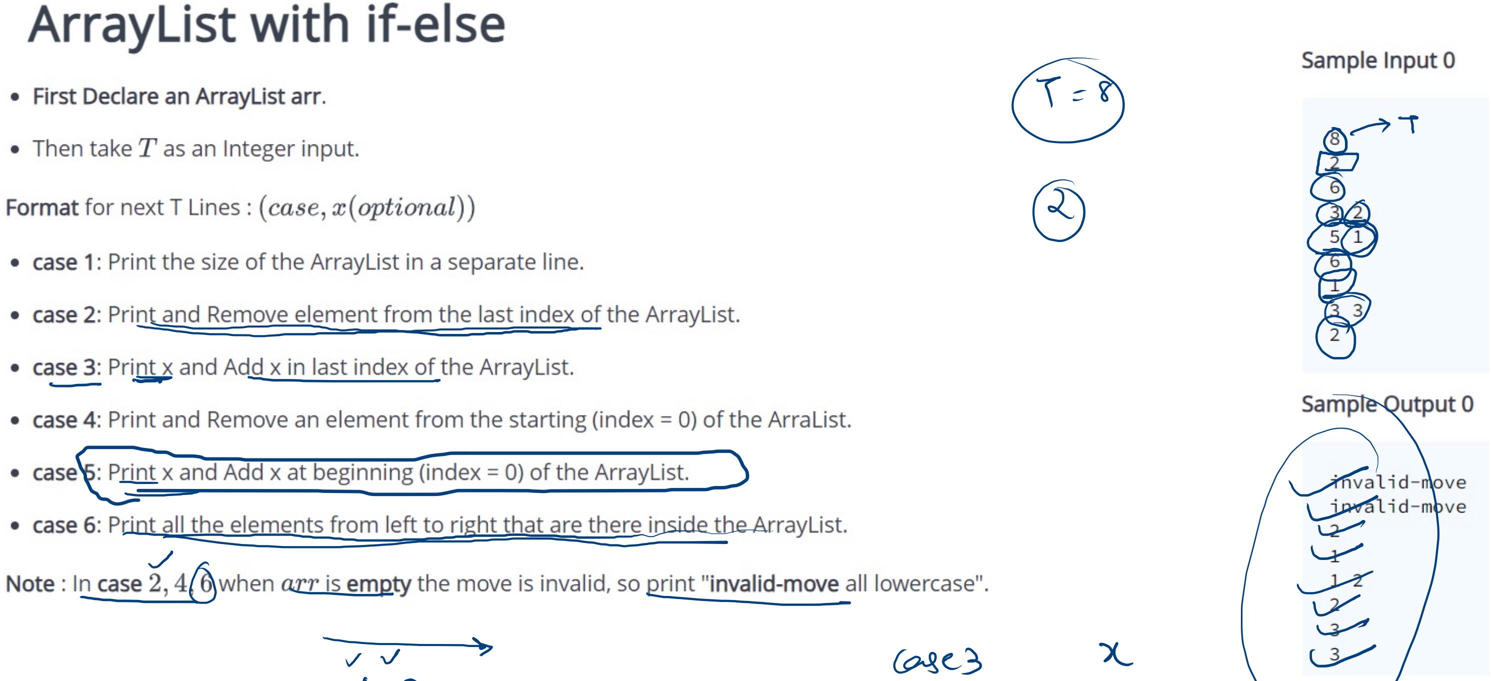


ArrayList with if-else

- First Declare an ArrayList arr.
- Then take T as an Integer input.

Format for next T Lines : $(case, x(optional))$

- case 1: Print the size of the ArrayList in a separate line.
- case 2: Print and Remove element from the last index of the ArrayList.
- case 3: Print x and Add x in last index of the ArrayList.
- case 4: Print and Remove an element from the starting (index = 0) of the ArraList.
- case 5: Print x and Add x at beginning (index = 0) of the ArrayList.
- case 6: Print all the elements from left to right that are there inside the ArrayList.

Note : In case $2, 4, 6$ when arr is empty the move is invalid, so print "invalid-move all lowercase".

