Arrays 1

Content

- Find maximum subarray sum.
- Perform multiple queries.
- Rain water trapping.

Rules ⟶
1. Private chat for answers
2. Use question
3. Be patient.

Maximum subarray Sum

continuous part of Arr

Given int[] A. Find max subarray sum of all subarrays

	0	1	2	3	4	5	6	
A =	-2	3	4	-1	5	-10	7	ans = 11

	0	1	2	3	4	5	6	
A =	-3	4	6	8	-10	2	7	ans = 18

Quiz

A = 4 5 2 1 6 ans = 18

Observation — $\forall_i$ $A_i > 0$ entire array sum = ans.

A = -4 -3 -6 -9 -2

Observation — $\forall_i$ $A_i < 0$ pick me largest no.

Bruteforce

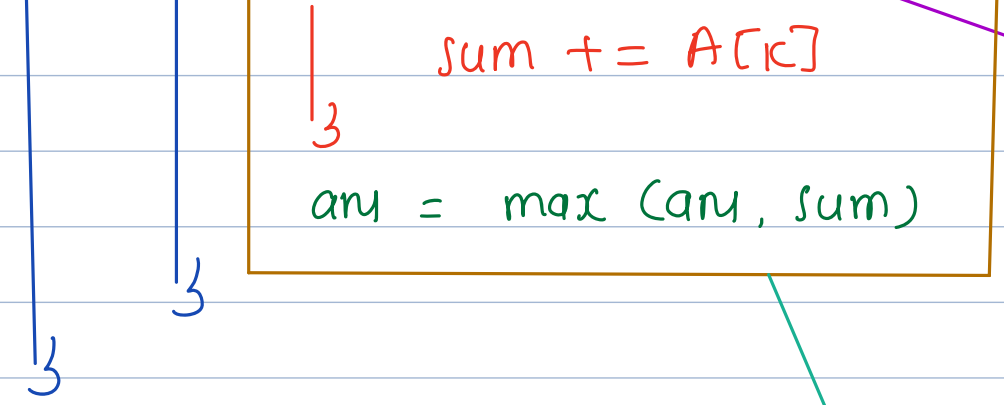
Calculate sum of all subarrays and check max

```
ans = -∞   // INT_MIN

for i ⟶ 0 to N-1 {          // start index of sub
    for j ⟶ i to N-1 {      // end index of sub
        sum = 0
        for k ⟶ i to j {
```

N

N



TC : $O(N^3)$

SC : $O(1)$

Can be optimised via prefix sum.

TC : $O(N^2)$

SC : $O(N)$

Carry forward ⟶ calculate & use immediately.

```
ans = -∞
for i ⟶ 0 to N-1 {
    sum = 0  // carry forward
    for j ⟶ i to N-1 {
        sum += A[j]
        ans = max(ans, sum)
    }
}
print(ans)
```

i: -1 10 -2

j:

