CS401 Project

1. Project Overview

Create an interactive menu driven GPAMS (Grade Point Average Management) system. The project requirement does not ask GUI (Graphic User Interface) at this time but if you want to pursue, please do so. If your GUI is easy to operate and has nice layout, you may be qualified to earn up to 5% extra credit. Command window GUI is not considered as real GUI.

2. Project Requirements

This project must be done under following environments:

Programming language: Java ONLY

Virus Free Project

Under IIT department of Computer Science Policy, any project submission with virus should be excluded from the evaluation. It means, TA will not continue to evaluate your project material. The final grade of project will be 0. To avoid any potential debate, everyone should follow the department rule. Before you submit your project, you have to check your project package (files) with the latest antivirus software with virus-free computers. TA has rights to withdraw a project with computer virus.

Academic Honesty

IIT expects students maintain high standards of academic integrity. This project must be planned, implemented, and completed by an individual or a team's efforts. Do not use someone's idea or online resources available in the Internet. If TA finds similarity of codes or behavior with other's work, those will fail the course as well as reporting to the CS department and the graduate school.

3. Detail Description of the Project

Initial Menu

When a GPAMS starts, it shows the following menu:
(* GUI project should have your own creative user interface. Creative interface does not limit but it must consider easiness to use.)

//command line sample of GPAMS menu (not GUI version)

```
IIT CS01 GPAMS Main Menu
    1. Roster Creation
    2. Grade Input
    3. Record Search
    4. Show list
    5. Change record*
    6. Statistics*
    7. Exit

Select?
```

- Option 5 and 6 are for team project

Overall Menu operation:

Each menu should provide convenient or acceptable movement from one menu selection to other menu selection.

Menu 1

When number one from the menu is chosen, create a roster of a course such as:

// number of student per sections, number of homework, number of exams, number of project, etc.

// sample of manual input of menu 1

```
1  Manual Data Feed
2  Default
     Select?  1

*** Manual Data Input ***
Numbers of site?  3      // there are three sites such as live(01), local online(02), other country
                         // site(03)

Numbers of student per site 01?  10
Numbers of student per site 02?  5
Numbers of student per site 03?  5

Total class size: 20

Numbers of Assignment?  10
Numbers of Project?   1
Numbers of Exam?   2
```

Create a proper data structure.

// sample of default input of menu 1

```
1    Manual Data Feed
2    Default
     Select?  2
```

Create a data structure from a data file (above manual example shows default values).

Menu 2

When number one from the menu is chosen, it prompts the user to enter the data file:

//example of menu 2
Enter data file:

The GPAMS (GPA Management System) reads the following data from a given data file:

Data Field	Data Max. Length (Byte)	Value
First Name	20	Any
Last Name	20	Any
*SID	10	Any
*Class Number	5	Any (less than or equal 5 char code)
*Site	3	Any
HW1 .. HWn*	INT	0 .. 10

Project	Float	0 .. 20
Midterm	INT	0 .. 100
Final	INT	0 .. 100

*Note: SID = Student ID (like social security number)
*Note: Class Number such as CS401, CS402, CS542, etc.
*Note: Site (We assume that a group of people have the same Site)
*Note: HWn = The maximum number of homework will be determined automatically.
How? Think about it.

First, GPAMS reads data from a file. The default data filename is ***'grades'*** but the data filename can be different (e.g. dt, cs401grade, etc.). The default location of data file is the same location of GPAMS; however, it can exist anywhere in the user's file system (`e.g. dt, c:\cs401\project\dt, etc`.).

Each row of the data file shows scores for one person. For example,

//example of data file
Michael Jordan 9998887777 CS401 01 10 9 8 10 9 9 8 7 7 9 10 9 18 88 95
Cindy Crawford 9995671234 CS401 03 9 8 7 6 6 7 8 9 9 10 8 8 10 97 85
…

Next, read the data from the file and make a list.
Use an array implementation.
Menu 1 and 2 can be selected multiple times. Different class records can be appended separately. If default data is already built in, the next data file name should be different than default file name. Append the new data entries after the end of current list.

