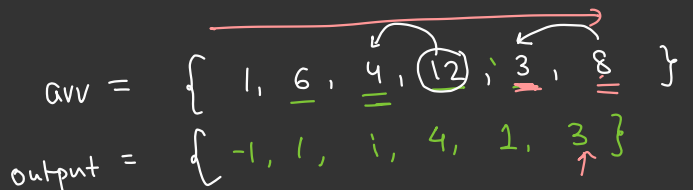


In prev class -

① Find smaller element to the left for every array element

O(N) {
- O(N) algo using stack.
- Smaller on the right
- find big element on the left
- " " " " " right (for (i=n-1 — 0)

⇒ Loop & comparison condition

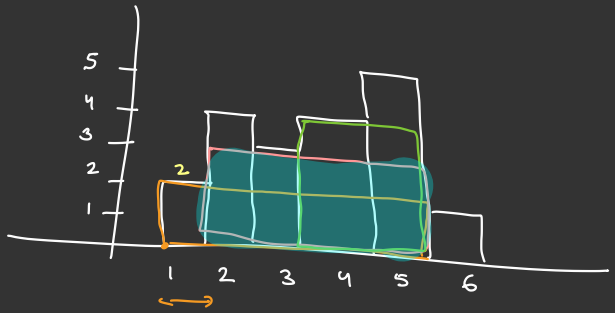
Classical Problem : Largest Area under **Histogram** [Data Visualisation]

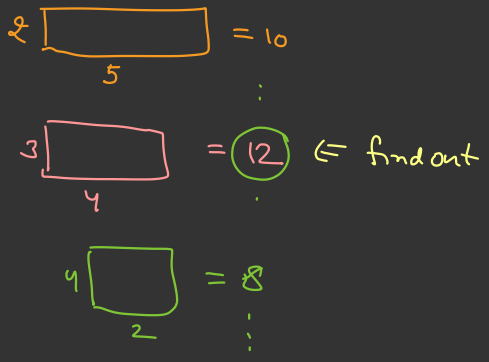
↓

graph to represent data / bar graph

Challenge — Largest Rect Area under Histogram (No re-ordering)

⇒ Area under largest rect block that ~~goe~~ lies inside histogram bars.





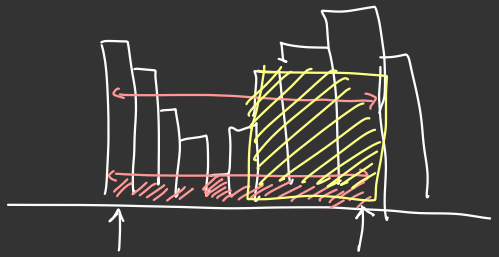
Input

1D array

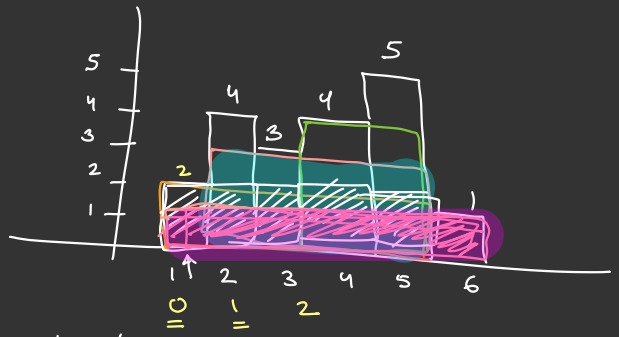
arr = [2, 4, 3, 4, 5, 1]

B1 B2 B3 B4 B5 B6

Output



Brute Force Logic –



		H	W	Area
S=1	e=1	2	1	2
S=1	e=2	2	2	4
S=1	e=3	2	3	6
S=1	e=4	2	4	8
S=1	e=5	2	5	(10)
S=1	e=6	1	6	6
S=2	e=2			
	e=3			
	e=4			
	⋮			

largest = 0

```
for(s → every bar){
  for(e → every bar >= s){
      w = e-s+1
      H = min( A[s], .... A[e] )
      area = W,H
```