sum = ∅ -2

ans = ~~-∞~~ ~~-1~~ 9

10

TC : $O(N^2)$

SC : $O(1)$

Kadane's

Case 1> all no. are +ve

A = 1 2 3

+ve +ve +ve

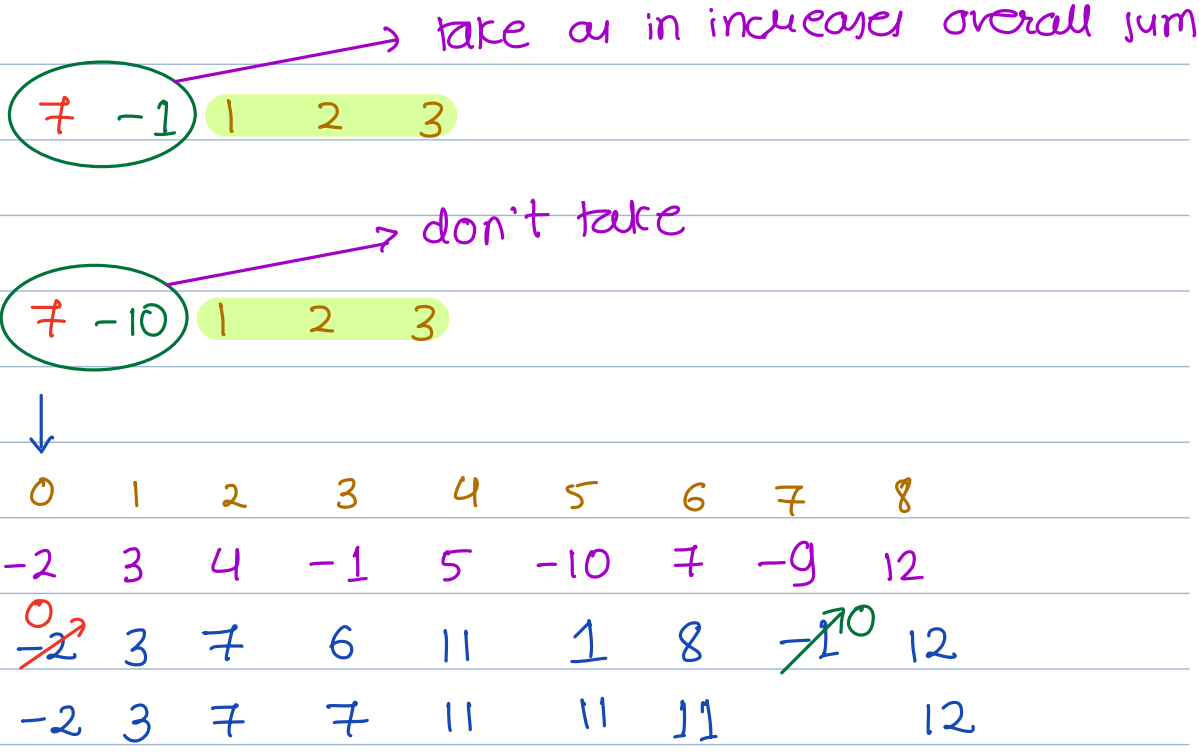
Case 2> all no. are -ve

A = -1 -2 -3

pick the largest

Case 3> -ve -ve -ve +ve +ve +ve -ve -ve -ve

Case 4> Generic case



Pseudocode

TC : O(N)

SC : O(1)

```
ans = -∞
sum = 0
for i ⟶ 0 to N-1 {
    sum += A[i]
    ans = max(ans, sum)
    if (sum < 0) {
        sum = 0
    }
}
return ans
```

	0	1	2	3
A =	-3	-5	-2	-6
sum = 0	~~-3~~ 0	~~-5~~ 0	~~-2~~ 0	~~-6~~ 0
ans = ~~-∞~~	-3	-3	-2	-2

		0	1	2	3	4	5	6	7	8
A =		6	-3	9	-15	3	12	0	-24	3
sum	0	6	3	12	~~-3~~ 0	3	15	15	~~-9~~ 0	3
ans	-∞	6	6	12	12	12	15	15	15	15

-2 3 4 -1 5 -10 7

ans = $-\infty$

sum = 0 , start → 0 , L , R → final ans

```
for i ⟶ 0 to N-1 {
    sum += A[i]
    if (sum > ans) {
        ans = sum
        L = start
        R = i
    }
    if (sum < 0) {
        sum = 0
        start = i+1
    }
}
```

// Print the array b/w L to R

Perform multiple queries from i to last index

Given an int[] A where every element is 0
Return the final array after performing multiple queries

Query (i, x) ⟶ Add x to all no. from i to N-1

	0	1	2	3	4	5	6
A[]	0	0	0	0	0	0	0
i x							
Query (1, 3)		+3	+3	+3	+3	+3	+3
Query (4, 2)					+2	+2	+2
Query (3, 1)				+1	+1	+1	+1
	0	3	3	4	6	6	6

Bruteforce ⟶

For every query
Add x to all no. from i to N-1

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

	0	1	2	3	4	5	6
A[]	0	0	0	0	0	0	0
i x							
Query (1, 3)		+3					
Query (4, 2)					+2		
Query (3, 1)				+1			
	0	3	0	1	2	0	0

Prefix sum → 0 3 3 4 6 6 6

Pseudocode

Queries [ ] [ ] { i , x } { 0 , 1 }

```
for q ⟶ 0 to Q-1 {
    i = Queries[q][0]
    x = Queries[q][1]

    A[i] += x
}
```

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

```
for ( i ⟶ 1 to N-1 ) {
    A[i] = A[i] + A[i-1]
}

print (A)
```

Perform multiple queries from i to j index

Given an int[] A where every element is 0
Return the final array after performing multiple queries