Sample Input 0

```
8
2
6
3 2
5 1
6
1
3 3
2
```

→ T

Sample Output 0

```
invalid-move
invalid-move
2
1
1 2
2
3
3
```

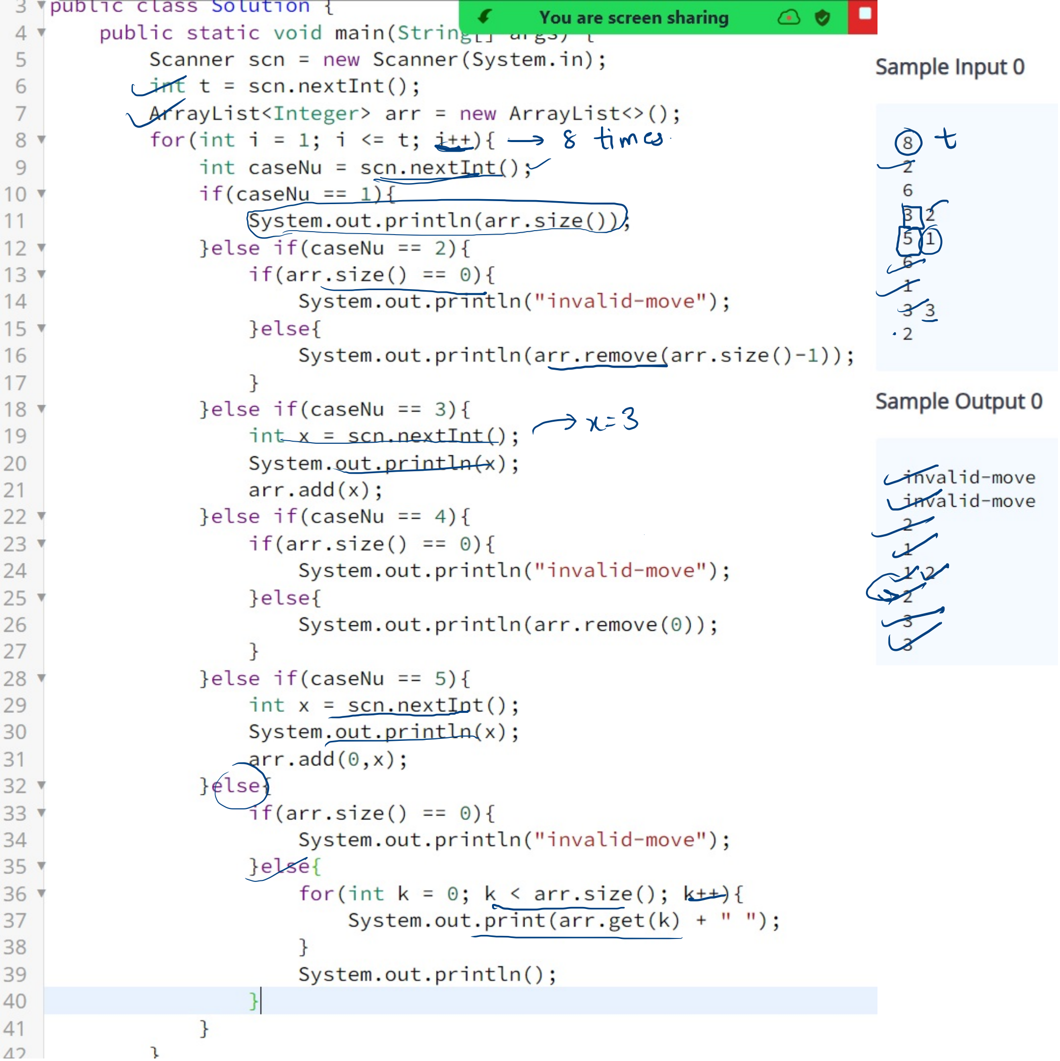
T = 8

2

✓ ✓

< 1 2 >

case3	x
3	2
5	1
3	3



```java
public class Solution {
    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int t = scn.nextInt();
        ArrayList<Integer> arr = new ArrayList<>();
        for(int i = 1; i <= t; i++){
            int caseNu = scn.nextInt();
            if(caseNu == 1){
                System.out.println(arr.size());
            }else if(caseNu == 2){
                if(arr.size() == 0){
                    System.out.println("invalid-move");
                }else{
                    System.out.println(arr.remove(arr.size()-1));
                }
            }else if(caseNu == 3){
                int x = scn.nextInt();
                System.out.println(x);
                arr.add(x);
            }else if(caseNu == 4){
                if(arr.size() == 0){
                    System.out.println("invalid-move");
                }else{
                    System.out.println(arr.remove(0));
                }
            }else if(caseNu == 5){
                int x = scn.nextInt();
                System.out.println(x);
                arr.add(0,x);
            }else{
                if(arr.size() == 0){
                    System.out.println("invalid-move");
                }else{
                    for(int k = 0; k < arr.size(); k++){
                        System.out.print(arr.get(k) + " ");
                    }
                    System.out.println();
                }
            }
        }
    }
```

