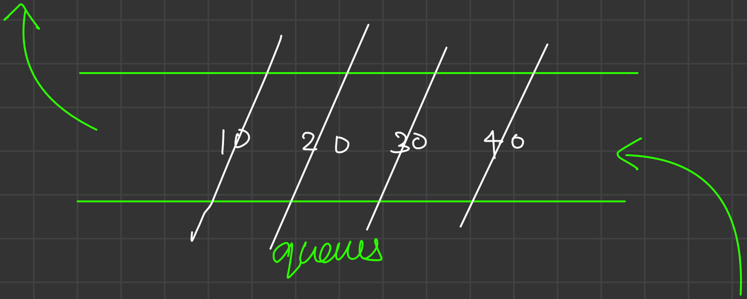
Queues.

↳ linear data structure



add	remove
10	10
20	20
30	30
40	40
↓ order of addition	↓ first in first out (FIFO)

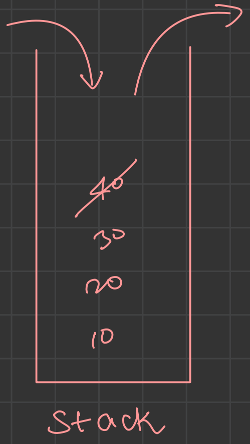
Stack

↳ linear data structure

↳ it follows last in first out (LIFO)

→ Methods

1. push TC: O(1)
2. pop TC: O(1)
3. peek TC: O(1)
4. size TC: O(1)



push	pop
10	40
20	30
30	20
40	10
↓ order of push	↓ order of pop

enter + queue

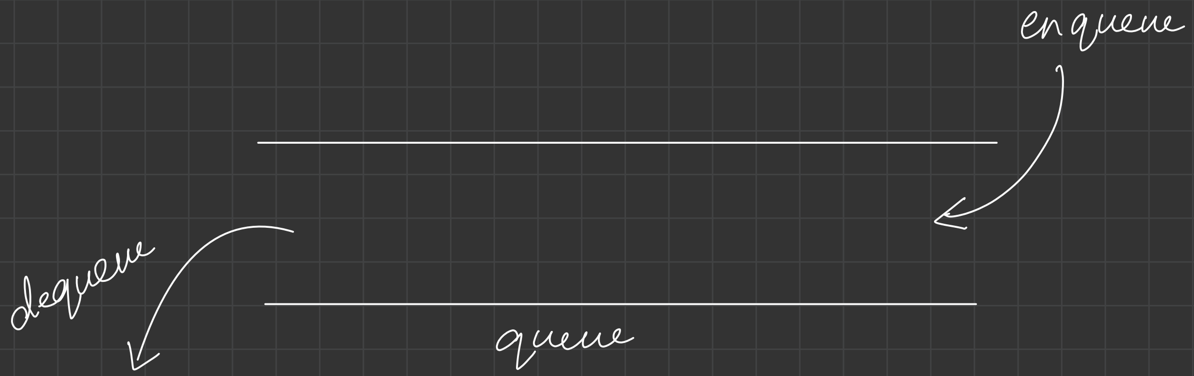
enqueue → Adding element in a queue

TC: O(1)

