

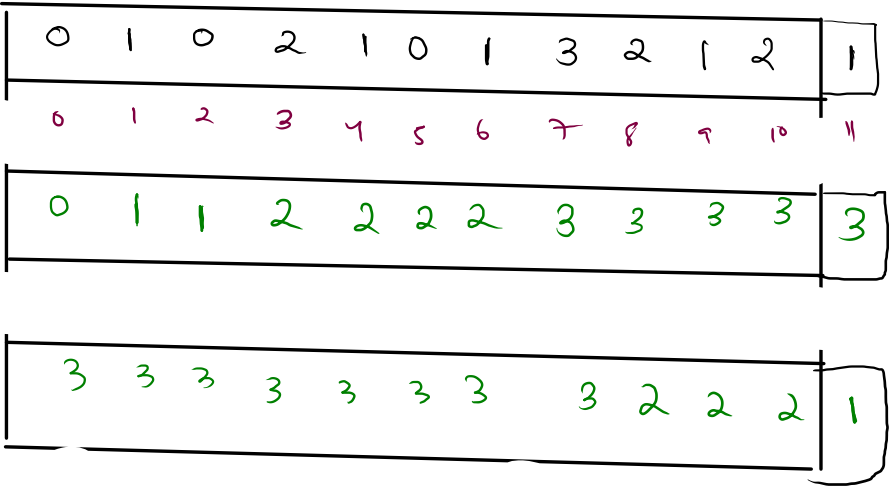
Store Maximum



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

Sample Input 0

```
12
0 1 0 2 1 0 1 3 2 1 2 1
```



$n=12$

0	1	0	2	1	0	1	3	2	1	2	1
0	1	2	3	4	5	6	7	8	9	10	11

$l2r$
(n)

✓0	1	1	2	2	2	2	3	3	3	3	3
----	---	---	---	---	---	---	---	---	---	---	---

→

[0 11]

$$l2r[0] = A[0]$$

$$\rightsquigarrow \underline{l2r[i]} = \text{Max}(l2r[i-1], A[i])$$

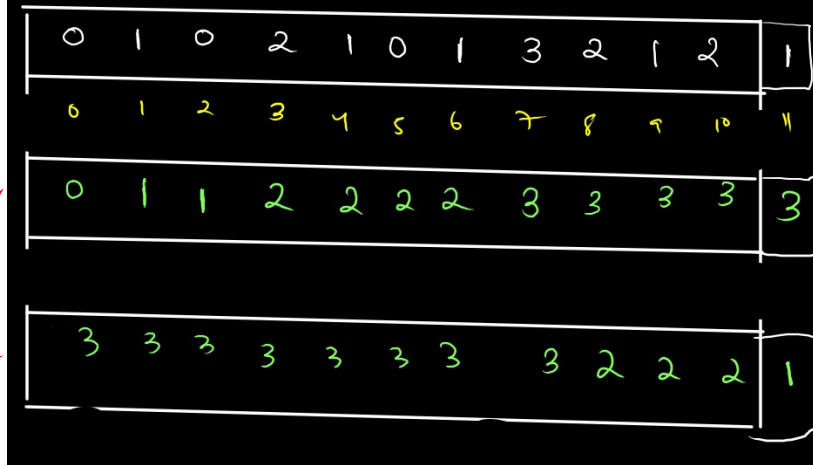
					3	3	3	2	2	2	1
--	--	--	--	--	---	---	---	---	---	---	---

```

4 public class Solution {
5     public static int trap(int[] height) {
6         int n = height.length;
7         int [] l2r = new int[n];
8         l2r[0] = height[0];
9         for(int i = 1; i < n; i++){
10             l2r[i] = Math.max(l2r[i-1], height[i]);
11         }
12
13         int [] r2l = new int[n];
14         r2l[n-1] = height[n-1];
15
16         for(int i = n-2; i >= 0; i--){
17             r2l[i] = Math.max(height[i], r2l[i+1]);
18         }
19
20         int ans = 0;
21         for(int i = 0; i < n; i++){
22             ans += Math.min(l2r[i], r2l[i]) - height[i];
23         }
24         return ans;
25     }
26 }
27

```

height



```
1 import java.io.*;
2 import java.util.*;
3
4 public class Solution {
5     public static int trap(int[] height) {
6         int n = height.length;
7         int [] l2r = new int[n];
8         l2r[0] = height[0];
9         for(int i = 1; i < n; i++){
10             l2r[i] = Math.max(l2r[i-1], height[i]);
11         }
12         int [] r2l = new int[n];
13         r2l[n-1] = height[n-1];
14
15         for(int i = n-2; i >= 0; i--){
16             r2l[i] = Math.max(height[i], r2l[i+1]);
17         }
18         int ans = 0;
19         for(int i = 0; i < n; i++){
20             ans += Math.min(l2r[i], r2l[i]) - height[i];
21         }
22         return ans;
23     }
24
25     public static void main(String[] args) {
26         Scanner scn = new Scanner(System.in);
27         int n = scn.nextInt();
28         int [] A = new int[n];
29         for(int i = 0; i < n; i++){
30             A[i] = scn.nextInt();
31         }
32         int ans = trap(A);
33         System.out.println(ans);
34     }
35 }
```

```

1 import java.util.*;
2 public class Main
3 {
4     public static void main(String[] args) {
5         Scanner scn = new Scanner(System.in);
6         int n = scn.nextInt();
7         for(int i = 0; i < n; i++){
8             System.out.println(i);
9         }
10    }
11 }
12