→ 8 times

→ x=3

Sample Input 0

```
8
2
6
3 2
5 1
6
1
3 3
2
```

t

Sample Output 0

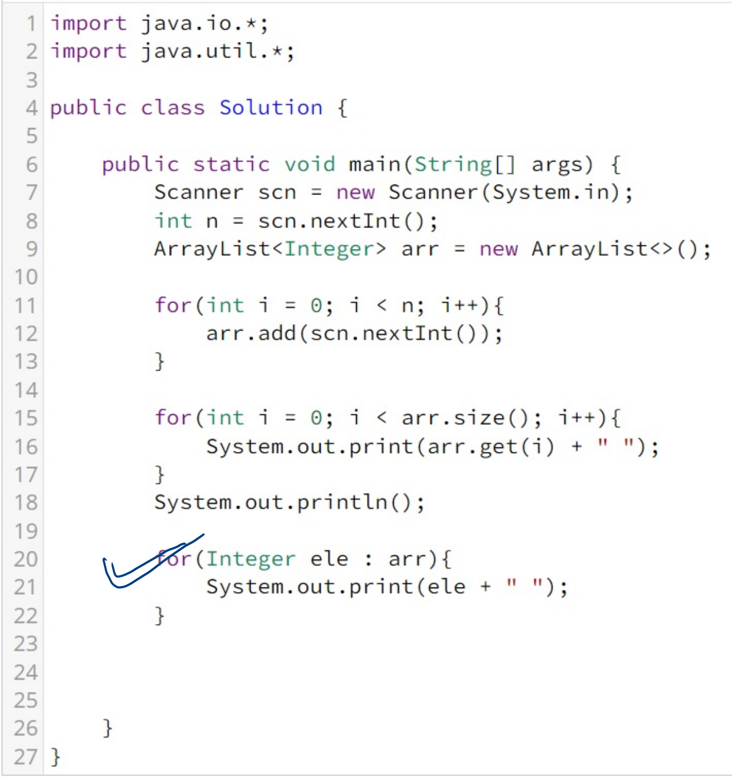
```
invalid-move
invalid-move
2
1
1 2
2
3
3
```

