v

CS222: Data Structures and Algorithms
Fall Semester (Aug - Dec 2023)
FINAL PROJECT

STATEMENT ABOUT ACADEMIC HONESTY AND INTEGRITY

Academic honesty and integrity are very important at Ashesi and central to the achievement of our mission: To train a new generation of ethical and entrepreneurial leaders in Africa to cultivate within our students the critical thinking skills, concern for others, and the courage it will take to transform a continent. As this mission is our moral campus, we recommend you take it seriously in this course without any exceptions at all.

Ashesi therefore does not condone any form of academic dishonesty, including plagiarism and cheating on tests and assessments, amongst other such practices. Ashesi requires students to always do their own assignments and to produce their own academic work unless given a group assignment.

As stated in Ashesi's student handbook, Section 7.4:

"Academic dishonesty includes plagiarism, unauthorized exchange of information or use of material during an examination, unauthorized transfer of information or completed work among students, use of the same paper in more than one course, unauthorized collaboration on assignments, and other unethical behaviour. Disciplinary action will be taken against perpetrators of academic dishonesty."

All forms of academic dishonesty are viewed as misconduct under Ashesi Student Rules and Regulations. Students who make themselves guilty of academic dishonesty will be brought before the Ashesi Judicial Committee and such lack of academic integrity will have serious consequences for your academic records.

OBJECTIVE:

This FINAL project is not only part of the final evaluation of the course but aimed at encouraging you to think critically about data structure and algorithm choices, giving you practical experience with complex data organization, and teaching you about the design considerations regarding the data structures and algorithms behind a project. The three-week time frame should be sufficient for groups of four members to divide tasks and integrate your work into a cohesive application.

INSTRUCTIONS:

1. This is a group project. You should work in groups of four to do the project and are free to select your group members. However, a mix of good, average and weak programmers is advised in each group.
2. A group can choose a project from any of the five project specifications given in this document. Every project given will involve an equal amount of work so it is encouraged to pick any project that interests you.
3. The Java programming language should be used to perform the project.
4. All the projects are in-memory projects and there is no need to use persistent memory or database.
5. A simple text-based (CLI) interface that allows users to interact with the system would be fine.
6. The in-built data structures of the Java collections framework as appropriate can be used.
7. **Evaluation Criteria**:
    a. Effective use of data structures to store, manage, and search
    b. Functionality as given in the specification and the expected outcomes.
    c. Code quality and documentation, including clear variable names, modular functions, and comments explaining the logic.
    d. A presentation and demonstration that reflects the contribution of every group member
8. **Deliverables**:
    a. Source code of the complete project.
    b. A README file that includes instructions on how to use the project.
    c. A document that discusses the choice of data structures, algorithms, and their complexity.
9. Submit all the deliverables as a single zip file.
10. The deadline for submission: **01 December 2023, 11:59 PM** (Hard Deadline).
11. The presentation schedule will be announced once the final exam schedule is communicated by the registry.

Project 1: Virtual File System Organizer

Project Description:
This project will simulate a virtual file system organizer to manage a collection of virtual files and directories. The application will allow users to create, delete, move, and search for files or directories within the system. It will not involve any real file manipulation on the disk, but will instead use in-memory data structures to simulate these operations.

Core Features, Ideas, and Hints:

- ✓ Directory Structure: Use a binary tree to maintain a hierarchical directory structure.
- ✓ File Searching: Implement a hashing mechanism to quickly search for files by their names.
- ✓ Path Resolution: Use stacks or queues to resolve file paths and navigate through the directory tree.
- ✓ File Manipulation: Implement functionalities to create, delete, and move virtual files or directories within the tree structure.
- ✓ Storage Visualization: Provide a method to display the file system tree, showing all directories and files.

Expected Outcomes:

- ✓ You should be able to choose the appropriate data structure and/or algorithm for each functionality.
- ✓ The application should efficiently handle the basic file system operations mentioned above.
- ✓ You should document the time complexity for each operation and discuss why certain data structures were chosen.

Project 2: Event Scheduler and Calendar

Project Description:
This project allows users to create, modify, delete, and view events within a virtual calendar. The application will not require any persistent storage and will store all data in memory using appropriate data structures.

