INPUT CASE

1. Request an Elevator: [**<TIME_INSTANCE>,<FLOOR>,<DESTINATION_FLOOR>]**
   Time, floor, Destination_floor
   [1 ,   1,   4]
   [2,   2,   1]
   [4,   3,   9]

Here there can be 3 ways in which lift can move.

1st Approach:

(When the lift moves unidirectional (Up & down) & doesn't pick a down request when moving upwards & vice versa)

Here the lift will move from ground floor to 9th floor but while moving upwards the door will not open for person 2 (as his request is for the opposite direction) and he will be picked when the lift moves down from the 9th floor.

Movement Flowchart:

```
 0   ->  1     ->   ->   ->     ->      3    ->   -> -> 4      ->   ->   -> 9-> ->  ->   -> 2  ->   ->   1
      (picks        (doesn't Pick    (picks        (drops             (drops        (picks        (drops
    person1)       person  2)        person 3)    person 1)          person 3)     person 2)    person 2)
```

Drawbacks:
Response time for person 2 will increase and total working time for lift also **increases.**
Starvation may Occur for person 2.

2nd approach:

When the lift moves unidirectional (Up & down) but picks a down request when moving upwards & vice versa.

Here the lift will move from ground floor to 9th floor but will also pick person 2 (even if his request is of opposite direction) and will stop at the 4th floor. Now at 4th floor it has 2 requests pending 2->1 and 3->9 but as the lift is unidirectional so it will first complete request 3->9 and will move to 9th floor and then comes back to 1st floor to complete the process.

Movement Flowchart:
0 -> 1 - > -> -> 2 -> -> -> 3 -> -> ->4 -> -> -> 9 -> -> -> 1
(picks (picks (picks (drops (drops (drops
Person 1) person 2) person 3) person1) person 3) person 2)

Drawback:
Person 2 will keep on waiting inside the lift even though his request was earlier and will increase its turnaround time.
Starvation may occur for requests.

Approach 3 : Best approach (Implemented approach)

When the lift drops the person based upon the priority of request received but picks / drops a person , in between also, if it is in the direction of move.

Here the lift is designed as a smart lift which works in a way so as to reduce the response and turnaround time for a request.
Now here the lift will move from 0->1st floor and will pick 1st person. After this the lift will move to the 2nd floor and pick the 2nd person. Then it moves to the 3rd floor and picks the 3rd person. Then the lift will move to the 4th floor to complete 1st request . At the 4th floor lift has two pending requests i.e., 2->1 and 3->9. Here as the 2nd request was received earlier than 3rd request , the lift will move down to floor no 1 to first complete the 2nd request and then move to 9th floor. This will not only decrease the waiting time for person 2 but also will result in less response and turnaround time.

Here the priority is given to the request earlier received.

Movement Flowchart:
0 -> 1 - > -> -> 2 -> -> -> 3 -> -> -> 4 - > -> -> 1 -> -> -> 9
(picks (picks (picks (drops (drops (drops
Person 1) person 2) person 3) person 1) person 2) person 3)

There are least chances of starvation.
The detailed calculations on the above three approaches are attached as Excel sheet.
From it we can conclude that approach 3 is best as it has least response and turnaround time and I have designed the algorithm accordingly.