Data Structures and Algorithms
CS222 - Capstone Project 2022

Total Points:	50
Assigned:	Tuesday, March 22, 2022

Instructions

- For your capstone project, you are highly encouraged to work in pairs or a team of not more than 3 members. Make sure, you partner with those in the same cohort as you, for assessment purposes. Your goal is to solve an interesting problem using data structures. A few sample problems are given below, however, you are flexible to propose your own.

Project Requirements

- Your solution must use **two** or more data structures, at least one should be a linear data structure (Array, List, Stack, Queue e.t.c) and one should be a non-linear (Tree, Graph, Hash Table, Maps e.t.c). You are free to use data structures implemented in the Java Collections API, data structure implementations from the textbook, or implementations we have used in lab/class or from scratch.
- You may not search for an implementation (source code) of a solution to your problem on the Internet or in any other source, nor may you refer to such an implementation if you happen to come across one. However, you are free to (and should) read about your problem and the general approach to solve it. You must cite (APA-style) any references you use.
- In deciding what data structure(s) to use, you should think about **efficiency**. You may find that it sometimes make sense to store the same data in different kinds of data structures, to enable efficient implementation of various operations. For example, I might chose to store student names & data in a hashtable (to enable quick lookup), but also store the names in a list (to keep track of the order in which they joined the class)
- Each member of the team must participate in all aspects of the project –research, design, implementation, and reporting. Although the project will be submitted as a team, each member will also submit an individual "portfolio" showcasing about a page or two of code that they have written and are most proud of. This code should be well structured and have some significant logic and/or use of data structures. (For instance, a collection of only get & set methods for a class would not be considered significant logic).

Sample Project Ideas / Problems

Project Idea 1: Research and application of a new data structure

This class covers basic data structures used as a foundation of problem-solving in computer science. There are many more specialized data structures that are used widely for various tasks in computer science. Pick one of these more specialized/advanced data

structures, not covered in the class, conduct research on how it works and what it is used for, and demonstrate its application to a given interesting problem. Some examples are below:

- **Graphs**. Used to model any real-world network (e.g., a social network, road network, airline network, the internet, water distribution network, etc.) Various graph algorithms can be used to answer questions such as the shortest distance between two points, the maximum and minimum amounts that can flow between a network, etc. For example, given a maze or a network of roads, you could implement Dijkstra's algorithm to find the shortest path from a start location to a goal location. There's a good chapter on graphs in our textbook.

- **Huffman trees.** A type of tree called a Huffman tree can be used to encode text to compress it. You can implement a system to encode text for compression and decode it for decompression.

- **Quadtrees**. Applications include detecting collisions in two-dimensional space (e.g., for computer games and robotics); storing sparse data (e.g., for a spreadsheet); spatial indexing in spatial databases such as GIS, etc.

- **Bloom filters**. Can be thought of as a probabilistic version of a hash table that can be used to store information about huge amounts of data, to answer the question of whether a particular piece of data is present. Often used to implement caches in computers.

- **Trie**. A type of tree often used in text prediction algorithms. It can be used to implement a phone contacts app which displays all matching names/numbers when you type a partial name/number.

- **Rope.** A type of tree often used to store long strings that might be edited. Often used in a word processing applications.

Project Idea 2: Game AI

The textbook, in section 8.4.2, describes how a breadth-first traversal of a game tree can be used to find a solution to a game. This problem requires you to apply this concept to solve a 1-person game (puzzle) such as an 8-puzzle[1] or Sudoku[2].

Project Idea 3: Model a Social network

Model a social network (such as Facebook/Twitter), which holds information about and connects individuals (e.g., individuals may have friends and/or followers). You should support functionality such as having a new user join, having new connections between users (e.g., new friendships), being able to determine the number of degrees of separation

[1] If you are not familiar with the 8-Puzzle, you can learn by playing it here: http://www.himanshug.info/8puzzle/8puzzle.html
[2] You can learn about Sudoku here: http://www.sudokukingdom.com/rules.php

between two users on the network, etc. While your programme is running, the information must be stored using appropriate data structures to ensure efficient implementation of the functionality supported by your information system. When the programme exits, data should be written to a file so that it can be read in next time the programme starts. In that manner, information is persisted between runs of the programme.

Project Idea 4: Modeling and simulation

Create a programme that simulates various processes in a particular context. E.g., you might simulate a grocery store, which involves processes such as allowing customers to queue at various check out points. Similarly, you could simulate processes at a restaurant, cafeteria, factory, or airport. You could also simulate networks such as an airport network, a public transportation network, or the Internet itself. You could create a traffic simulation which engineers can use to investigate the effect of changing various road infrastructure.