After manual data input, go back to main menu (or different in GUI implementation)

Menu 3

//sample of menu 3

Menu 3
1. **Search by first name**
2. **Search by last name**
3. **Search by SID**
4. **Traveling list**
5. **Main Menu**

Select?

If menu 3.1 is chosen, it shows the full data of people who have first name as the keyword. If there are more than one same first name, the result shows all of them, but they are sorted by the last name. If there are multiple same first and last names, then the result is sorted by Site number.
Menu 3.2 follows the same rule as menu 3.1 but last name is the keyword to search.
Menu 3.3 searches by student ID.
Menu 3.4 creates a linked list and it needs two parameters and they are the position number with the forward or backward direction (+ or -).

* How to search from which class? You can design your own way.

* If Peter and Peterson exist in a class, should search the shortest matching name. Thus if you search “Peter”, Peterson is not returned.

*NOTE: GUI implementation project can have different manners to traveling. The below example shows command line implementation case.

```
Current location is: Foster, Jody 01

Menu 3.4 where to travel? 3 +  //for example

Travel: Foster, Jody 01 >> Foster, Larry 01 >> Jobs, Steve 01 >> Lancaster, Burt 02
```

//other sample

```
Current location is: Foster, Jody 01

Menu 3.4 where to travel? 2 -

Travel: Crawford, Cindy 01 << Dodson, Nina 01 << Foster, Jody 01
```

If the reference reaches the end node or the head node before moving forwards or backwards to the designated position, output shows an error message like “Sorry, it’s beyond the scope.” and shows list up to the end (NULL for the last node, listData for the first node). You can have your own error message(s).

//sample for forwards

```
Current location is: Jobs, Steve 01

Menu 3.4 where to travel? 4 +

Sorry, can’t go that far. Burt Lancaster is the last member of the list.
Travel: Jobs, Steve 01 >> Lancaster, Burt 02 >> NULL
```

//sample for backwards

```
Current location is: Foster, Jody 01

Menu 3.4 where to travel? 4 -

Sorry, it’s beyond the scope. Cindy Crawford is the first element of the list.
Travel: listData << Crawford, Cindy 01 << Dodson, Nina 01 << Foster, Jody 01
```

Menu 3.5 returns to the main menu.

Menu 4

```
Menu 4
    0.  Scoring Weight per item
    1.  Sorted list by last name
    2.  Rankings by total score
    3.  Rankings by homework average
    4.  Rankings by project score
```

5. Rankings by grade
6. Main Menu

Select?

Menu 4.0 is a special case.
Menu 4.0 can be selected once. If weight factors of item enter once, all sub-menus of Menu 4 applies the same rule until menu 4.0 modifies new weights.
If total of weight (percentage) of inputs is not equal to 100%, then the system asks any extra item such as quiz or participation, or brings an exception case, etc.
Here is an example:

Final Exam	**30**
Midterm Exam	**30**
Project	**20**
Homework	**20**
Total	**100**

This example means homework takes 20% of overall score, etc.
How to assign weight factors? Up to your design. You can have more items such as class participants, quizzes, extra credits, etc.
*** If these extra items are added, how to handle them with current data structure with default data file that may not have those items? You need to solve it with your brilliant imagination. ***

The whole printing list shows last name and first name in that order.
Menu 4.1 results sorted list (in ascending order) by Class, Site, and last name, which means ascending order of class code first and the smaller Site group must be shown first. Menu 4.1 shows the order of data as given below:

Last name, first name, SID, Site, homework average, project, midterm, final, total score, Grade

*Grade: A >= 90, 80 <= B < 90, 70 <= C < 80, E < 70
* Grade can be automatically assigned by total score.

There is no need to show all homework scores. TS (Total score) will be calculated by the weight factor of menu 4.0:

Menu 4.2, 4.3, and 4.4 are sorted in descending order by TS, HA (Homework Average), and Project scores, which mean that the highest score shows first. If there are multiple tie scores by the key, then the result will be sorted by ascending order of Site and last name.

