

Stack

1. Nearest greater to left
2. Nearest greater to right
3. Nearest smaller to left
4. Nearest smaller to right

①) Nearest Greater to right

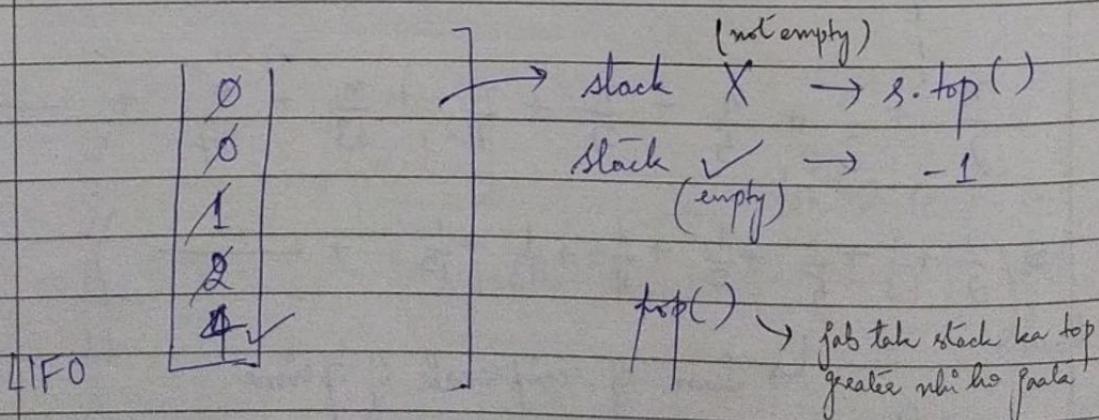
0	1	2	3	4	5	6	
arr [] :	1	3	0	0	1	2	4
	↑	↑	↑	↑	↑	↑	↑
o/p :	3	4	1	1	2	4	-1

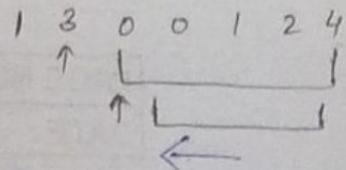
Bente face for (int i=0; i<n-1; i++) {
 for (int j=i+1; j<n; j++) {

j depend on i $O(n^2)$ → iske koi marks nahi milne wale

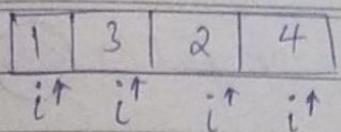
Matlab stack You really need to optimize
 ka question

1 3 0 0 1 2 4
 ↑





PAGE NO. icon
DATE: / /



List · list

list : [-1 | 4 | 4 | 3]

↓
reverse list

↓
O/P

* Stack empty \rightarrow list.add (-1)

s.top() > arr[i] \rightarrow s.top()

Code s.top() \leq arr[i] \rightarrow pop stack empty
s.top() greater than
arr[i]

Code :-

list<integer> list = new ArrayList<>();

Stack<integer> st = new Stack<>();

③ 2 I
② for(int i=0; i<size; i++)

for(int i = size-1; i>=0; i--) {

if (st.size() == 0) {
list.add (-1);

? else if (st.size() > 0 & st.peek() > arr[i]) {
list.add (st.peek());

1 3 2 4
←

< ③ III

else if (st.size() > 0 && st.peek() <= arr[i]) {
 } $\Rightarrow \textcircled{3} \text{ III}$

while (st.size() > 0 && st.peek() <= arr[i]) {
 } $\Rightarrow \textcircled{3} \text{ III}$

st.pop();

}

if (st.size() == 0) {

list.add(-1);

}

else {

list.add(st.peek());

}

}

st.push(arr[i]);

$\rightarrow \textcircled{2} \text{ No need to reverse } \textcircled{3} \text{ II}$