delete + queue

dequeue → remove element from a queue

TC: O(1)



```
Queue <G> que_name = new ArrayDeque<>();
```

Methods

① add() ↓ enqueue

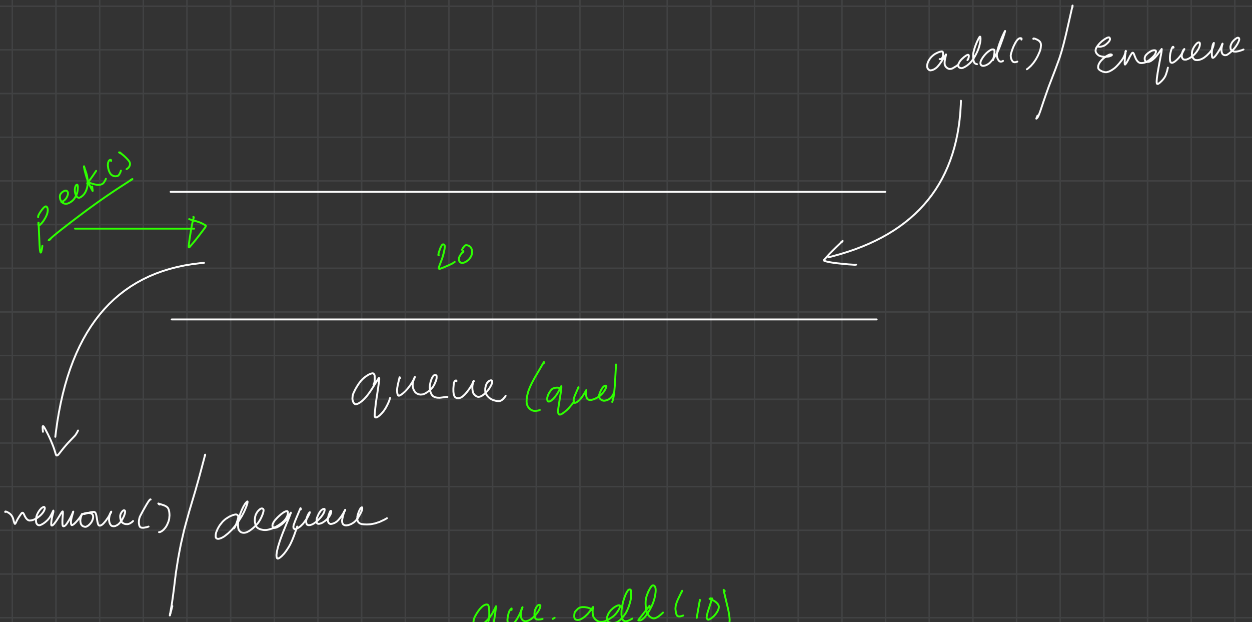
② remove() ↓ dequeue

③ peek() ↓ to see the front ele

④ size() ↓ size of queue

```
Queue<G> que_name = new LinkedList<>();
```

① offer() ② poll() ③ peek () ④ size()



```
que.add(10)
que.add(20)
que.peek() → 10
que.remove() → 10
que.size() → 1
```

queues

- ↳ linear data structure
- ↳ follows fifo (first in, first out)

Methods

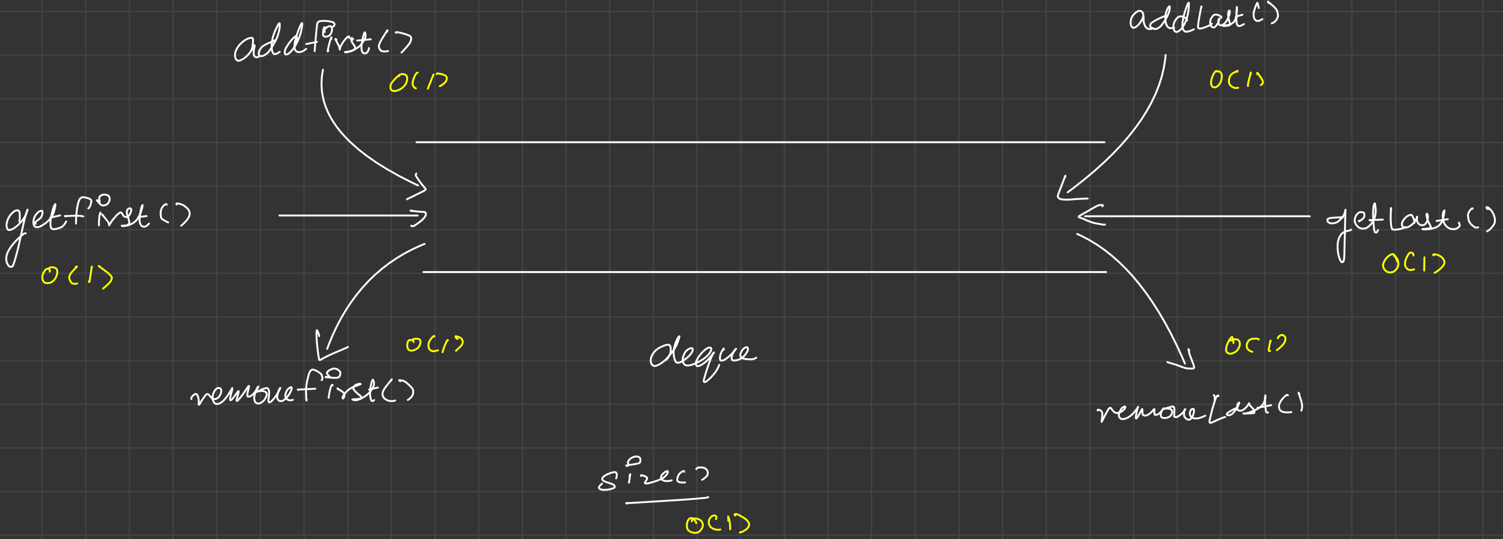
- ↳ add(), remove(), peek(), size()

