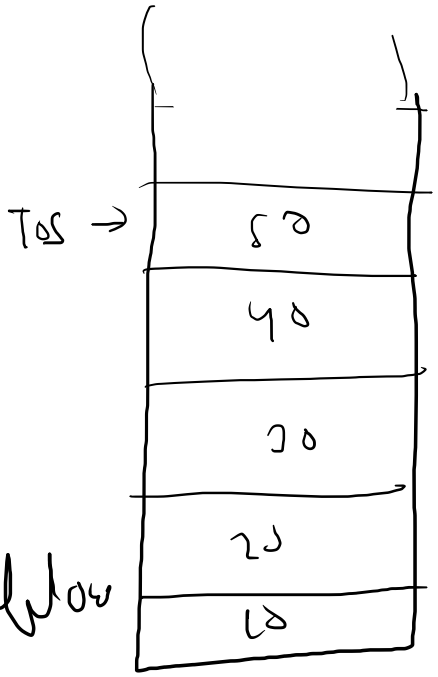
Normal Stack

Capacity = 5 → units

Push

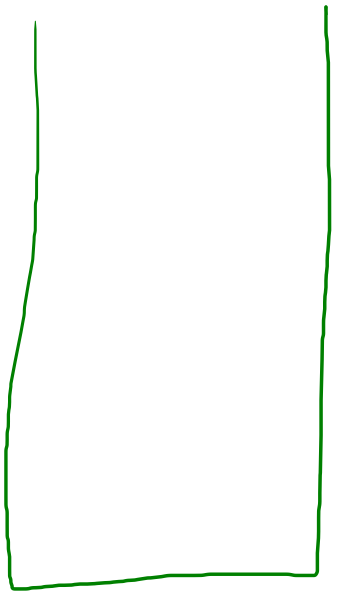


Push(60)

Stack full

→ Stack overflow error

Pop / Peek



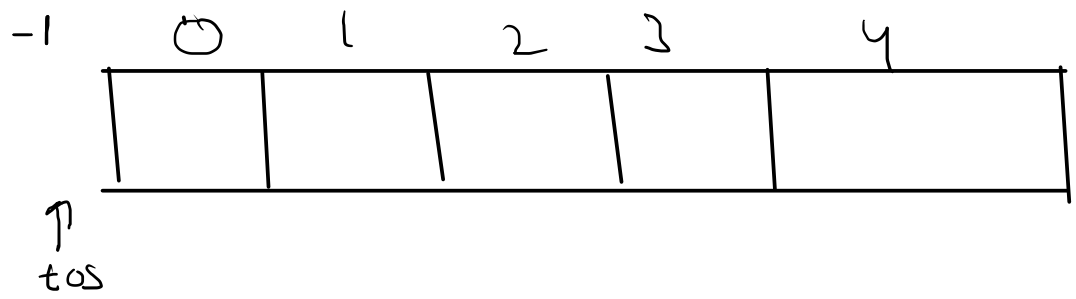
St = empty

St.Pop()

