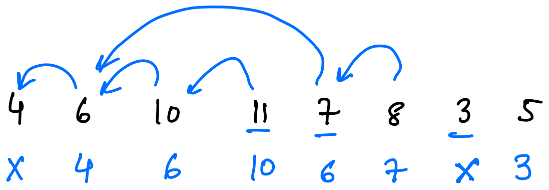
- Nearest Smaller Element

For every index i, Find the nearest element on left which is smaller than arr[i].

0	1	2	3	4	5
4	5	2	10	8	2
□ INT_MAX	4	☒	2	2	☒



4	6	10	11	7	8	3	5
X	4	6	10	6	7	X	3

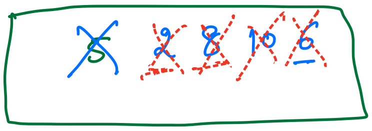
B.F

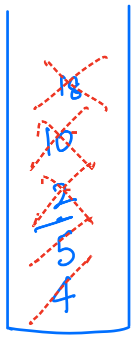
for every index i, traverse from i-1 → 0 until you get a smaller elem than arr[i].

T.C: $O(n^2)$

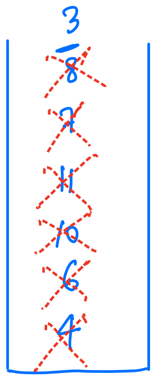
5	2	8	10	6	1	.....
X -1	X	2	8	2	X -1	

stack →





4	5	2	10	18	2
-1	↑	↑	↓	↓	
x	4	-1	2	10	-1
	↓ current top				



0	1	2	3	4	5	6	7	
4	6	10	11	7	8	3	5	
↓	↓	↓	↓	↓	↓	↓	↓	
-1	4	6	10	6	7	-1	3	
-1	0	1	2	1	4	-1	6	} indices

-1 for no available

```
stack<int> st;      ans[ ];

      for( i=0; i<n; i++)
      {
            while( !st.empty() && st.top() >= arr[i])
                  st.pop();
            if( st.empty())   ans[i]=-1;
            else              ans[i]= st.top();

            st.push(arr[i]);
      }
```

>= → <=

T.C: O(n)

nearest smaller element on right → iterate from right

nearest greater element on left

nearest greater on right → iterate from right

```
for( i=0; i<n; i++)
{
                    !st.empty() &&
          while( arr[st.top()] >= arr[i])
                    st.pop();
          if( st.empty())  ans[i] = -1;            ans[i] = n;
          else        ans[i] = st.top();

          st.push(~~arr[~~i]);
                       i
}
```

Q Find the sum of max-min for all subarrays.

0	1	2
2	5	3

$3*4/2=6$ $n \downarrow \frac{n*(n+1)}{2}$

	max	min	diff
[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
			8

BF ?

$\approx n^2 * n = O(n^3)$

$\sum \text{maxs} - \text{mins}$

$\sum \text{maxs} - \sum \text{mins}$

$$\boxed{\sum \text{maxs} = \sum_{i=0}^{n} \text{count in how many subarray arr[i] is max} * \text{arr[i]}}$$

	0	1	2	3	4	5	
[	2	13	8	5	4	7	]
-1		2*5		1*2 = 2		3*1	

nearest larger on left

contribution of i^{th} element $= (i - j) * (k - i) * arr[i]$

j: nearest largest index on left

k: nearest large on right

$\sum nums = \sum_{i=0}^{n}$ for how may subarr i^{th} element is min * arr[i]

T.C: O(n) = n+n+n+n+n

S.C: O(n) = $\underline{n+n}+\underline{n+n}$

```
// Build nearest smaller arrays for left & eight
// Build nearest larger arrays for left & right
     ans=0
   for i=0 ⟶ n
{
     ans += ((i-j)*(k-i) - (i-j')*(k'-i))* arr[i];
}
```

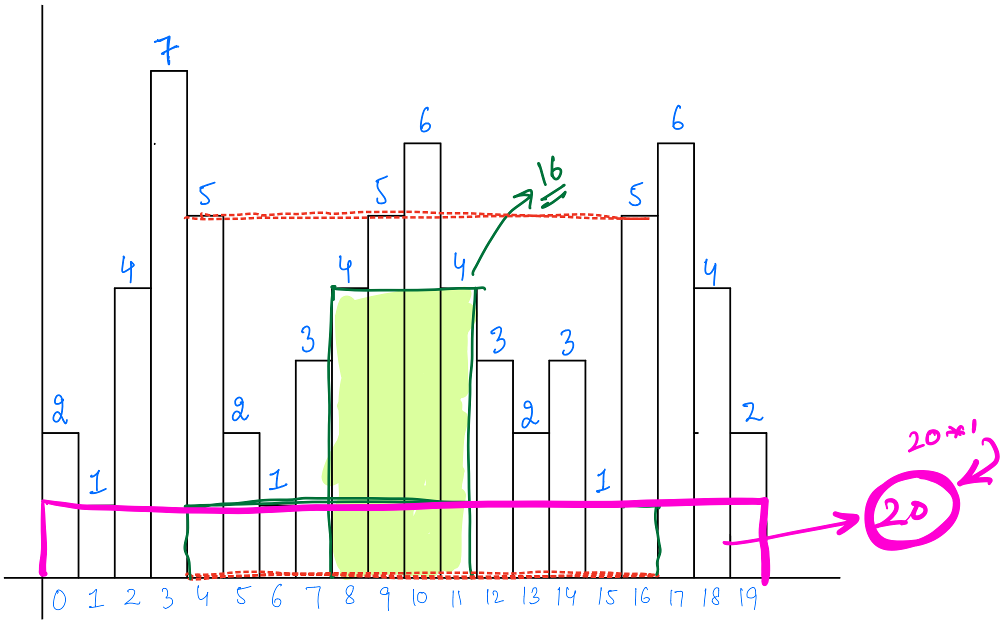
nearest smaller on left

nearest larger on left

eight

right

- Histogram - Find the largest rectangle formed by continuous histogram bars.