$O(N^3)$

H = min(A[s], ... A[e]) → Loop O(N)

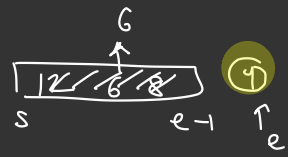
→ Carry Forward O(1) at each position

largest = max (largest, area)

}

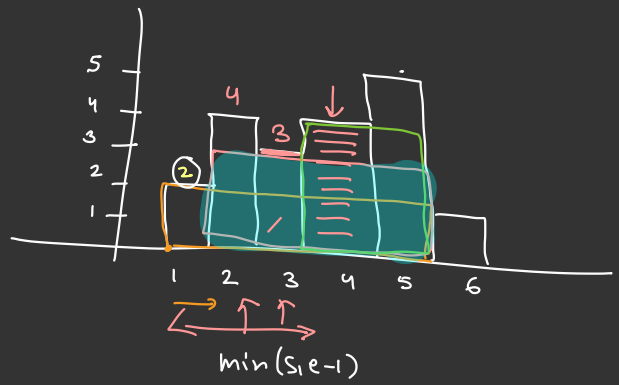
}

print (largest)



$$min[s,e] = min(a[e], min[s,e-1])$$

O(N²) using carry forward technique to find the minima



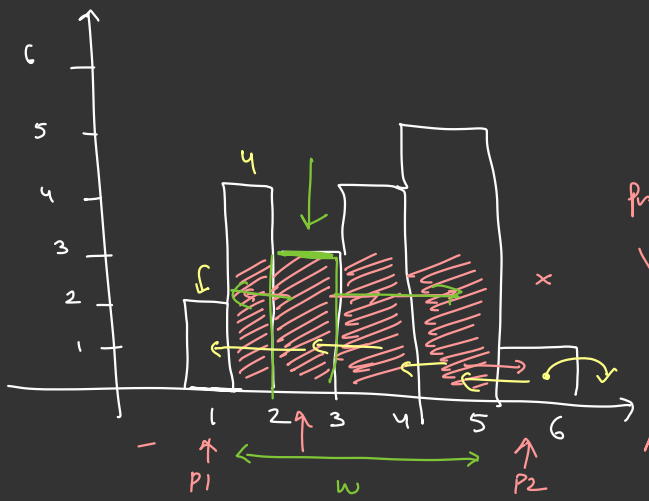
O(1)

		min	W	A
S=1	e=1	2	1	1
	e=2	(2,4)	2	4
	e=3	(3,2)	3	6
	e=4	(4,2)	4	8

2

$$\underline{min(s,e)} = min\left(\underline{a[e]}, \underline{min(s,e-1)}\right);$$

⋮



bar[3] is included in the ans.

Precompute

~~Iterate of every i~~ O(N)

(P1) → idx of bar on the left which is less than b[i]

(P2) → idx of bar on right which is less than Current bar

input (2, 4, 3, 4, 5, 1)

P1 idx = [-1, 0, 0, 2, 3, -1] O(N)

P2 idx = [ ] O(N)

w = P2 − P1 − 1 = 6 − 1 − 1 = 4

H = arr[3] = 3

Area = 3 × 4

P1 = }
P2 = } Compute using Stack Algo

$O(N+N+N) = O(N)$

for(every bar b[i]){

Stack Algo

P1 → (get left) P1[i] ⇒ O(1)

P2 → (get Right) P2[i] ⇒ O(1)

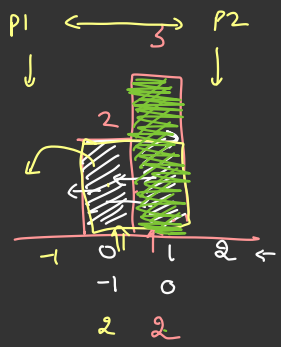
W = (P2 − P1 − 1)

H = b[i]

A = W * H

Largest = max (A, Largest)

O(N)



(Put N if Right Small doesn't exist)

