## Queue

- Queue using Array

```
public class QueueB {
      public static boolean isEmpty() {
      public static boolean isFull() {
         return rear == size-1;
          if(isFull()) {
          if(isEmpty()) {
              System.out.println("empty queue");
```

```
    rear-;
    return front;
}

public static int peek() {
    if(isEmpty()) {
        System.out.println("empty queue");
        return -1;
    }

    return arr[0];
}

public static void main(String args[]) {
    Queue q = new Queue(5);
    q.add(1);
    q.add(2);
    q.add(3);
    System.out.println(q.remove());
    System.out.println(q.peek());
}
```

## Circular queue using array

```
//circular queue using array
public class Queue {
    static class Queue {
        static int arr[];
        static int size;
        static int front = -1;
        static int rear = -1;

        Queue(int size) {
            this.size = size;
            arr = new int[size];
        }

        public static boolean isEmpty() {
            return rear == -1 && front == -1;
        }
}
```

```
if(front == -1) {
   if(isEmpty()) {
       front = (front+1)%size;
public static int peek() {
   if(isEmpty()) {
       System.out.println("empty queue");
```

```
return arr[front];
}

public static void main(String args[]) {
    Queue q = new Queue(5);
    q.add(1);
    q.add(2);
    q.add(3);
    q.add(4);
    q.add(5);
    System.out.println(q.remove());
    q.add(6);
    System.out.println(q.remove());
    q.add(7);

while(!q.isEmpty()) {
        System.out.println(q.remove());
    }
}
```

## - Queue using Linked List

```
//queue using Linked List
public class QueueB {
    static class Node {
        int data;
        Node next;
        Node(int data) {
            this.data = data;
            next = null;
        }
    }
    static class Queue {
        static Node head = null;
        static Node tail = null;
        public static boolean isEmpty() {
            return head == null && tail == null;
        }
}
```

```
Node newNode = new Node (data);
   if(isEmpty()) {
   if(isEmpty()) {
       System.out.println("empty queue");
public static int peek() {
   if(isEmpty()) {
Queue q = new Queue();
```

```
while(!q.isEmpty()) {
         System.out.println(q.peek());
         q.remove();
}
```

Java Collection Framework

```
Queue<Integer> q = new ArrayDeque();
while(!q.isEmpty()) {
    System.out.println(q.peek());
   q.remove();
```

- Queue using 2 stacks

```
//queue using 2 stacks
import java.util.*;

public class QueueB {
   static class Queue {
```

```
static Stack<Integer> s1 = new Stack<>();
   public static boolean isEmpty() {
       return s1.isEmpty();
       while(!s1.isEmpty()) {
           s2.push(s1.pop());
       s1.push(data);
       while(!s2.isEmpty()) {
           s1.push(s2.pop());
       return s1.pop();
       return sl.peek();
public static void main(String args[]) {
   Queue q = new Queue();
   while(!q.isEmpty()) {
       System.out.println(q.peek());
```