Programming Fundamentals

Individual Project (20%)

[Deadline: 12:00:00 noon 21st April 2022 (Thu)]

Objectives

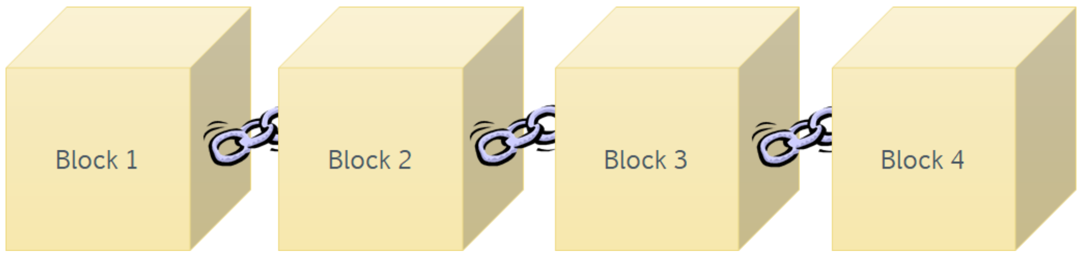
1. Acquire the skills and experience of implementing a medium-scale C++-based software system.
2. Apply sound methodologies and practices to solving problems using C++.
3. Experience how a system can be commented and documented properly for references by potential audiences (e.g., developers, users and operators).
4. Experience how to disseminate ideas efficiently through program demonstration.

Topics

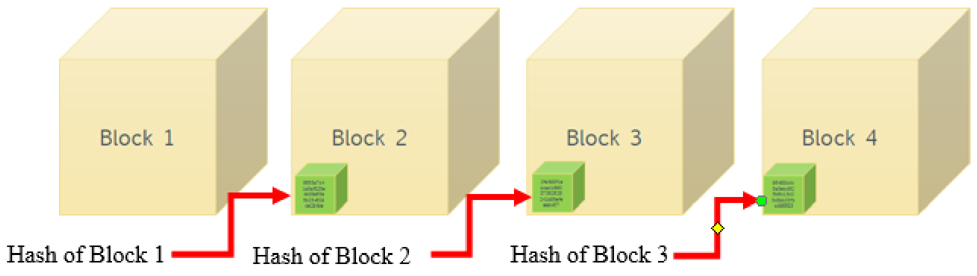
Choose ONE of the following three topics:

Topic 1 A Blockchain Implementation

Blockchain, a cryptographic tool, is extensively used in Fintech applications nowadays for protecting chronological order of transactions.



The blocks, essentially contains information, are “linked” together by enforcing a specific relationship between blocks. That is, a block must contain a “fingerprint”, which is a hash value, of the data of the previous block. A hash function can condense arbitrary message (the block information) to a fixed size (e.g., 160 bits) and produces a fingerprint of the message.



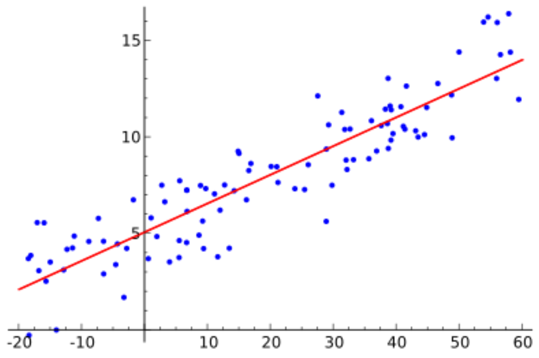
After a block is added to the end of the chain, it can neither be removed or modified. If there are malicious changes to the transaction data, it can be detected by checking the hash value stored in the next block.

In short, blockchain is essentially a linked list data structure, which links the transaction data stored in blocks. And the integrity of the blocks can be verified by using hash values.

Develop a C++ program that allows the addition of new blocks, each includes certain information, defined by yourself, to the chain. Also, the program should be able to display the chain data in a number of ways, (e.g., by block number, the data of the whole chain, the hash values of a particular block). Also, provide a function that allows the users to check the integrity of the data stored in the chain. For the hash function, adopt the source code here.

Topic 2 Learning Linear Regression

Linear regression is a mathematical technique that helps find out the relationship between one or more predictor variable(s) and one outcome variable. One usage is that in supervised machine learning, the model is trained to find the best-fit line between the independent and dependent variable.



(Source: https://en.wikipedia.org/wiki/Linear_regression#/media/File:Linear_regression.svg)

Develop a C++ program that will introduce the concept of linear regression and give at least 3 examples (from beginner level to advanced level) for illustration purpose. Imagine that you are teaching your classmates, but you are only allowed to use your program to teach them on behalf of you. Your program should allow certain interactivity, which may include allowing users to choose the (sub-)concepts to learn, and inputting data (with guidelines by the program) and have a visual representation of the calculation and the outcomes.

