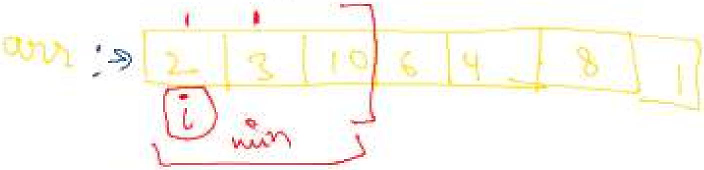
Question   Max diff

Given an array **arr[]** of integers, find out the **maximum difference** between any two elements such that **larger element appears after the smaller number**.



3 - 2

(1 > 0)

18 - 2

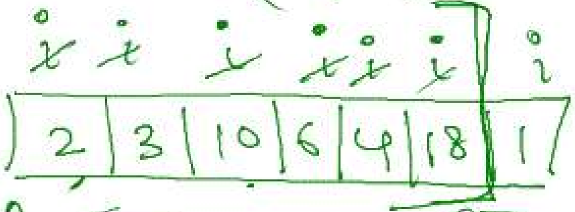
= 16

min

max diff ->

min = integer.MAX VALUE   2

✓ maxdiff = ~~0~~ ~~1~~ ~~8~~ 16



min ←

6 - 2 = 4

10 - 2

= 8

```
for( i - 0 to arr.length) {
    if (arr[i] < min) {        (2 < +∞){
        min = arr[i];
    }
    if ( arr[i] - min > maxDiff) {
        max = arr[i] - min;
    }
}
sysO ( maxDiff)
```

max

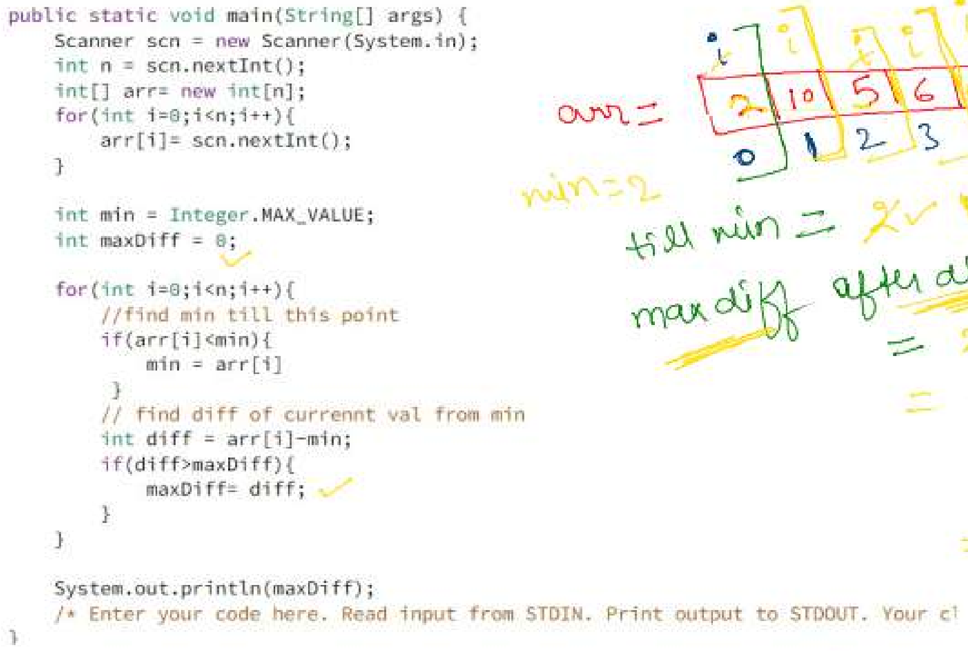
16

2

① find min till that point.

② substract arr[i] from min to check max diff.

