

- 1.) It follows first in first out approach
- 2.) .add() → It will add my element in the last
- 3.) .remove() → It will remove my head from queue and also return.
- 4.) .size() → It tells us about size of queue
- 5.) .isEmpty() → It will return T/F based on size of queue.
- 6.) .peek() → It just return the head of Head.

1. add 10 ✓
2. add 20 ✓
3. add 30 ✓



4. remove() ✓ → 10
5. peek() ✓ → 20
6. size() ✓ → 2

7. add 40 →
8. remove() ✓
9. remove() ✓
10. size() ✓ → 1
11. peek() = 40

Syntax  
 Queue<Integer> que = new ArrayDeque<>();

Queue<>() α  
 ArrayDeque<>();

That cannot create constructor for interface

normal class

```

class Solution {
    int a, int b;
    p.s.v. max() {
        //
    }
    p.s.v. getmax() {
        //
    }
    getval {
        //
    }
}
  
```

Interface Queue {

```

    p.s.v. getmax();
    p.s.v. add();
    p.s.v. remove();
}
  
```

Class ArrayDeque extends Queue {

```

    max() {
        //
    }
    add() {
        //
    }
    remove() {
        //
    }
}
  
```

Class Bird {

✓ fly() {

Class Eagle extends Bird {

Eat() {

```

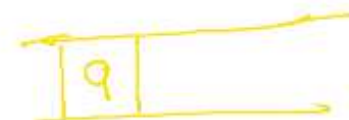
    Bird b = new Bird(); ✓
    Eagle e = new Eagle();
    Bird b = new Eagle();
    Eagle e = new Bird(); α
  }
  b.fly();
  
```

# Ques Queue Implementation

```
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    int n = scn.nextInt();
    int i=0;

    Queue<Integer> que = new ArrayDeque();

    while(i<n){
        int val = scn.nextInt();
        if(val==1){
            System.out.println(que.size());
        }else if(val==2){
            if(que.size()>0){
                que.remove();
            }else{
                System.out.println("-1");
            }
        }else if(val==3){
            int x = scn.nextInt();
            que.add(x);
        }else{
            if(que.size()>0){
                System.out.println(que.peek());
            }else{
                System.out.println("-1");
            }
        }
        i++;
    }
}
```



0  
-1  
9  
1

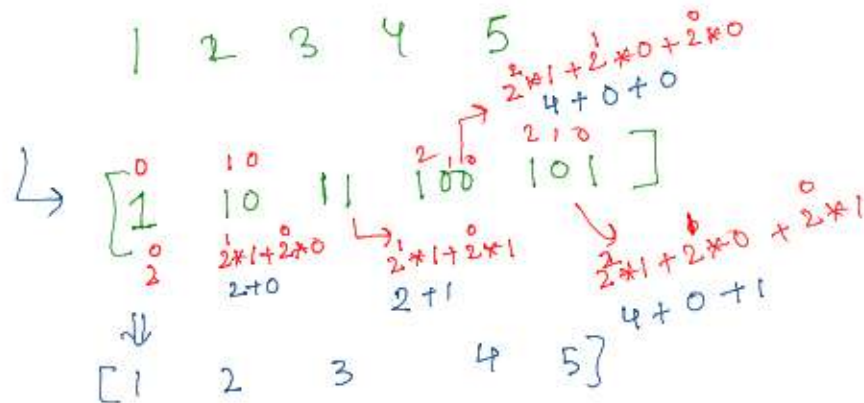
/\* Enter your code here. Read input from STDIN. Print output to STDOUT. Your class

Que: Binary no.

n = 5

$\neq (0, 1)$

1 2 3 4 5



①

11

$[1, 10, 11, 1]$

①

②

③

101 110 111 1000 1001 1010 1011

11 1000 1010 1001 1011

queue = a

"0"),  
"1"),

$[1, 10, 11, 100, 101]$

public class Solution {

```
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    int n = scn.nextInt();
    Queue<String> que = new ArrayDeque<>();
    que.add("1");
    for (int i = 0; i < n; i++) {
        String val = que.remove();
        System.out.print(val + " ");

        que.add(val + "0");
        que.add(val + "1");
    }
}
```

/\* Enter your code here. Read input from STDIN. Print output

101 110 111 1000 1001 1010 1011

val = 1010  
1011

$[1, 10, 11, 100, 101]$

Que

Array = [2, -1, 3, -4, 5, -6, 7, 8]  $k=3$

$i=0$

for ( $i < k$ ) {  
 if ( $arr[i] < 0$ ) {  
 q.add(arr[i]);  
 }  
}



[-1, -4, -6]

① check if element is inside window or not

② add negative to queue

③ sys(peek());

```
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    int n = scn.nextInt();
    int[] arr = new int[n];
    int k = scn.nextInt();
    for (int i=0; i<n; i++) {
        arr[i] = scn.nextInt();
    }

    Queue<Integer> que = new ArrayDeque();

    for (int i=0; i<k; i++) {
        if (arr[i] < 0) {
            que.offer(arr[i]);
        }
    }

    if (!que.isEmpty()) {
        System.out.print(que.peek() + " ");
    } else {
        System.out.print("0" + " ");
    }

    for (int i=k; i<arr.length; i++) {
        if (!que.isEmpty() && que.peek() == arr[i-k]) { // to remove element outside window
            que.poll();
        }
        if (arr[i] < 0) { // add only -ve value
            que.offer(arr[i]);
        }
        if (arr[i] < 0) { // add only -ve value
            que.offer(arr[i]);
        }
        if (!que.isEmpty()) { // print first -ve value of window
            System.out.print(que.peek() + " ");
        } else {
            System.out.print("0" + " ");
        }
    }
}

/* Enter your code here. Read input from STDIN. Print output to STDOUT. Your class is */
```