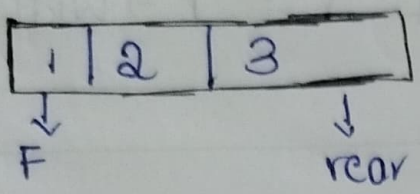
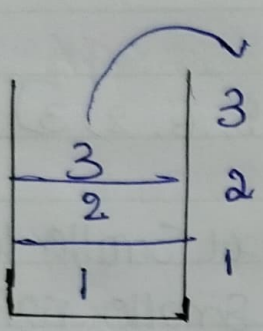
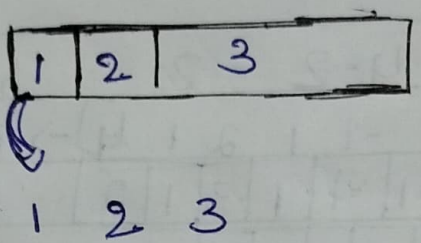
QUEUE

↳ FIFO (First IN First Out



difference between stack and queue

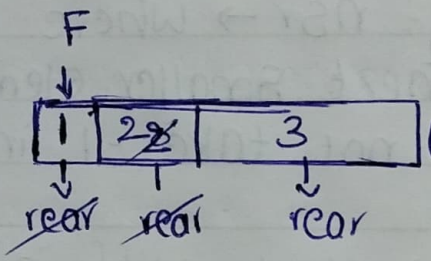




Operations

1) Add → O(1) → Ex:

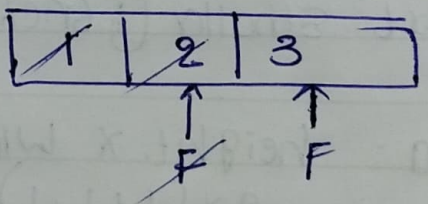
Enqueue



(insert at rear)

2) Remove → O(1) → Ex:

Dequeue



(Remove at F -one)

3) Peek → O(1) → Ex: Top of the element

Date: .....

Expt. No.: ..... Page No.: .....

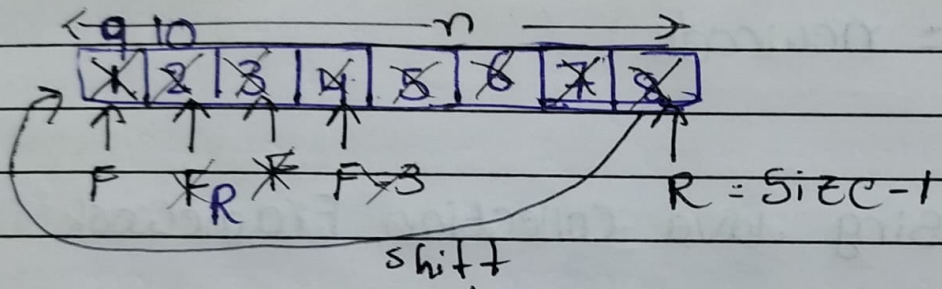
Implementation

Array ✓ LinkedList Stack

- fixed size "n"
- remove O(n)

Circular queue

By using Circular queue → O(1)



rear = rear + 1

rear = rear + 1 % Size (insert) → 9

= 7 + 1 % 8 = 8 % 8 = 0

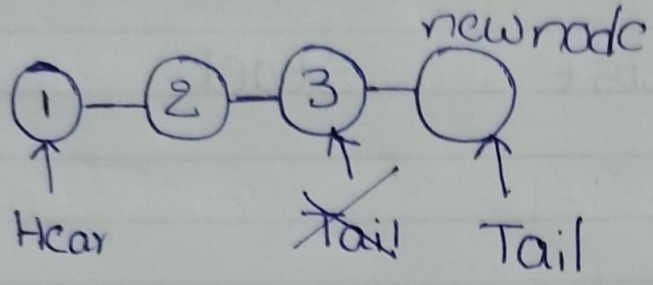
Front

Front = Front + 1 % Size

Teacher's Signature _____

**

8.) Queue using Linked List



Approach

tail.next = newnode

tail = newnode

Note: Similar to insert at newnode last

Queue using Java Collection Framework

We can use import java.util.*;

Queue<Integer> q

↓

interface

Queue<Integer> q = new LinkedList<>();

For Fast we use ArrayDeque

Queue<Integer> q = new ArrayDeque<>();

Note: In this null value not store

Date: .....

Expt. No.: ..... Page No.: .....