< 1 2 >

0 1

k = 0
1
2

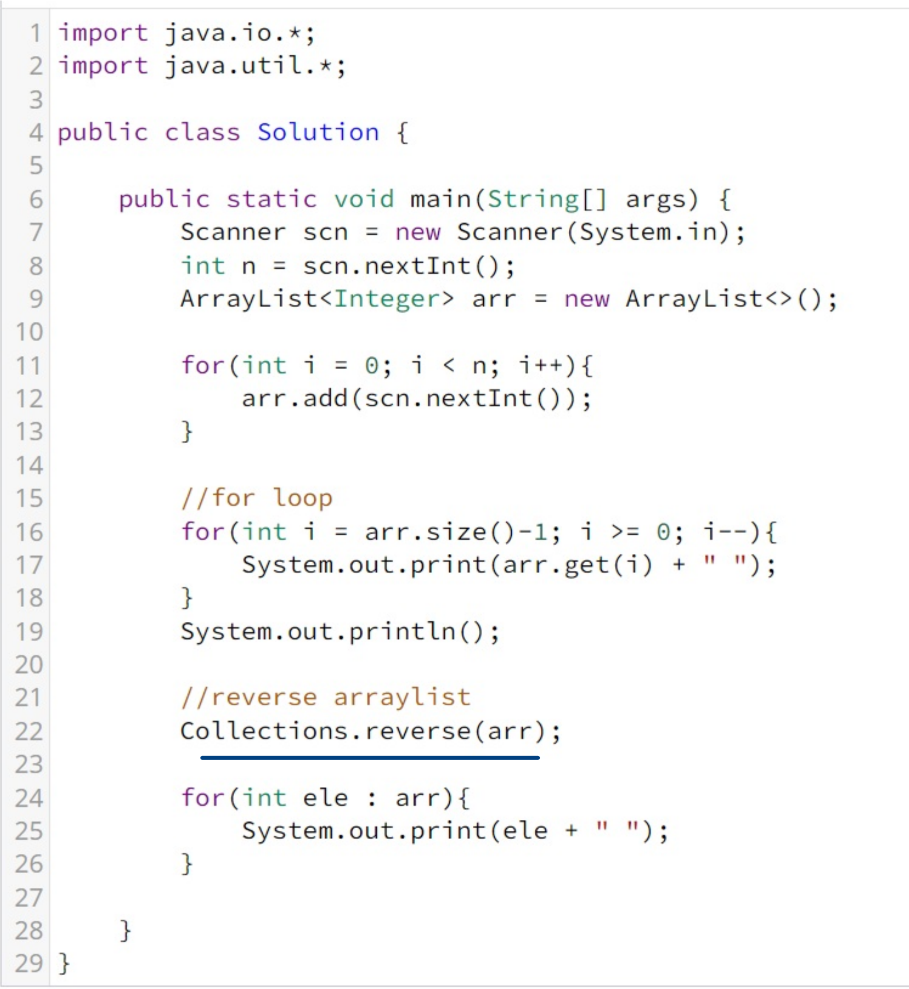
0 < 2
1 < 2 ✓
2 < 2 ✗



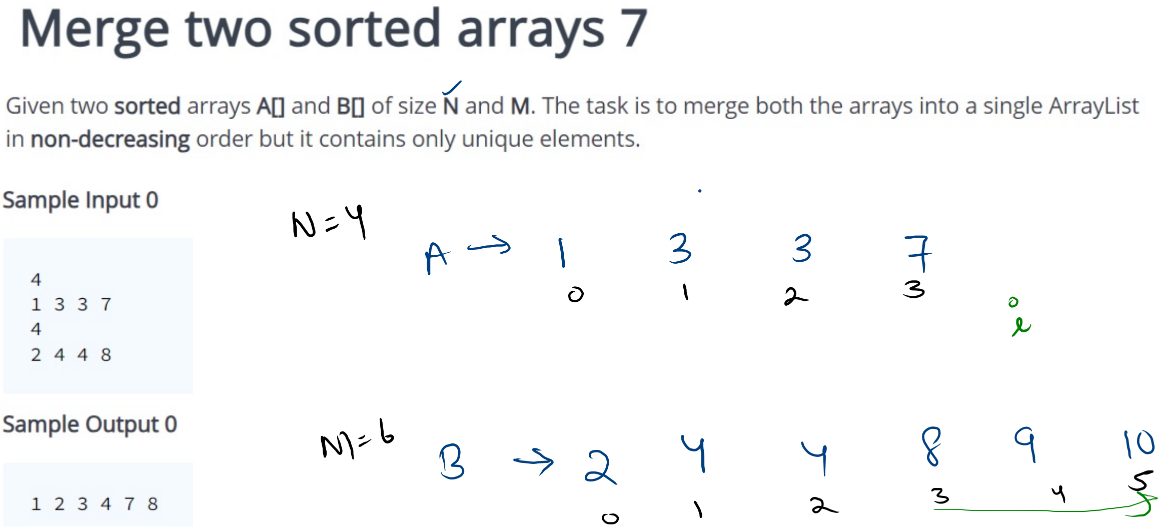
```
import java.io.*;
import java.util.*;

public class Solution {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();
        ArrayList<Integer> arr = new ArrayList<>();

        for(int i = 0; i < n; i++){
            arr.add(scn.nextInt());
        }

        for(int i = 0; i < arr.size(); i++){
            System.out.print(arr.get(i) + " ");
        }
        System.out.println();

        for(Integer ele : arr){
            System.out.print(ele + " ");
        }


    }
}
```

4

10 20 30 40



```java
import java.io.*;
import java.util.*;

public class Solution {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();
        ArrayList<Integer> arr = new ArrayList<>();

        for(int i = 0; i < n; i++){
            arr.add(scn.nextInt());
        }

        //for loop
        for(int i = arr.size()-1; i >= 0; i--){
            System.out.print(arr.get(i) + " ");
        }
        System.out.println();

        //reverse arraylist
        Collections.reverse(arr);

        for(int ele : arr){
            System.out.print(ele + " ");
        }

    }
}
```



Merge two sorted arrays 7

Given two **sorted** arrays **A[]** and **B[]** of size **N** and **M**. The task is to merge both the arrays into a single ArrayList in **non-decreasing** order but it contains only unique elements.

