Agenda :

- Sum of Subarray Minimums → Hard
- Trapping Rain Water → Hard
- Minimum Stack → Med

Sum of Subarray Minimums

int[] arr = {3, 4, 2, 5, 6, 7, 1}

ans = $\sum$ minVals

Subarray

{3} 3
{3, 4} 3
{3, 4, 2} 2
{3, 4, 2, 5} 2
{3, 4, 2, 5, 6} 2
{3, 4, 2, 5, 6, 7} 2
{3, 4, 2, 5, 6, 7, 1} 1

{4} 4
{4, 2} 2
{4, 2, 5} 2
{4, 2, 5, 6} 2
{4, 2, 5, 6, 7} 2
{4, 2, 5, 6, 7, 1} 1

{2} 2
{2, 5} 2
{2, 5, 6} 2
{2, 5, 6, 7} 2
{2, 5, 6, 7, 1} 1

{5} 5
{5, 6} 5
{5, 6, 7} 5
{5, 6, 7, 1} 1

{6} 6
{6, 7} 6
{6, 7, 1} 1

{7} 7
{7, 1} 1

{1} 1

Brute Force

- Nested loop : try to get all subarray sum
  - inc. the sum.

TC: $O(N^2)$

SC: $O(1)$

$$\text{arr} = \{\overset{0}{2}, \overset{1}{4}, \overset{2}{5}, \overset{3}{3}, \overset{4}{6}, \overset{5}{7}, \overset{6}{9}, \overset{7}{1}\}$$

nseli i nseri

$$\sum \text{min of Subarray} = 2*a + 4*b + 5*c + 3*d \ldots\ldots$$

No. of subarray ele is min.

