Stacks -2

Lecture-46

Raghav Garg

Ques: Balanced Brackets

'(' , ')'

'Easy'

→ String s = "()()()"; true

String s = "(())()"; true

string s = "()(()()"; false

string s = ")()("; false

string s = "(((()))"; true

pop()
push()
top()

Ques: Balanced Brackets

String S = "()()()";
i

Steps

1) If you see opening bracket
→ push

2) If you see a closing bkt,
st → top ko dekho, agar → '('
st.pop()

→ If st.size()==0, true

Ques: Balanced Brackets

string s = "(())()"; true

i

Steps

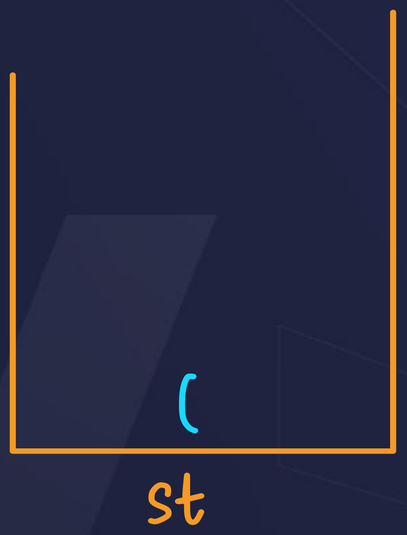
1) If you see opening bracket
   → push

2) If you see a closing bkt,
   st → top ko dekho, agar → '('
   st.pop()

→ If st.size() == 0, true

Ques: Balanced Brackets

```
string s = ")(()()"
            i
```



Steps

1) If you see opening bracket
   → push

2) If you see a closing bkt,
   st → top ko dekho, agar → '('
   st.pop()

→ If st.size() == 0, true
   else false

Ques: Balanced Brackets

string s = "())()(";
i

Steps

1) If you see opening bracket
→ push

2) If you see a closing bkt,
st → top ko dekho, agar → '('
st.pop(), agar st empty → false

→ If st.size() == 0, true
else false

Ques: Remove Consecutive Duplicates in a string

String s = "aaabbcddaabffg";

str = "g f b a d c b a"

rev ↳ "a b c d a b f g"

Ques: Next greater element

Problem Statement :

arr	3	1	2	5	4	6	2	3
ans	5	2	5	6	6	-1	3	-1

	T.C.	S.C
Brute Force Solution :	$O(n^2)$	$O(1)$

$O(n)$

Method-2 'Using stack'

	i							
arr	3	1	2	5	4	6	2	3
ans	5	2	5	6	6	-1	3	-1

```
ans[n-1] = -1;
st.push(arr[n-1])
```


```
while (st.top() <=arr[i]) st.pop();   POP
ans[i] = st.top;                      ANS
st.push(arr[i]);                      PUSH
```

pop, ans, push

Example

arr =	4	1	2	5	4	3	4	6	2	7
ans =	5	2	5	8	8	4	8	-1	7	-1

pop

ans

push

Ques: Previous greater element

arr	3	1	2	5	4	6	2	3
pge	-1	3	3	-1	5	-1	6	6

pop

ans

push

Ques: Stock Span problem

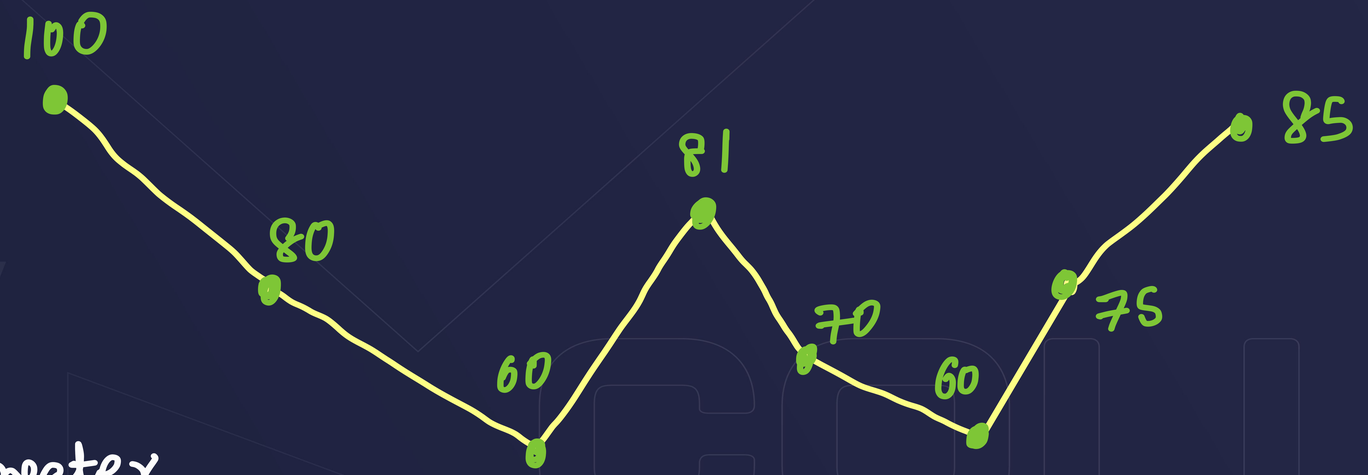
Input – {100,80,60,70,60,75,85}

Output – {1 , 1 , 1, 2 , 1, 4 , 6}

span, pehle se lekar continuously

	0	1	2	3	4	5	6
arr	100	80	60	70	60	75	85
ans	1	1	1	2	1	4	6

	0	1	2	3	4	5	6	7
arr	100	80	60	81	70	60	75	85
pge	-1	100	80	100	81	70	81	100
ans	1	1	1	3	1	1	3	7



