STACK - ADITYA VERMA

IDENTIFICATION ?

① ARRAY ✓ → Stack
→ Heap (Sort)

② BRUTE FORCE → O(n²)

```
for ( int i=0; i<n ; i++)
    for (int j=0; j<n ; j++)
  ↗
Simple O(n²)
```

```
for (int i=0; i<n; i++)
    for (int j=0; j<i ; j++)
            ← dependent O(n²)
[j = fun (i)]
```

j → 0 to i j++
j → i to 0 j--
j → i to n j++
j → n to i j--

→ $\boxed{O(n^2) + j = f(i)}$

→ 100% $\boxed{\text{Stack}}$

eg.①

Next largest element :- Nearest greater to right.

Q. arr : 1 3 2 4
↑ ↑ ↑ ↑
[3 4 4 -1] → o/p

eg.2. arr : 1 3 0 0 1 2 4
↑ ↑ ↑ ↑ ↑ ↑ ↑
o/p → [3 4 1 1 2 4 -1]

Brute force :-

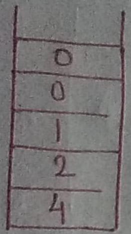
```
for (int i=0; i<n-1; i++)
    for (int j= i+1 ; j<n ; j++)
        ↙                 ↘
    O(n²)             j=fun(i)
            ↘       ↙
             STACK
        (Identification) ✓
```

eg. 1 3 ~~0~~ ~~0~~ ~~1~~ ~~2~~ 4 ✓

↑
i

→

mow, we know, we have to use stack.

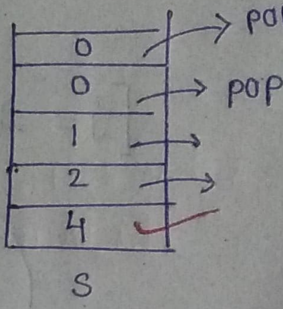


there fore '0' should be at the top as we have to compare the right most element first. Again stack uses LIFO concept.

∴ To treat '0' first we must input '0' at the last. ∴ we have to traverse the array from the back.

1	3	0	0	1	2	4

i ↑



```
if (arr[i] ≥ s.top())
    s.pop();
else
    ans.push_back(s.top())
    s.push_back(arr[i])
    i--;
```

```
*
[ if (s.empty())
  ans.push_back(-1)
  s.push_back()
  i--;
```

complexity: $O(n^2) \rightarrow O(n)$ optimal

At the end we will reverse the array (ans_array) as we are computing the ans in the reverse fashion.

8.2

Stock - Span Puroblem :-

eg. arr: 100 80 60 70 60 (75) 85

$\underbrace{60 \uparrow \quad 70 \uparrow \quad 60 \uparrow}_{4} \quad 75 \uparrow$ day 6

no. of days consecutive smaller or equal including that day on which stock prices varry.

o/p: 1 1 1 2 1 4 6

arr[]: 100 80 60 70 60 75 85

× ✓ ✓ ✓ ✓

0 1 2 3 4 5 6

(stop) ← nearest greater to left

Brute force: for (int i=0; i<n; i++)

for (int j=i ; j≥0; j--)

$O(n^2)$ + $j = f(n)\ i$ → STACK

Identification ✓

As we can clearly see that the whole programe is similar. Hence to calculate the no. of days, we will simply calculate the difference of the current day to the ngl index.

eg. for day 5 → 75

ans [5] = 5 - (Stop → index)

