPA (>4 or <0) your program should display an appropriate error message.
- To calculate the GPA check the Course Points and GPA Calculations section
- After your program shows the output it must wait until the user press “Enter” to show the main menu

Option 4: Add a course to a student.

This option should ask the user for an ID if the ID was found your program should ask for the course details (Course code, course name, credit hours, grade) then add it to that student and shows a success message.

- If the input was not found your program should display an appropriate error message.
- After your program shows the output it must wait until the user press “Enter” to show the main menu

Option 5: Drop a course from a student.

This option should ask the user for an ID if the ID was found your program should ask for the course code and then remove it from that student.

- If the input (ID or course code) was not found your program should display an appropriate error message.

- After your program shows the output it must wait until the user press “Enter” to show the main menu

Option 6: Add a student info.

This option asks the user for student details(name, ID, Major) without any courses and add it to the lists/dictionaries and shows a success message

- If the entered ID is not valid ID (Check ID Guidelines) your program should display an appropriate error message.
- If the entered ID already exists, your program should display an appropriate error message.
- After your program shows the output it must wait until the user press “Enter” to show the main menu

Option 7: Delete a student info.

This option asks the user for a student ID and remove it and the correspondent courses then shows a success message.

- If the ID was not found, your program should display an appropriate error message
- After your program shows the output it must wait until the user press “Enter” to show the main menu

Option 8: Save data to the file.

This option saves the modification(add course, drop course, add student, remove student) that happened in this session into the same text file using the same format described in File Structure

Option 9: Quit.

File Structure

The file is structured such that each student has his own line where his information is separated by comma “,”, and each registered course is separated from the other by semicolon “;”

```
Student Name, ID, Major, Course Code 1, Course Name 1, Credit hours 1, Course_1 Grade; Course Code 2, Course Name 2, Credit hours 2, Course 2 Grade
```

A Sample of student line:

```
Ahmed Khaled,202012340,ME,MATH101,Calculus I,3,A;ICS104,Intro to Prog in Python & C,3,B+
```

Notes:

1. Although this pdf file shows the sample in 2 lines, in real life there is no lines between the text and it is only a problem in formatting
2. Each course must have a grade (assuming that all info is for previous terms, not the current term

Course Points and GPA Calculations

- Course points calculation formula
  - $letter\ grade\ weight\ \times\ course's\ credit\ hours$
- Letter grades weights:

Letter Grade	Weight
A+	4.00
A	3.75
B+	3.50
B	3.00
C+	2.50
C	2.00
D+	1.50
D	1.00
F	0.00

- Calculation Example
  - 3 credit hours course with letter grade "B+" → (3.5 * 3 = 10.5)
  - 4 credit hours course with letter grade "A" → (3.75 * 4 = 15)
  - $\boldsymbol{Total\ GPA} = \frac{\text{Total Course points}}{\text{Total Credit hours}}$
    - (10.5 + 15) / (4 + 3) = 25.5 / 7 = 3.64 (round to 2 digits)

ID Guidelines

The student ID must be <u>9 digits length </u>and <u>start with 2 and ends with 0</u>

The following items must be observed when you write your code:

- Comments are important they are worth. (worth 5%)
- The code must use meaningful variable names and modular programming (worth 10%)
- Global variables are not allowed. You should learn how to pass parameters to functions and receive results.
- You must submit a working program. Non-working parts can be submitted separately. If a team submits a non-working program, it loses 20% of the grade.
- User input must be validated by the program i.e. valid range and valid type
- Your code has to be limited to the material covered in the lectures and Lab.
- The deadline for submitting the lab project is **Friday December 4 before midnight**.
- Submitting Saturday before midnight will lead to 5% penalty
- Submitting Sunday before midnight 15% penalty

Deliverable:

Each team has to submit:

- The code as a Jupyter notebook
- The report as part of the Jupyter notebook or as a separate word file. The report will describe how you solved the problem. In addition, you need to describe the different functions with their task and screen shots of your running code. (worth 5%)

Lab demo:

- The week of **December 6 to 10** will be used for lab project demos.
- A slot of 15 minutes will be allocated to each team for their demo and instructor questions
- Students who do not appear for lab demo will get 0 even if his teammate attended.