ENGG1340 / COMP2113 Course Project (2023 2BC)

Introduction

In this project, you will apply what you have learned in this course (e.g., github usage, C/C++ programming, etc.) and **implement a text-based mini-game of your own choice**. A text-based game is a console application that can be accessed using the Linux terminal and via SSH. Please pay attention to the project requirements, scope, and submission details in this instruction.

Please also notice the late submission and plagiarism policies available in the course information on Moodle.

Milestones and Timeline

- Stage 1: Grouping (due: March 16, ~3 weeks from now)
- Stage 2: Project submission (due: April 27, ~9 weeks from now)

Check what you need to submit and note carefully the specific requirements for the project.

Form Your Group

You must work in a group of 2 - 5 for the project. Please make sure that you have indicated your grouping on Moodle, as some submissions would be done on a group basis on Moodle. Grouping cannot be changed after the grouping deadline. You may make use of the Moodle forum to find project partners.

Please indicate your grouping on this Moodle page and have your project partners join the same group.

Examples for Your Reference

There are several examples of text-based games available for your reference. You should not copy any code from the Internet directly.

Caveat: Plagiarism is strictly prohibited. Please ensure that all your project submissions are plagiarism-free, or you risk losing all marks in the project. Please refer to "What is plagiarism?" - HKU T&L for the definition and scope of plagiarism.

- Honkonmon Go
- Blackjack
- Minesweeper

Requirements

- **Code Requirement.** Your implementation should include all the following coding elements:
  - Generation of random game sets or events
  - Data structures for storing game status (e.g., arrays, STL containers)
  - Dynamic memory management (e.g., dynamic arrays, linked lists, STL containers)
  - File input/output (e.g., for loading/saving game status)
  - Program codes in multiple files (recall separate compilation)
  - Proper indentation and naming styles
  - In-code documentation
- All project files should be committed to a private GitHub repo.
- **Contributions:** Each student should contribute certain added (+) lines of the code and documents to their project, gauged by GitHub's "Contributors Graph".
  - For a group of 2, each student should contribute at least 25%.
  - For a group of 3, each student should contribute at least 20%.
  - For a group of 4 - 5, each student should contribute at least 10%.
  - If you cannot meet the contribution percentage and fail to provide reasonable excuses in the self-evaluation, you will receive no mark for the project.
- Commit comments should not be empty and should be written sensibly.
- For each function, comments on "what it does", "what the inputs are", and "what the outputs are" are needed.
- You may use any C/C++ libraries that don't require any additional installation on the grader's side. That means the grader won't download any libraries from the internet for you, and your program will be considered not compliable because of missing dependencies. To use a library that requires additional installation, you should either put the necessary files from a library together with your code so that an additional installation is not needed, or you shouldn't use the library at all.
- You should make sure your code can be successfully compiled in the Computer Science Department's "academy" server.

Stage 1: Grouping

The purpose of this milestone is to help you define the project scope and consolidate your ideas before you start to implement.

What to do for this milestone:

- To form a project group and indicate your group members as mentioned in the previous section.
- To identify a text-based game type that you would like to implement. Or design your own game.
- To decide what features your text-based game would have.
- To set up a private GitHub repo where your work will be hosted.

What to do (by each group)

- Set up a private Github repo that includes all group members and the teaching team account engg1340comp2113.
- You can refer to Inviting collaborators to a personal repository GitHub Docs for the procedure to add collaborators.
- The teaching team will accept the invitation within 7 days. If we fail to do so, please add again.

Reference:

- Getting started with writing and formatting on GitHub;
- About READMEs

Stage 2: Project Submission

What to do for this milestone:

- To implement the text-based game.
- To document your code.

What to submit (by each group)

- In the same Github repo that you set up in Stage 1.
  - Create a README.MD file in the repo that contains:
    - Identification of the team members.
    - A description of your game and introduce the game rules.
    - A list of features that you **have implemented** and explain how each coding element 1 to 5 listed under the **coding requirements** aforementioned support your features.
    - A list of non-standard C/C++ libraries, if any, that are used in your work and integrated to your code repo. Please also indicate what features in your game are supported by these libraries.
    - Compilation and execution instructions. This serves like a "Quick start" of your game. The teaching team will follow your instructions to compile and run your game.
  - Your code including **the makefile and all source files (.h / .cpp / .c)**
- Submit a link to the repo to Moodle. (Refer to the details on the submission page)
- A **video** (at most 3 minutes long) demonstrating a gameplay and the implemented features of your program.

    - You can either submit a link to the video or upload it to Moodle, and add that in the README.MD file.
    - Video clips after 3 mins would be ignored if your demo is longer than 3 minutes.

What to submit (by each individual student)

- Submit on Moodle (Refer to the details on the submission page)
- A list of work that you have done for the project.
- A peer evaluation of your project partners in your group.
- To declare your own contribution with explanations if the Contributions Graph in GitHub does not reflect the real contribution distribution.

Weighting

Your work will be assessed with regard to the following aspects: Project total (15%)

- Problem statement (including setting & assumptions) (1.5%)
- Implementation (including items 1 to 5 listed under coding requirements) (7.5%)
- Program functionality and special features (including creative and fun elements) (2%)
- Documentation (including README.MD and in-code comments) and coding style (1.5%)
- Video demonstration (1%)
- Collaboration (1.5%)