TC: O(1)

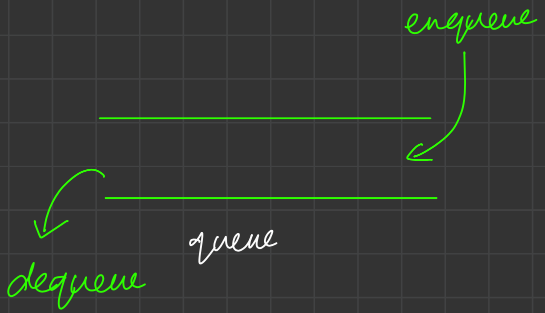
deque → doubly ended queue

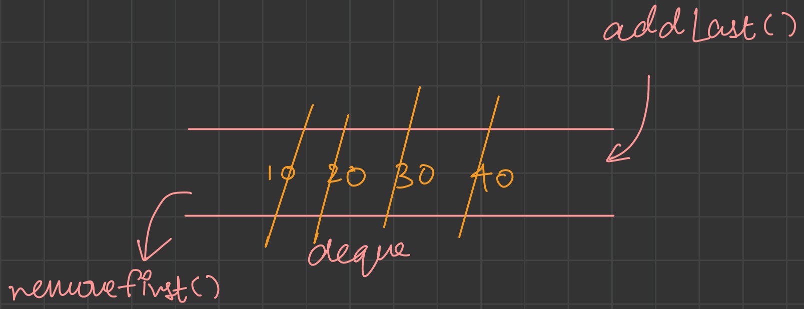
linear data structure

implemented using doubly linkedList internally



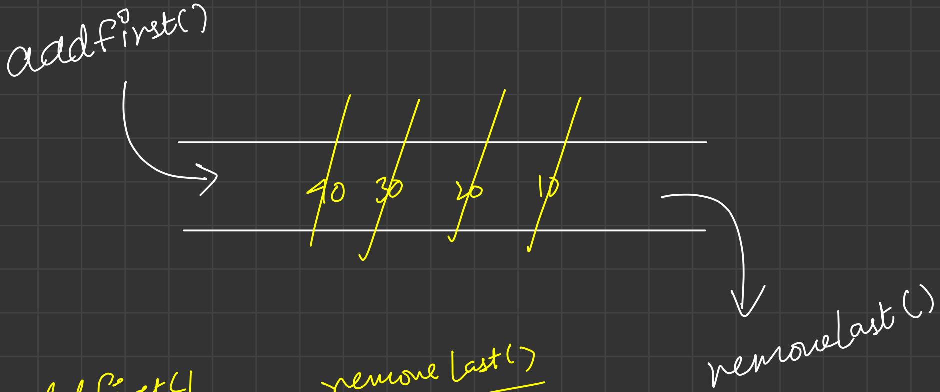
Q Can you implement a queue using a deque?





