1. Overview.

- You know
- You don't know.

It's a software sim that can be used in malls/theatre to generate bills for vehicles parked.

class diagram or entire working system.

How the user will interact with → Hard Code in main

Srorage → Memory.

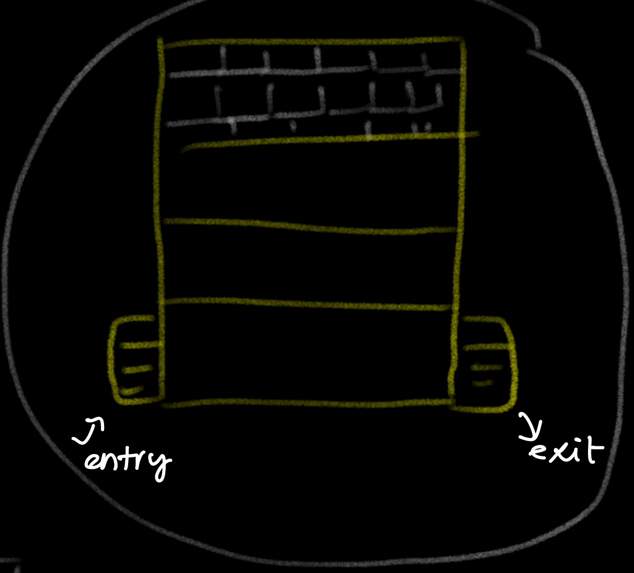
2. Requirement gathering

1. Diff parking slots for diff type of vehicle.
2. single floor / multifloors.
3. Multiple entry/exit gates
4. A token should be given at the time of entry.
5. Payment is taken at the end [bill]
6. Payment charging
   - hourly
   - fixed
   - day

   } Fee Calculation Strategy
7. Payment via online/offline mode.
8. Admins will use the software.
9. Vehicle is entered, we can assign an empty slot.
10. Don't allow other type of vehicles to be parked at different spot.

11. Each floor can have multiple types of parking slots.

Visualise



Parking lot

- list<floor>
- list <entry/exit> gates
- allowed vehicle Type
- status

ParkingFloor

- list<parkingSlots>
- floorNumber

Gate

- type
- number
- operator
- status

ParkingSlot

- vehicle type
- parking status
- number (slot)
- floor

Operator

- emp Id
- name

Vehicle

- Vehicle type
- number
- nameOfOwner
- owner Contact

Ticket

- number
- entry time
- vehicle
- parkingSlot
- gate
- operator

Bill

- exitTime
- ticket
- amount
- gate (exit)
- operator
- status (paid/not)
- payment

Payment

- refNumber
- Status
- time
- mode

Ext^n: Allow partial payment.

100 via cash
100 via Gpay } bill -> 200Rs.

Bill

- exitTime
- ticket
- bill Amount
- gate (exit)
- operator
- status(paid/not)
- List<payment>

Payment

- refNumber
- status
- time
- mode
- amount

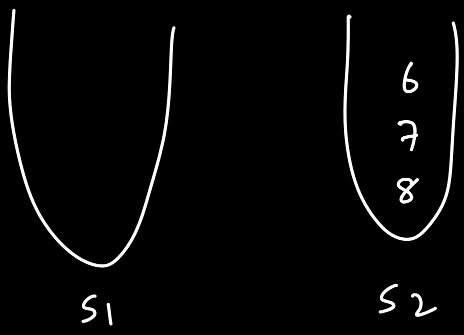
Queue using stacks.

↳ 2 stacks.

1 2 3 4

1, 2, 3, 4, dq, dq, 5, 6, dq, dq,

7, 8, dq.
5

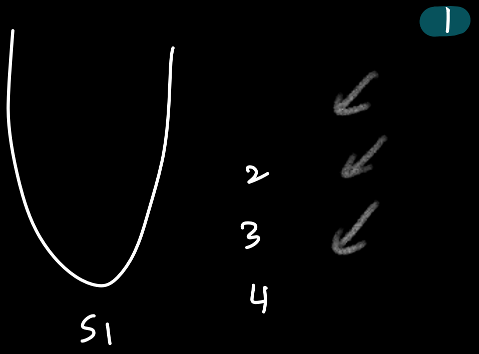


TC: enque: O(1)

deque: O(1) in average.

=) dq.

↳ 1 stack.



```
enqueue()
    push to S1.
```

```
int dequeue()
    if (S1.size()==1)
        return S1.pop()

    top = S1.pop()
    output = deque();
    S1.push(top);
    return output
```