Nearest Smaller Element

Given an integer array A.

$\forall i \Rightarrow$ find the index of the nearest smaller element on the left of i.

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

O/P $\Rightarrow$ [-1, -1, 1, 2, 2, 2, 1, 6] (index)

Soln 1) Brute force

$\forall i \Rightarrow$ iterate from (i-1) to 0 & find the first smaller element than A[i].

Code

NSE[N] =

```
NSE[0] = -1;

for (i = 1; i < N; i++) {
        ans = -1;
        for (j = i-1; j >= 0; j--) {

            if (A[j] < A[i]) {
                ans = j;
                break;
            }
        }
        NSE[i] = ans;
}
```

T.C. = $O(N^2)$

S.C. = $O(1)$

Quiz $A = [\underset{0}{8}, \underset{1}{x}, \underset{2}{x}, \underset{3}{x}, \underset{4}{x}, \underset{5}{5}, \underset{6}{x}, \underset{7}{x}, \underset{8}{x}, \underset{9}{x} \ldots]$

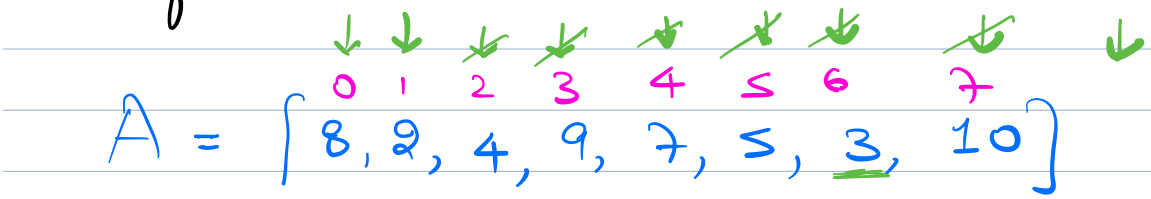
x at index 1 $\rightarrow$ 10, x at index 2 $\rightarrow$ 9

x at index > 5 $\rightarrow$ NSE[i] == 0

NO

2) Optimised Solution

⇒ Store all potential answers for future elements.



NSE ⇒ [-1, -1, 1, 2, 2, 2, 1, 6]

Potential answers = ~~8~~, 2(1), ~~4(2)~~, 9(3), ~~7(4)~~, ~~5(5)~~, 3(6), 10(7)

insert insert

Remove Remove Remove Remove

⇓

Which DS ?? → Stack

Code

```
NSE[N]
NSE[0] = -1;
Stack<int> st;          // Potential answers
st.push(0);

for(i=1; i<N; i++){

    while(!st.isEmpty() && A[i]<=A[st.peek()]){
        st.pop();
```

```
    }

    if (st.isEmpty()){
        NSE[i] = -1;
    }else{
        NSE[i] = st.peek();
    }
    st.push(i);
}
return NSE;
```

$$T.C. = O(N)$$

$$S.C. = O(N)$$

Variations of NSE on left $\forall i$

1) $\forall i \Rightarrow$ find the nearest smaller or equal element to the left of i.

```
while(!st.isEmpty() && A[i] < A[st.peek()]){
```

2) $\forall i \Rightarrow$ find the nearest greater element to the left of i.

```
while(!st.isEmpty() && A[i] >= A[st.peek()]){
```

3) $\forall i \Rightarrow$ find the nearest greater or equal element to the left of i.

```
while(!st.isEmpty() && A[i] > A[st.peek()]){
```

4) $\forall i \Rightarrow$ find nearest smaller element on the right of i.

	0	1	2	3	4	5	6	7
A =	8	2	4	9	7	5	3	10
NSE[] =	2	-1	3	7	5	3	-1	-1
PSE =	~~10~~	~~2~~	~~2~~	~~7~~	~~7~~	~~5~~	2	8

```
NSE[N]
NSE[N-1] = -1;
Stack<int> st;        // Potential answers
st.push(N-1);

for(i = N-2; i >= 0; i--){
        while(!st.isEmpty() && A[i] <= A[st.peek()]){
                st.pop();
        }

        if (st.isEmpty()){
                NSE[i] = -1;
        } else {
                NSE[i] = st.peek();
        }
        st.push(i);
}
return NSE;
```

5) Nearest smaller or equal in right

6) Nearest greater in right

7) Nearest greater or equal in right.