Q



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

    int min = Integer.MAX_VALUE;
    int maxDiff = 0;

    for(int i=0;i<n;i++){
        //find min till this point
        if(arr[i]<min){
            min = arr[i]
         }
        // find diff of currennt val from min
        int diff = arr[i]-min;
        if(diff>maxDiff){
            maxDiff= diff;
        }
    }

    System.out.println(maxDiff);
    /* Enter your code here. Read input from STDIN. Print output to STDOUT. Your c
}
```

arr =

0	1	2	3	4	5	6
2	10	5	6	3	11	1

min=2

till min = ~~2~~ 1

max diff after all subtraction

= 2-2 > 0

= 10-2 > 0

8>0

= 13-2

11>8

2	10

max diff = ~~0~~ ~~8~~ 11

→ main

arr

2	10	5	1	6	13	4

8

2	10	5	1	13

min = 1

max = ~~8~~ 12

min = ~~2~~ 1

diff = arr[i] - min = 13 - 1 = 12

max diff = ~~0~~ ~~8~~ 12

max diff

10	1	3

10-1 = 9 ✗

10	1	3

= 2

min = ~~10~~ 1

arr[i] = ~~10~~ 3

max diff = ~~0~~ 2

Question

arr ⇒

1	1	1	2	3	3	3	4	3	4

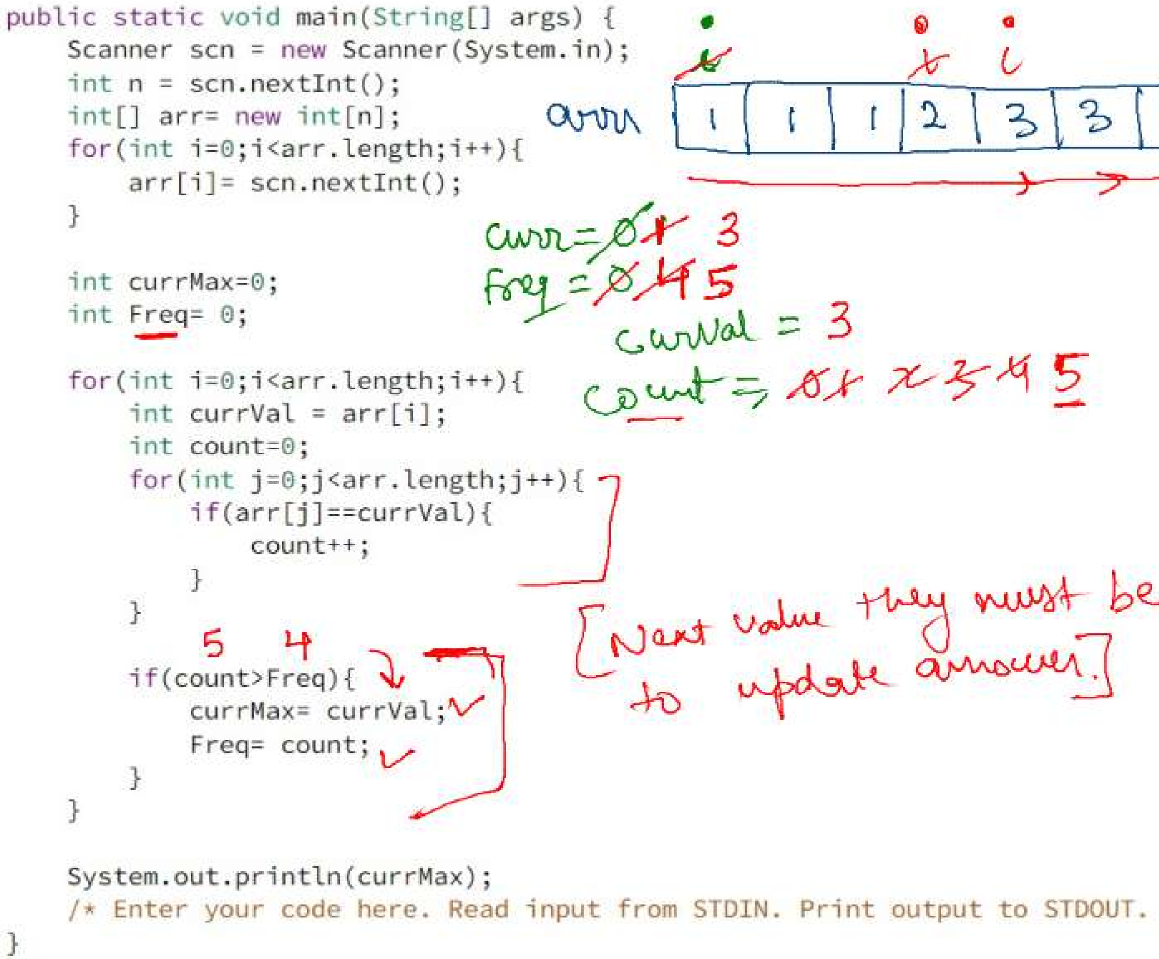
i i i i

max repeatitive no: 3

max = 4

arr[i] = 1

ans update बदलो नहीं



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

    int currMax=0;
    int Freq= 0;

    for(int i=0;i<arr.length;i++){
        int currVal = arr[i];
        int count=0;
        for(int j=0;j<arr.length;j++){
            if(arr[j]==currVal){
                count++;
            }
        }
              5   4
        if(count>Freq){
            currMax= currVal;
            Freq= count;
        }
    }

    System.out.println(currMax);
    /* Enter your code here. Read input from STDIN. Print output to STDOUT.
}
```