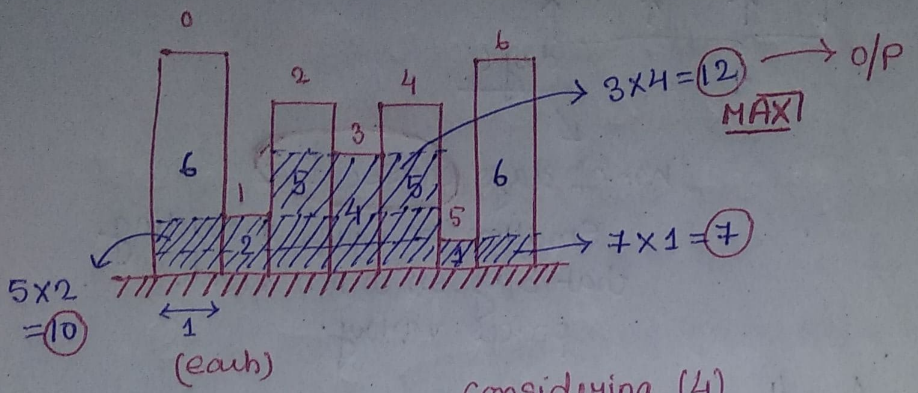
= 5 - 1

= 4

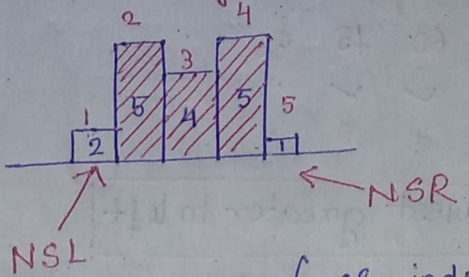
we can simply achieve this storing pairs.

Q.3 # Maximum Area Histogram:- MAH

eg. arr[]: 6 2 5 4 5 1 6 → height



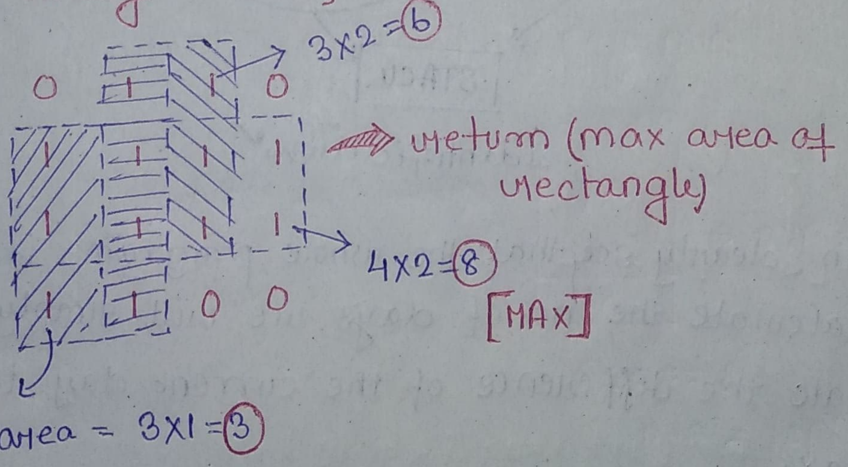
considering (4)



area = (NSR_index − NSL_index −1) * arr[i]

Q.4 # Maximum Area in a binary Matrix:-

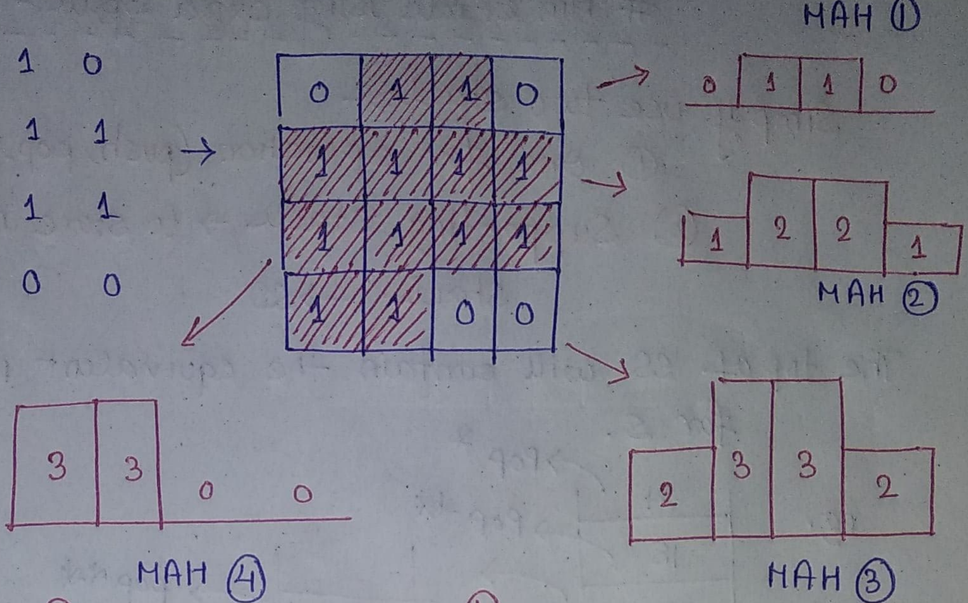
given a binary matrix;