Underflow error

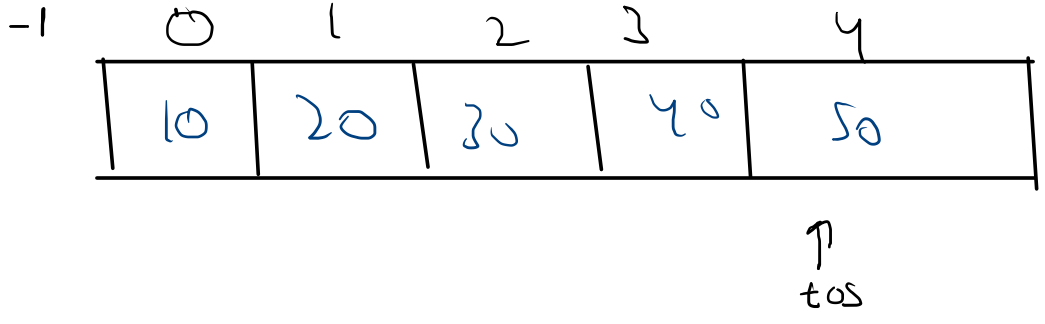
stack => 5

Initial



tos
= -1

Push Operation



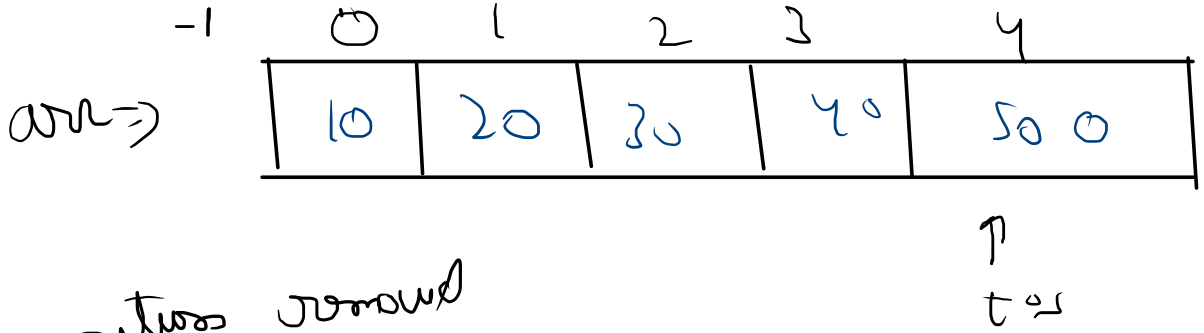
int tos
= -1

Push 10
20
30
40
50

(1) arr.l -1 == tos
→ full → overflow error

(2) tos++
arr[tos] = val

Pop



Pop() → return removed val

① if (tos = -1) {
underflow

② val = data(tos)
tos --

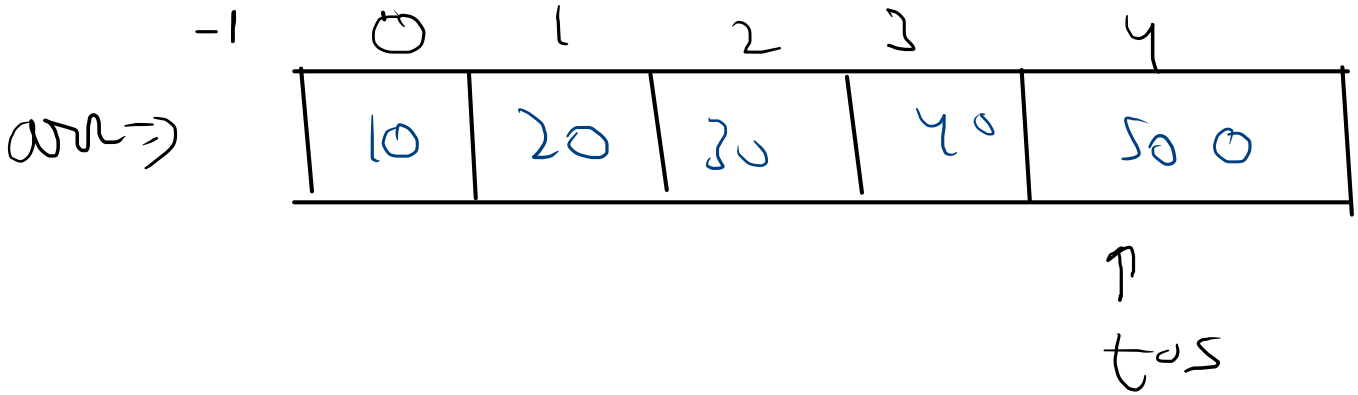
Push(500)

Peek()

1. data [tos]