i i i

arr

1	1	1	2	3	3	4	4	3	5	3	8	1

curr = ~~0~~ ~~1~~ 3

freq = ~~0~~ ~~4~~ 5

currVal = 3

count = ~~0~~ ~~1~~ ~~2~~ ~~3~~ ~~4~~ 5

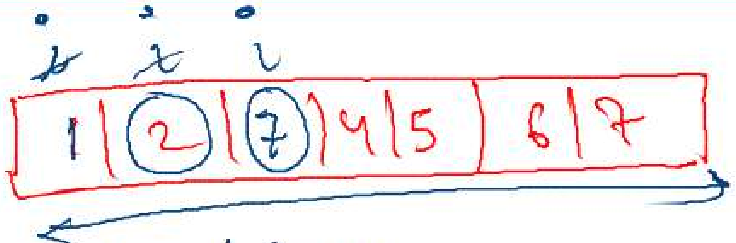
1 , 4 ✓

3 , 5

[Next value they must be greater in count to update answer]

Q dulicate

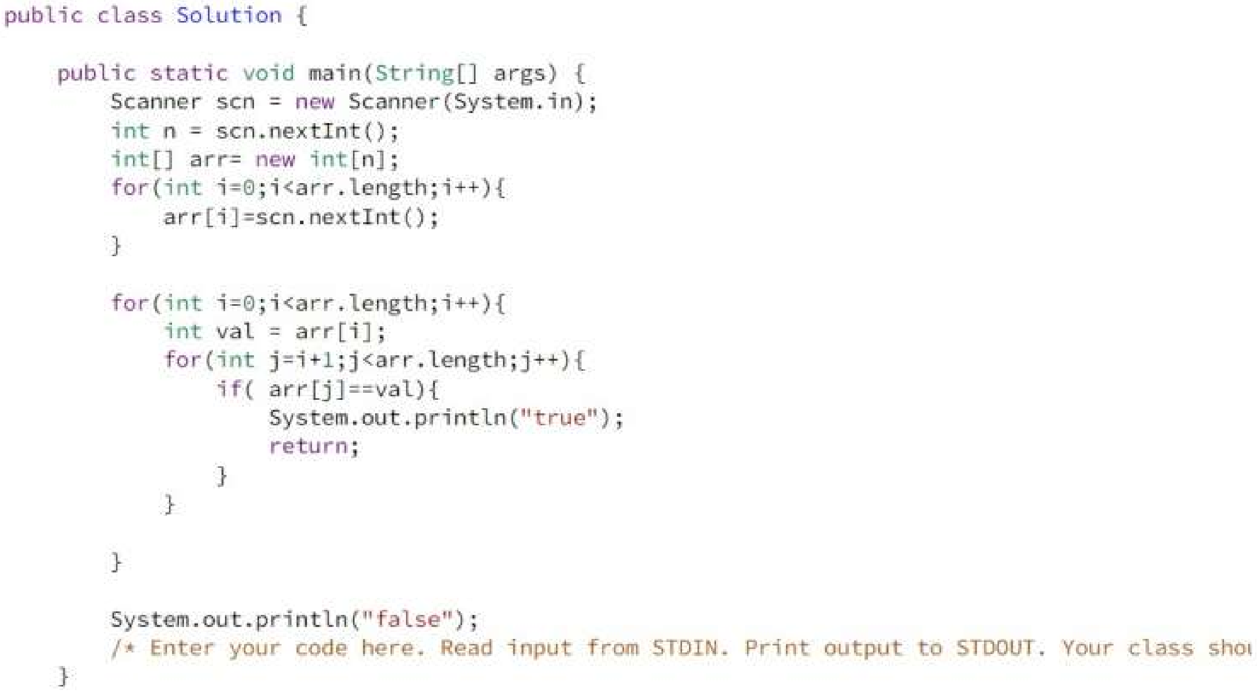
arr



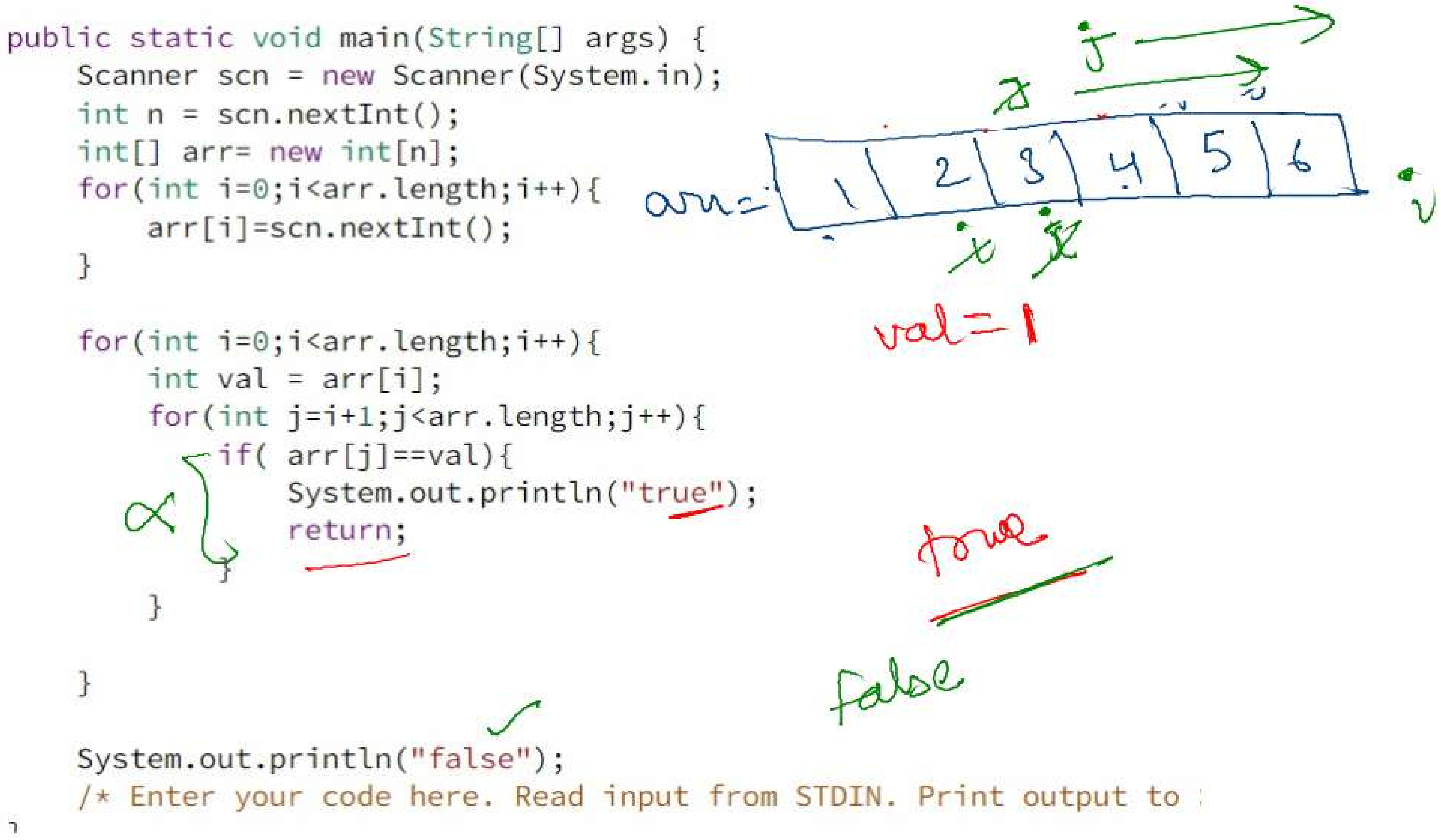
true

false

ans = false | true

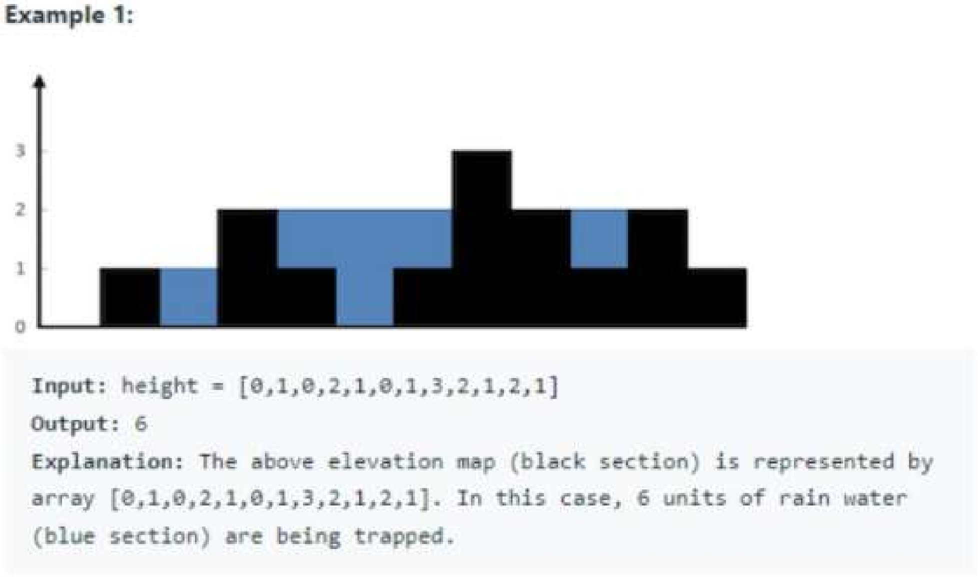


```
public class Solution {

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

        for(int i=0;i<arr.length;i++){
            int val = arr[i];
            for(int j=i+1;j<arr.length;j++){
                if( arr[j]==val){
                    System.out.println("true");
                    return;
                }
            }

        }

        System.out.println("false");
        /* Enter your code here. Read input from STDIN. Print output to STDOUT. Your class shou
    }
```



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

    for(int i=0;i<arr.length;i++){
        int val = arr[i];
        for(int j=i+1;j<arr.length;j++){
            if( arr[j]==val){
                System.out.println("true");
                return;
            }
        }