$\Rightarrow \textcircled{1} \text{ // like base return the REVERSED LIST}$

$\textcircled{2}$ Nearest greater than left^{to}

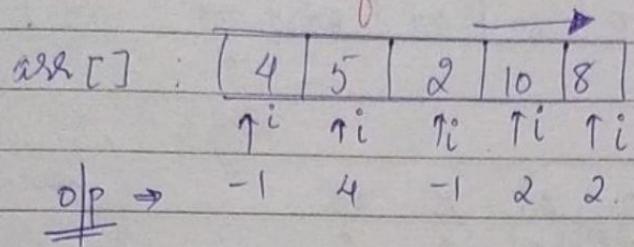
Left to Right

Change $\textcircled{1}$ Loop \Rightarrow left to right

$\textcircled{2}$ Don't reverse the list

$\textcircled{2} \text{ II}$

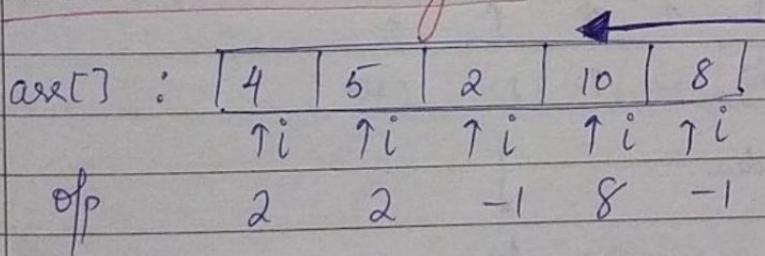
(3) Nearest smaller to left



```
for (int i = 0; i < n; i++)
    for (int j = i-1; j >= 0; j--)
        j depends on i ⇒ [STACK]
```

- Changes
- ① Less than ③ III
 - ② Reverse X ③ II
 - ③ Left Traverse ③ I

(4) Nearest Smaller to right



```
for (int i = 0; i < n; i++)
    for (j = i+1; j < n; j++)
        j depends on i ⇒ [STACK]
```

- Changes
- ① Loop : right to left
 - ② Smaller
 - ③ Reverse

Stock Span Problem

0	1	2	3	4	5	6
arr : [100]	80	60	70	60	75	85

arr [] :	100	80	60	70	60	75	85
	↑	↑	↑	↑	↑	↑	↑
o/p ⇒	1	1	1	2	1	4	6

Day 6

consecutive smaller
or equal

For 75 → 60, 70, 60 are smaller & counting itself (75)
 $3+1 = 4$

Nearest greater to left (kahan stop kr rhe h? jab
greater element mil jha hai
on left)

0	1	2	3	4	5	6
100	80	60	70	60	75	85
↑	↑	↑	↑	↑	↑	↑
-1	100	80	80	70	80	100

Index

1

$$5 - 1 = 4 \rightarrow 0/9$$

75
pair
(100, -1)

(100, 0)

(80, 1)

(60, 1)

(70, 3)

(60, 1)

(85, 0)

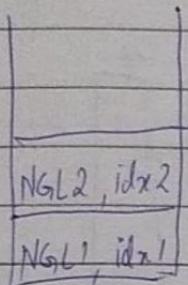
arr ka

index

nearest greater to left
ka index

(NG, index)

`Stack< pair< Integer, Integer >> st = new Stack<>();`



(NGL
element , index)

Changes :-

- ① `st. add (st. top(). second);`
- ② `st. push ({ arr[i], i });`
- ③ `st. top() → st. top(). first`

list [-1 | 0 | 1 | 1 | 3 | 1 | 1 | 0]

④ at end : `for(int i = 0 ; i < v.size() ; i++) {`
 $v[i] = i - v[i];$
`}`
 return $v;$

list \Rightarrow [-1 | 0 | 1 | 1 | 3 | 1 | 0]

$\downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow \quad \downarrow$

index of -1 100 80 80 70 80 100

$0 - (-1) = 1$
 $1 - 0 = 1$
 $2 - 1 = 1$
 $3 - 1 = 2$
 $4 - 3 = 1$
 $5 - 1 = 4$
 $6 - 0 = 6$