Largest Rectangle in Histogram

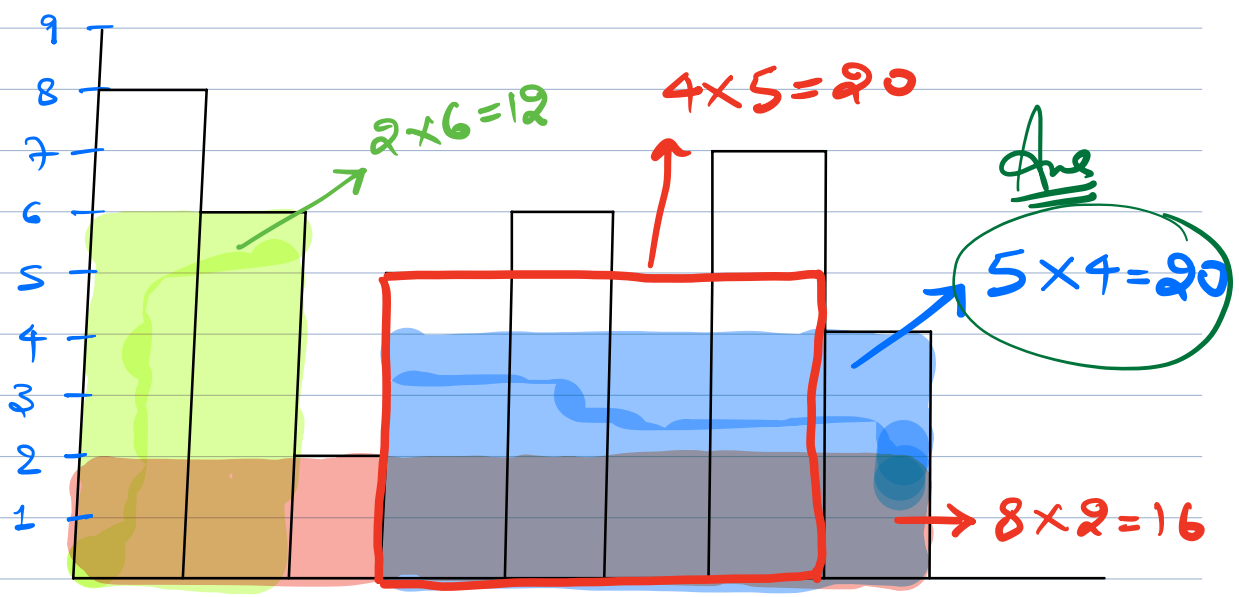
Q. Given an integer array A where

A[i] ⇒ Height of the i^{th} bar in the Histogram

Width of each bar is 1

Find the area of the largest rectangle formed by continuous bars.

$A = [8, 6, 8, 5, 6, 5, 7, 4]$



8 6 2 5 6 5 7 4

Sol^n 1) Brute force

∀ subarrays from i to j ⇒ calculate area & find max

⇓

minHeight × (j − i + 1)