Query (i, j, x) ⟶ Add x to all no. from i to j

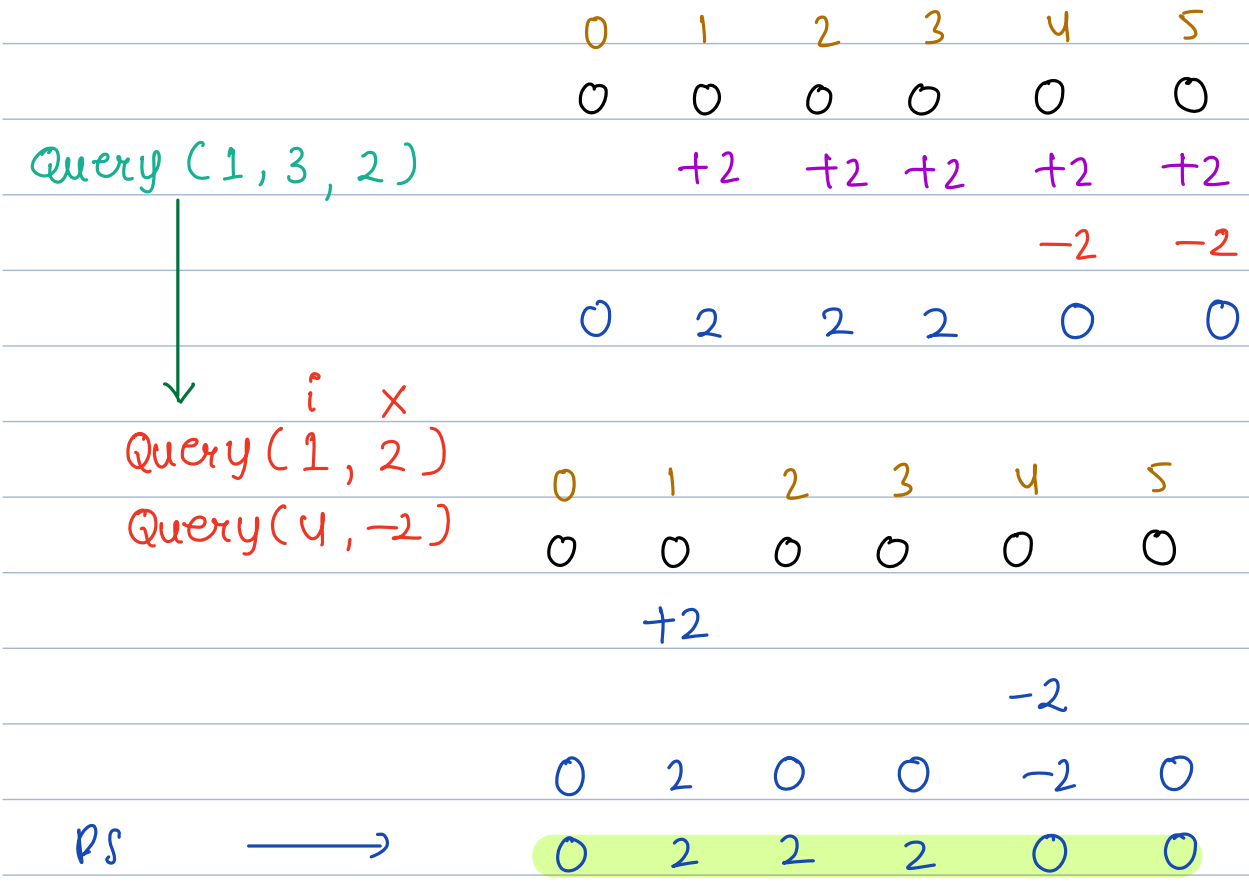
	0	1	2	3	4	5	6
A[]	0	0	0	0	0	0	0
i j x							
Query (1, 3, 2)		+2	+2	+2			
Query (2, 5, 3)			+3	+3	+3	+3	
Query (5, 6, -1)						-1	-1
	0	2	5	5	3	2	-1

	0	1	2	3	4	5	6	7
	0	0	0	0	0	0	0	0
i j x								
1 4 3		3	3	3	3			
0 5 -1	-1	-1	-1	-1	-1			
2 2 4			4					
4 6 3						3	3	3
	-1	2	6	2	5	3	3	0

Bruteforce

For each query ⟶
Add x from index i to j

TC : O(QN)
SC : O(1)