```

i/p	o/p
5	5
500	500
1000	1000
10 ⁵	10 ⁵

depends
on
i/p

```

1 import java.util.*;
2 public class Main
3 {
4     public static void main(String[] args) {
5         Scanner scn = new Scanner(System.in);
6         int n = scn.nextInt();
7         for(int i = 0; i < 10; i++){
8             System.out.println(i);
9         }
10    }
11 }
12

```

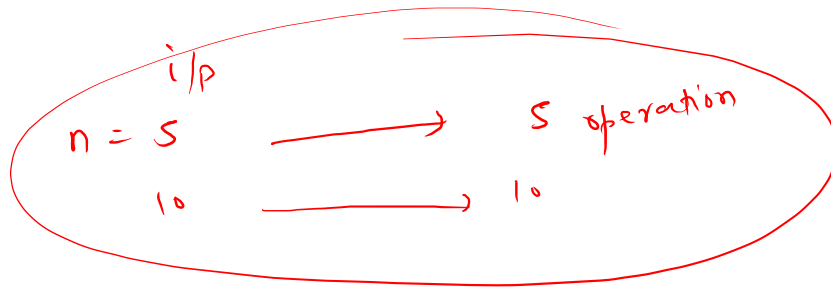
i/p	o/p
5	10
500	10
1000	10
10 ⁵	10

constant itr

Time Complexity (TC)

* $TC \neq$ time to run your program.

* TC: any maths funcⁿ to represent relationship
between no. of i/p & operations



TC ?

why??

(A)

(TC) ↓

SC ↓

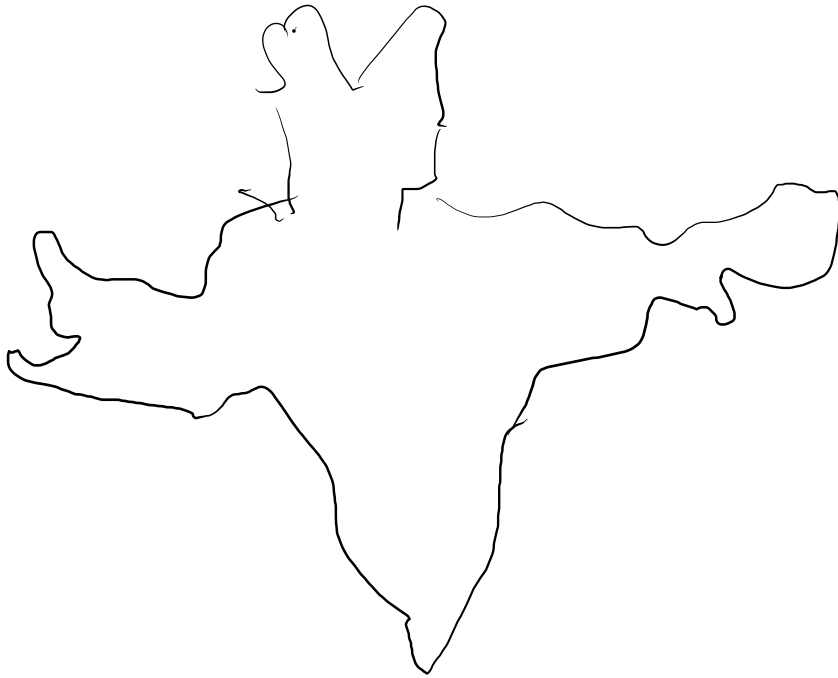
(B)

TC ↓

TC

Notations

$O(1)$



worst \rightarrow Sagar
 O \hookrightarrow at least 1 state

avg \rightarrow 5 states.
 Θ

best \rightarrow 29 states
 ω
omega

Constant

$O(1)$

$O(k)$

$O(1 \times k)$

$k(O(1))$

maths (+ - * / %)

int age = 52;

sys0("Aman")

i/p

$$y = 2x + \textcircled{k} \quad \begin{array}{l} \text{any.} \\ 1 \\ 10 \\ 6 \\ 50 \end{array}$$