    }

    System.out.println("false");
    /* Enter your code here. Read input from STDIN. Print output to
}
```

Qus trap water



Example 1:

Input: height = [0,1,0,2,1,0,1,3,2,1,2,1]

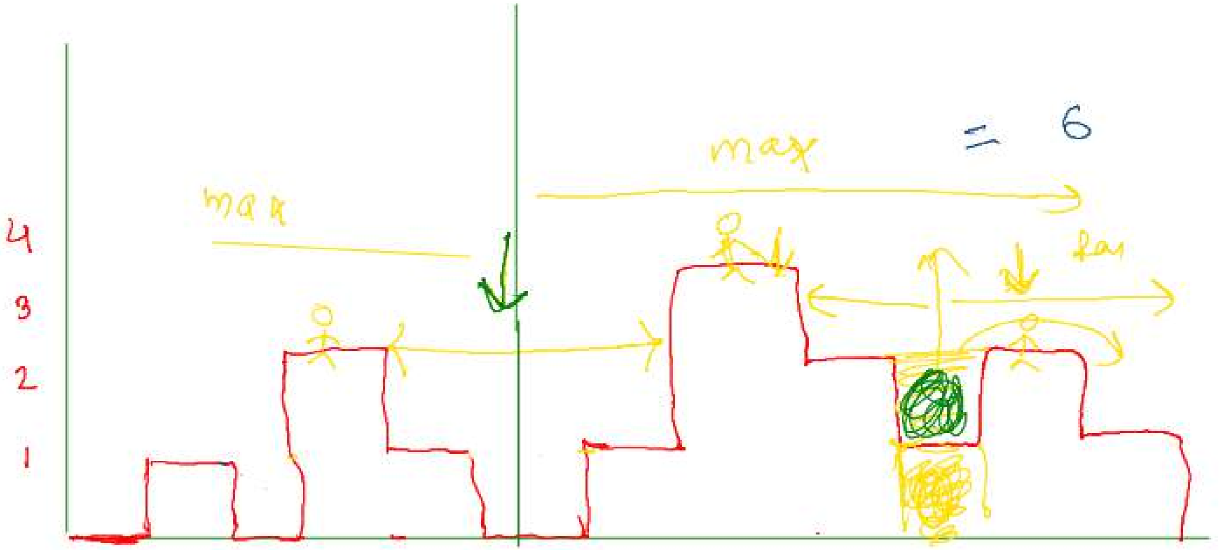
Output: 6

Explanation: The above elevation map (black section) is represented by array [0,1,0,2,1,0,1,3,2,1,2,1]. In this case, 6 units of rain water (blue section) are being trapped.

0, 1, 0, 2, 1, 0, 1, 3, 2, 1, 2, 1

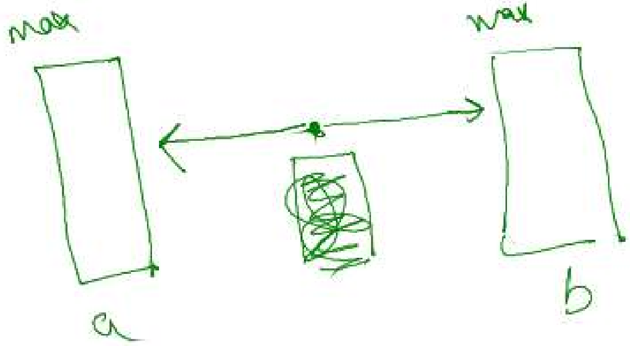
Bruteforce

6

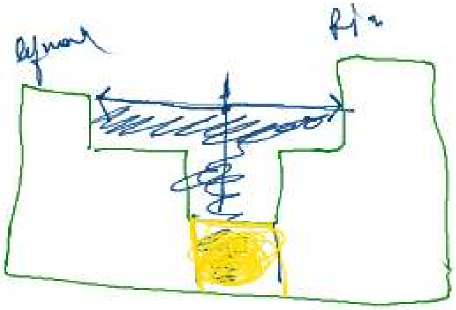


min (left max, Right) - arr[i]

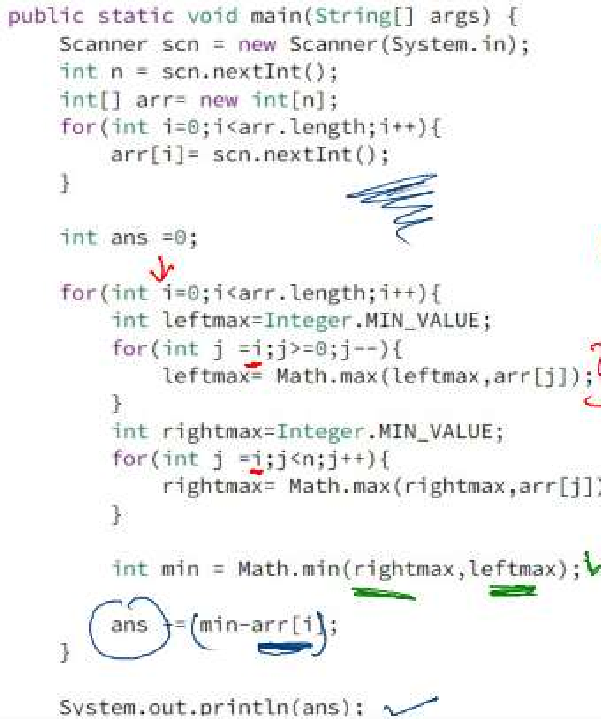