Sample Input 0

```
4
1 3 3 7
4
2 4 4 8
```

Sample Output 0

```
1 2 3 4 7 8
```

N=4

A → 1 3 3 7

0 1 2 3

i

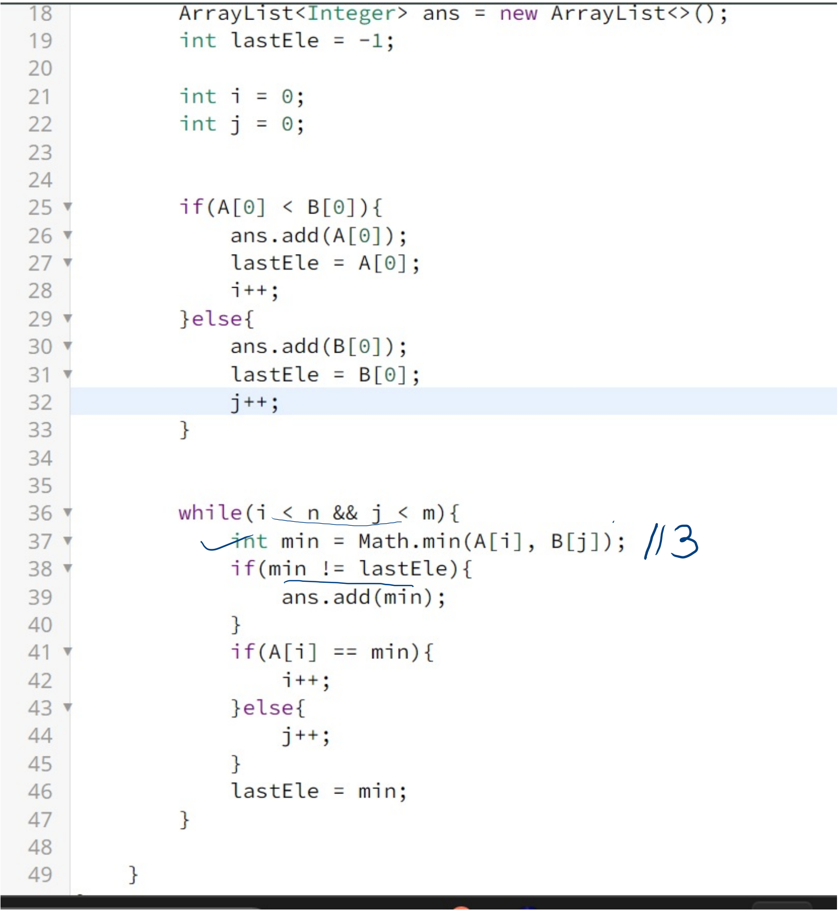
M=6 B → 2 4 4 8 9 10

0 1 2 3 4 5

j

lE = ~~8~~ 9

< 1 2 3 4 7 8 9 10 >



```
        ArrayList<Integer> ans = new ArrayList<>();
        int lastEle = -1;

        int i = 0;
        int j = 0;


        if(A[0] < B[0]){
            ans.add(A[0]);
            lastEle = A[0];
            i++;
        }else{
            ans.add(B[0]);
            lastEle = B[0];
            j++;
        }


        while(i < n && j < m){
            int min = Math.min(A[i], B[j]); //3
            if(min != lastEle){
                ans.add(min);
            }
            if(A[i] == min){
                i++;
            }else{
                j++;
            }
            lastEle = min;
        }

    }
```