Why Linked List → slow

It has Prev. next. data → in one node

Question 2

Queue using 2 Stacks

↙ FIFO ↓ LIFO

Push O(n) (or) POP O(n)

add ↗ remove ↗

remove, POP } O(1) remove, Peek, add } O(1)

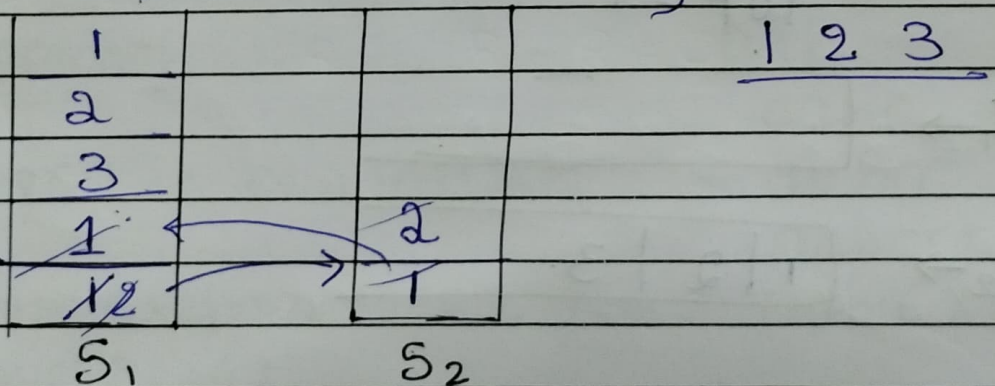
3 steps Process

① add

S1 → S2 }
S1 Push
S2 → S1

② remove

S1 → POP
peek
S1



1 2 3

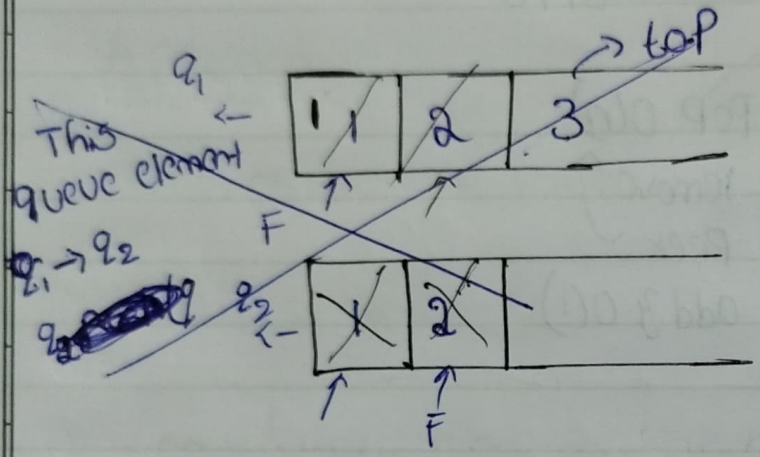
Teacher's Signature _____

Question 3

Stack using 2 queues

Push O(n) or Pop O(n)

Push [add - O(1)]



Note: In queue delete the element at front.

Top = 3 → then return

Top = 2

Top = 1

$q_1 \rightarrow$

$q_2 \rightarrow$ | 1 | 2 | 3 |

Date: .....

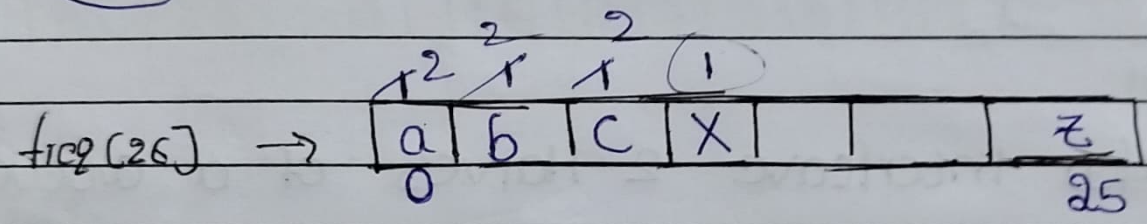
Expt. No.: .....

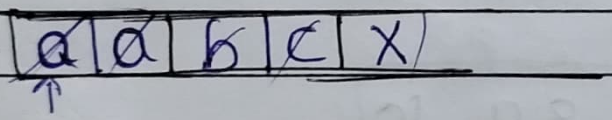
Page No.: .....