Code

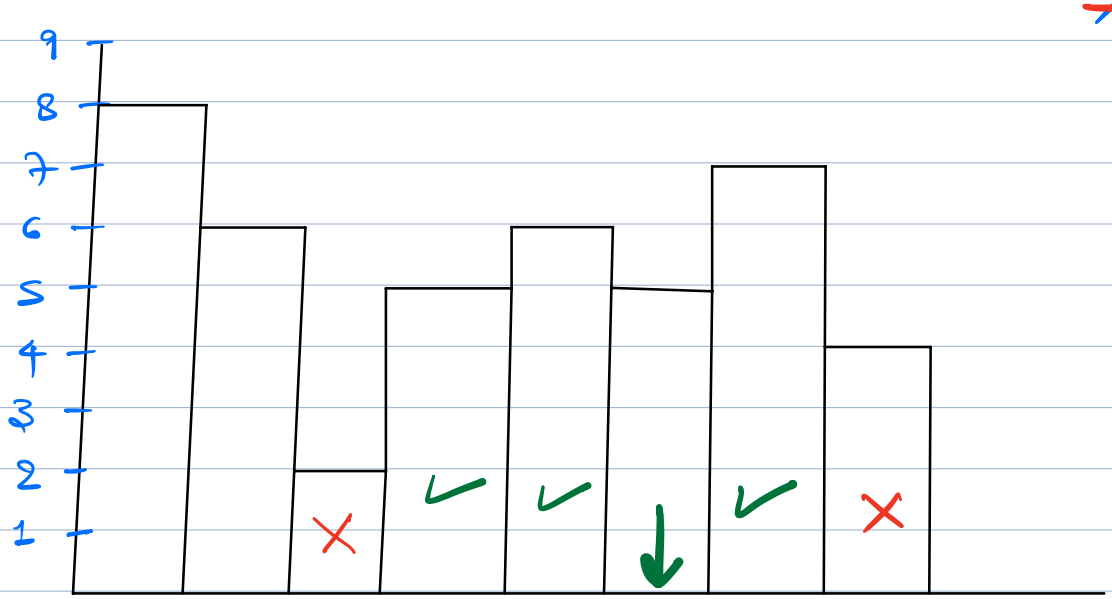
```
ans = 0;
for (i = 0; i < N; i++) {
    minHeight = INT_MAX;
    for (j = i; j < N; j++) {
        minHeight = min(minHeight, A[j]);
        area = minHeight * (j - i + 1);
        ans = max(ans, area);
    }
}
return ans;
```

T.C. = $O(N^2)$

S.C. = $O(1)$

2) Optimised

Area = Width × Height



(Nearest Smaller Element on Left)

NSEL

$\forall i \Rightarrow$ Left = (NSEL + 1)

Right = (NSER − 1)

⇓

Steps

1) Calcate Neerest smeller element on Left $\forall i$

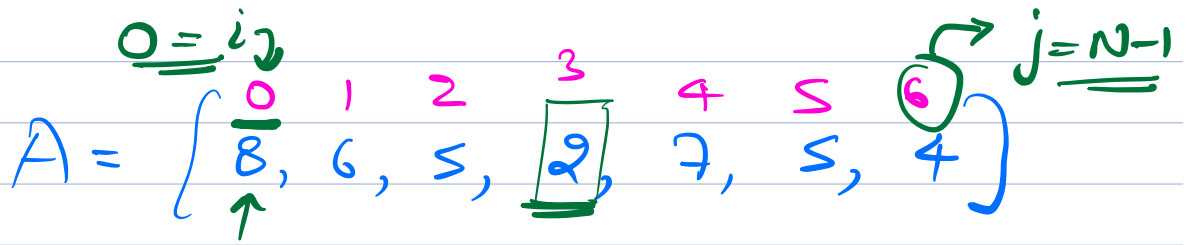
2) Calcate Neerest smeller element on Right $\forall i$

3) $\forall i \Rightarrow$ Largest Rectayle with Height A[i] $\Rightarrow$

i (Left Index) = NSEL[i] + 1

j (Right Index) = NSER[i] − 1

Area = (A[i]) × (j − i + 1)