(a , b) ⇒ b- arr[i];



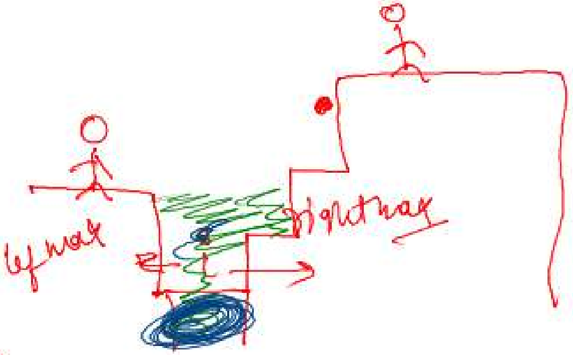
(b) - arr[i]

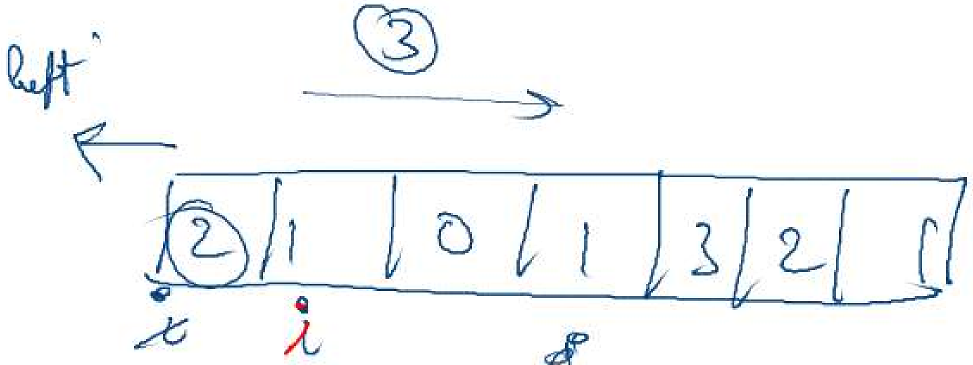


min(leftmax- rightmax)- arr[i]



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

    int ans =0;

    for(int i=0;i<arr.length;i++){
        int leftmax=Integer.MIN_VALUE;
        for(int j =i;j>=0;j--){
            leftmax= Math.max(leftmax,arr[j]);
        }
        int rightmax=Integer.MIN_VALUE;
        for(int j =i;j<n;j++){
            rightmax= Math.max(rightmax,arr[j]);
        }

        int min = Math.min(rightmax,leftmax);

        ans += min-arr[i];
    }

    System.out.println(ans);
```