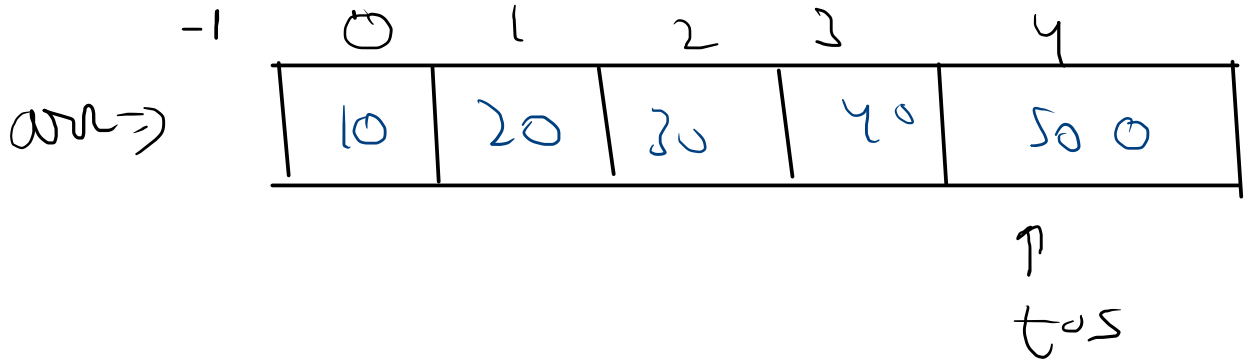
2. if (tos = -1)
   → underflow

display

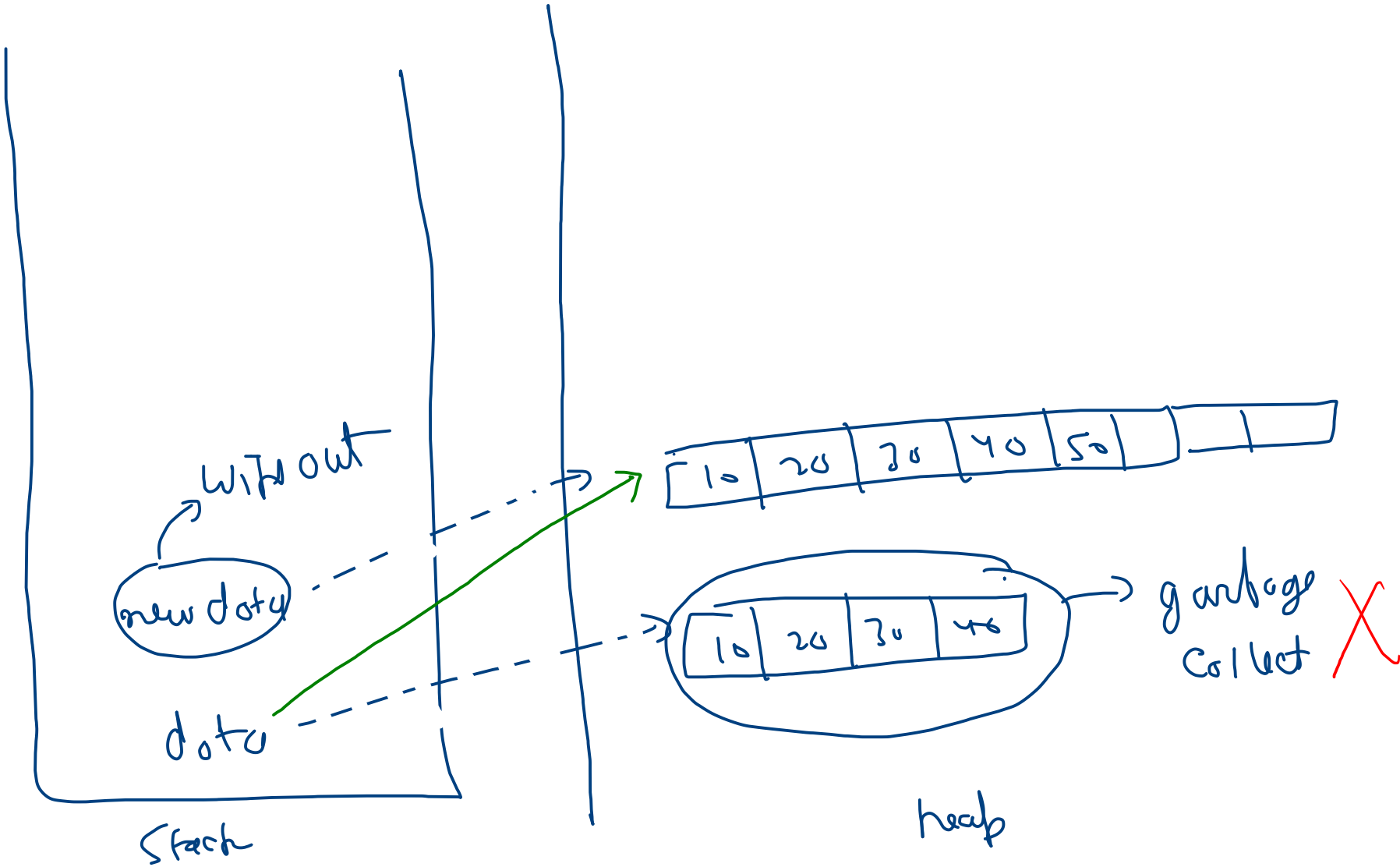


Print from tos to 0