Pseudocode Queries [ ][ ] {i, j, x}

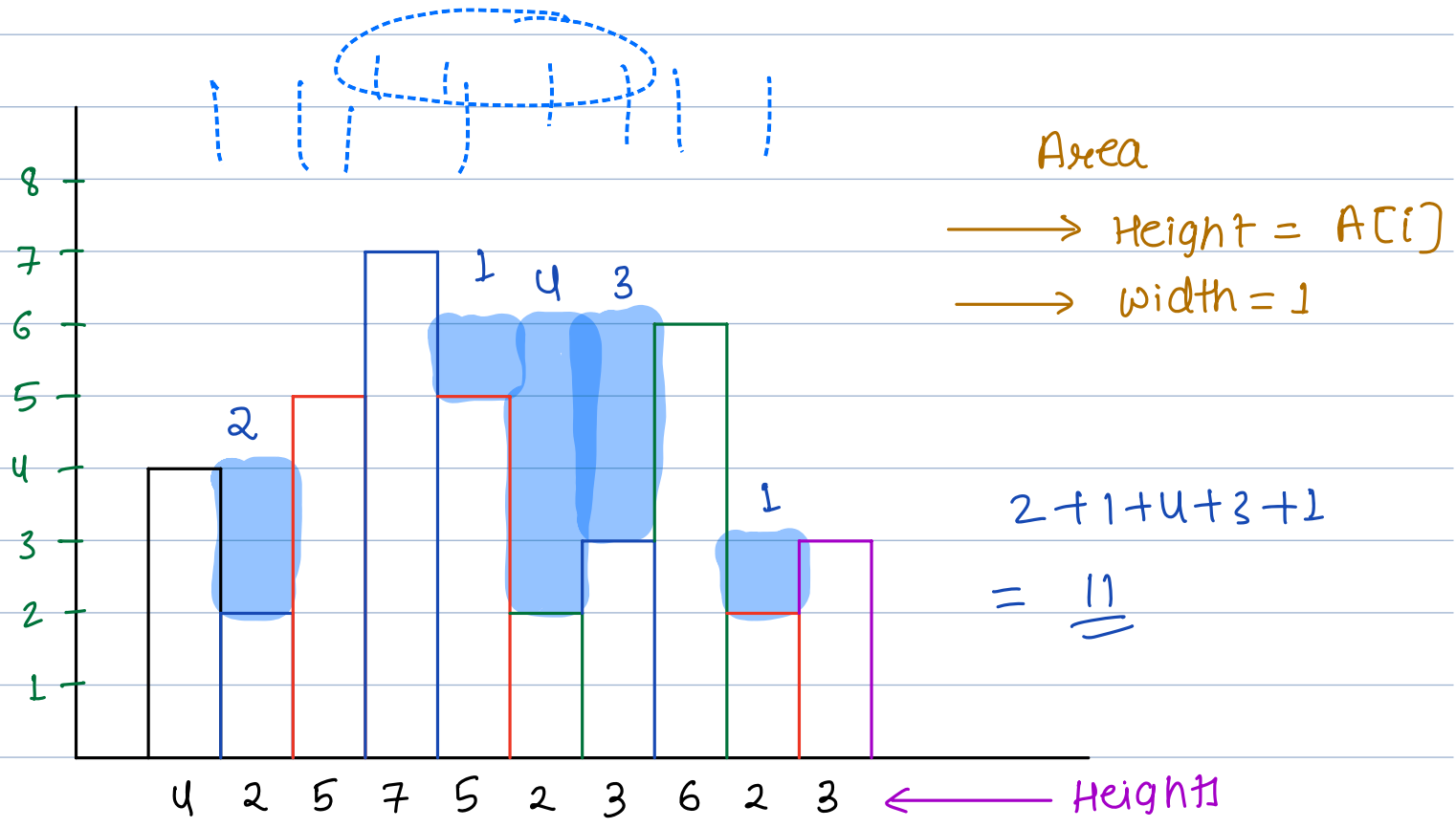
```
for q ⟶ 0 to Q-1 {
    i = Queries[q][0]
    j = Queries[q][1]
    x = Queries[q][2]
    A[i] += x
    if(j+1 < N) A[j+1] -= x
}
```