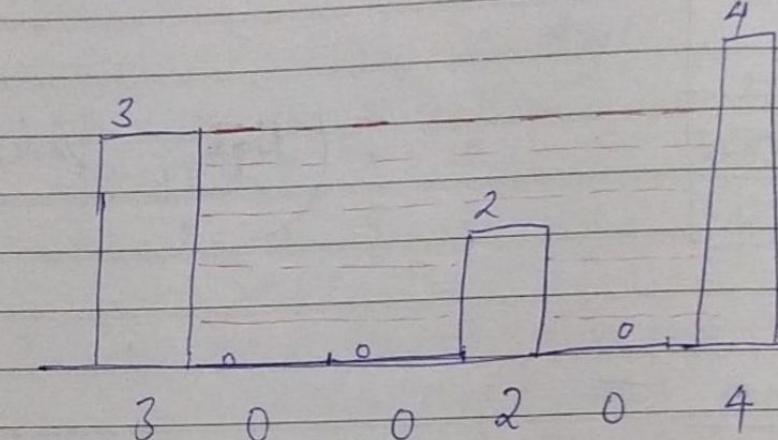
of p

Rainwater Trapping Problem

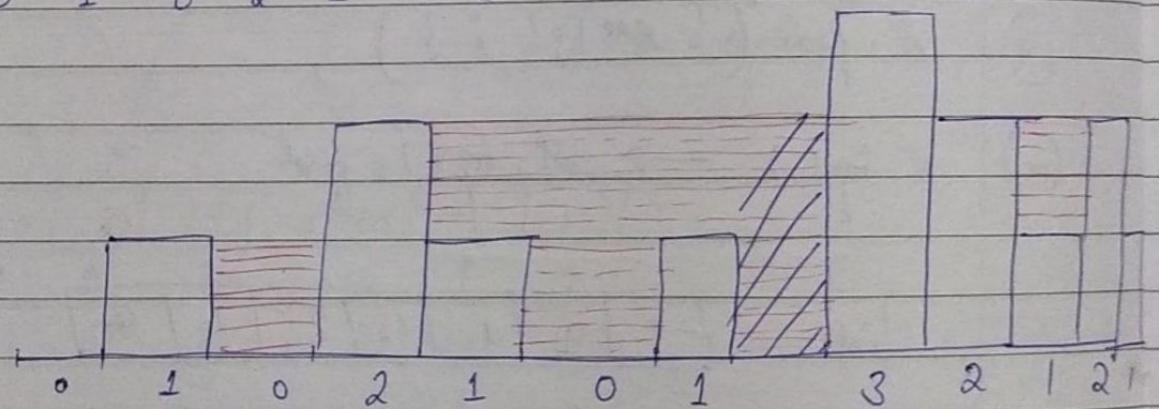
Eg. arr: [3 | 0 | 0 | 2 | 0 | 4]

s/p:

$$\begin{array}{r} 3+3+1+3 \\ = 10 \end{array}$$



Eg. 0 1 0 2 1 0 1 3 2 1 2 1



s/p: 1 + 1 + 2 + 1 + 1 = 6

Idea: How building be like

uski left array
ka man

uski right array
ka man

M/N

@ any i

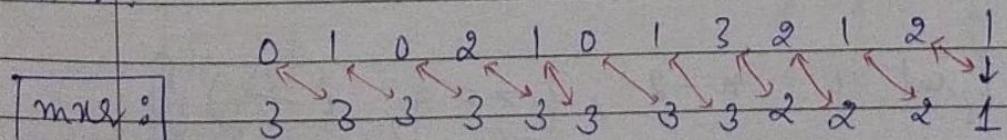
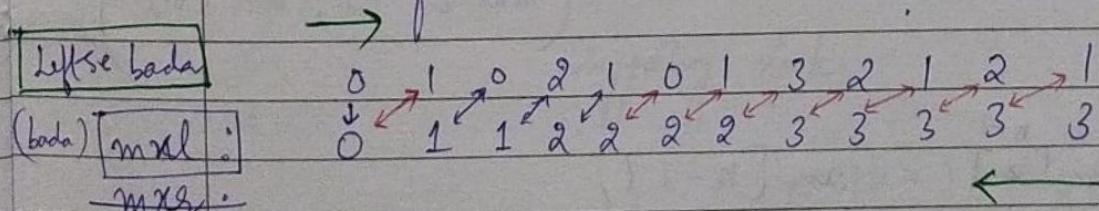
$$\text{water}[i] = \min(\text{mnL}, \text{mnR}) - \text{arr}[i]$$

$$\begin{aligned} \text{water}[i] &= \min(3, 4) - 2 \\ \text{arr}[i] &= 2 \\ &= 3 - 2 = 1 \end{aligned}$$

	3	0	0	2	4	
mnL :		0	1	3		
mnR :		1	4	1	0	

$$\text{water}[i] = \min(\text{mnL}[i], \text{mnR}[i]) - \text{arr}[i]$$

Q/H how to find mnL & mnR?



