Building Design

This is a complicated project. The purpose of this design step is to help you succeed in this project. We have asked you to build a UML diagram of the entire class structure.

Include the UML Question 9 and 10 will assess this.

Answer the following questions in this document and upload with your UML diagram.

1) How are you storing your elevators in your Building model.

I use an ArrayList 'elevators: List<Elevator>'.

2) How are you storing the incoming requests so you can distribute them to the elevators.

I use ArrayLists 'upRequests: List<Request>' and 'downRequests: List<Request>'. I will use index to get the elements and make the ArrayList work as Queue.

And in Building Interface there is a method 'addRequest(Request request): boolean' to add the request to the list.

3) How are you distributing your downRequests and your upRequests to the elevators?

In my Building class, I design a private method 'distributeRequestToElevator(): void'. For each request, it will check the elevator one by one to see if it is eligible to take the request. If the elevator is taking requests, for elevators waiting on ground floor, up requests will be distributed according to available capacity. And for elevators waiting on top floor, down requests will be distributed according to available capacity.

4) How are you removing all requests when a takeOutOfService request is received.

I will call stopElevatorSystem() method. In this function, it will modify the elevator system status to 'stopping', remove all elements in 'upRequests: List<Request>' and 'downRequests: List<Request>' using .clear(), and take each elevator out of service.

5) How does your step method handle updating the elevators?

In step() method, it will use multiple if statement to check the status of the elevator system.

If the system is out of service, it will return and do nothing.

If the system is stopping, it will check whether all elevators are back to ground floor. If yes, it will set the system to out of service. Otherwise it will call step() method on each elevator and let them move.

If the system is running, it will distribute requests to elevator and step each elevator.

6) How do you start processing requests?

After receiving requests, it will first break into multiple single requests and be stored into 'upRequests: List<Request>' and 'downRequests: List<Request>'. Then 'distributeRequestToElevator(): void' method will distribute the requests to elevators. Each elevator will call 'processRequests(List<Request>): void' method to change the direction of the elevator. And after calling step() method, the elevator will move accordingly.

7) How do you take the building out of service?

Call 'stopElevatorSystem(): void' method in Building class, and it will change the system status to stopping, remove all requests, and call 'takeOutOfService(): void' method in Elevator class to stop each elevator. When all elevators are back to the ground floor, the system status will be changed to out of service.

8) How do you take the elevators out of service?

After stop the elevator system by calling 'stopElevatorSystem()' method, it will call 'takeOutOfService(): void' method for each elevator. This method will change the elevator's status in its ElevatorReport. And it will turn the direction down and bring all elevators to the ground floor and open doors.