Q4) First non-repeating letter in a Stream of characters

"aabccxb" → String

		Print
a	→	a
aa	→	-1
aab	→	b
aabc	→	b
aabcc	→	b
aabccx	→	b
aabccxb	→	x





String	Print
a	a
aa	-1
aab	b
aabc	b
aabcc	b
aabccx	b
aabccxb	x

Approach

```
int freq[] = new int[26];

<Queue>Character> q1 = new
  Queue<>();

for (int i=0; i<Str.length(); i++) {
```

Teacher's Signature _____

```
char ch = str.charAt(i);
freq[ch-'a']++;
q.remove(); → If till found 1st non-repeating character
freq[q.peek()] = 1
if(isEmpty()){
    return -1;
}
```

Q5)

~~Reverse~~ Interleave 2 Halves of a queue (even length)

Ex:-

1	2	3	4	5	6	7	8	9	10

Answer:-

1 6 2 7 3 8 4 9 5 10

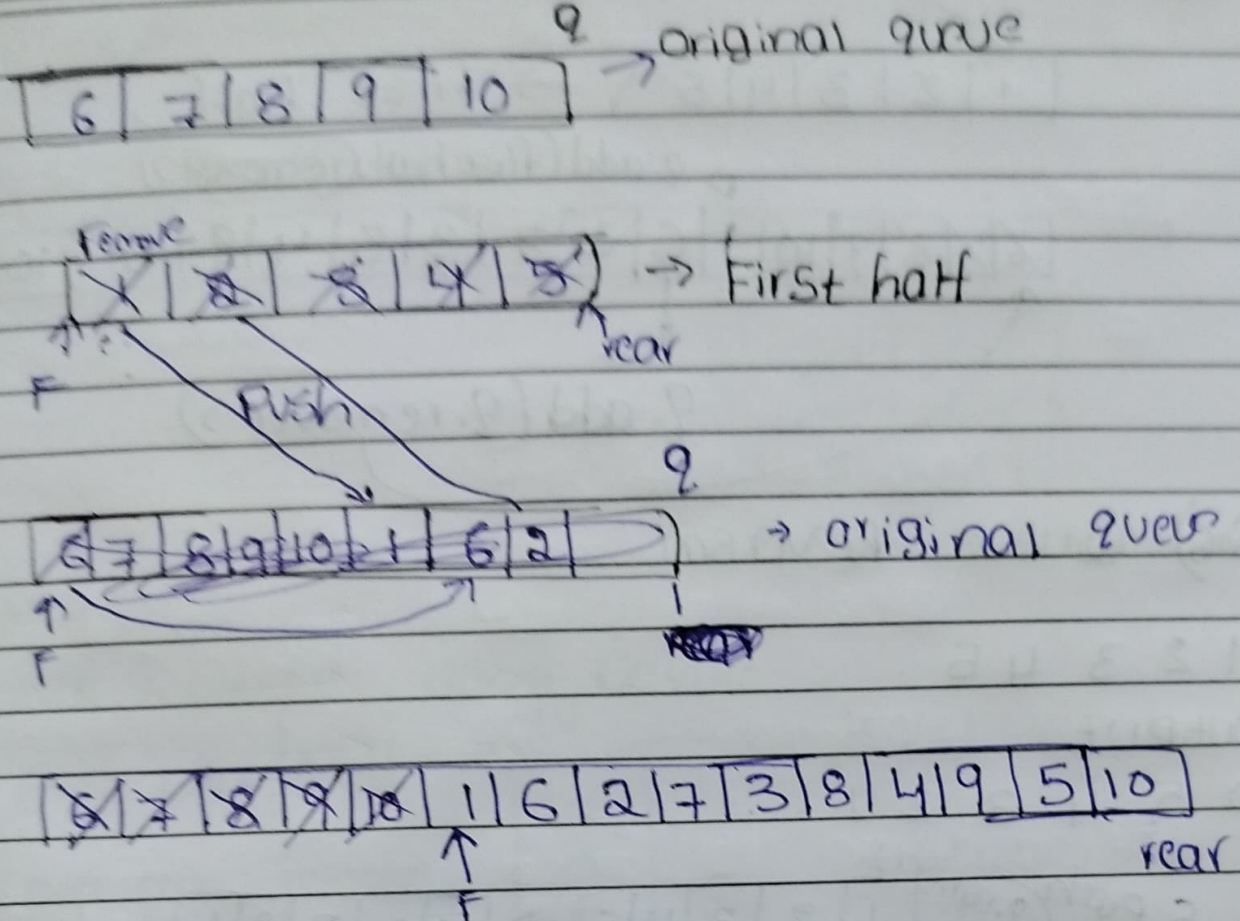
Size = n

Size/2 = 5

1st half queue2 (first)

Date: .....