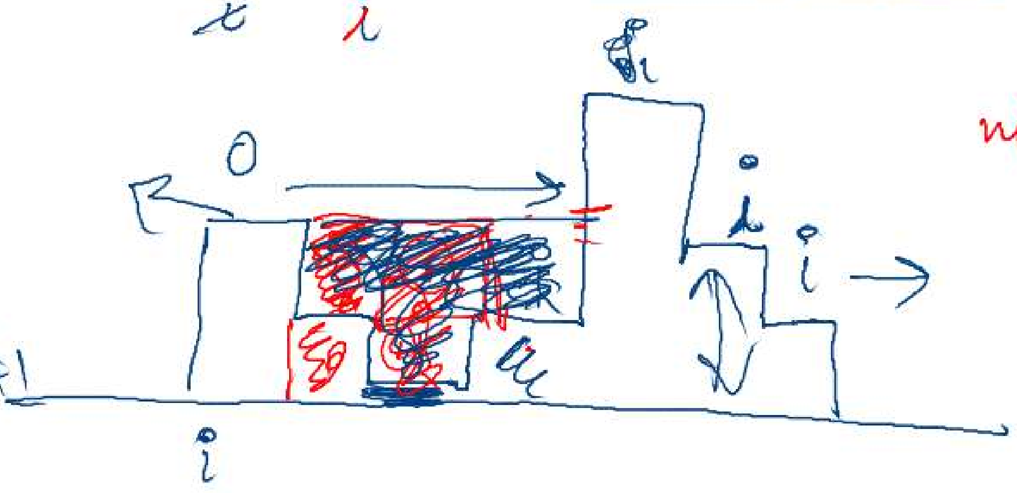


$\min(3, 2)$

$2 - 2$

$= 0$

②



$1 + 2$
$= 3 + 1$
$= 4$

$\min(2, 3) - 0$

$= 2 - 0$

$= 2$

$(3, 1)$

$1 - 1$

$= 0$

$(2, 3) - 1$

$2 - 1 = \boxed{1}$

$\min(2, 3) - arr[i]$

$= 2 - 1 = 1$