Previous Greater Element

	0	1	2	3	4	5	6	7	i
arr	100	80	60	81	70	60	75	85	
pgi	-1	0	1	0	3	4	3	0	
span	1	1	1	3	1	1	3	7	

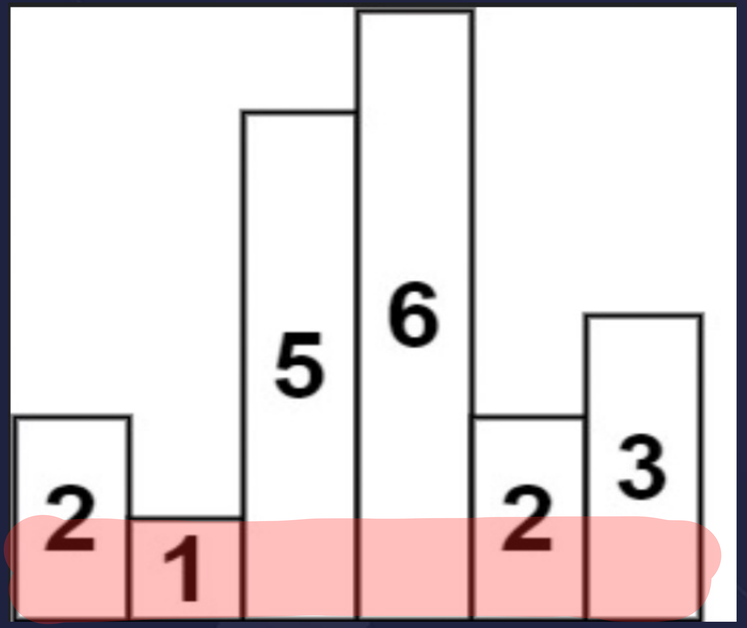
pop → while (arr[st.top()] <= arr[i])

ans

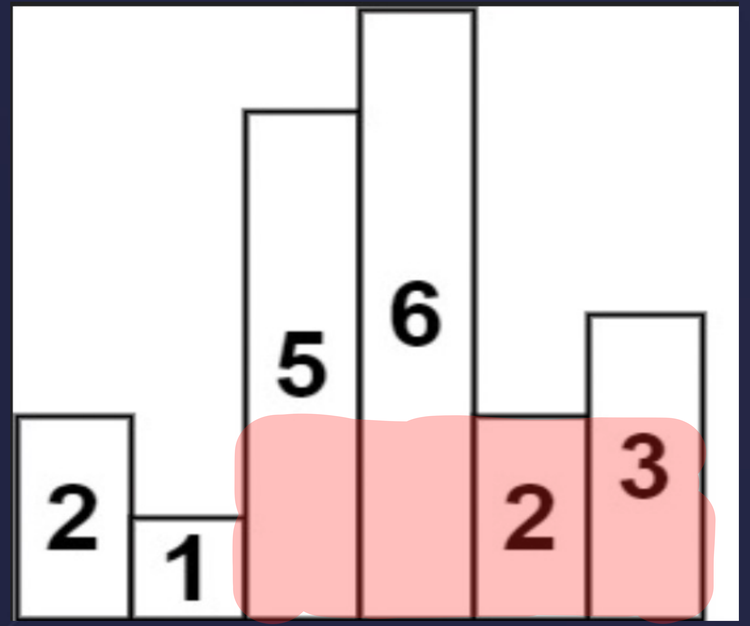
push

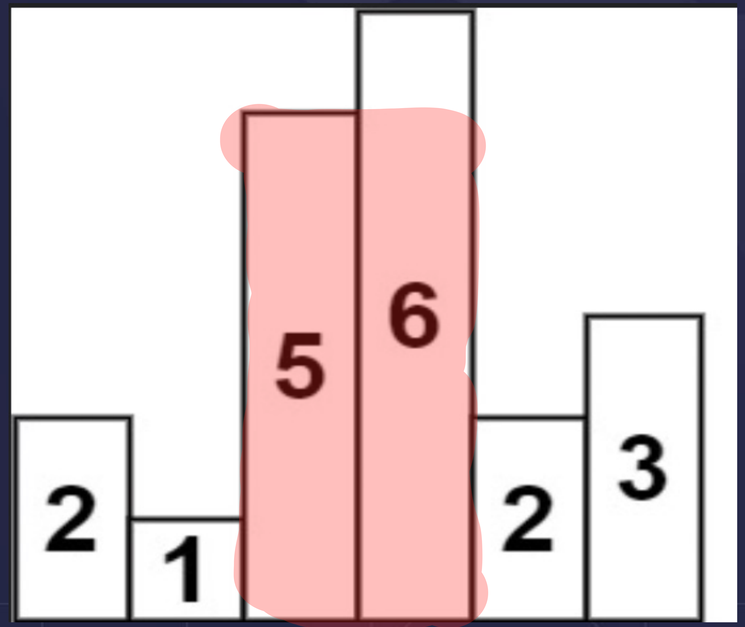
Ques: Largest Rectangle in Histogram

[Leetcode - ~~99~~ 84]