Expt No.: ..... Page No.: .....



Approach

We create one more queue in fun → first half

Size = q.size

```
for(int i=0; i<size/2; i++) {
    firsthalf.add(q.remove)
}
```

Then

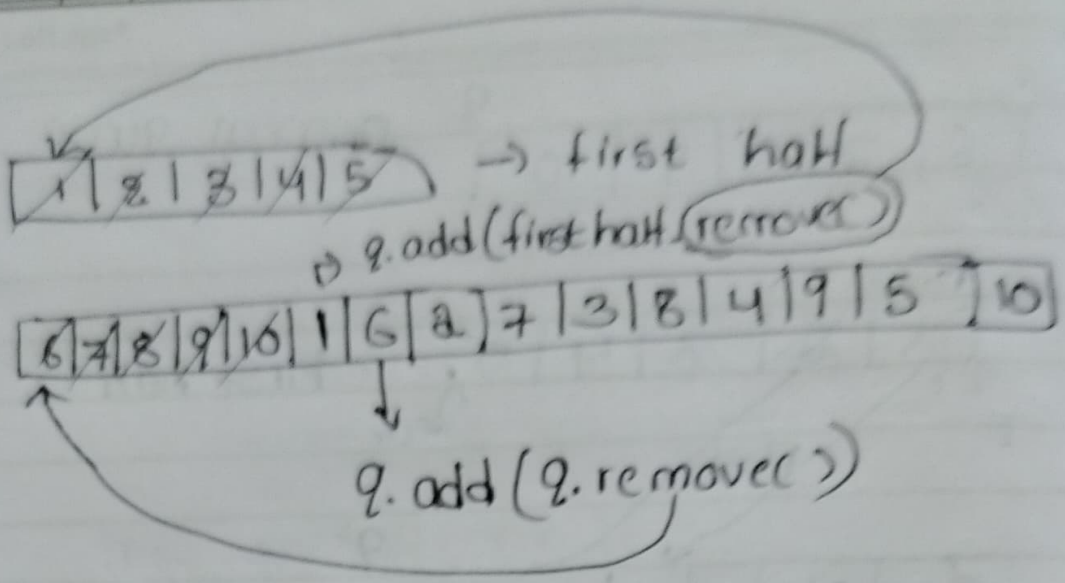
first half : 1, 2, 3, 4, 5

q : 6, 7, 8, 9, 10

```
While(!firsthalf.isEmpty()) {
    q.add(firsthalf.remove());
    q.add(q.remove());
```

Teacher's Signature _____

Then

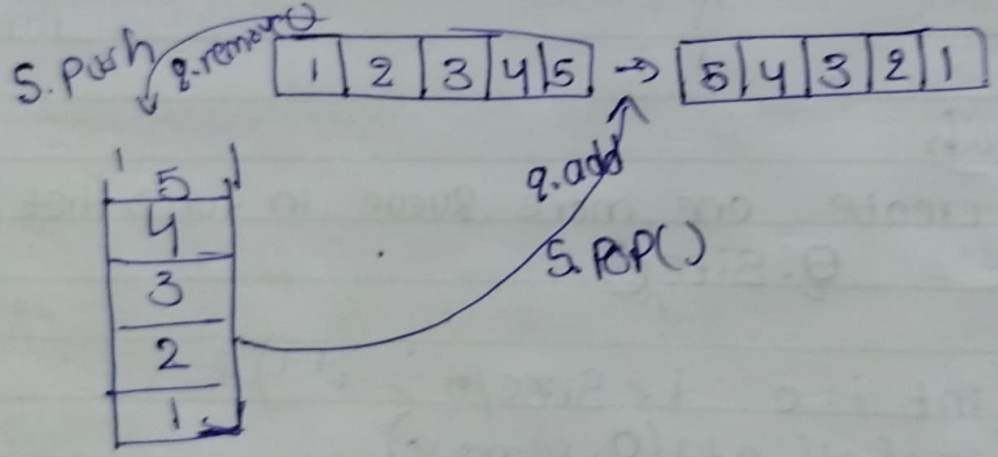


Q6) Queue Reversal

1 2 3 4 5

output

→ 5 4 3 2 1

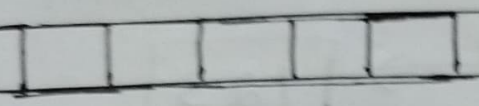


Date: .....

Expt No: ..... Page No.: .....

