FINAL GROUP ASSIGNMENT (35%)

Due: 22 September 2023, 11:59 PM (GMT +7)

INTRODUCTION

Your task for this assignment is to design, code (in C++) and test a program for motorbike rental in Viet Nam.

OBJECTIVES

This assignment provides a chance for students to practice and apply all concepts learned so far. It is also an opportunity to get familiar with analysis, design and development of a software application with a practical project idea to get ready for life and work. For simplicity, all interaction with the application will be done via a simple text interface (no GUI is required). However, it will provide all basic functionalities of a practical application (may be further developed after completion).

DESCRIPTION

APP FOR MOTORBIKE RENTAL

Assume that you are involved in a start-up who came up with an idea to make an app for people to rent motorbike. The app will allow registered members to rent motorbike of other members.

There is an initial entry fee of $20 (pay to the system) when **registering** as a member, which earns the new member *20* ***credit points (Note that $1 is equivalent to 1 credit point)***. Initial registrations must capture all the information necessary to be a member, including his/her personal info (*username, full name, phone number, id type (Citizen ID/Passport), id/passport number, driver's license number, expiry date*), and the info of his/her motorbike if available (*model, color, engine size, transmission mode, year made, description*).

Each member ***earns credit points*** when the motorbike is used by another member and vice versa, and can ***use credit points*** **to rent an available motorbike**. If member does not have enough credit points, he/she needs to **perform a top up** of the credit points using cash ($1 = 1 credit point).

The system will track two separate **score ratings** which can vary from **1** to **10**.

- The **motorbike-rating score** will reflect the condition of the motorbike based on average of ratings of the renters who have used the motorbike in the past.
- The **renter-rating score** is derived by averaging the ratings of all motorbike owners who have rented their motorbike to that renter (how well that renter has taken care of the motorbike).

Any member can **list his/her motorbike** to be used in a particular **period** (start time – end time), **consuming points** (per day), and with or without specifying ***minimum required renter-rating*** for potential renters. The *minimum renter-rating* can thus be used to prevent the motorbike being rented to poor rating renters or those for which no history is available. Of course, the member can **unlist the motorbike** if needed.

Any member can **view all available motorbikes** in a **specified time** (start time – end time), and **city in which the motorbike is located**. The display list should consist of only those motorbikes for which the member has *adequate credit points and renter-rating*.

The information should also include the *motorbike-rating score* of the motorbike. Members should also be given the option to **view the reviews** (score and comments) on any of the listed motorbikes.

Any member can **request to rent** any motorbike for which they are eligible by using their credit points.

The motorbike owner can **view all requests for his/her motorbike**, including *info and rating of the interested renters,* and may **choose to *a request to accept*** *(reject all other requests with overlapped time)*.

Non-members can **view all motorbike details** (but not their reviews, i.e. score and comments), to encourage non-members to join. Non-members can only rent the motorbike if they register as a member.

CONSTRAINTS:
For the initial version of the app, some constraints are applied as below:

1. Each member can only add one motorbike to his/her account.
2. Each member can only occupy one motorbike at a time.
3. Members are not allowed to cancel if the request is already accepted.
4. The application is only available to users in Ha Noi and Sai Gon.

ASSUMPTIONS:
Transaction that involves the top up of credit points using cash can be authorized with the member's password only.

PROCESS AND GUIDE

A. **Class Diagram:** Identify classes, attributes, methods and relationships between classes. Sketch a class diagram for the application.

B. ***Basic Features:***
Implement and test each feature. The application should have the following basic features:
Note: The program should validate user's input accordingly. All attributes should be private for data encapsulation.

1. A non-member can register to become a member (information is recorded).
2. A non-member can view all motorbike details (but not their reviews).
3. A non-member can rent the motorbike if he/she has registered as a member.
4. A member can login with the registered username and password, and can view all of his/her information.
5. An admin can login with predefined username and password, and can view information of all members and motorbikes.
6. A member can list his/her motorbike available to be rented (with consuming points and rental amount per day, and minimum required renter-rating), and unlist it if wanted.
7. A member can search for all available **suitable** motorbikes for a particular **city** (suitable with his current credit points and rating score).
   Example:
   My credit is 20 and my rating score = 6
   There is a motorbike with consuming point: 8 credit points/day and minimum required score is 7.
   - ➔ This motorbike won't be available in my search (since I do not have enough rating score for it).