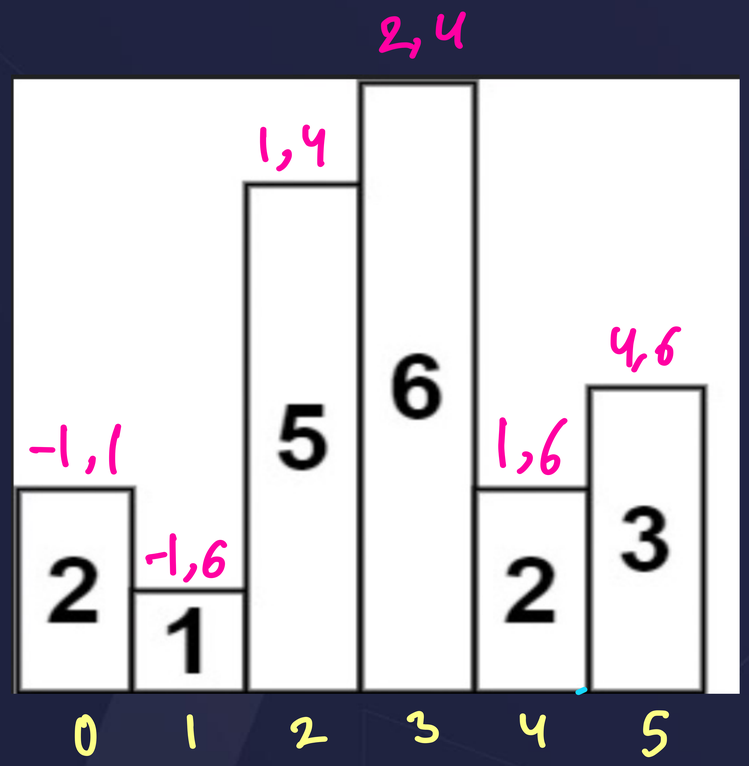


{ 2 , 1 , 5 , 6 , 2 , 3 }





Ques: Largest Rectangle in Histogram [Leetcode - ~~85~~ 84]



2	1	5	6	2	3
2	6	10	6	8	3

Hint : Nse , Pse

Area = height * breadth

↓

nsi[i] - psi[i] - 1

Next Smaller Element

-1	0	1	2	3	4	5	6
	2	1	5	6	2	3	
nse	1	-1	2	2	-1	-1	
nsi	1	6	4	4	6	6	

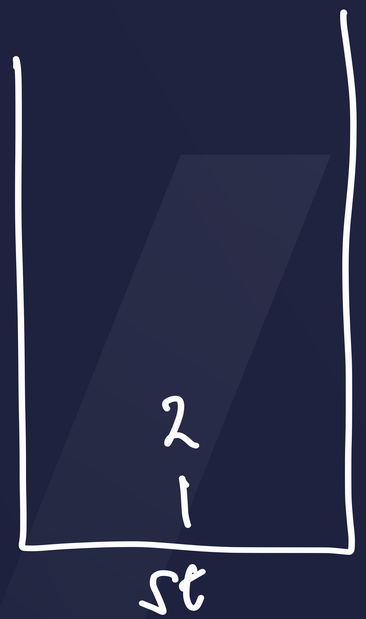
pop → apne se bade element

ans

push

Previous Smaller Element

	-1	0	1	2	3	4	5	6
arr		2	1	5	6	2	3	
pse		-1	-1	1	5	1	2	
psi		-1	-1	1	2	1	4	



pop

ans

push

```
int n = arr.size();
int nsi[n];
stack<int> st;
nsi[n-1] = n;
st.push(n-1);
for(int i=n-2;i>=0;i--){
    while(st.size()>0 && arr[st.top()]>=arr[i]) st.pop();
    if(st.size()==0) nsi[i] = n;
    else nsi[i] = st.top();
    st.push(i);
}
```

Right ✓

	-1	0	1	2
arr		2	4	
nsi		2	2	
psi		-1	0	

```
int psi[n];
stack<int> gt;
nsi[0] = -1;
st.push(0);
for(int i=1;i<n;i++){
    while(gt.size()>0 && arr[gt.top()]>=arr[i]) gt.pop();
    if(gt.size()==0) psi[i] = -1;
    else psi[i] = gt.top();
    gt.push(i);
}
```

psi Right

0 1

```
int maxArea = 0;
for(int i=0;i<n;i++){
    int height = arr[i];
    int breadth = nsi[i] - psi[i] - 1;
    int area = height * breadth;
    maxArea = max(maxArea,area);
}
return maxArea;
```

THANK YOU!