Core Features, Ideas, and Hints:
- ✓ Event Creation: Users can create events with a title, description, date, and time.
- ✓ Event Modification: Events can be edited or deleted after they are created.
- ✓ Daily View: Display all events for a single day, using a list or stack to manage daily events.
- ✓ Monthly View: Use a hash table to quickly access events on any given day within a month.
- ✓ Event Conflicts: Implement an algorithm to detect and warn about event scheduling conflicts.
- ✓ Event Reminders: Use a priority queue to manage and display upcoming event reminders.

Expected Outcomes:
- ✓ ou should demonstrate the use of different data structures to handle various aspects of the application.
- ✓ The application should provide quick access to events based on dates and handle conflicts efficiently.
- ✓ You should explain your choice of data structures for the calendar's requirements and functionalities.

Project 3: Interactive Restaurant Reservation System

Project Description:
This project involves creating a simulation of a restaurant reservation system that allows users to book tables, cancel reservations, and view reservation statuses. The system will operate entirely in memory without the need for persistent storage, relying on data structures to manage the state and operations.

Core Features, Ideas, and Hints:

- ✓ Table Management: Implement a binary tree to represent table availability based on seating capacity.
- ✓ Reservation Bookings: Allow users to book a table with specific criteria (e.g., date, time, size).
- ✓ Cancellation and Rescheduling: Enable users to cancel or reschedule their reservations.
- ✓ Reservation Lookup: Use a hash table to quickly find and view reservations by the customer's name.
- ✓ Waitlist Management: Implement a queue to manage the waitlist for customers without reservations.
- ✓ Occupancy Overview: Provide a visual (text-based) display showing the current occupancy status of the restaurant.

Expected Outcomes:

- ✓ You should be able to select and implement suitable data structures for booking and managing reservations.
- ✓ The system should efficiently handle the dynamics of restaurant reservations and waitlisting.
- ✓ You should provide complexity analysis for your algorithms, especially for insertion and deletion operations.

Project 4: Library Management System (LMS)

Project Description:
Develop a simple in-memory Library Management System that allows users to interact with a library catalogue. The system will allow for adding books, checking them in and out, searching for books, and managing book availability.

Core Features, Ideas, and Hints:
- ✓ Catalogue Management: Use a hash table to manage the book catalogue for quick retrieval.
- ✓ Checkout System: Implement a queue system to handle book checkouts and returns.
- ✓ Search Functionality: Enable searching for books by title, author, or ISBN using efficient search algorithms.
- ✓ Book Availability: Use a binary tree to keep track of available and checked-out books.
- ✓ User Interface: A simple text-based (CLI) interface that allows users to interact with the system.

Expected Outcomes:
- ✓ You will be able to demonstrate the appropriate use of data structures for managing books and user interactions.
- ✓ The system should handle book checkouts and availability updates in real time.
- ✓ You will provide documentation of your design choices and complexity analysis for the implemented features.

Project 5: Traffic Simulation System

Project Description:
You will create a simulation of a traffic system to manage vehicle flow at a four-way intersection. The system will simulate traffic lights, vehicle queues, and the logic that determines when vehicles pass through the intersection. The project will not require persistent storage and will use in-memory data structures.

Core Features, Ideas, and Hints:
- ✓ Intersection Management: Use queues to manage the vehicles waiting at each direction of the intersection.
- ✓ Traffic Light Timing: Implement a system to simulate traffic light changes and manage the flow of vehicles accordingly.
- ✓ Vehicle Arrival: Randomly generate vehicles arriving at the intersection and determine their direction.
- ✓ Traffic Statistics: Maintain a hash table to record statistics like the number of vehicles passed, average wait time, etc.
- ✓ Priority Rules: Implement rules for emergency vehicles, where they are given priority to pass the intersection.

Expected Outcomes:
- ✓ You will be able to demonstrate the use of queues for vehicle management and hash tables for statistics tracking.
- ✓ The system should be able to simulate the traffic flow and provide real-time updates based on traffic light changes.
- ✓ You should provide a complexity analysis of the simulation algorithms.