Right side
bada

$$\begin{array}{c}
 \begin{array}{r|ccccc}
 & 3 & 0 & 0 & 2 & 0 & 4 \\
 \text{mnl:} & 3 & 3 & 3 & 3 & 3 & 4 \\
 \text{mxs:} & 4 & 4 & 4 & 4 & 4 & 4 \\
 \text{min:} & \hline
 \end{array} \\
 \begin{array}{r}
 \xrightarrow{-} 3 \quad 3 \quad 3 \quad 3 \quad 3 \quad 4 \\
 - \quad 3 \quad 0 \quad 0 \quad 2 \quad 0 \quad 4 \\
 \hline
 \end{array}
 \end{array}$$

$$\sum 0 + 3 + 3 + 1 + 3 + 0 = \textcircled{10} \text{ op Ans}$$

Code:

```

int n = arr.length;
if mnl[n] == max func.
int mxs[n];

```

```

int[] mnl = new int[n];
int[] mxs = new int[n];
mnl[0] = arr[0];
for (int i=1; i < n; i++) {

```

$mnl[i] = \max(mnl[i-1], arr[i]);$

$mxs[n-1] = arr[n-1];$

```

for (int i = n-2; i >= 0; i--) {

```

$mxs[i] = \max(mxsl[i+1], arr[i]);$

3

int water[] = new int[n];

for (int i=0; i < n; i++) {

 water[i] = min (max[i], max[i]) - arr[i];
 }

int sum = 0;

for (int i=0; i < n; i++) {

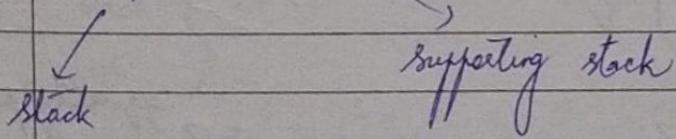
 sum = sum + water[i];

}

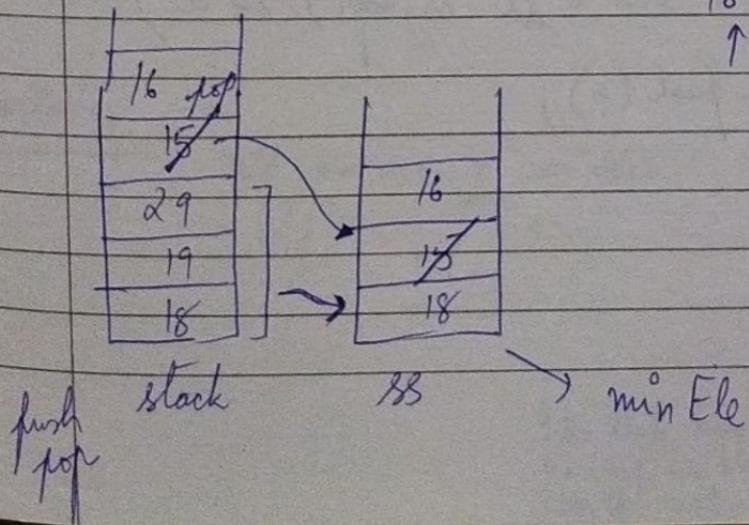
return sum;

Min Stack

- ① push
- ② minElement()
- ③ pop

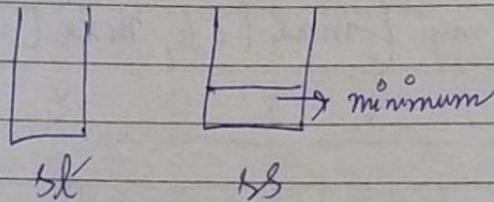


18 19 29 15 16 P P
 ↑ ↑ ↑ ↑ P P
 P P



Min Stack Implementation with Extra Space

18 19 29 15 16



* Single element is
always minimum
(obviously)

```
Stack < Integer > st = new Stack <>();
```

```
Stack < Integer > ss = new Stack <>();
```

```
int getMin() {
```

```
    if ( ss.size() == 0 )  
        return -1
```

```
    return ss.peek();
```

```
}
```

```
void push( int a ) {
```

```
    st.push( a );
```

equally probable

```
    if ( ss.size() == 0 || ss.top() >= a )
```

```
        ss.push( a );
```

```
    return;
```

```
}
```

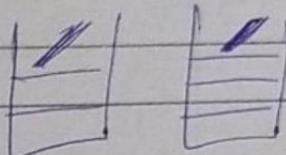
```
int pop() {
```

```
    if (st.size() == 0)
```

```
        return -1;
```

```
    int ans = st.top();
```

```
    st.pop();
```



```
    if (ss.peek() == ans)
```

```
        ss.pop();
```

```
    return ans;
```

```
}
```