1 3 3 5

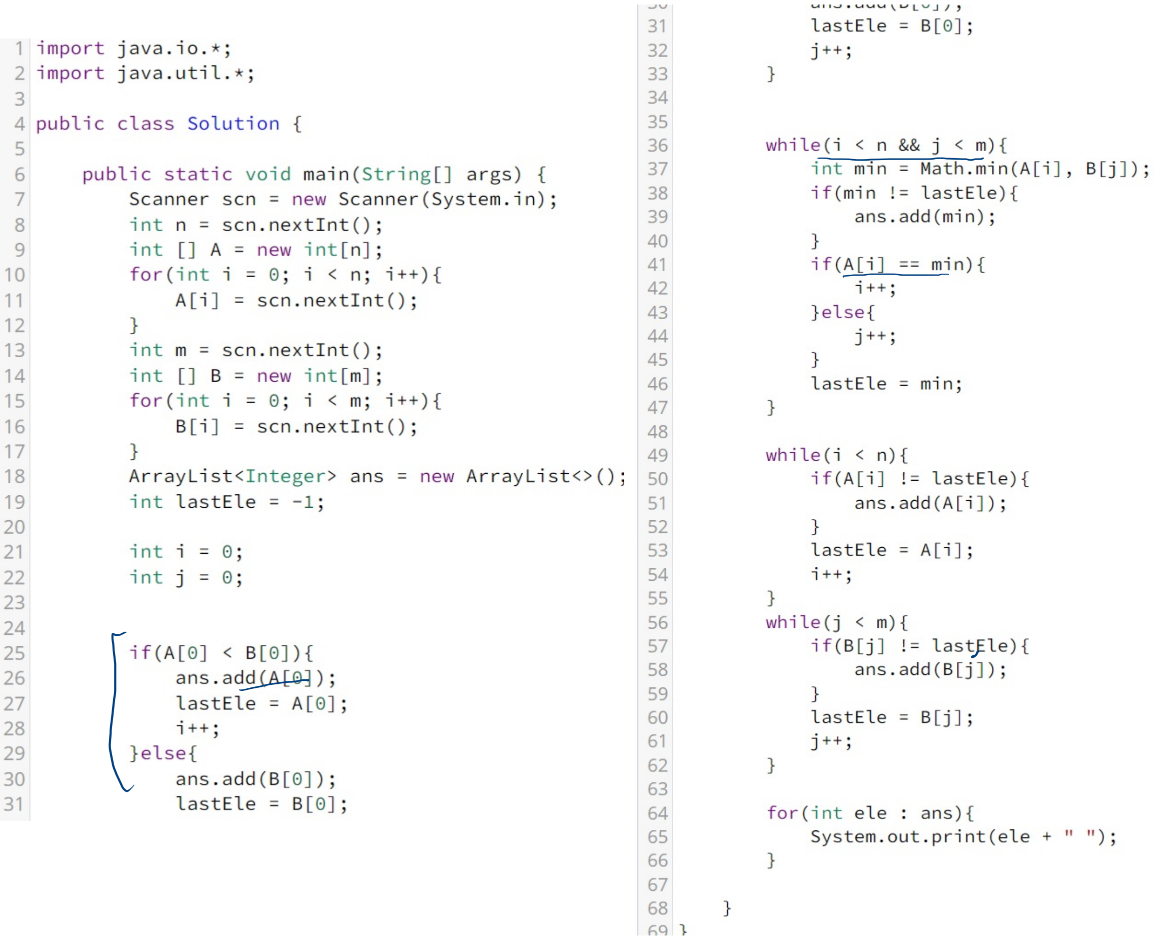
0 i

2 4 4 5 6 7

j

lE = ~~-1~~ ~~1~~ ~~2~~ ~~3~~ ~~4~~ 5

{1 2 3 4 5



```
import java.io.*;
import java.util.*;

public class Solution {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();
        int [] A = new int[n];
        for(int i = 0; i < n; i++){
            A[i] = scn.nextInt();
        }
        int m = scn.nextInt();
        int [] B = new int[m];
        for(int i = 0; i < m; i++){
            B[i] = scn.nextInt();
        }
        ArrayList<Integer> ans = new ArrayList<>();
        int lastEle = -1;

        int i = 0;
        int j = 0;


        if(A[0] < B[0]){
            ans.add(A[0]);
            lastEle = A[0];
            i++;
        }else{
            ans.add(B[0]);
            lastEle = B[0];
            j++;
        }


        while(i < n && j < m){
            int min = Math.min(A[i], B[j]);
            if(min != lastEle){
                ans.add(min);
            }
            if(A[i] == min){
                i++;
            }else{
                j++;
            }
            lastEle = min;
        }

        while(i < n){
            if(A[i] != lastEle){
                ans.add(A[i]);
            }
            lastEle = A[i];
            i++;
        }
        while(j < m){
            if(B[j] != lastEle){
                ans.add(B[j]);
            }
            lastEle = B[j];
            j++;
        }

        for(int ele : ans){
            System.out.print(ele + " ");
        }

    }
}
```

A→ 1 2 2 3
0 1 2 3
i

B→ 2 4
0 1
j

< 1 2 3 4 >

lE = ~~-1~~ ~~1~~ ~~2~~ ~~3~~
4