Tal Malchi 208278556
Yuval Ben Simhon 318916335

Object-Oriented Programming-Task Number 0+1

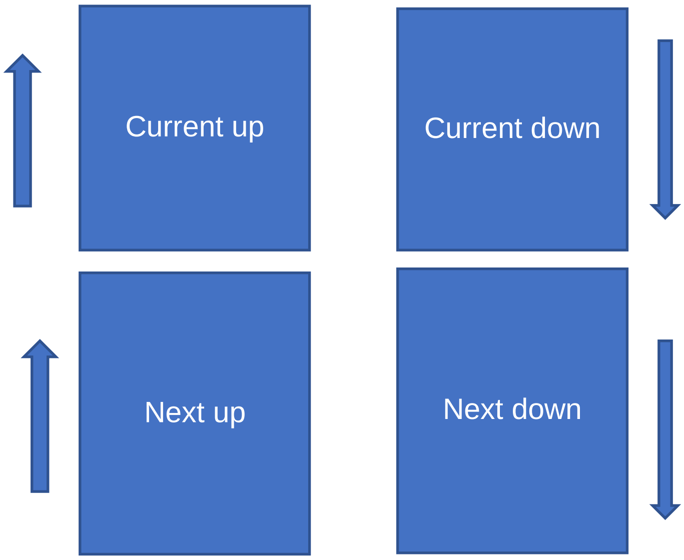
1. The Most Relevant Links we saw during our work:

https://www.youtube.com/watch?v=xOayymoIl8U
https://www.youtube.com/watch?v=siqiJAJWUVg
https://thinksoftware.medium.com/elevator-system-design-a-tricky-technical-interview-question-116f396f2b1c
https://elevation.fandom.com/wiki/Destination_dispatch#System_principle
https://everythingwhat.com/what-is-a-destination-oriented-elevator
https://peters-research.com/index.php/papers/understanding-the-benefits-and-limitations-of-/destination-control

2. Online Algorithm:

In case of single elevator:

First of all , we will have 4 array lists, with length as the number of the floors of the building.



- Elevator_route_function:
  - The elevator move over all the floors in the 'current_up' arraylist.
  - Once it will arrive to the upper floor in the 'current_up' arraylist. It copy the 'next_up' arraylist to the 'current_up' arraylist.
  - And start to move over 'current_down' arraylist.
  - Once it will arrive to the lower floor in the 'current_down' arraylist. It copy the 'next_down' arraylist to the 'current_down' arraylist.

- If there was a new call
  - We will check if it is in the same direction
    - If it is **in the same direction** && the elevator is in the **UP** direction && the current location of the elevator is **lower** than the source floor of the new call, then we will add source&destination floors to 'current_up' array(current_up[floor]=True →means the elevator will stop in this floor).

- If it is **in the same direction** && the elevator is in the **UP** direction && the elevator **higher** than the source floor of the new call, then we will add the source&destination floors to ‘next_up’ array (next_up[floor]=True →means the elevator will stop in this floor).
- If it is **in the same direction** && the elevator is in the **DOWN** direction && the current location of the elevator is **higher** than the source floor of the new call, then it will stop in this floor- means we will add source&destination floors to ‘current_down’ array(current_down[floor]=True →means the elevator will stop in this floor).
- If it is **in the same direction** && the elevator is in the **DOWN** direction && the elevator **lower** than the source floor of the new call, then we will add the source&destination floors to ‘next_down’ array (next_down[floor]=True →means the elevator will stop in this floor).
- If it is **not in the same direction**
  - If the elevator is up and got call to down
    - We will add source&destination floors to ‘current_down’ array (current_down[floor]=True →means the elevator will stop in this floor).
  - If the elevator is down and got call to up
    - We will add source&destination floors to ‘current_up’ array (current_up[floor]=True →means the elevator will stop in this floor).

In case of multiple elevator:

- for each call
  - for each elevator:
    - Now we will calculate the whole route time include the new stops
  - The chosen elevator will be the elevator with the shortest time that calclated

3. **offline Algorithm:**

In case of single elevator:

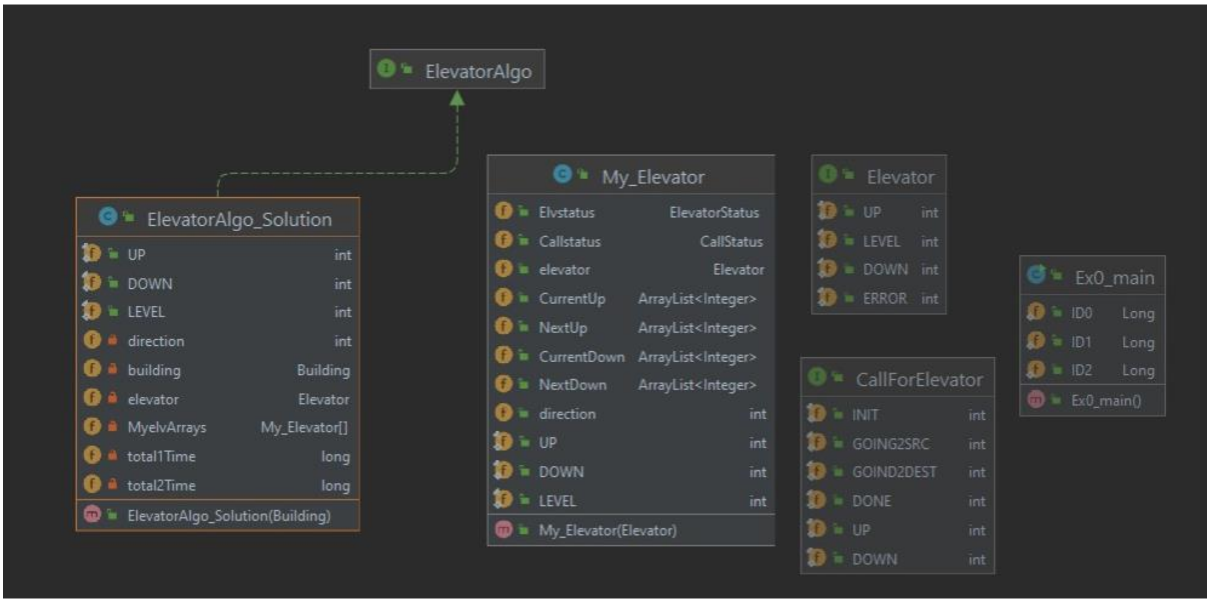
- if it the elevator move up, than put in 'current up'
- if it down- put in 'current down'
- write the entire route, and then do the route

In case of multiple elevator:

same as we wrote above

Tal Malchi 208278556
Yuval Ben Simhon 318916335

4. **UML:**



5.JUNIT Test:

We assume that we will need to use the functions below, therefore we will need to build the next tests(for example):

i. Calculate the time between the floor of new call and the floor of the destination of the new call .
(easy to make test- take 2 different floor – also negative floors -and compute the difference between them, and multiply it with the elevator speed. After that we will add all additional variables – such as the acceleration & declaration , time of closing &opening doors).

ii. When elevator get new call , we will need function that calculate the time of all elevators in the building with all exicting calls **in addition** of the new call– and than it will choose the elevator with the minimmal total time .
(test will be like that: we will create array of elevators, with the next calls for example:
Elevator1:0->1->2->3
Elevator2:-1->0
Elevator3:4
Than we will enter new call from 4 to 0- and it is very clear that the elevator that should be choose will be Elevator3).
**Of course there are more options to make tests, but it depends how we will implement the code.

NOTES

- The reports of the algorithm's results time is additional to our ZIP file.