*Ques***

Min Stack Implementation in O(1) Space

$O(1)$ ✓

Problem kya aayi?

arr
stack
list

$\begin{bmatrix} 7 \\ 5 \end{bmatrix}$ if greater ele at
top → NO
 $\min = 5$ Issues

containers

$\begin{bmatrix} 3 \\ 5 \end{bmatrix}$ smaller element
at the top
 $\min = 3$

Variables

If you use n variables

$O(1)$

Now if $\text{pop}()$ is called
on stack

then $\min = 5$

but how to retrieve
that old value from
 $\min \Rightarrow \text{NOT POSSIBLE}$

Iks thi kaise
ke liye we store
corrupt value

Wapas min to 3 se 5 kena
IMPOSSIBLE

int minElem = Integer.MAX_VALUE;

int getMin() {
 if (s.size() == 0)
 return -1;
 return minElem;

}

void push (int x) {

 if (s.size() == 0) {
 s.push(x);
 minElem = x;

}

else {

 if (x >= minElem) {
 s.push(x);

}

else if (x < minElem) {

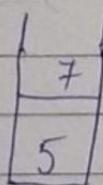
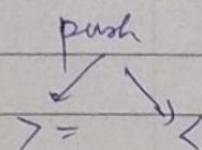
 s.push (2 * x - minElem);

 minElem = x;

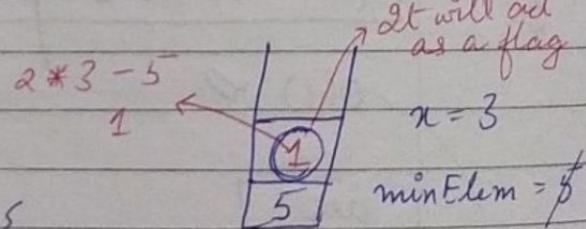
}

3

3



$$\text{minElem} = 5$$

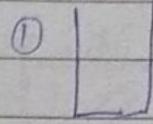


$$x = 3$$

$$\text{minElem} = 1$$

$$3$$

void pop () {



if (st.size() == 0) {

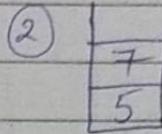
return -1;

}

else {

if (s.top() >= minElem)

s.pop();



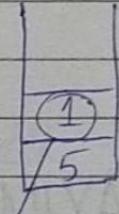
minElem = 5

7 > 5

If so, then
it can be safely
popped, bcz
minElem = 5
hi raha

elseif (s.top() < minElem) {

③



minElem = 3

minElem = 2 * minElem -

s.pop();

flag → yeh bata raha h
ki agar 1 ho pop kya
tht minElem change
hoga

2ME-y

→ s.pop()

}

3

3

int top () {

if (s.size() == 0) {

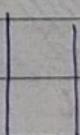
return -1;

}

else {

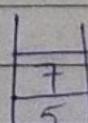
7 5

①



return -1

②



minElem = 5

*if (st.top() >= minElem) {
 return s.top(); }*

3

*elseif (s.top() < minElem) {
 return minElem; }*

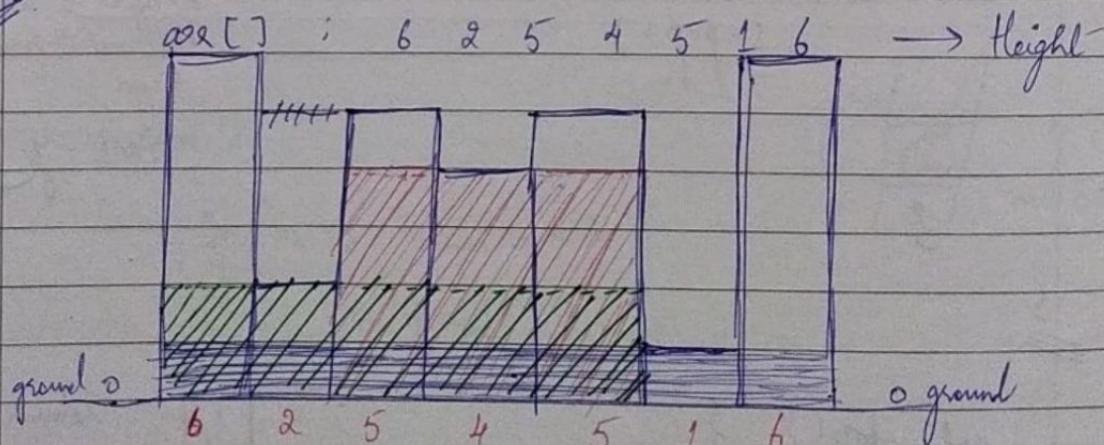
3

3

*kyunki MF stack ka hi MF hai means st.top() fay hai
agar st.top() minElem se bhi chota hai, that means st.top() fay hai*