$1 + 0\times1 + 1\times2 + 0\times1\times2$
$= 1+2$
$= 3$

6

7

7, 9

3

5

4, 5

6

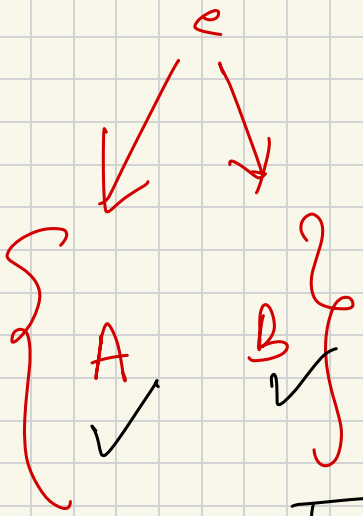
6, 7

6, 7, 9

No of Subarray

$= 1 + 2\times1 + 1\times3 + 2\times1\times3$

$= 1 + 2 + 3 + 6 = 12$

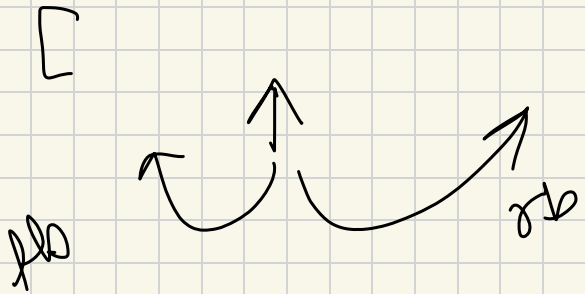


Total Subarray

$$X = 1 + A \times 1 + 1 \times B + A \times 1 \times B$$

Sum 2

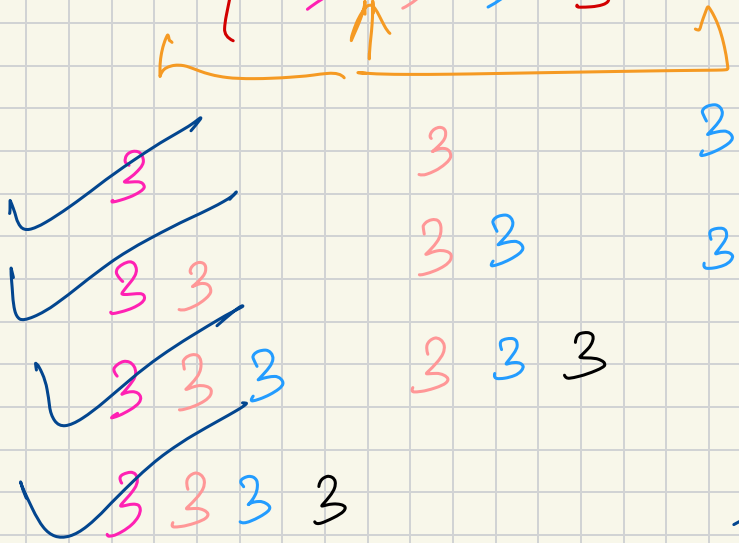
$$\sum_{i=0}^{n} [X_i \times arr[i]]$$



No. of inclusion from left = $i - lb - 1$

from right = $rb - i - 1$

int[] = { 3, 3, 3, 3 }



1 + 1×1 + 1×2 + 1×1×2

= 1 + 1 + 2 + 2

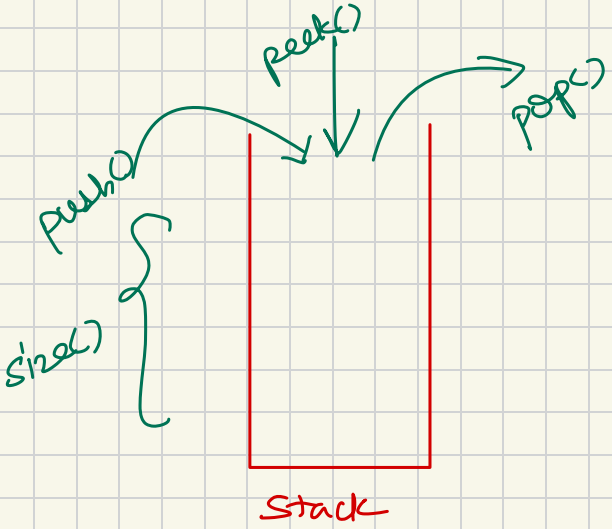
= 6

Total = 1 + 0×1 + 1×3 + 0×1×3

= 4

- Next Smaller and equal element on left.
- Next Smaller element on right.

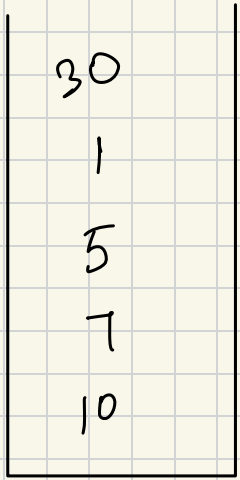
Min stack



getMin() : Method

↓

returns Min value present in the stack in O(1)



Test Case:

push(10)
push(7)
push(5)
push(2)
getMin() ⇝ 2
pop()
getMin() ⇝ 5
push(1)
getMin() ⇝ 1
push(30)
getMin() ⇝ 1

push(10) ✓

push(7) ✓

push(5) ✓

push(2) ✓

getMin() ⇝ 2 ✓

pop() ✓

getMin() ⇝ 5 ✓

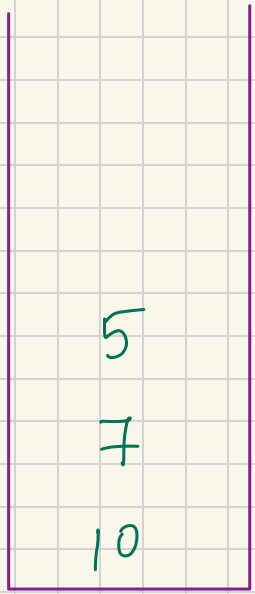
push(1)

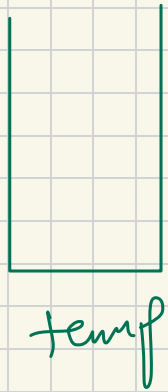
getMin() ⇝ 1

push(30)