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

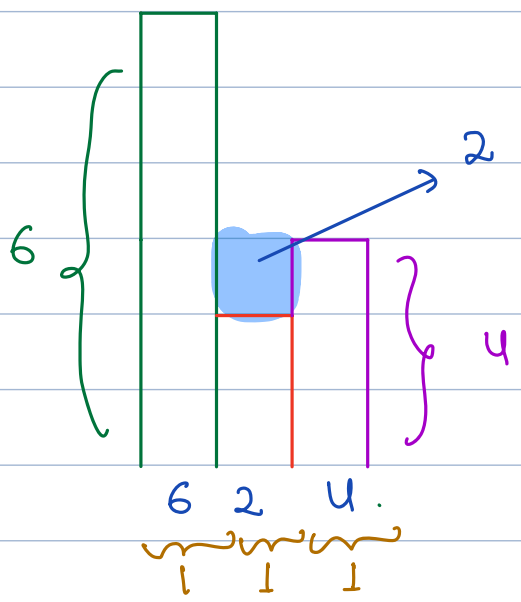
Break
22:38

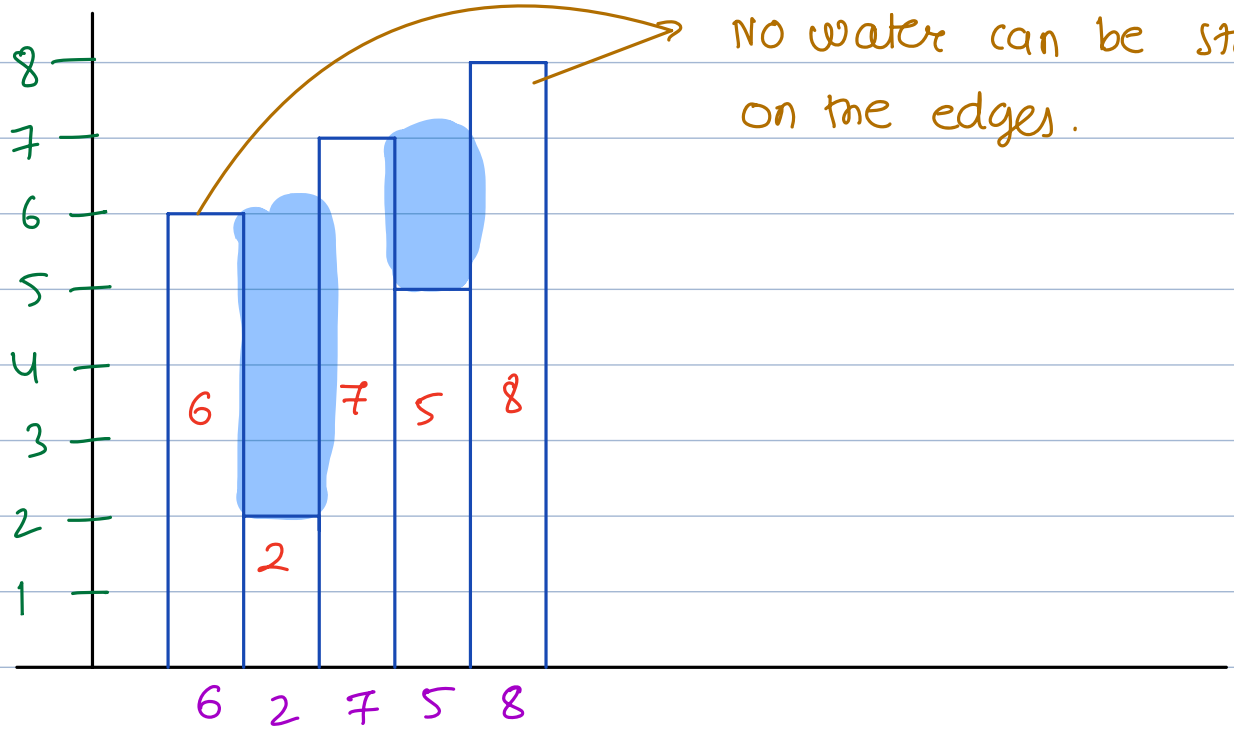
```
for(i ⟶ 1 to N-1) {
    A[i] = A[i] + A[i-1]
}
print(A)
```

Q> Rain Water Trapping



Given A[N] each value represents building height calculate the total water accumulated co2 of rain