* Famous popular asked

MAXIMUM AREA HISTOGRAM (MAH)



Red
 $\text{Area} = 3 \times 4 = 12$

Blue
 $1 \times 7 = 7$

Green
 $2 \times 5 = 10$

MAXIMUM

Index 4

NSR	5
NSL	1
	—
	4
	-1

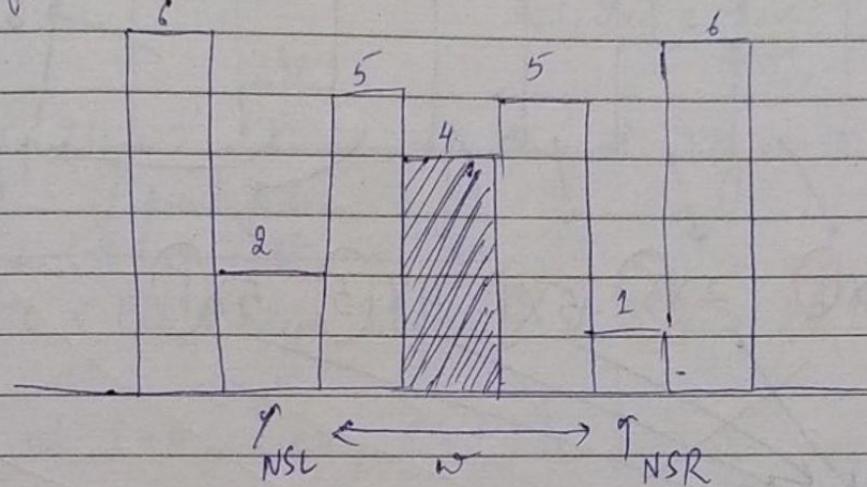
= ③ width

0 1 2 3 4 5

6 | 2 | 5 | 4 | 5 | 1 | 6

Linden NSR \rightarrow right : 1 5 3 5 5 ~~17~~ ~~17~~

Linden NSL \rightarrow left : -1 -1 1 1 3 -1 5



Area tak tak expand ho sakla hain fab
tak building ke left & right me koi
chhota element building na mil jaaye