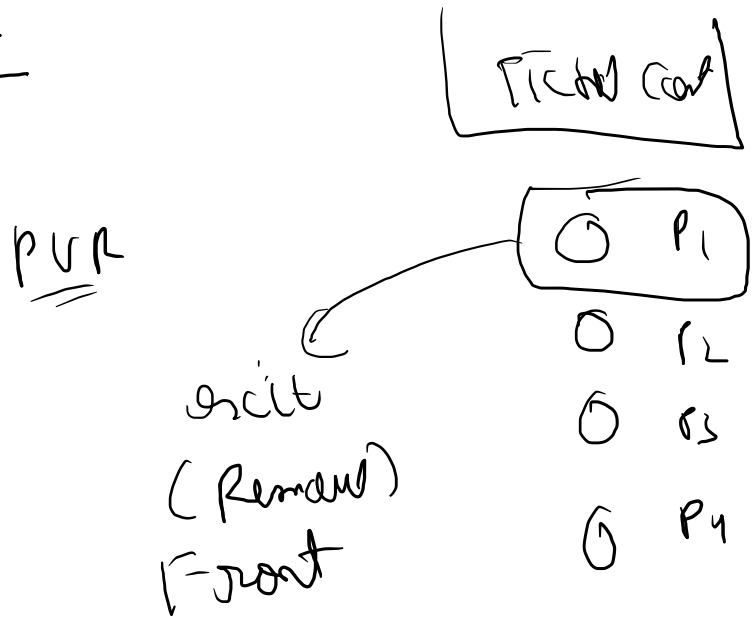
Dynamic Stack



Push(100)


without
newdata
data
10
20
30
40
50
10
20
30
40
garbage
Collect
Stack
heap

Q ....