Deque

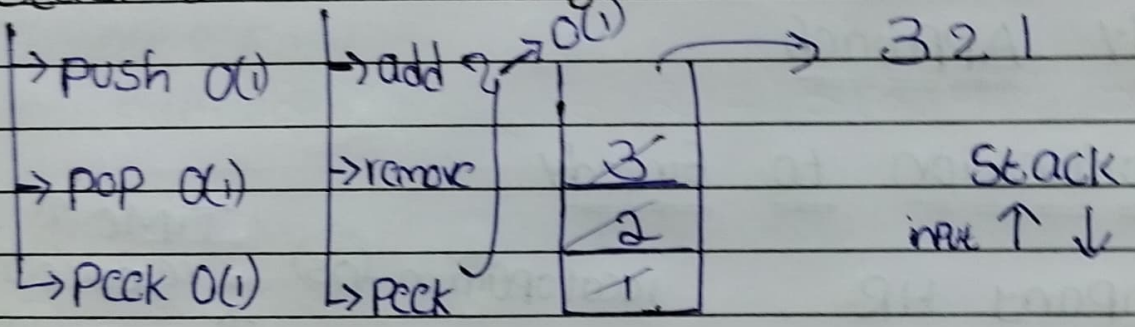
Double ended queue

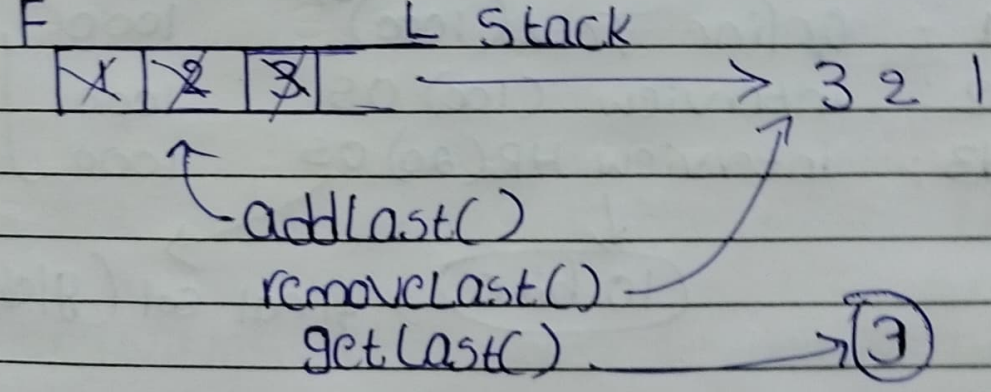


addFirst()
addLast()
removeFirst()
removeLast()
getFirst()
getLast()

Q7)

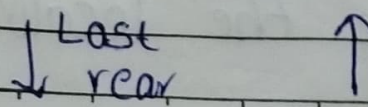
Stack & Queue using Deque



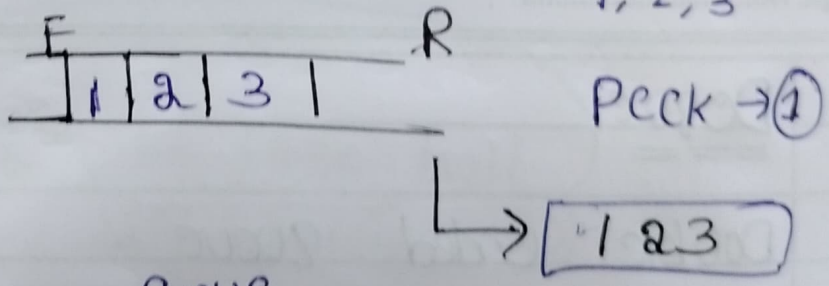


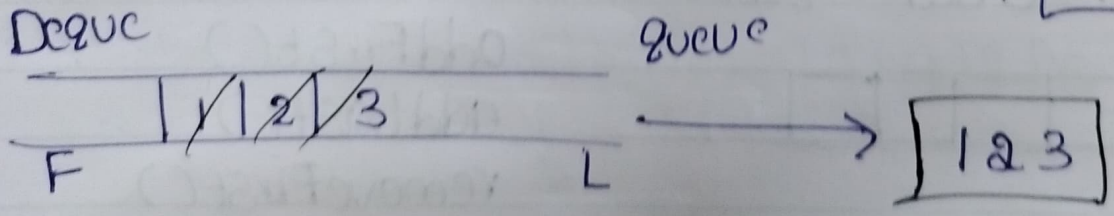
Queue

FIFO



Teacher's Signature _____





add → addLast()
remove → removeFirst()
Peek → getFirst() → 1