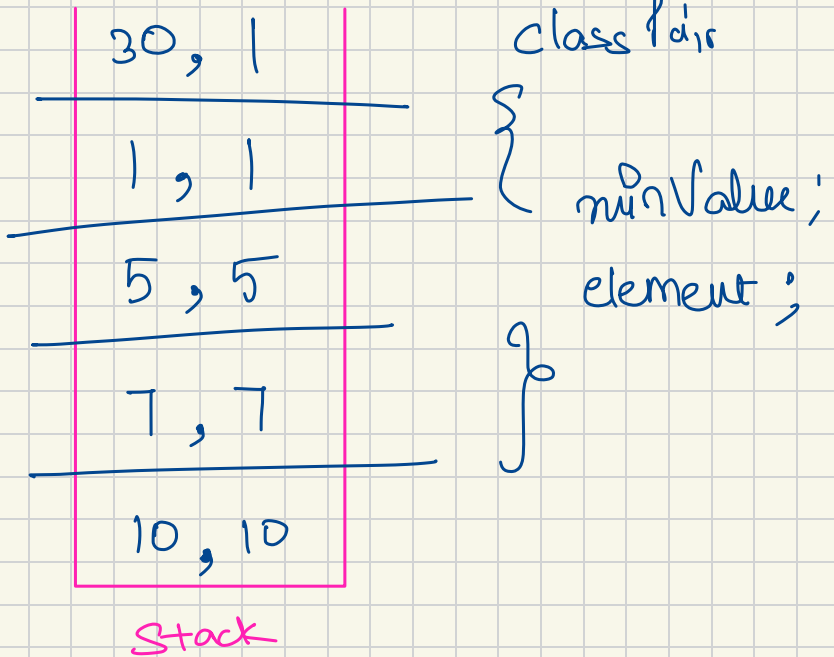
getMin() ⇝ 1

TC: O(N)

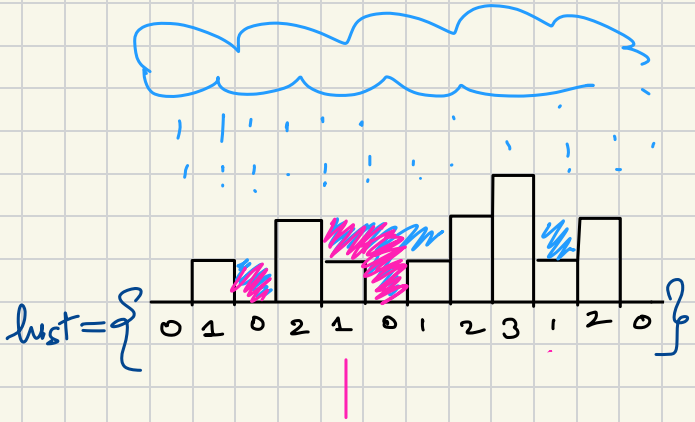




push(10)
push(7)
push(5)
push(2)
getMin() ⇝ 2
pop()
getMin() ⇝ 5
push(1)
getMin() ⇝ 1
push(30)
getMin() ⇝ 1



Trapping Rainwater.