$2x + k_1$
 $2x + k_2$

th

$$4x + \textcircled{k}$$

$n = 5$

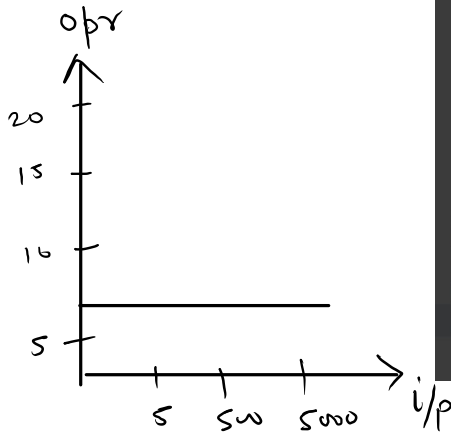
{ 4 mit

$n = 500$

{ 4 4 mit

$n = 500000$

{ 4 mit



```
1 import java.util.*;
2 public class Main
3 {
4     public static void main(String[] args) {
5         Scanner scn = new Scanner(System.in);
6         int n = scn.nextInt();
7         int a = 10;
8         a = 17;
9         int b = 20;
10        System.out.println(a + b);
11    }
12 }
13
```

↓
O(1)

$y = \textcircled{K}$ } math

$O(K) \Rightarrow K \cdot O(1)$
 $= \textcircled{O(1)}$

$$\begin{cases} y = x^2 + 2x + 5 \end{cases}$$

highest power = 2

quadratic

$$\Rightarrow O(\underbrace{n^3 + 2n + 5}_{\downarrow O(n^3)})$$

$O(n)$ \rightarrow linear \cdot (Tc)

i/p

5

50

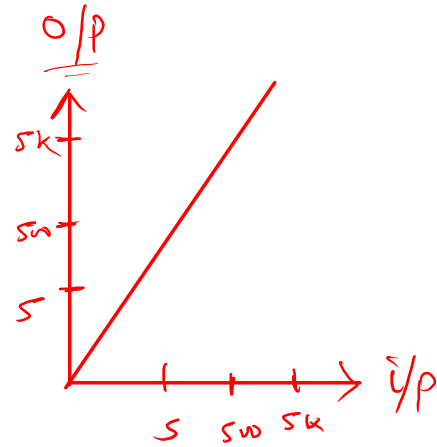
500

o/p

5

50

500



main.java

```
1 import java.util.*;
2 public class Main
3 {
4     public static void main(String[] args) {
5         Scanner scn = new Scanner(System.in);
6         int n = scn.nextInt();
7         for(int i = 0; i < n; i++){
8             System.out.println(i);
9         }
10    }
11 }
12
```

7
52
502

$n + 2$

$n + 2$

$O(n+2)$
 $O(n)$

```
6 public static int search(int [] A, int x){
7     for(int i = 0 ;i < A.length; i++){
8         if(A[i] == x){ h
9             return i;
10        }
11    }
12    return -1;
13 }
```

n

$O(n)$

$$k_1n + k_2$$

$$2x + 3$$

degree = 1

$$TC = \frac{k_1 + n \times k_2}{= O(n \times k_2)}$$

$$= k_2 O(n)$$

$$= O(n)$$

```
1 import java.util.*;
2 public class Main
3 {
4     public static void main(String[] args) {
5         Scanner scn = new Scanner(System.in);
6         int n = scn.nextInt();
7         for(int i = 0; i < n; i++){
8             int x = scn.nextInt();
9             System.out.println(x);
10        }
11    }
12 }
13
```

} k_1

} k_2

```

1 import java.util.*;
2 public class Main
3 {
4     public static void main(String[] args) {
5         Scanner scn = new Scanner(System.in);
6         int n = scn.nextInt();
7         for(int i = 0; i < 10; i++){
8             int x = scn.nextInt();
9             System.out.println(x);
10        }
11    }
12 }
13

```

500

5000

5×10^1

$\times 10^2$

$\times 10^3$

$\times 10^4$


```
1 import java.util.*;
2 public class Main
3 {
4     public static void main(String[] args) {
5         Scanner scn = new Scanner(System.in);
6         int n = scn.nextInt();
7         for(int i = 0; i < n; i++){
8             System.out.println(i);
9         }
10
11         for(int i = 0; i < n; i++){
12             System.out.println(i);
13         }
14     }
15 }
16
```



```

1 import java.io.*;
2 import java.util.*;
3
4 public class Solution {
5
6     public static int search(int [] A, int x){
7         for(int i = 0 ;i < A.length; i++){
8             if(A[i] == x){
9                 return i;
10            }
11        }
12        return -1;
13    }
14    public static void main(String[] args) {
15        Scanner scn = new Scanner(System.in);
16        int n = scn.nextInt();
17        int [] A = new int[n];
18        for(int i = 0; i < n; i++){
19            A[i] = scn.nextInt();
20        }
21        int x = scn.nextInt();        //tar to be matched
22        int ans = search(A, x);
23        if(ans == -1){
24            System.out.println("False");
25        }else{
26            System.out.println("True");
27        }
28    }
29 }

```

$$k_1 + n + n + k_2$$

$$\begin{aligned}
 & k_2 + k_1 + n + n \quad 2n + (k_1 + k_2) \\
 & 2n \\
 & O(n) \quad 2n + k \\
 & \alpha(n)
 \end{aligned}$$