NSEL -1

NSER -1

Code

```
NSEL [N]
NSEL[0] = -1;
Stack <int> st;        // Potential answers
st.push(0);

for(i=1; i<N; i++){

    while(!st.isEmpty() && A[i]<=A[st.peek()]){
        st.pop();
    }

    if (st.isEmpty()){
        NSEL[i] = -1;
    } else {
        NSEL[i] = st.peek();
    }
```

```
        st.push(i);
}

NSER[N]
NSER[N-1] = -1;
Stack<int> st;          // Potential answers
st.push(N-1);

for(i= N-2; i>=0; i--){
        while(!st.isEmpty() && A[i]<=A[st.peek()]){
                st.pop();
        }
        if (st.isEmpty()){
                NSER[i]= -1;
        }else{
                NSER[i]= st.peek();
        }
        st.push(i);
}

ans = 0;

for (i=0; i<N; i++){
        if (NSEL[i]==-1){
                s= 0;
```

```
} else {
        s = NSEL[i] + 1;
}

if (NSER[i] == -1) {
        e = N-1;
} else {
        e = NSER[i] - 1;
}

area = A[i] x (e-s+1);

ans = max(ans, area);
}
return ans;
```

T.C. = O(N)

S.C. = O(N)

H.W. → S.C. = O(1)

Given an integer array of size N.

∀subarrays → find (mex − min)

&

return the sum of all

$$A = \begin{matrix} 0 & 1 & 2 \\ [2, & 5, & 3] \end{matrix}$$

s	e	Max	Min	Max − Min
0	0	2	2	0
0	1	5	2	3
0	2	5	2	3
1	1	5	5	0
1	2	5	3	2
2	2	3	3	0
		⇓	⇓	Add 8 Ans

Sum of max of all subs − Sum of min of all subs

Soln) Brute force

⇒ ∀ subarrays → find min & max
find max − min & add it.

Code

```
Sum = 0;
for (i=0; i<N; i++) {
    maxEle = INT_MIN;
    minEle = INT_MAX
    for (j=i; j<N; j++) {
```

```
maxEle = max(maxEle, A[j]);
minEle = min(minEle, A[j]);

sum = sum + (maxEle - minEle);
        }
    }
return sum;
```

T.C. = $O(N^2)$

S.C. = $O(1)$

2) Optimal Sol^n (Contribution Technique)

⇓

elements ⇒

⇓

Contribution = A[i] × (No. of sub. A[i] is max − No. of sub. A[i] is min)

⇓

S	E	Max	Min	Max − Min
0	0	2	2	0
0	1	5	2	3
0	2	5	2	3
1	1	5	5	0
1	2	5	2	3

1	1	3	3	0
2	2	3	3	0
		Sum of max of all subarr	Sum of min of all	Add
				8 Ans

$A = [2, 5, 3]$ (indices 0, 1, 2)

$i = 0 \Rightarrow 2 \times (1 - 3) = -4$ (-2)

$i = 1 \Rightarrow 5 \times (4 - 1) = 15$ (3)

$i = 2 \Rightarrow 3 \times (1 - 2) = -3$ (-1)

8

Q) In how many subarrays $A[i]$ is max

$A = [8, 6, 3, 1, 2, 5, 4, 2, 7]$ (indices 0 1 2 3 4 5 6 7 8; i = 5)

$S' \le S$

$e \ge S$

NGEL

$S = NGEL + 1$

e = NGER − 1

NGER

$S = [NGEL+1, i] \Rightarrow (i - NGEL)$

$e = [i, NGER-1] \Rightarrow (NGER - i)$

Subarrays = $(i - NGEL) \times (NGER - i)$;

Subarrays where A[i] is min = $(i - NSEL) \times (NSER - i)$;

Steps

1) Calculate NSEL — N, N
2) " NSER — N, N
3) " NGEL — N, N
4) " NGER — N, N
5) Calculate contribution of each i. N

T.C. = O(N)

S.C. = O(N)

s a e m a d a m z x y

$x_i \Rightarrow$ ______

$O(N) \times O(N) = O(N^2)$

⇓

DSA ⇒ SQL, LLD, HLD, Project

Time Spent on Ass / H.W. ⇓

GOAL ⇒ Attend as many live sessions as possible

&

Solve as many recent assignment question as possible

⇓

70% PSP

LRU