SCALER

Stacks - 2

TABLE OF CONTENTS

Notes

Nearest Smaller Element

Q1 → Given an integer array find the index of nearest smaller element on left of i $\forall A[i]$.

	0	1	2	3	4	5	6	7
A =	8	2	4	9	7	5	3	10
	-1	-1	1	2	2	2	1	6

$\forall i$, find max j s.t $A[j] < A[i]$ & $j < i$.

	0	1	2	3	4	5	6	7
A =	4	6	10	11	7	8	3	5
	-1	0	1	2	1	4	-1	6

	0	1	2	3	4	5
A =	4	5	2	10	8	2
	-1	0	-1	2	2	-1

Bruteforce → TC = $O(N^2)$ SC = $O(1)$

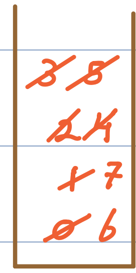
	0	1	2	3	4	5	6	7
A =	8	_	_	_	_	5	_	_

For any $i > 5$, can index 0 be the ans ?

No, $\forall i > 5$, if $8 < A[i] \Rightarrow 5 < A[i]$

& closer index will be 5.

	0	1	2	3	4	5	6	7
A =	4	6	10	11	7	8	3	5
	-1	0	1	2	1	4	-1	6



checking → latest index first ⇒ LIFO ∴ use <u>stack</u>.

```
// stack st
for i → 0 to (N-1) {
    while (!st.isEmpty() && A[st.peek()] >= A[i]) {
        st.pop()
    }
    if (st.isEmpty())   ans[i] = -1
    else    ans[i] = st.peek()
    st.push(i)
}
```

> <= <

TC = <u>O(N)</u>   SC = <u>O(N)</u>

Q2 → Find nearest smaller or equal on left.

Q3 → Find nearest greater element on left.

Q4 → Find nearest greater or equal on left.

Q5 → Find nearest smaller element on right.

```
// stack st
for i → (N-1) to 0 {
    while (!st.isEmpty() && A[st.peek()] >= A[i]) {
        st.pop()
    }
    if (st.isEmpty())   ans[i] = -1
    else    ans[i] = st.peek()
    st.push(i)
}
```

>   <=   <

TC = O(N)   SC = O(N)

Q6 → Find nearest smaller or equal on right.

Q7 → Find nearest greater element on right.

Q8 → Find nearest greater or equal on right.

- **A person uses Google Maps to find the nearest restaurants and picks one based on it's proximity. Unfortunately, after visiting, they realised that the restaurant didn't meet their expectations.**