Topic 3 Simulation for Social Distancing

Suppose we have a type of virus that causes severe respiratory syndrome and can transmit to other people if they get close. The virus causes severe illness that is more likely in elderly patients and those with certain underlying medical conditions.

Develop a C++ program for scientists to study the impact of vaccination by developing a simulator. In the simulator, N users will walk in an area of X meter * X meter. Some users randomly walk (i.e., random direction with random step length), and others walk along straight lines. When a user reaches the boundary of the area, the user may change the walking direction to a random direction within the area and continue.

When two users move closer within C meters, the virus can transmit to another user. Initially we have R contagious users and V vaccinated users among the total of N users. We assume that vaccinated users will not be transmitted or become contagious (CLAIM: which may not be the case in reality).

In your program, it receives inputs from scientists, e.g., N, X, C, R, and V. You may add other variables for scientists, e.g., walking speed, social distancing rules, percentage of users that violates the social distancing rules, possibility of virus transmission within C meters, etc. The program receives inputs from scientists and visualizes users with different colors indicating their status (e.g., vaccinated, contagious, others). (You can use external libraries for text/background coloring for this topic) The users' locations, their statuses, and total number of users in different statuses should be updated in the simulation.

Requirements

Your program MUST be implemented using text-based mode only. No GUI is allowed.

You are only allowed to use the techniques and libraries introduced in this course. For string manipulation, you are allowed to use any of the C-string or string object approaches. If you need to use other build-in/external libraries other than those specified above, seek the consent from the lecturers first and provide with proper justification.

Deliverables

(1) C++ Program

Put your source files (e.g., .cpp and .h) in a single folder, which is named as your student ID, that can be recognized properly in VS Code. The source file that is the entry point (i.e., it contains the `main()`) of your program should be named as `main.cpp`.

(2) Report

It should contain a description of the project, including the problem statement, the objectives of the program, program design, and structure of the program. Justify your design properly. Also, include a user manual so that "non-programmer" user knows how to use your program. The report should contain at least 5 pages and at most 10 pages.

(3) Demonstration

A 5-minute video that demonstrates and describes your implementation and how it can achieve the project outcome(s).

Assessment Criteria

This project is assessed based on the following rubrics:

	Excellent	**Good**	**Satisfactory**	**Weak**	**Fail**
System Implementation (50 marks)	Develop an application that satisfies all the system requirements. The program logic is well-commented. (41-50 marks)	Develop an application that satisfies most of the system requirements. The program logic is properly commented. (31-40 marks)	Develop an application that satisfies some of the system requirements. The program logic is fairly commented. (15-30 marks)	Develop an application that satisfies a few of the system requirements. The program logic is poorly commented. (1-14 marks)	Unable to develop an application that satisfies the system requirements and/or the source code(s) cannot be compiled successfully. (0 mark)
Report (30 marks)	Identify all requirements and provide a comprehensive system design and sound justifications. (24-30 marks)	Identify most of the requirements, and provide good system design and justifications. (17-23 marks)	Identify some of the requirements, and provide reasonable system design and justifications. (8-16 marks)	Identify a few of the requirements, and provide poor system design and justifications. (1-7 marks)	No requirements, system design and justifications are provided. (0 mark)
Demonstration (20 marks)	The presentation has an attention-grasping introduction and clear demonstration, excellent time management and visual aids. (16-20 marks)	The presentation has a clear demonstration, good time management and visual aids. (11-15 marks)	The presentation has a basic, but reasonable demonstration, fair time management and visual aids. (6-10 marks)	The presentation has a poor demonstration and uses time poorly. The student has trouble bringing visual aids smoothly into the presentation. (1-5 marks)	The presentation does not have a demonstration. (0 mark)

Submission

Follow the steps below:

1. Create a folder and name it as Project_<student no>_<your name>.
   E.g., Project_12345678d_CHANTaiMan
2. Put the three deliverables to the folder.
3. Compress the folder (.zip, .7z, or .rar).
4. Submit the file to Blackboard.

Any wrong file naming and submission will be given ZERO mark for the whole project. If your program cannot be compiled, ZERO mark will be awarded for Deliverable (1).

A maximum of 5 attempts for submission are allowed. Only the last attempt will be assessed.

The deadline of this assignment is **12:00:00 noon 21st April 2022 (Thu)**. No late submission is allowed.

This assignment is an individual work. All work must be done on your own. Plagiarism is serious offence. The Moss (https://theory.stanford.edu/~aiken/moss/) system will be adopted for plagiarism checking. Submissions with high similarity, in terms of code patterns and structures, in addition to direct-copy-and-paste, will be treated as plagiarism. Copying code from web resources is prohibited as well. Any plagiarism cases (both copier and copiee) will be given ZERO mark in this assignment.