area = base * height
↳ maximise

go to each bar → fixing the height

i^{th} bar → nearest smaller on left ≡ l
nearest smaller on right ≡ r

↳ $(r-l-1) * arr[i]$
↑
max

•

$\underbrace{8 * 5}_{40} + 4$ $+4 = 44$

$\overrightarrow{8 + 5 - 4}$ $= \boxed{9}$

infix expression

$\boxed{\text{operand} \quad \text{op} \quad \text{operand}}$

• precendence of operators

$*, / \leftarrow +, -$

• L R associativity
↓
same precendence
calculate from
L to R

$10 + 3 * 4 - 7$ $O(n^2)$

infix notation don't have any info order of execution of operators

$A + B$

postfix

operand opera op

$AB+$

prefix
↓
op operand operand

$+AB$

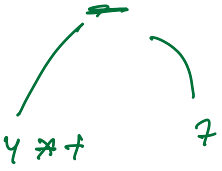
have operators in order of execution

• $A + B \Longrightarrow AB+$

• $A + \underbrace{B * C}_{BC*} \Rightarrow ABC*+$

- 10 + 3 * 4 - 7

  10 34* +



1034*+7-

x op y ⇒ x y op

- (10 + 3) * 2 - (7-6) * (4+8)

  103+ * 2 - 76- * 48+

  103+ 2 * - 76- 48+ *

10 3 + 2 * 7 6 - 4 8 + * -

A + B * C

↓

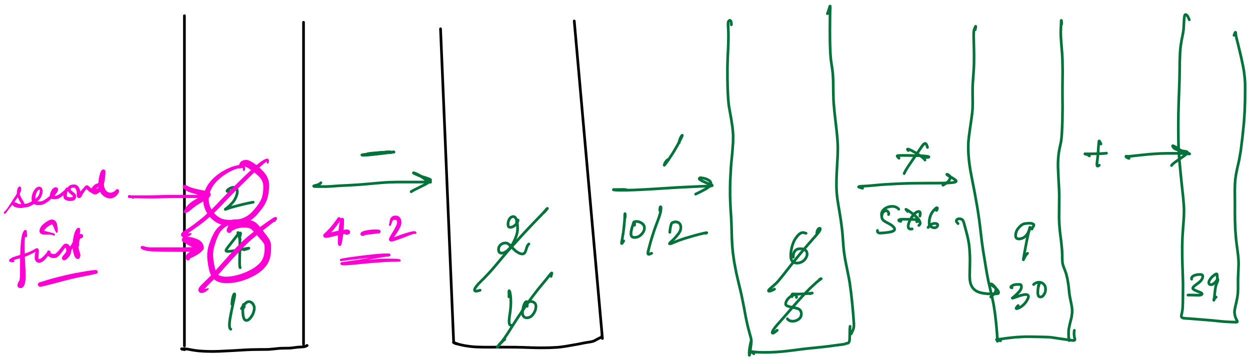
A B C * +

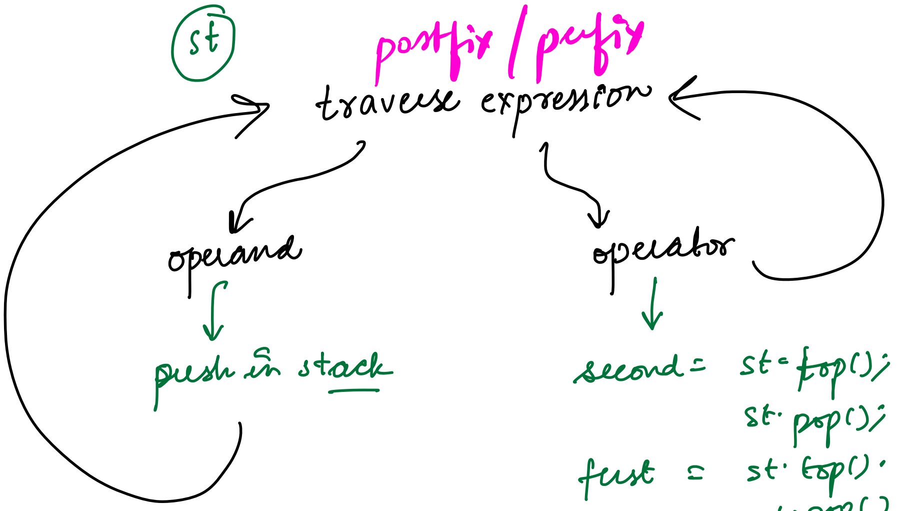
A ~~B C~~ $\overset{*}{\text{value}}$ +

B*C = value

10 / (4 - 2) * 6 + 9

10 4 2 - / 6 * 9 + → postfix





st

postfix / prefix

traverse expression

operand

push in stack

operator

second = st.top();
st.pop();

first = st.top().
st.pop()

push→(first op second) stack

ans = st.top()