MAH →	Natural no.	Binary no.
	1D	2D
	MAX AREA	MAX AREA

2D → 1D

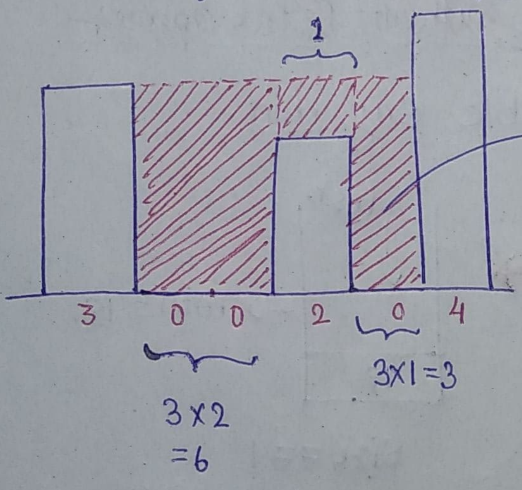
$$\begin{matrix} 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 \end{matrix}$$



$$\text{Ans} \rightarrow \max\left(\overset{②}{\text{MAH ①}},\ \overset{④}{\text{MAH ②}},\ \underset{⑧}{\text{MAH ③}},\ \underset{⑥}{\text{MAH ④}}\right) \longrightarrow ⑧$$

Rain Water Trapping

eg. arr[] : { 3, 0, 0, 2, 0, 4 }



→ 6+3+1 = ⑩ unit water

→ Solution:-

we will calculate the water over each building and then calculate their sum.

for calculating water over each building:

$$\min(\text{max\_left}, \text{max right}) - \underset{\text{height of building}}{\text{arr[i]}}$$

Min Stack using Extra Space:-

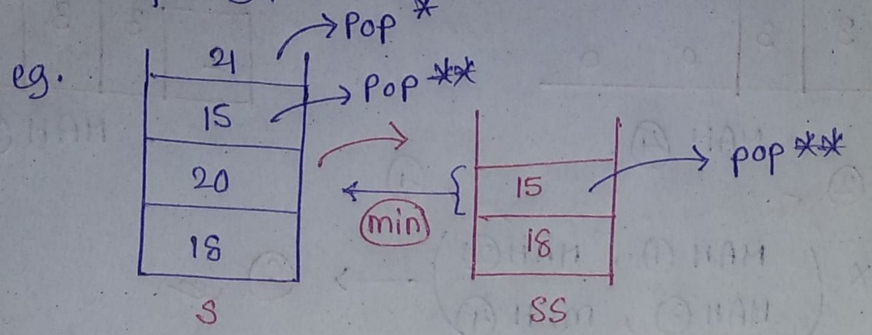
Simply use two stacks -

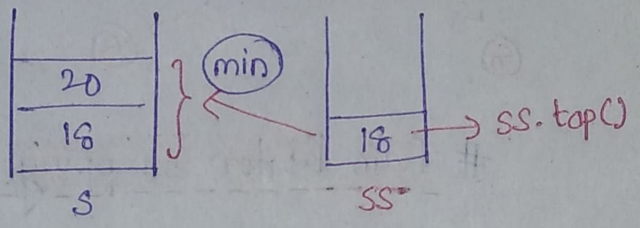
① One for operation (push, pop, top) → S

② Supporting stack → to store min of the stack → SS

The top of SS will contain the equivalent min value for S.

