OOP Course – Assignment 1

Leead Jacobowitz – 313308785

Arieh Norton - 31507493

Part 1

1. These three websites helped us better understand the functionality of the "Destination dispatch" algorithm in an elevator.

a) https://en.wikipedia.org/wiki/Destination_dispatch
b) https://www.geeksforgeeks.org/smart-elevator-pro-geek-cup/
c) https://www.popularmechanics.com/technology/infrastructure/a20986/the-hidden-science-of-elevators/

2. Off-Line algorithm

- The Off-Line algorithm requires all information before the algorithm starts. In order to use the off-line algorithm, we will get a list of the calls. We then will convert the call list to an ArrayList call_list.

- Each cell in call_list will have a call with three parameters: (source, destination, direction)

- Let's set pointer S to point at the first index (which will be the first call source) in call_list, and pointer D to pint at the second index (which will be the first call destination) in the cal_list.

- If the current direction of the elevator is UP then S will move along the call_list, stopping only at the cells with the direction UP. S will finally stop at the lowest source within the UP calls.

- Next the D pointer will also loop around the UP calls in the call_list and find the lowest destination value.

-The elevator will now leave the source and go to the destination, and update the current source as the current destination. The D pointer now will look for a new lowest destination according to the current source.

- This UP stage will happen until the elevator destination will be the highest floor in the building, thus preventing the elevator to go any higher.

- Once the elevator reaches the highest floor, the D pointer will point at the current floor and the S pointer will now run through the DOWN calls in the call_list.

- S will find the highest floor that is lower than the current D and will move the elevator from floor D to S.

- The elevator will continue doing this until it has reached the lowest floor.

- Each time the elevator moves from S to D the algorithm removes S from the call_list.

- If their numerous elevators, there will be a main call_list and each one will be running the algorithm, while updating the main list after every stop.

3. On-Line algorithm

- In the Online algorithm we are presented the calls at each time stamp as they arrive.

- This algorithm initializes the direction of the elevator, has two Arrays of Array-Lists (for the UP-calls and DOWN-calls), where each cell in the list is also an Array-List. The algorithm also stores another Array called ElevatorList which stores the amount and indexes of the elevators given from the building.

-The algorithm implements the "ElevatorAlgo" interface and imports the "Building", "CallForElevator" and "Elevator" interfaces.

OOP Course – Assignment 1

Leead Jacobowitz – 313308785

Arieh Norton - 31507493

- The algorithm starts with the "allocate" function. This function determines which elevator will be assigned for each single call c. The algorithm, is split to one elevator or more.

-If there is only one elevator the algorithm, designates the calls to the single elevator and places a track of a call depending on its call type(UP or DOWN).

-If there are numerous elevators then the algorithm calls upon a sub-function called "FindBest", which allocates the best suited elevator according to the call.

- The "FindBest" function prioritizes which elevator should be assigned to each call using a set of pre-determent cases. If the cases do not match, the algorithm, will choose a random elevator with the same direction.

- Once the elevator was chosen, the cmd function sends the elevator to the source or destination and is in charge of removing the call from the call-list(UP or DOWN).

- The cmd will also check if on the way to the wanted source or destination there are any other relevant calls to take care of. If so, the cmd will take care of them.

4**. The UML is attached within the zip file.**

5. Junit test

Our tests are meant to check if the main functions work. We want to test "FindBest", "cmd", and "allocatElevator".

FindBest

1) We would flag an elevator knowing it's the best fit and would want to check if the function picks that elevator.
2) We would want to check that if there is no suited elevator that random would still pick from the right

CMD

1) We would want to test the movement of the elevator and make sure that the cmd moves the elevator accordingly.
2) We would want to test that the cmd moves the correct calls according to the UP and Down list.

AllocatElevator:

1) We would want to make sure that our calls were added to the right list(P or DOWN).
2) Make sure that we call correctly the "FindBest" function.