Bar 0

W = P2 − P1 − 1
= 2 − (−1) − 1
= 2 + 1 − 1
= 2

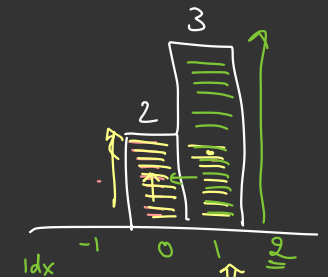
H = 2

A = 4

Bar 1

(2 − 0 − 1) × 3
= 1 × 3
= 3

→ everything upto N−1 can be included with current bar



N = 2

P1[ ] = -1 0

P2[ ] = 2 2

N N

2 - 0 - 1

= (1) × 3

= 3

for (bar) {

w1 =

A1 = 4

A2 = 3

2 + 1 - 1

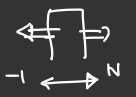
= 2 × 2

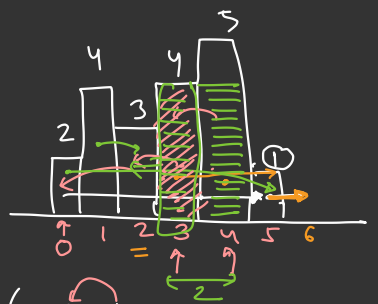
P2 - P1 + 1

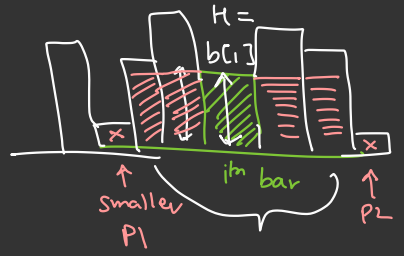
=

Q

1035







$w = P2 \to P1 - 1$

$H = b[i]$

input (2, 4, 3, 4, 5, 1)

Left P1 = (-1, 0, 0, 2, 3, -1)

Right P2 = (5, 2, 5, 5, 5, 6)

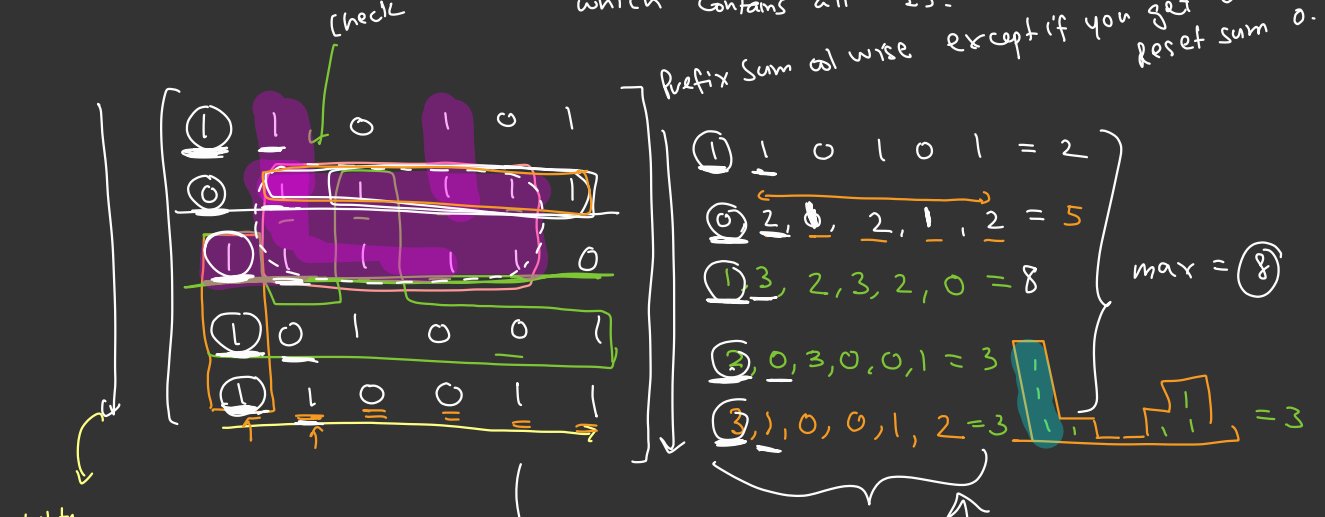
O(N) stack Algo.

w = (5, 1, 4, 2, 1, 6)

H = (2, 4, 3, 4, 5, 1)

A = (10, 4, 12, 8, 5, 6)

Q2 2D Matrix of 0's & 1's find out max Rect area, which Contains all 1's.