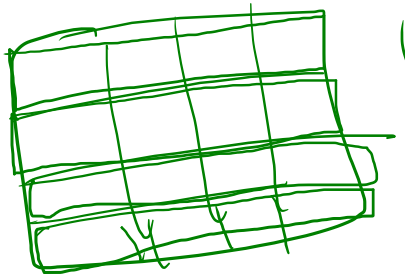
PVR

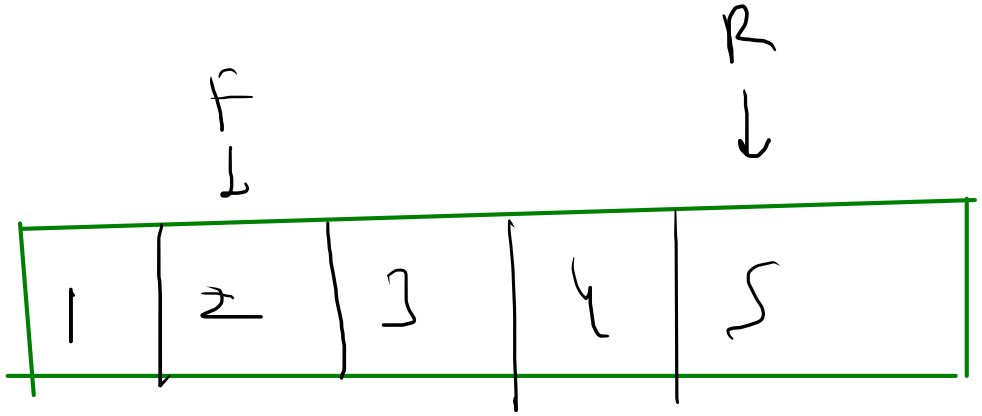
→ FIFO

P7 → add → Lost

→ FIFO

→ Linear DS → 1D Fashion





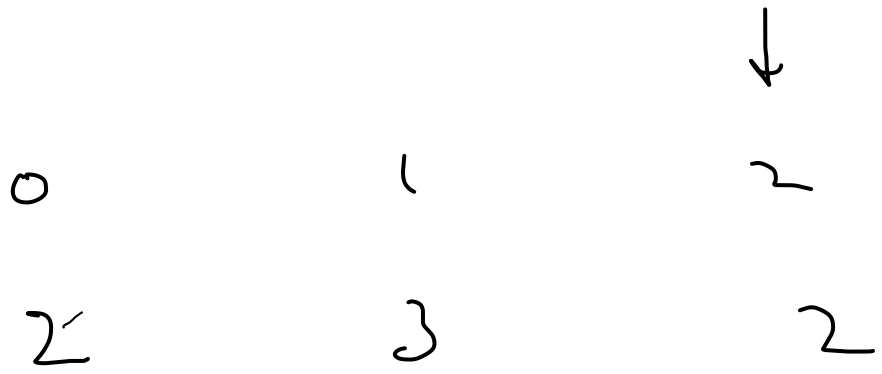
2

add → overflow

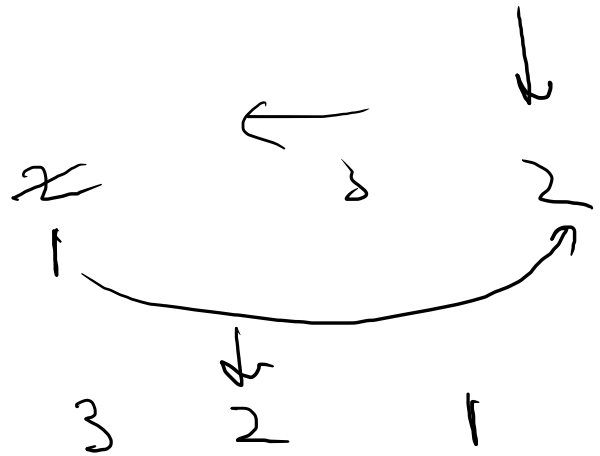
Remove → underflow

LinkedList () → Iterator

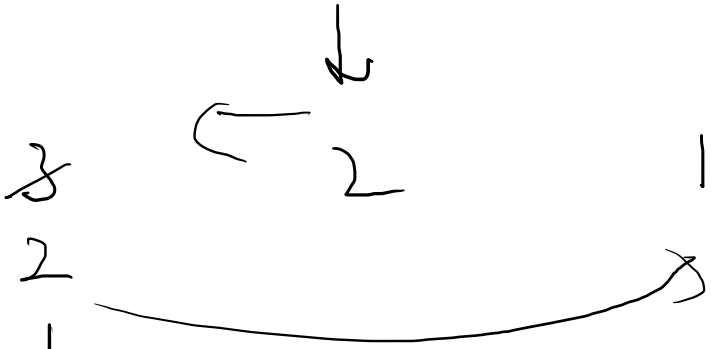
Time need to Buy Ticket



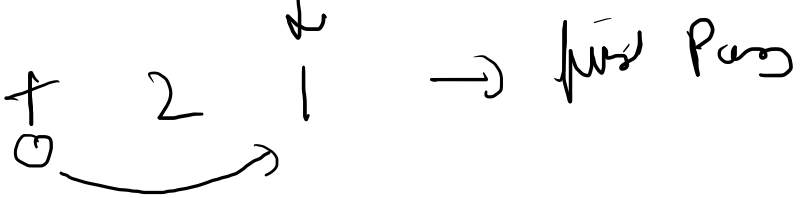