$$\text{width} = 5 - 1 = 4$$

$$4 - 1 = 3$$

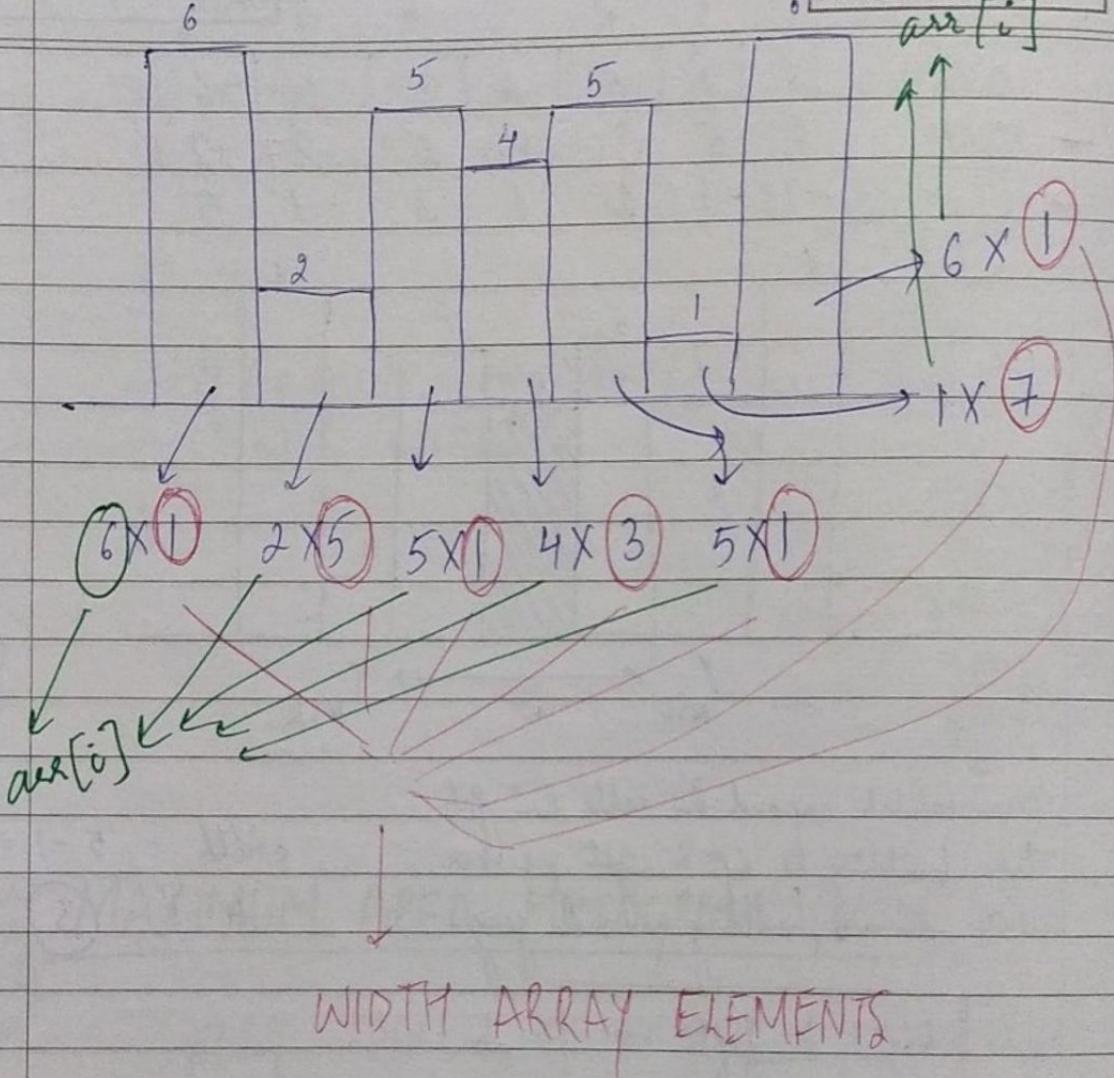
right - left - 1 \Rightarrow width

$$\boxed{\text{width}[i] = \text{right}[i] - \text{left}[i] - 1}$$

\rightarrow iss formula se

width : 1 5 1 3 1 7 1

arr[i]



$$\text{arr}[0] : 6 \quad 2 \quad 5 \quad 4 \quad 5 \quad 1 \quad 6$$

$$\text{width}[i] : 1 \quad 5 \quad 1 \quad 3 \quad 1 \quad 7 \quad 1$$

$$\boxed{\text{Area}[i] = \text{arr}[i] * \text{width}[i];}$$

Area[7] : [6 | 10 | 5 | 12 | 5 | 7 | 6]

Find \max^m
=) ANSWER

Code :-

left → NSL Index

NSL, idx

pair

left

List <Integer> list = new ArrayList <Integer>();
int pseudoIndex = -1;

Stack <Integer> st = new

Stack <Pair <Integer, Integer>> st = new Stack <>();

for (int i=0; i<n; i++) {

if (st.size() == 0) {

left.list.add (F/N); pseudoIndex

}

else if (st.size() > 0 && st.peek().first < arr[i]) {

left.add (st.pop().second);

}

else if (st.size() > 0 && st.peek().first >= arr[i]) {

while (st.size() > 0 && st.peek().first >= arr[i]) {

st.pop();

}

if (st.size() == 0) {

left.add (-1);

}

else {

left.add(st.push(), second);

}

3

st.push(area[i], i);

}

return left;

3rd by → right → NSL index

int pseudosIndex = 7; (arr.length + 1)

for (int i=0; i < size; i++) {

width[i] = right[i] - left[i] - 1;

}

for (int i=0; i < size; i++) {

area[i] = area[i] * width[i];

}

Find max in area[i]

ans

ans

Stack Problem!!!