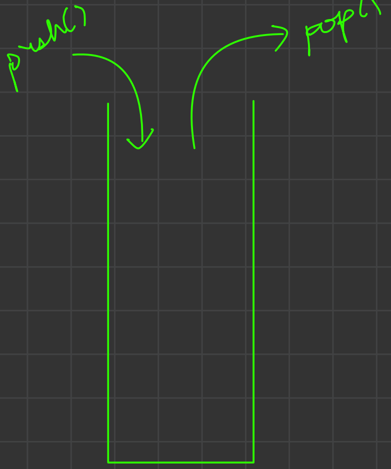
add	remove
10	10
20	20
30	30
40	40

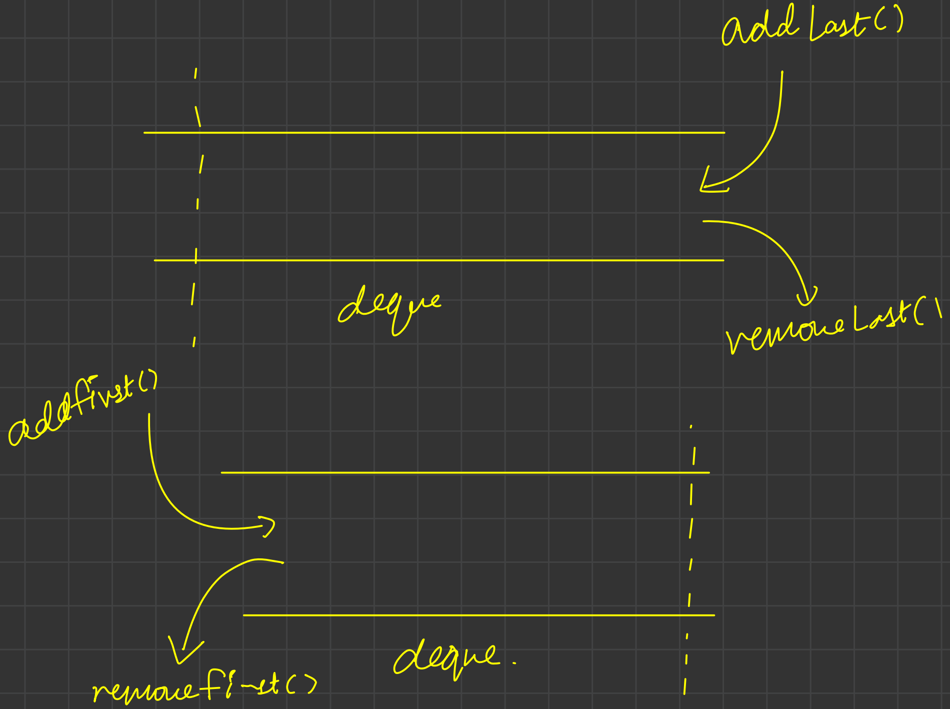
addLast()	removeFirst()
10	10
20	20
30	30
40	40



addfirst()	remove last()
10	10
20	20
30	30
40	40

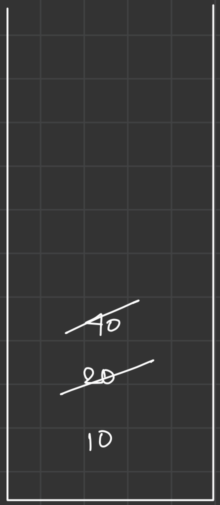
Q Can you implement a stack using deque?

