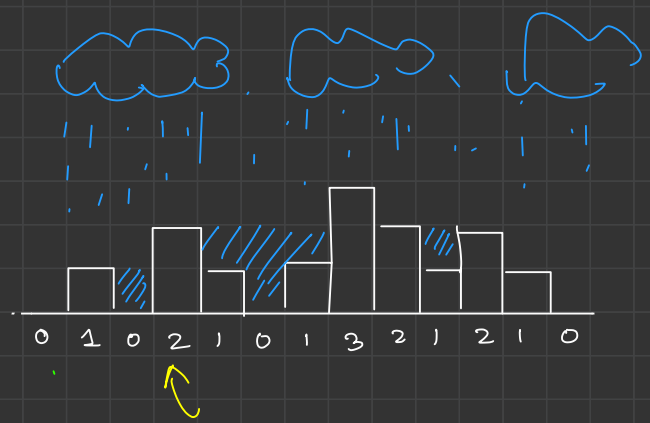
Trapping Rain Water



Brute force

→ go to each building

→ get tallest in left side

→ get tallest in right side

→ Height of water = min(LB, RB) − height of building.

→ Add all

TC : $O(N^2)$

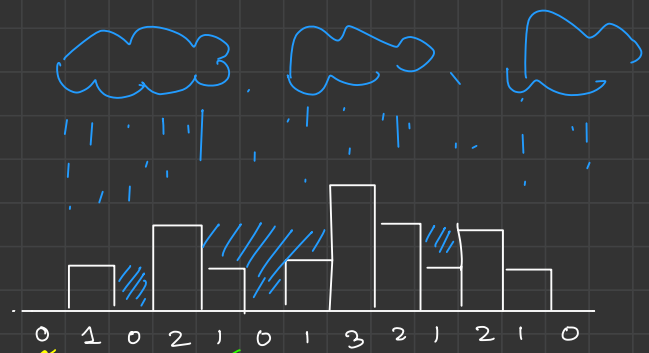
SC : $O(1)$

```
for(i = 0 → n)
{
    for(j = 0 → i-1)
        max LB

    for(j = i+1 → n)
        max RB

    hw = min(LB, RB) - hB;

    sum += hw;
}
```

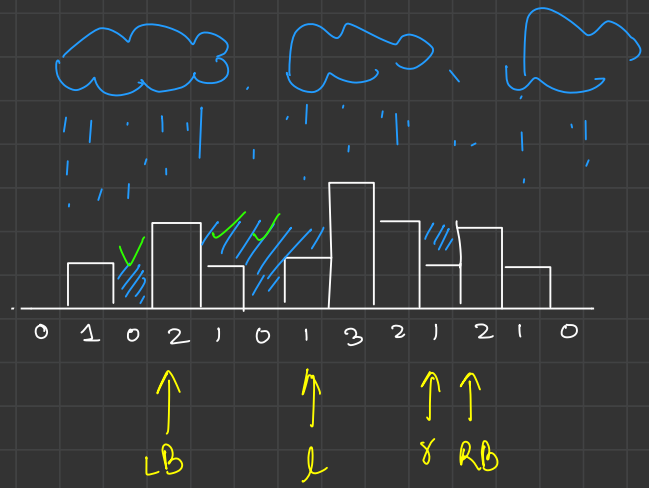


lmax	0	0	1	1	2	2	2	2	3	3	3	3	3
rmax	3	3	3	3	3	3	3	2	2	2	1	0	0

TC: O(N)

SC: O(N)

```
int[] lmax = new int[n];
lmax[0] = 0;
for (i = 1 → n)
{
    lmax[i] = max(arr[i-1], lmax[i-1]);
}
```



TC: O(N)
SC: O(1)

LB <= RB

```
{
    if (arr[l] < LB){
        water += (LB - arr[l]);
    } else
    {   LB = arr[l];
    }
    l++;
}
```

RB < LB

```
{   if (arr[r] < RB){
        water += (RB - arr[r]);
    } else
    {   RB = arr[r];
    }   r--;
}
```

Min Stack

- push
- pop
- peek
- size
- getMin → return minimum in a stack

push(4) ✓

push(1) ✓

push(3) ✓

getMin → 1

push(0) ✓

getMin → 0 ✓

pop() → 0 ✓

getMin → 1

pop() →

pop() →

st

minFree = ∞

```java
void push(int x){
    //Write Code here
    if (st.size() == 0) {
        st.push(minEle);
        st.push(x);
        minEle = x;
    } else {
        if (x <= minEle) {
            st.push(minEle);
            st.push(x);
            minEle = x;
        } else {
            st.push(x);
        }
    }
}

int pop(){
    //Write Code here
    if (st.size() == 0) {
        return -1;
    } else {
        if (st.peek() == minEle) {
            int x = st.pop();
            minEle = st.pop();

            return x;
        } else {
            return st.pop();
        }
    }
}
```

```java
int getMin(){
    //Write Code here
    if (minEle == Integer.MAX_VALUE) {
        return -1;
    } else {
        return minEle;
    }
}
```

push(4)
push(5)
push(20)
push(4)

4
4
20
5
4
∞

minEle = ~~∞~~ ~~4~~ 4

Sum of Subarray Minimums

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

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

3 2 2 1 1

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

{2} {2, 4} {2, 4, 1} {2, 4, 1, 5} {2, 4, 1, 5, 2}

2 2 1 1 1

{4} {4, 1} {4, 1, 5} {4, 1, 5, 2}

4 1 1 1

{1} {1, 5} {1, 5, 2}

1 1 1

{5} {5, 2} {2}

5 2 2

ans = 36

Brute force

TC: $O(N^2)$ → calc. min of all subarray and add them

SC: $O(1)$

0	1	2	3	4	5	6	7

arr[] = { -2, 3, -1, 4, 2, 3, 7, -10 }

{2} → {2}, {4,3}, {2,3,7}

{4,2} → {4,3}, {2,3,7}

$\boxed{2 \times 3} = 1$

{2}

{2,3}

{2,3,7}

{4,2}

{4,2,3}

{4,2,3,7}

	0	1	2	3	4	5
arr[] =	3,	2,	4,	1,	5,	2
nseli[]	-1,	-1,	1,	-1,	3,	3 ✓
nseri[]	1,	3,	3,	6,	5,	6 ✓

TC: O(N) SC: O(N)

$3 - (-1) = 4$

$6 - 3 = 3$

12

no. of left subarray = idx − nseli

or right = nseri − idx

$\sum_{i}^{i}$ total × val