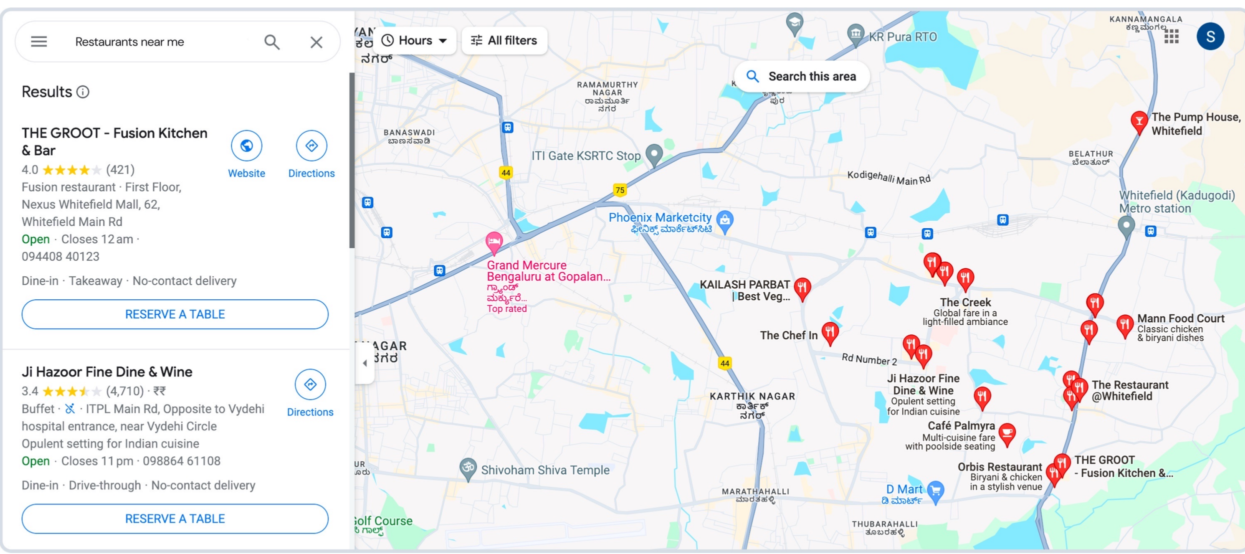
Task

Let's break it down with a simple example. You have a list of restaurants and their ratings. For each restaurant, we're going to find the next restaurant to the right on the list that's not just close but also has a higher rating than the current one. If there's no better option on the list, we'll say there's none available.

Problem

Given a sequence of restaurants listed on Google Maps with their ratings, create a tool that helps users discover the rating of the next higher-rated restaurant to the right for each listed establishment.

Nearest greater on right.

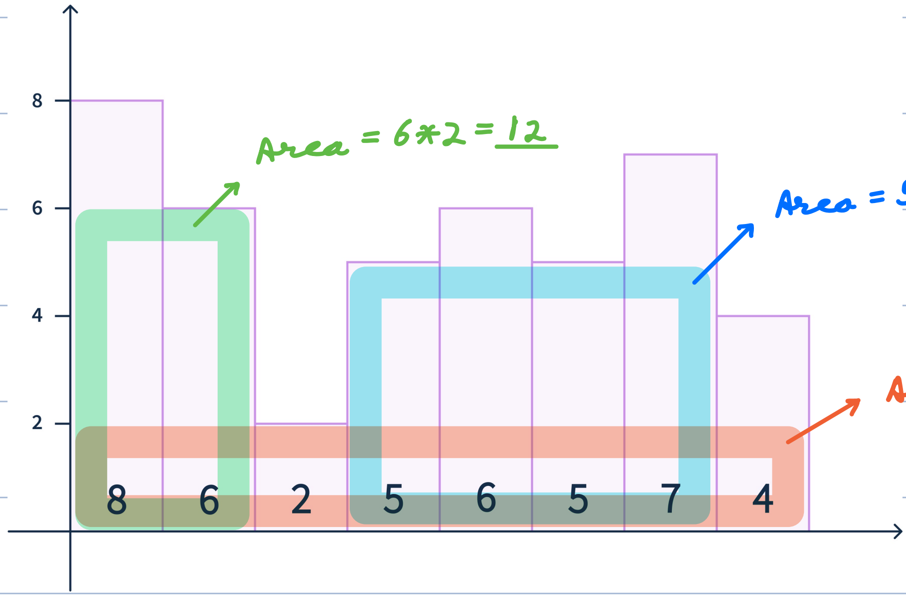


< Question > : Find the largest area of rectangle (formed by continuous bars) in histogram.

H * W

A[i] → Height of i^{th} bar

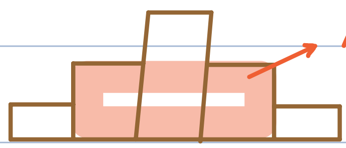
Width = 1 ∀ bars



Area = 5 * 4 = 20 ✓

Area = 2 * 8 = 16

	0	1	2	3	4
A =	[1	2	3	2	1]



Area = 2 * 3 = 6 ✓

Idea → Find continuous bars that give largest rectangle.