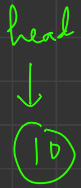




Q design a stack using linked list

push(10)
push(20)
push(30)
pop() → 30
push(40)
pop() → 40
pop() → 20

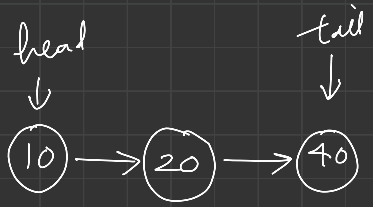




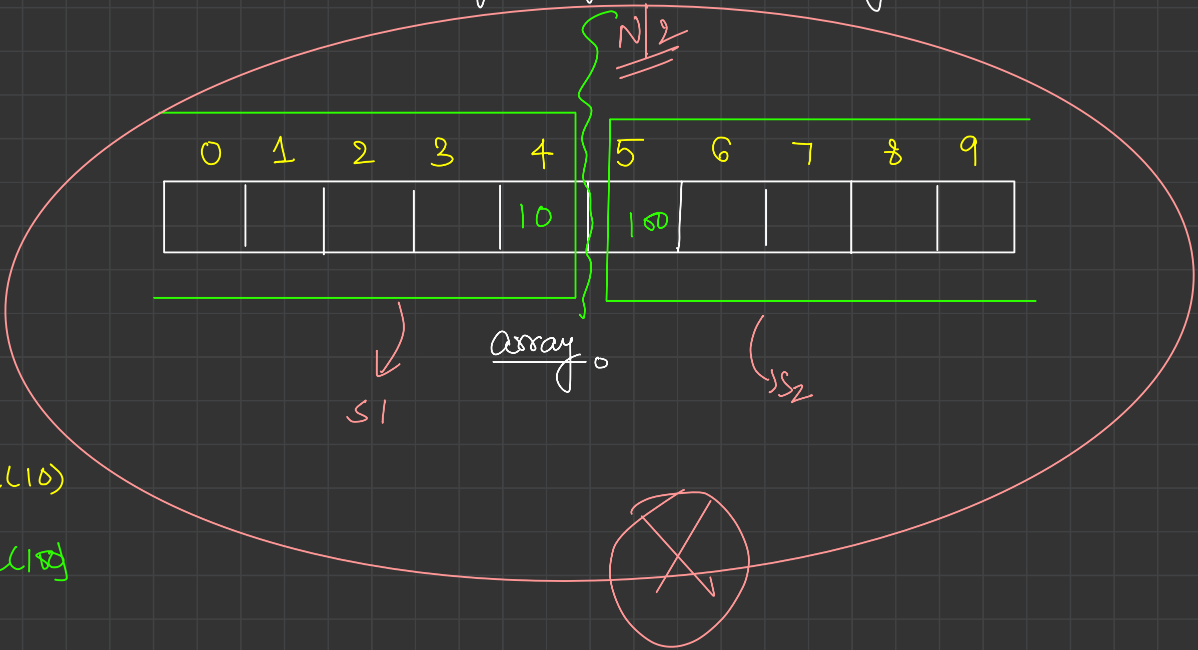
push()
→ addFirst in a linked list

pop()
→ removeFirst ll.