k=2 1st 1st



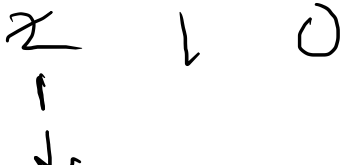
2 Sec



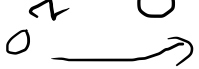
3 Sec



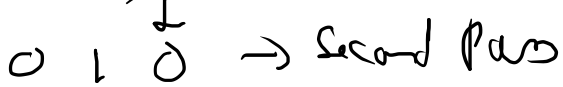
4th Sec

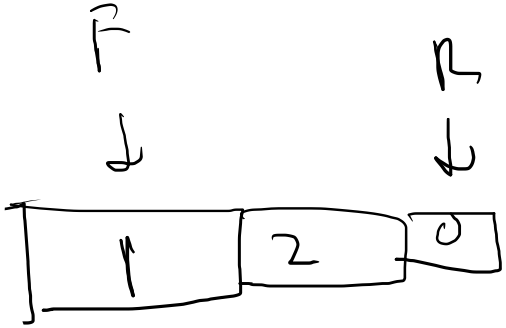


5th Sec

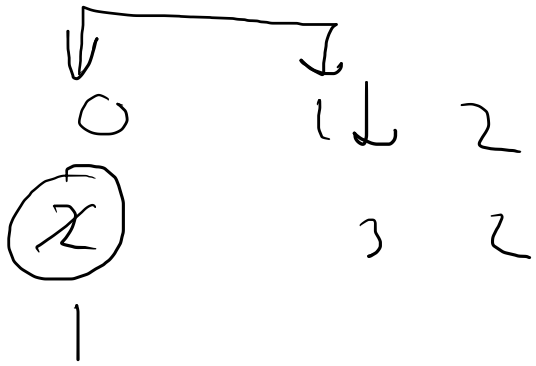


6th Sec

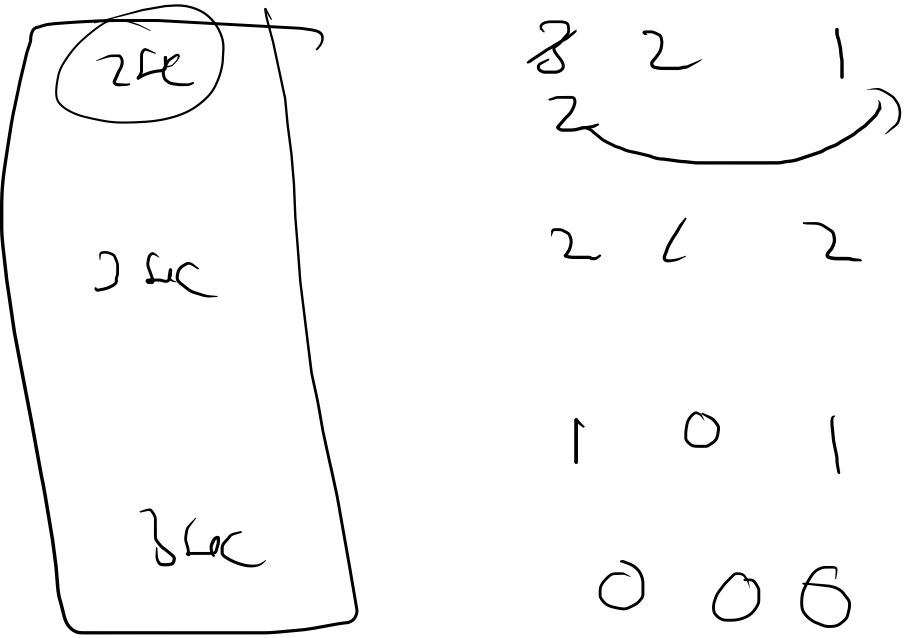



F
R
1
2
0

2



8



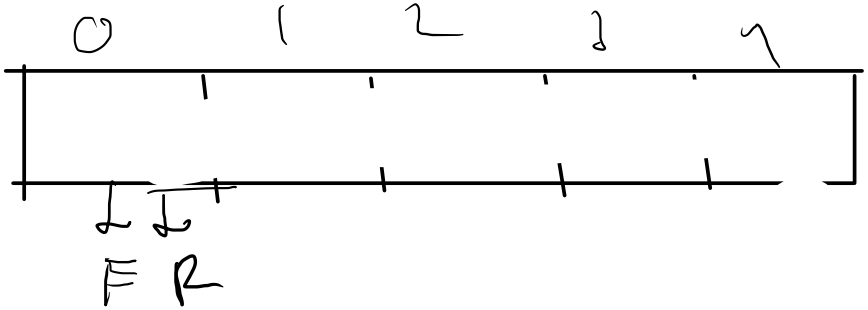
2 1 2

1 0 1

0 0 0

Push → O(1)