Menu 4.2 shows per class:
Last name, first name, Site, TS

Menu 4.3 shows per class:
Last name, first name, Site, HA

Menu 4.4 shows per class:
Last name, first name, Site, Project

Menu 4.5 shows list per grade and per course if there are more than one course data.

Menu 5

```
Menu 5
    1. Add record
    2. Remove record
    3. Change record
    4. Main Menu

    Select?
```

Menu 5.1 adds a new student record to the list by manual user interaction.
Menu 5 prompts the input for the following order: HW1 .. HWn accepts homework scores.

```
Last name:
First name:
SID:
Class Number:
Site
HW1:
…
HWn:
Project:
Midterm:
Final:
```

The system interaction interface can be user defined. You can build your own nice interface to user.
After inserting a student record, verifies the new record into the screen and returns to the main menu.

Menu 5.2 deletes a person from the list. Finding a person by different keys is what you can define such as name, SID, or others.

Menu 5.3 changes score(s) of a record(s). It's also a free design part.

Menu 6

```
Menu 6 Statistics
    1. Average Score of class
    2. Percentage of grade
    3. Main Menu

    Select?
```

Menu 6.1 shows averages of all categories:

//sample

```
Statistics per Class
============================
Average of Homework:        17.7
Average of Project:         13.8
```

```
Average of Midterm:          87.2
Average of Final:            82.9
Average of Total Score:      84.4

Statistics per Site
--------------------------------------------------
Class Site:                  01
Average of Homework:         18.7
Average of Project:          12.8
Average of Midterm:          89.2
Average of Final:            85.9
Average of TS:               87.4

Class Site:                  02
.....
```

Menu 6.2 shows the percentage of each grade category (A, B, C, E):

//sample

```
A       35.3%
B       52.9%
C       8.8%
E       0%
```

Menu 7
Exit the project.

Default log
Record all interactions on the screen as a log file: log.txt
After exiting the program by choosing menu 7, a log file will be created and the log file should be readable by normal ASCII editor (e.g. notepad in MS Windows).
The format and information of the logs are your own free design. The logs are used for maintenance purpose.

What to submit?

1. Source codes -- detail of comments/remarks/note
2. Manual document: How to use your project (in detail)
3. Screen shots of instruction in power point presentation file.
4. Final Design diagram document (or flow charts)
5. Self Evaluation form: What can be done, what cannot be done

All files are zipped to one file: **CS401_seatnumber_lastname_firstname_18F.zip**

Only zip file format is recommended. No other compression extension such as *.rar.

Final Project Due Date: ***November 18th Sunday midnight US CST, 2018*** or earlier

Submit to course blackboard Digiral Dropbox

No late submission is allowed at this time.

- **If the BB marks your submission as LATE, 0% of the project under any situation.**
- Strongly suggest to submit your project at least 12 hours earlier than the due date because many other course submission may be the same.

** Submit pre-project **

While you are doing your project, there is a pre-project submission to encourage your progress. You can practice early stages of Software Development Life Cycle. Among 9 steps, you need to practice first four steps. However, pre-project submission does not require that detail stages. If you follow the first four stages such as problem analysis, requirement elicitation, software specification, and Design, it's excellent opportunity for you. If you do, submit all stages documentation. Or you can simply submit your design document and project schedule. Of course, working professionals in the software industry maintain project management skill.

Your design document shows your understanding of the project and diagrams of your coding. It is not a formal design diagram but it shows overall relationships of objects or procedures. Your project schedule shows your progress. You simply need to show how many hours are required to complete each small task that you plan. Per your schedule plan, you need to show by when you can complete a certain part of project or the whole project, by when you can complete testing, documenting, evaluating, debugging, etc. You don't need to be accurate but try to keep it.

20% of total project score if you submit yours.

Due date: **November 1st, Thursday US CST midnight 2018 or earlier**