water stored = 6 sq. units

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

list[] = {0, 1, 0, 2, 1, 0, 1, 2, 3, 1, 2, 0}

Brute force

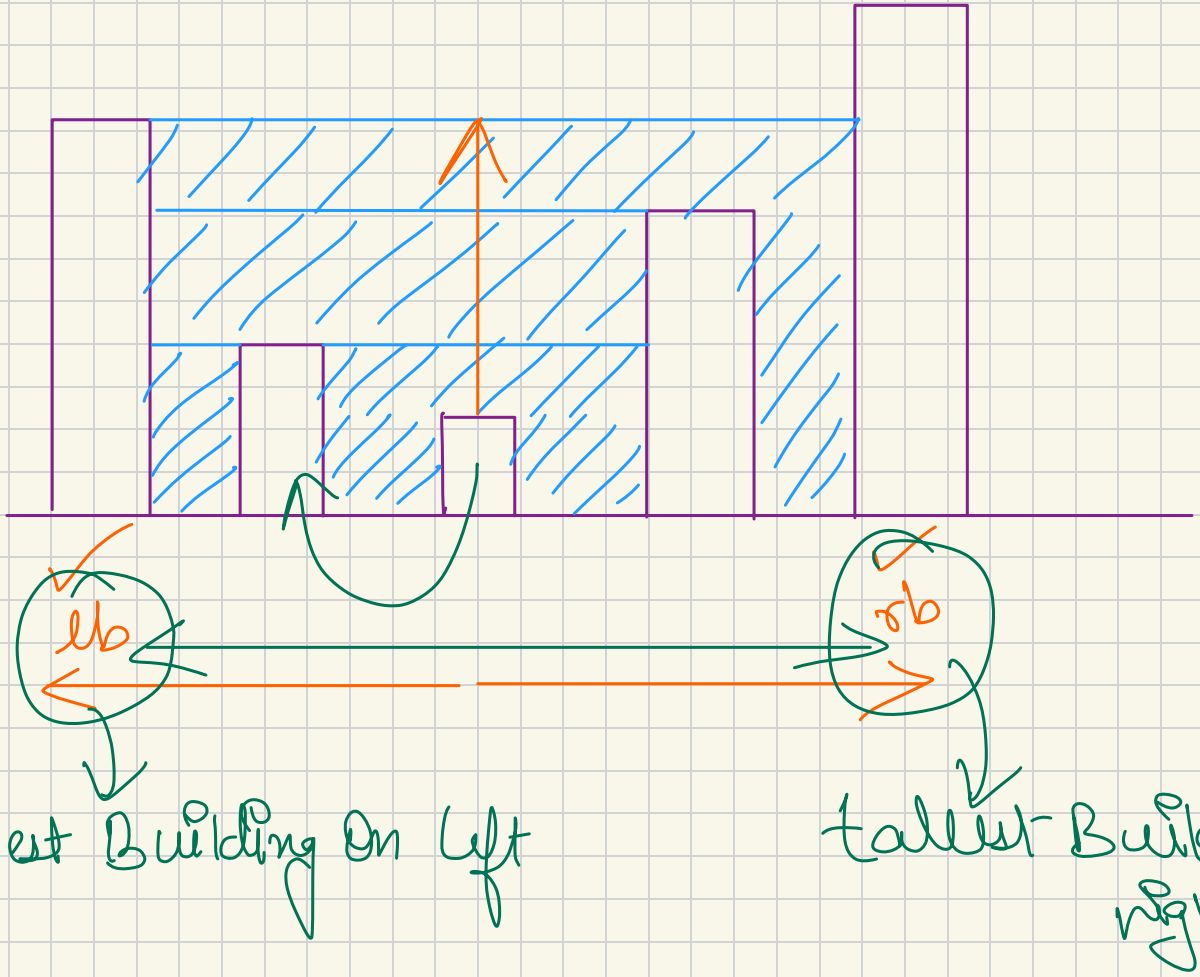
```
for(int i = 0 ⟶ n)
{
    getTallest(0, i-1)
    getTallest(i+1, n)
    Calc. water above you
}
```