Pop → O(n)

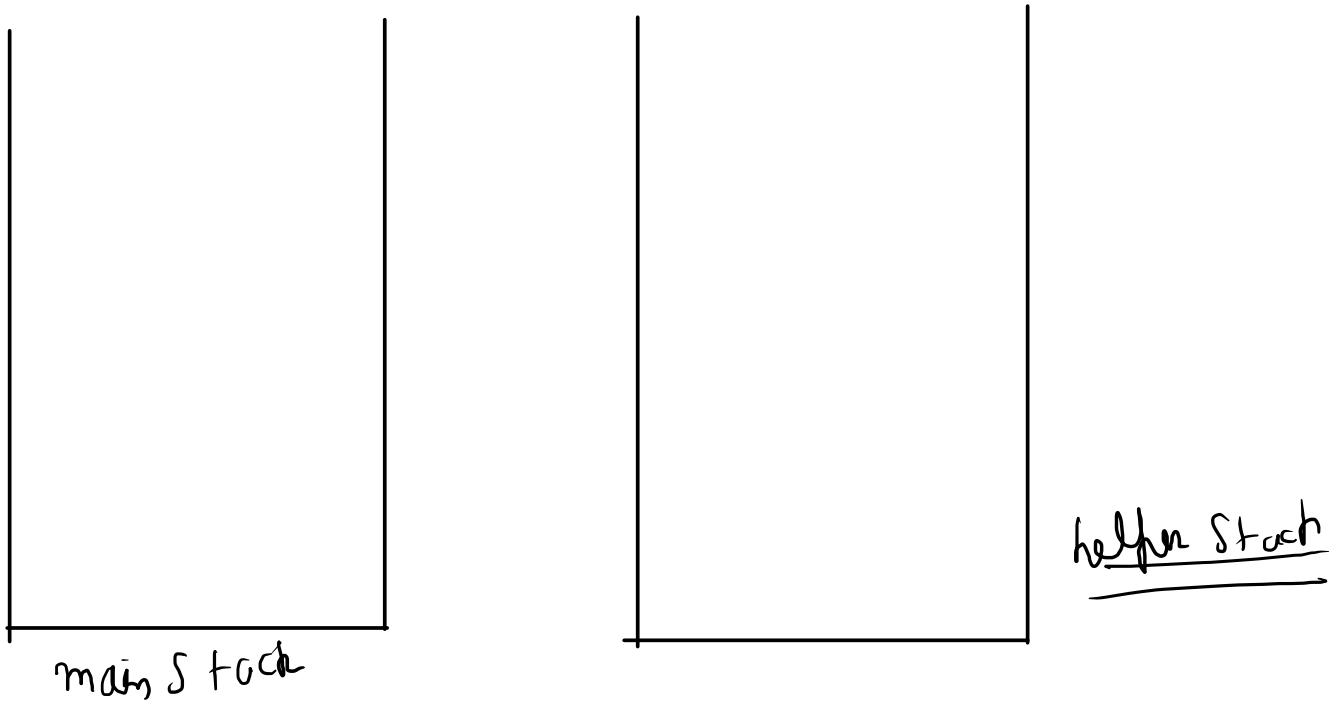


Push(con)

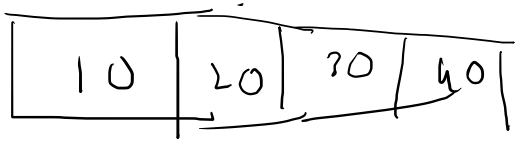
Pop O(1)

↓

Pop all



Report 2



Push(10)

Push(20)

Push(30)

Pop() → 10

Push(90)

Peek()
→ 20

Push → O(1)

Pop / Peek → O(n)