Bruteforce → ∀ subarray , calculate

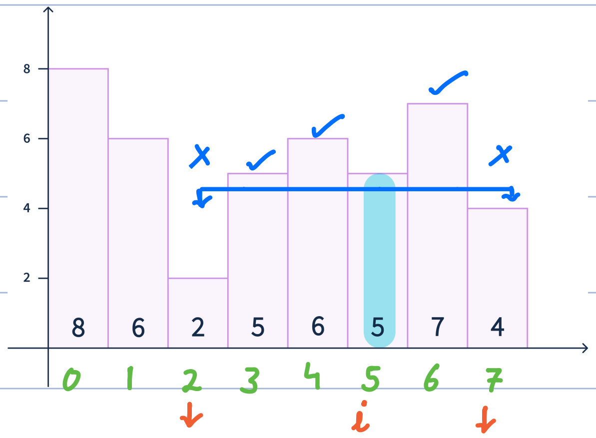
area = (min A[i]) * length of subarray.

carry forward   l — r ⇒ r - l + 1

TC = $O(N^3) \longrightarrow O(N^2)$ SC = $O(1)$

TC < $\underline{O(N^2)}$ ⇒ we cannot travel all subarrays i.e width.

<u>Idea 2</u> → ∀ height find max width.



$[j+1 \quad k-1]$

$\text{width} = k - \cancel{1} - (j+1) + \cancel{1}$

$= \underline{k - j - 1}$

Nearest smaller on left = j

Nearest smaller on right = k

$$\boxed{\text{Ans} = \max_{\forall i} \left(A[i] * (\text{nearestSmallerRight}[i] - \text{nearestSmallerLeft}[i] - 1) \right)}$$

$TC = O(N + N + N) = \underline{O(N)}$

$SC = \underline{O(N)}$

< Question > : Find sum of (max-min) for all subarrays. $1 \le N \le 10^5$

0 1 2
arr : [1 2 3]

	max	min	max-min
[1]	1	1	0
[1 , 2]	2	1	1
[1 , 2 , 3]	3	1	2
[2]	2	2	0
[2 , 3]	3	2	1
[3]	3	3	0
			4 (Ans)

Bruteforce → TC = O(N³) ⟶ O(N²) (carry forward)

SC = O(1)

0 1 2
A = [2 5 3]

2 ⟶ 2 − 2 = 0
2 5 ⟶ 5 − 2 = 3
2 5 3 → 5 − 2 = 3
5 ⟶ 5 − 5 = 0
5 3 ⟶ 5 − 3 = 2
3 ⟶ 3 − 3 = 0
8 (Ans)

2∗(1−3) = −4
5∗(4−1) = 15
3∗(1−2) = −3
8

3 7 8 9 11
✗ ✗ ✓ ✗ ✗

Contribution Technique

$Ans = \sum_{\forall i}$ contribution of $A[i]$

$A[i] * \left(\underbrace{\#\text{ subarrays } A[i] \text{ is max}} - \#\text{ subarrays } A[i] \text{ is min} \right)$

< Question > : In how many subarrays, ith element is the maximum element?

arr[ ] : [1 8 3 5 4 2 11 7 2]

1	8	3	5	4	2	11	7	2
0	1	2	3	4	5	6	7	8
	j		i			k		

j: Nearest greater on left

k: Nearest greater on right

[3 5] [5]

[3 5 4] [5 4]

[3 5 4 2] [5 4 2] 6 ✓

subarrays → (# start) * (# end) = $\underline{(i-j) * (k-i)}$

$[j+1 \quad i]$ $\quad [i \quad k-1]$

$i-(j+1)+1 = \underline{i-j}$ $\qquad k-1-i+1 = \underline{k-i}$

subarrays A[i] is min ?

j → Nearest smaller on left

k → Nearest smaller on right

$$\boxed{\begin{aligned} Ans = \sum_{\forall i} A[i] * (&(i - nearestGreaterLeft[i]) * \\ &(nearestGreaterRight[i] - i) - \\ &(i - nearestSmallerLeft[i]) * \\ &(nearestSmallerRight[i] - i)) \end{aligned}}$$

$TC = O(N + N + N + N + N) = \underline{O(N)}$ $SC = \underline{O(N)}$

H.W → How to solve if duplicates are present.