Group Collaboration Statement

YanBin Huang

- Implemented the MYLIFT algorithm, a custom scheduling method designed to optimize elevator efficiency by considering factors such as request wait time, direction matching, elevator occupancy, and floor priorities. This algorithm extends the shared Algorithm base class and is integrated into both single and multi-elevator modes.
- Developed a graphical user interface (GUI) demonstration to visualize elevator movement, request handling, and scheduling in real time. The GUI provides simulation controls such as start, pause, and reset, and supports both single and multi-elevator views.
- Designed and implemented a multi-elevator system, allowing multiple elevators to operate simultaneously. This involved integrating the RequestDispatcher to allocate floor requests to the most suitable elevator based on distance, direction, and availability.
- Implemented the DirectionalQueue data structure, which efficiently organizes floor requests based on travel direction to minimize elevator idle time and improve response speed.
- Provided necessary test files for validating algorithm performance, ensuring that MYLIFT, LOOK, and SCAN could be tested under various conditions, including different numbers of floors, request distributions, and elevator capacities.
- Created the README documentation, detailing the project structure, system functionality, and instructions for running the different simulation modes (console and GUI).
- Edited video demonstrations showcasing the performance of the implemented scheduling methods, highlighting differences in efficiency and movement logic between the algorithms.

Throughout the project, I applied many fundamental principles of object orientated programming, and utilised priority queues and request management structures to help finalise the project. I contributed to both technical implementation and documentation to ensure the system was fully functional and well-documented

Xuanjin Qu

- Worked on the PriorityQueue implementation, ensuring efficient handling of elevator requests by prioritizing them based on factors such as wait time, direction, and urgency. This helped optimize scheduling by ensuring high-priority requests were serviced first.
- Contributed to the development and refinement of the Request, Queue, PriorityQueue, and Adjustment modules, collaborating with the team to modify and adjust these components for improved functionality.
- Implemented the Request module, which defines the structure and attributes of elevator requests, ensuring that each request includes necessary details such as origin floor, destination floor, and request time.
- Developed and optimized the Queue module, which manages floor requests in a structured manner, allowing smooth integration with the elevator scheduling algorithms.
- Finalized adjustments and modifications to improve system efficiency and stability, working with the team to refine the queue-based request handling system.

Through my contributions, I focused on improving the efficiency and organization of elevator request handling by refining queue-based data structures and prioritization methods. By implementing and optimizing the PriorityQueue, Queue, Request, and Adjustment modules, I helped ensure that the system could process elevator requests smoothly and in an optimized manner. My work supported the integration of intelligent scheduling mechanisms, enhancing the responsiveness and reliability of the elevator control system.

Logan Westwood

My main responsibility in the group project was the completion of the Specification report that required explanation of data structures, algorithm design, and implementation. This required me to have a complete understanding of how our system functioned and how different scheduling algorithms, SCAN, LOOK and MYLIFT, dealt with lift movement and request processing.

I needed to ensure that our implementation details were correct but clear to every member of the team. For this, I wrote with clarity in mind and collaborated with our group's developers to ensure documentation reflected code. This project enhanced my technical writing skill, my proficiency in describing complex concepts, and my understanding of algorithmic efficiency.

Sam Harries

Implemented the ElevatorApp.java file to test and run the LOOK, SCAN and MYLIFT algorithms

- I designed the architecture to allow the program to read from an input file, then runs the current algorithm, and then simulates the elevators movement in a step wise way.

Helped to implement classes such as Building, FloorState, LiftState.

- In Building, I helped to debug issues with the code numerous times and implemented the method to read the building and its requests from the text file.
- For LiftState, I added key variables that allowed us to track the movements of the elevator with currentFloor, and goingUp, I also created the getter and setter methods for this class.

Created core planning documents to help with team organisation.

Implemented the LOOK algorithm, this algorithm extends a shared Algorithm abstract class, handling how the elevator moves and processes each floor request.

Adopted a fixed input file format (e.g., small_scenario.txt, medium_scenario.txt, large_scenario.txt) to define floors, lift capacity, and requests.

Created the Building.FromFile(String filename) method to parse each scenario file and build the corresponding Building object.

Addressed off-by-one indexing issues by subtracting 1 from floor numbers read from the file.

Implemented a ScenarioTestRunner class that:

- Reads in each scenario file
- Runs SCAN, LOOK, and MYLIFT on the same building data
- Tracks pass/fail outcomes, total steps taken, and whether requests are fully served within a maximum step limit
- Prints a final summary of test results for each algorithm and scenario

Collected metrics to track elevator performance (total steps, elevator moves, average/max wait times) for each scenario.

Created a CSV file that stores the elevator performance metrics.

Created charts to visualise the results of the elevator performance.

Created the Group Collaborative Statement, that allows each member to get credit for what they created

Throughout the project, I contributed to both technical development and documentation, ensuring that all features that I implemented were effectively used and tested.

Jacob Evans