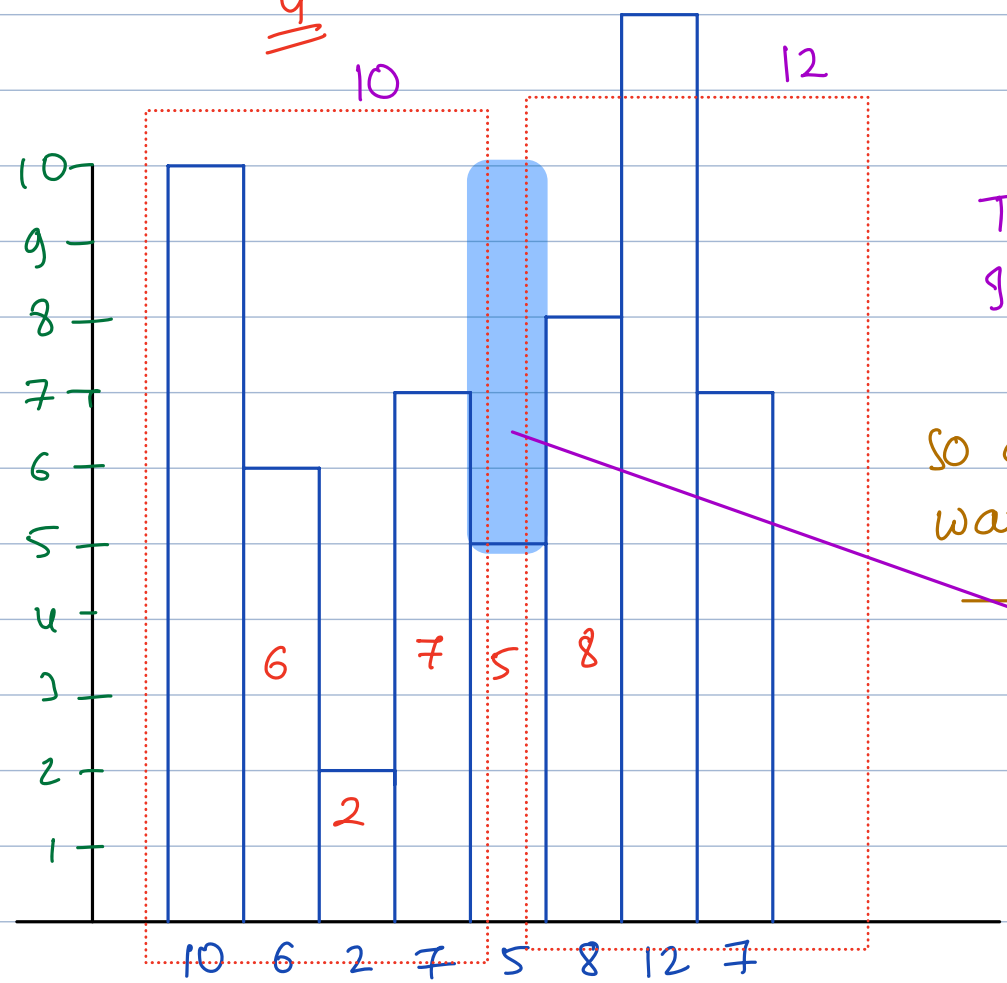


No water can be stored on the edges.

6 8

6 − 2

4



Till height 10 I can fill the water

So above building 5 water stored

$10 - 5 = 5$

Bruteforce

water = 0

TC $O(N^2)$

SC $O(1)$

```
for i ⟶ 0 to N-1 {

    leftMax = -∞   // 0 will work too.
    for j ⟶ i to 0 {
        leftMax = max(leftMax, A[j])
    }

    rightMax = -∞   // 0 will work too.
    for j ⟶ i to N-1 {
        rightMax = max(rightMax, A[j])
    }

    waterHeight = min(leftMax, rightMax)
    area = waterHeight - A[i]
    water += area

}
print(water)
```

	↓		
	6	2	4
leftMax	6	6	6
rightMax	6	4	4
waterHeight	6	4	4
area	6-6	2	4-4
water	0	2	2

prefix R → L = Suffix

Pseudocode

```
prefixMax = new int[N]
leftMax = -∞
for i ⟶ 0 to N-1 {
    leftMax = max(leftMax, A[i])
    prefixMax[i] = leftMax
}
```

```
suffixMax = new int[N]
rightMax = -∞
for i ⟶ N-1 to 0 {
    rightMax = max(rightMax, A[i])
    suffixMax[i] = rightMax
}
```

```
water = 0
```

```
for i ⟶ 0 to N-1 {

    leftMax = prefixMax[i]
    rightMax = suffixMax[i]
    waterHeight = min(leftMax, rightMax)
    area = waterHeight - A[i]
    water += area

}
```