Project Idea 5: Information System (e.g., student information system, customer information system, employee information system, etc.)

Create an information system that stores information about relevant entities (e.g., a student information system will store information about students). The information system must support standard functionality like being able to enter information about a new entity, being able to look up information about a given entity, and being able to delete information about a given entity, but it must also have specialized functionality relevant to the specific information system being implemented (e.g. with a student information system, you may wish to be able to display an alphabetical list of all students in a given class, or list all students with GPAs above a certain value). Like in the case of Sample Problem 2, information should be persisted between runs of the programme.

Project Deliverables and Timeline

Milestone 1: Monday, April 4, 2022

- Choice of project partner(s) and project idea (*Working topic*).

Milestone 2: Friday, April 8, 2022

- Problem description, high level overview description of solution architecture and algorithm (*Project Charter*).

Milestone 3: Wednesday, April 20, 2022

- First draft of Report covering these sections: - Problem description & detailed description of solution architecture and algorithm to be implemented.
- Code review of implementation progress

Milestone 4: Wednesday, April 27, 2022

- Completed code submission
- Final report submission
- Individual Portfolio submission

Specification of Report

Your report should have an informative title that reflects the problem you are solving. You should also put the names of the authors (First name(s) (*Case letters e.g. Peter John*), SURNAME (*in capital letters e.g. WILLIAMS*) below the title of the report.

The report should have the following sections:

- **Introduction**: Briefly describes the problem to be solved and why it is an interesting and/or important problem.
- **Background**: Presents, with references cited in the APA style, any background information needed to understand the problem or the approach. For example, the Game AI problem, an explanation of breadth first search in a game tree could go here.
- **Methodology**: Presents your high-level solution architecture and describes your approach to solving your problem. This includes a breakdown of your problem into key modules/pieces of functionality, a specification of the algorithm for solving each key piece (you may use structured pseudocode or a flowchart for this), an explanation of the data structures chosen, and why, and a specification of the classes comprising your implementation (you may use UML diagrams here)
- **Results**: The outcome of your work. What data did you test your solution with, and what was the outcome? Did things work as expected? What, if any, are the shortcomings or limitations of your approach?
- **Conclusion**: Summarize the project here, and identify any key lessons learned. You can also identify possible directions for future work.
- **References**: A list of any sources referred to in your text, in the APA format. It is expected that you did some background reading either about your problem or your approach or the data structure that you are going to use. As such, you should have at least 1 or 2 references here. Note, however, that academic work should not cite Wikipedia. You may, however, cite books, journal articles, online articles/blogs/websites with an identifiable author (either an individual or an organization), etc. YouTube channels are not also ideally respected sources of information as they are not peer-evaluated as per the academic expectations.

Grading Scheme

Problem Modeling, Solution Design/Methodology: 10 points
This portion of the grade evaluates your approach to the problem as judged both from the the "Methodology" part of your report, and from your implementation/code.

- Does the project meet the requirement of using at least 2 data structures, at least one of which is not a sequential/linear data structure?
- Is the choice of data structures appropriate for the problem being solved?
- Are the chosen data structures being used in a way that is appropriate for the problem being solved (i.e., to enable an efficient solution approach)?
- Are the algorithms / approaches being used to solve the problem correct and efficient?

Functionality & Implementation: 10 points

- How well does the implemented solution work? Does it solve the chosen problem?

- How well is the code structured? Is it well organized and modularized (i.e., broken down into appropriate classes and methods)?
- Does the implementation use good programming style, proper indentation, formatting, etc?

Report: 10 points
- Is the report well organized?
- Are the explanations clear, concise, and precise?
- Is the report free from grammatical and typographical errors?

Individual code portfolio: 20 points
Your individual code portfolio is to showcase about a page or two of code that you have written and are most proud of.
- Does the code have some significant logic and/or use of data structures?
- How well does the implemented solution work? Does it solve the chosen problem?
- How well is the code structured? Is it well organized and modularized (i.e., broken down into appropriate classes and methods)?
- Does the implementation use good programming style, proper indentation, formatting, etc.?

NOTE: Milestones 2 and 3 are aimed at helping you structure your work and give you an opportunity to get feedback early. They do not, on their own, carry any points. But *not* submitting work representing a good faith effort for those milestones can carry a penalty of up to 5 points for each milestone.