Q design a queue using linkedList

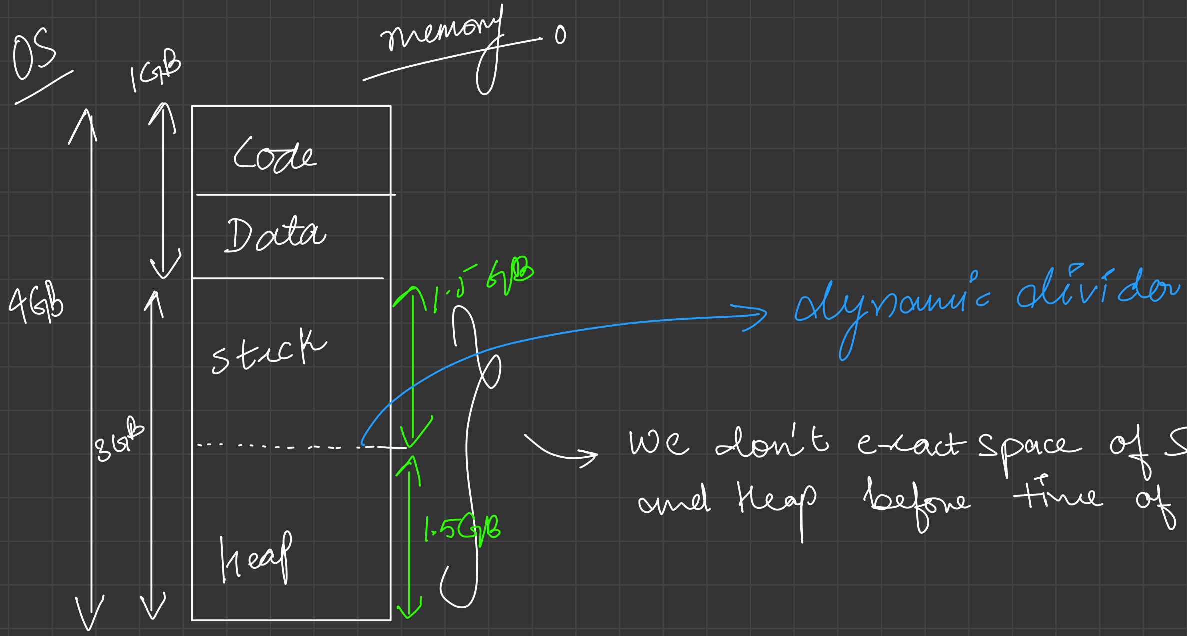


Q implement two stacks using single int array!

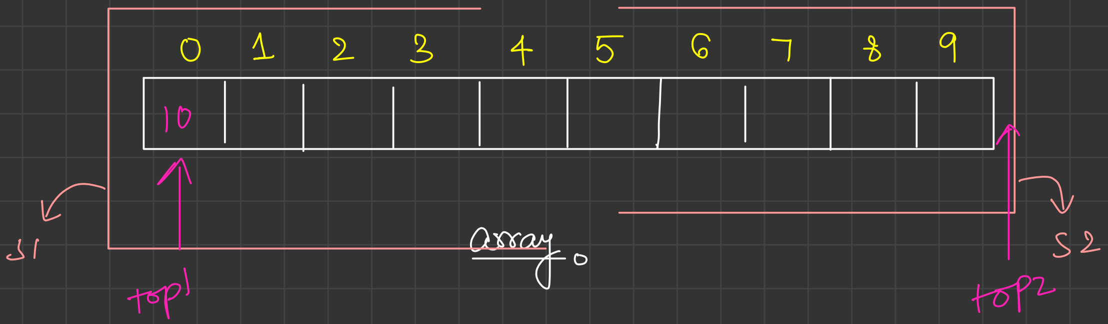


S1 push(10)

S2 · push(100)



We don't exact space of Stack and Heap before time of compilation



S1.push(10)

S1 push(20)

S1. push(30)

S1. push(40)

S1 push(50)

S1. push(60)

S2. push(100)

S_2. push (200)

Q Implement Queue using stacks (2stack)

① Enqueue

O(1)

② dequeue

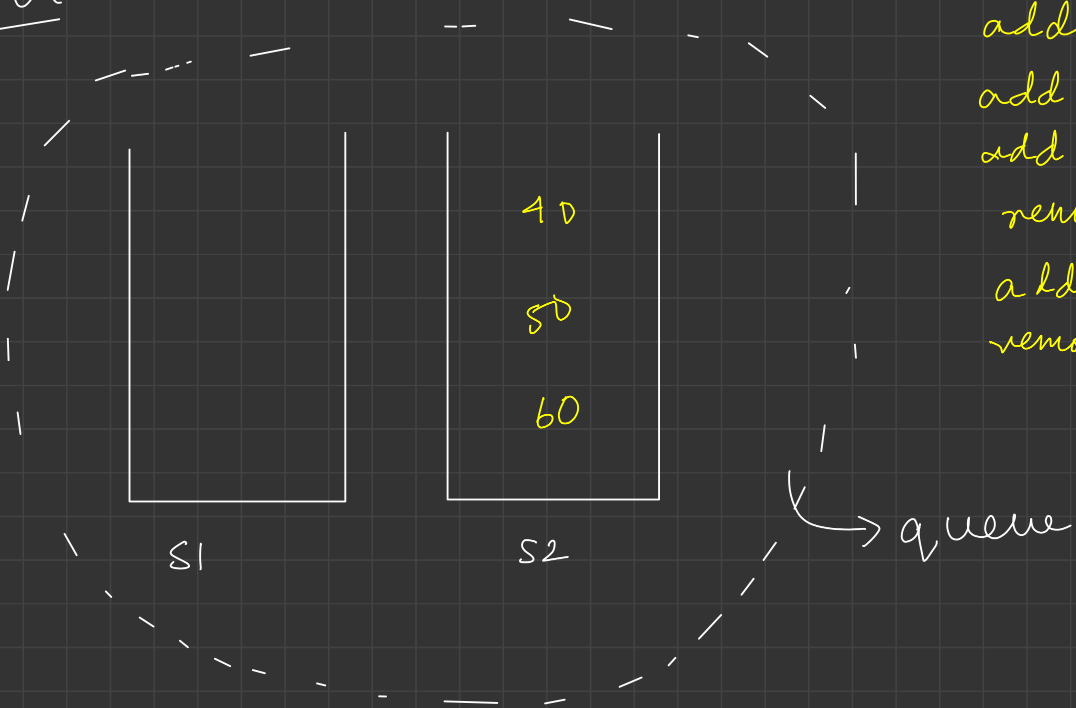
O(1)