8. A member can request to rent a motorbike.
9. A member can view all requests to his listed motorbike.
10. A member can accept a request (reject all other requests).
11. A member can rent the successfully requested motorbike.
12. A member can rate each of his/her rented motorbike (*score and comments*).
13. A member (motorbike owner) can rate each of the renters who had used his/her motorbike (*score and comments*).
14. All data must be saved into a **file (appdata.txt)** before the program is ended, and loaded into the program when it is started.

C. ***Time Period Feature***: if you already completed all basic features, improve the application by supporting time period matter for the program as in its description, especially for features 6-10 *(e.g. list the motorbike available for a time period; request to use the motorbike in a period; only automatically reject non-accepted requests with overlapped time, etc.).*

NOTE: For GROUPS OF THREE (most groups have 4 members, but there are groups with three members due to unforeseen circumstances),

1. motorbike-rating score and all features related to motorbike-rating score are exempted (not required to implement).
2. Time period feature is required to be implemented with ONLY ONE DAY (i.e. list the motorbike to be available in only one particular day, request to use the motorbike for only one day, etc.).

D. Welcome Screen

When completing your application, it should have a welcome screen with example content structure as below (you are free to adjust its format if prefer. Note: … in the example content means so on and so forth depends on what you want to show).

```
EEET2482/COSC2082 ASSIGNMENT
MOTORBIKE RENTAL APPLICATION

Instructor: Dr. Ling Huo Chong
Group: Group No.
sXXXXXXX, Student Name
sXXXXXXX, Student Name
sXXXXXXX, Student Name
sXXXXXXX, Student Name

Use the app as 1. Guest   2. Member   3. Admin

Enter your choice: 2
Enter username:
...

This is your menu:
0.  Exit
1.  View Information
2.  ...

Enter your choice: 1
...
```

IMPORTANT INSTRUCTIONS

1. You are required to write your program in C++ programming language only.
2. You will need to demonstrate the OOP skills that you have learned in the course.
3. You must thoroughly document your code using comments (// or /* … */).

4. Your code should be separated logically into multiple source code (.cpp) and header (.h) files that overall have a single purpose, i.e. your code should be structured in a way that each file has a clear scope and goal. Marks will be deducted if you put everything together into one single source code file. Make sure you use the header files and header guard correctly.

ACADEMIC INTEGRITY

This assessment requires that you meet RMIT's expectations for academic integrity. You must not ask for or accept help from anyone else on completing the tasks. You must not show your work to another student/group enrolled in this course who might gain unfair advantage from it. These things are considered plagiarism or collusion. Plagiarism is a form of cheating. It is the presentation of the work, idea or creation of another person as though it is your own. If you use any resources, write an acknowledgment in the file where the resources were used. Any submission found in whole or in part plagiarized will be considered as plagiarism. To be on the safe side, never ever release your code to the public. For example, if you use GitHub to store your code, make sure that you set the repository as private. For more information, visit RMIT's website for Academic Integrity and Consequence of Plagiarism.

The penalty for plagiarism is extremely severe at RMIT. If you are found to plagiarize, you will receive a mark of zero for an assessment or even an entire course. Repeated plagiarism will lead to exclusion from RMIT.

RUBRIC

Refer to the Rubric published in the Group Assignment page in the course Canvas.

SUBMISSION INSTRUCTIONS

As a group, submit following files separately (*only ONE member needs to do the submission***) to the course Canvas, DON'T zip them together:**

1. An executable file of your program, i.e. ***GroupX_Program.exe*** *(where X denotes your group number).*
2. A zip file of all source code including .h and .cpp files and data file, i.e. ***GroupX_Sources.zip.***
3. A report together with class diagram and explanation in PDF format, i.e. ***GroupX_Report.pdf***. Refer to the report template provided in the Group Assignment page in the course Canvas.
4. Record of Individual Contribution in PDF format, i.e. ***GroupX_Individual_Contribution.pdf.*** Refer to the Individual Contribution form available in the Group Assignment page in the course Canvas.

Note: *The files must be also submitted separately to TURNITIN, and make sure your executable file can run. If you do not submit it, or it fails to run/crashes, you may lose up to 50% of the grade (although some marks may be given for the effort).*

Late work: 10% penalty per day. Submission later than 03 days will not be accepted and a mark of zero will be given.

--- END OF ASSIGNMENT ---