PAGE NO. / /
DATE: / / icon

Max area Rectangle in Binary matrix

MAH (int arr[], int size) {

right[] → NSR (arr, size)

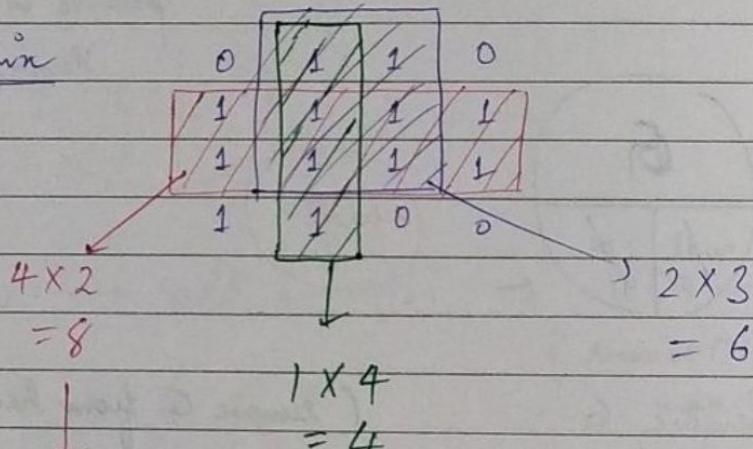
left[] → NSL (arr, size)

width[i] → right[i] - left[i] - 1

area[i] → arr[i] * width[i]

return max in area[i]

Binary matrix



MAXIMUM

Ans

(MAH → 1D
arr → 2D)

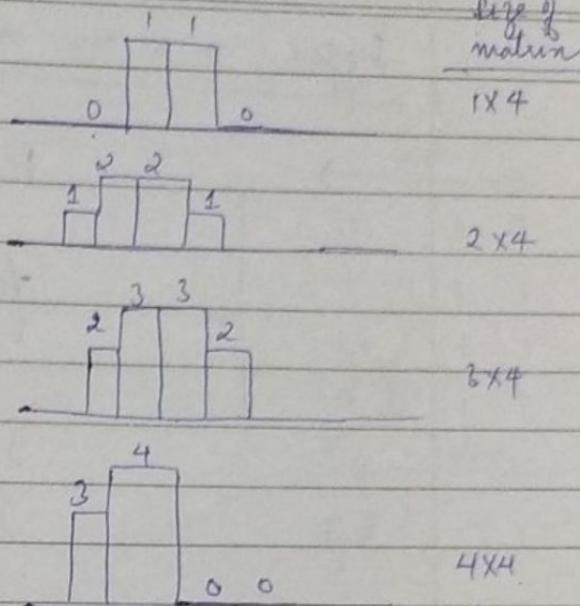
(2D → 1D me convert karna hain)

agar kardiga toh MAH already kar chuke hain

Joh ye que easily solve ho jaayeg.

0	1	1	0
1	1	1	1
1	1	1	1
1	1	0	0

ground



ans = Math.max (MAH(H1) → 2 → 1x4
 MAH(H2) → 4 → 2x4
 MAH(H3) → 8 → 3x4
 MAH(H4) → 6 → 4x4)

8 is minimum
Ans

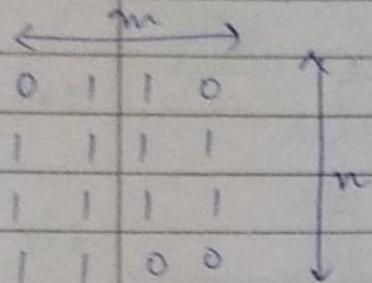
Code :-

0	1	1	0
1	1	1	
1	1	1	
1	1	0	0

H1 vector	=	0 1 1 0	→ MAH ✓
H2 vector	=	1 2 2 1	→ MAH ✓
H3 vector	=	2 3 3 2	→ MAH ✓
H4 vector	=	3 4 0 0	→ MAH ✓

* agar 1 hai toh add
 karuge, 0 hai toh
 sare 0 likh dunge
 vector me }

Suppose $n \times m$ matrix



list <int> ℓ_1 = new ArrayList<>();

for($j=0$; $j < m$; $j++$) {

ℓ_1 . add($arr[0][j]$);

}

int mn ^{^2} = MAH (ℓ_1);

for(int $i=1$; $i < n$; $i++$) {

 for(int $j=0$; $j < m$; $j++$) {

 if ($arr[i][j] == 0$) {

$\ell_1[j] = 0$

 } else {

$\ell_1[j] = \ell_1[j] + arr[i][j]$;

}

$mn = \max(mn, MAH(\ell_1))$;

8

return mn

}