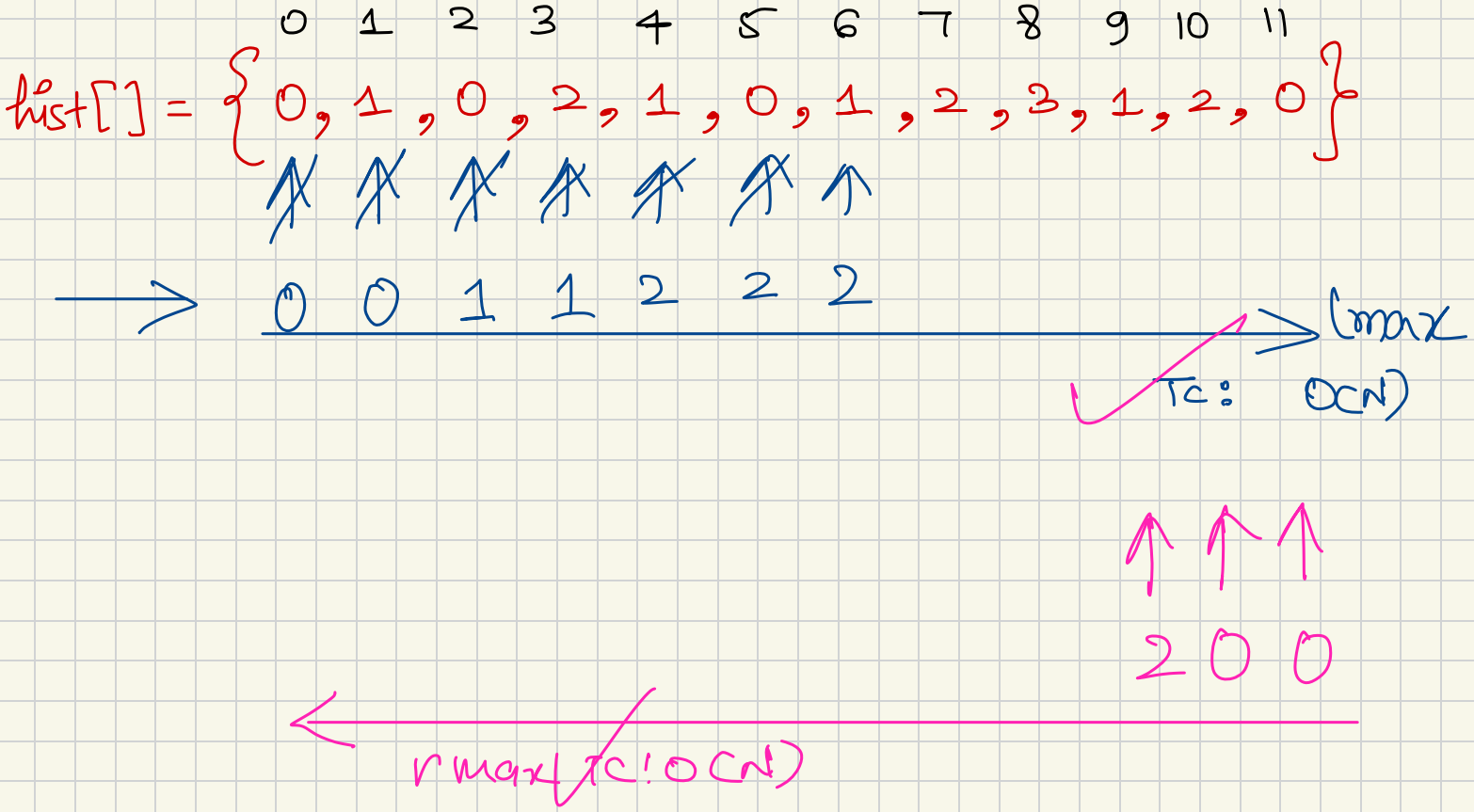
i = 0 : 2N, i = 1 : 2N, ... i = N+1 : 2N

Σ 2N = 2N × N = $2N^2$

TC : $O(N^2)$
SC : $O(1)$

$= O(N^2)$


lb
rb
tallest Building On Left
tallest Building On right


0 1 2 3 4 5 6 7 8 9 10 11
list[] = {0, 1, 0, 2, 1, 0, 1, 2, 3, 1, 2, 0}
0 0 1 1 2 2 2
max
TC: O(N)
2 0 0
rmax(TC: O(N))