Enqueue O(1)

10

20

30



add(10)

add(20)

add(30)

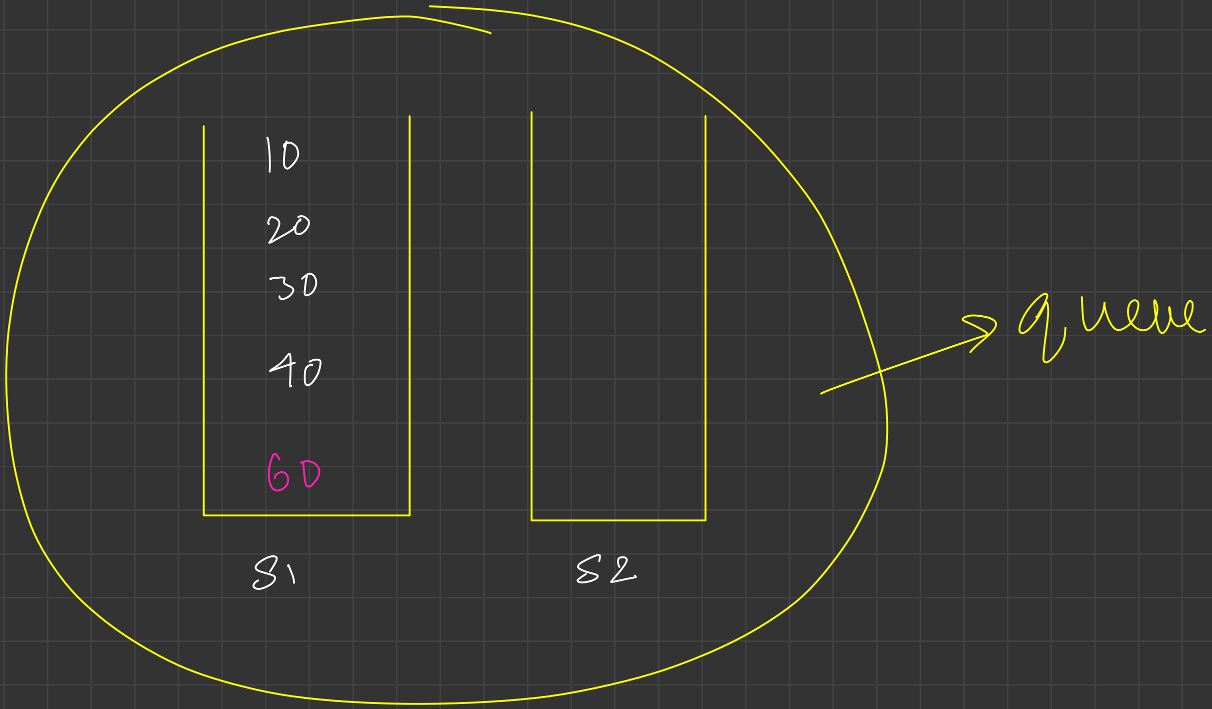
remove() → 10

add(40)

remove()

dequeue O(1)

10, 20, 30, 40, 60 → add


10
20
30
40
60
S1
S2
queue