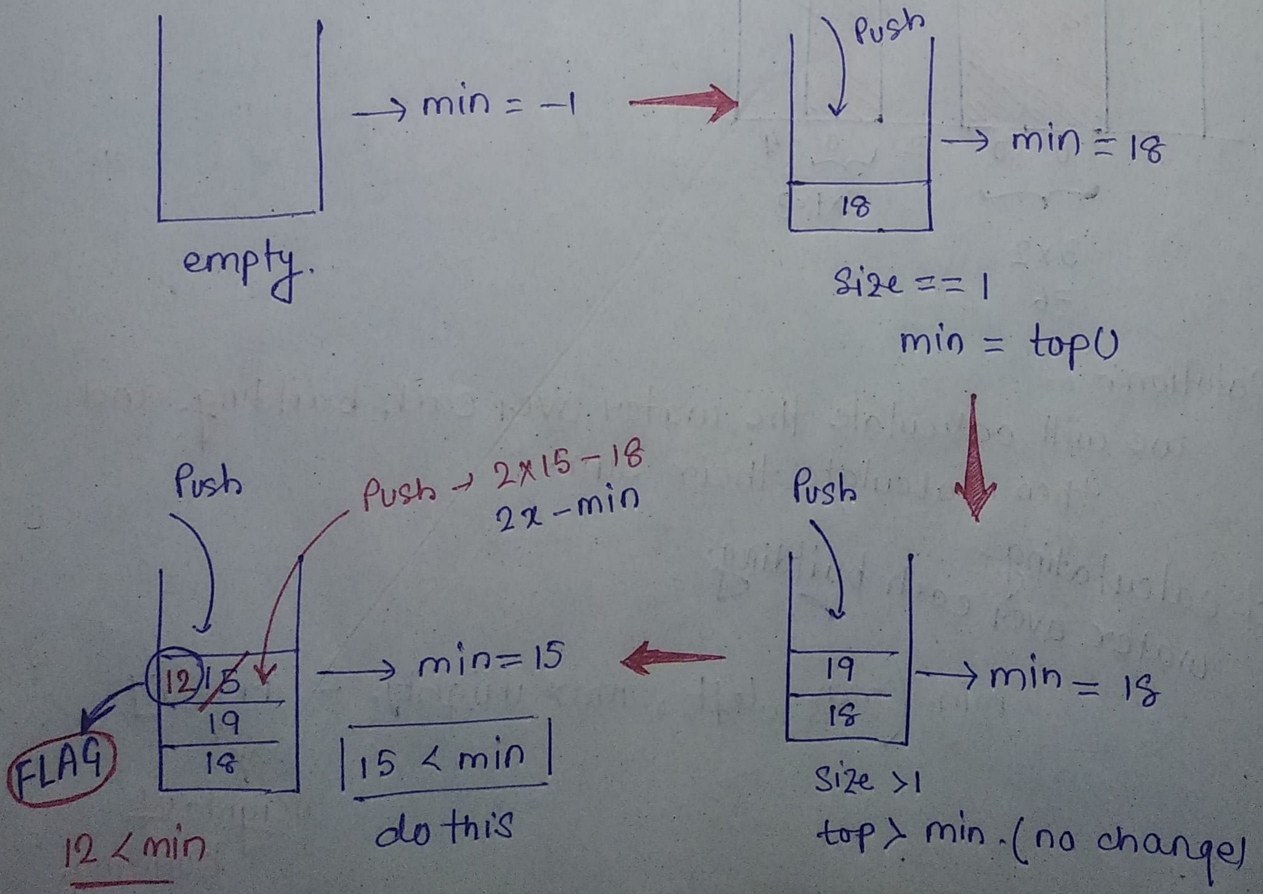
eg.

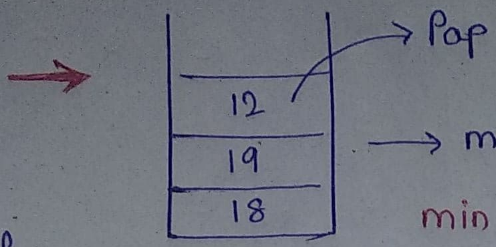




Min Stack using without Extra space:-

Solution → Having a Variable say min





→ min = 15

$\text{min} = 2 \times \text{min} - \text{top}()$

$= 2 \times 15 - 12$

$= 30 - 12$

$\boxed{\text{min} = 18}$

if

(S.top() < min)

← VALUE RESTORED

MORDERN
Problem
requires
MODERN
Solutions.