Brute force

$O(N^6)$ [Generate all rect $(x1, y1, x2, y2)$ $O(N^4)$ → Checking $O(N^2)$]

⇒ Six Nested loops

Matrix

$O(N^2)$ [② For every Row of PM Compute histogram $O(N)$ ③ Track the max for every Row

① Prefix Matrix $O(N^2)$

$O(N^2)$

final Question.

Given N Array elements, find **sum of max** of every subarray

[1, 4, 3]

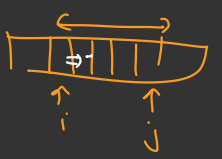
max

- [1] = 1
- [4] = 4
- [3] = 3
- [1,4] = 4
- [4,3] = 4
- ~~[1,4,3]~~
- [1,4,3] = 4

1 + 4 + 3 + 4 + 4 + 4

= 20

Brute force

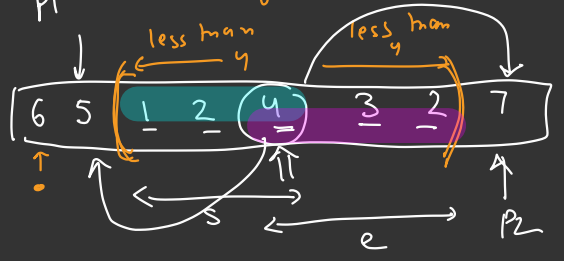



```
for(i)
  for(j)
    max ⇒ loop  O(N)
        or
        Carry forward O(1)
```

$O(N^3)$ or $O(N^2)$

Contribution of 4 in final sum

any subarray start $\leq$ 4th index



4 will be max in what subarrays

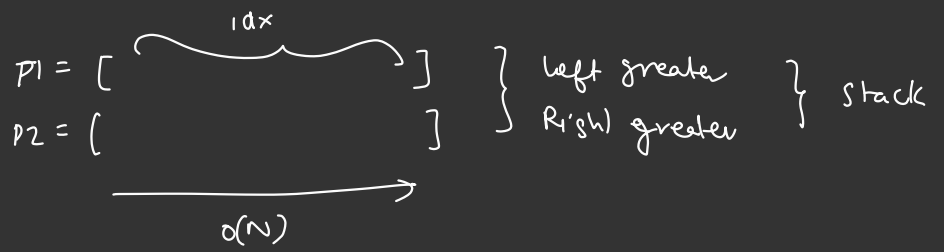
[1, 2, 4) = 4
[1, 2, 4, 3) = 4
(1, 2, 4, 3, 2] = 4
} = 3

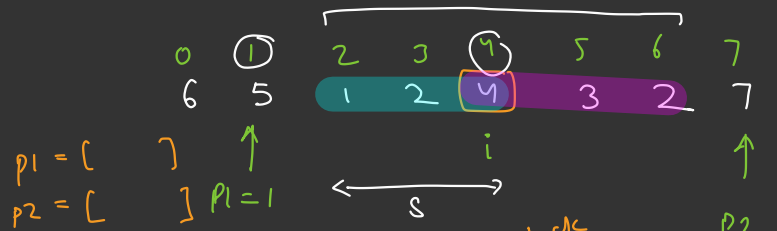
= 9

Range Range
[S, E]
=
3 x 3 Contribution
= 9 x 4
= 36

(4) } = 3
(4, 3)
(4, 3, 2)

(2, 4) = 4
(2, 4, 3) = 4
(2, 4, 3, 2) = 4
} = 3






```
for (every i) {
    start_i = (i - P1)
    end_i   = (P2 - 1)
    ans  +=   start_i x end_i x arr[i]
}
```

stack

O(N)