next Greater Element in Stack:

if no next greater → -1

ceil / Floor + NGE

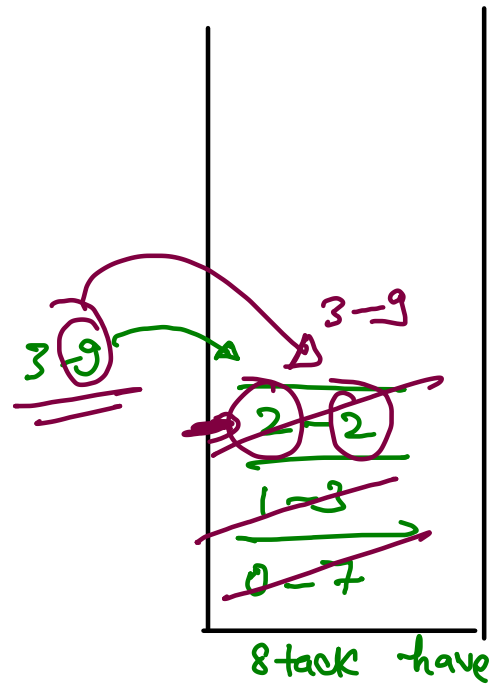
	0	1	2	3	4	5	6	7	8	9
arr →	7	3	2	9	4	6	1	8	5	3
ngi →	9	9	9	-1	6	8	8	-1	-1	-1
ngei →	3	3	3	10*	5	7	7	10*	10*	10*

(value)

(index)

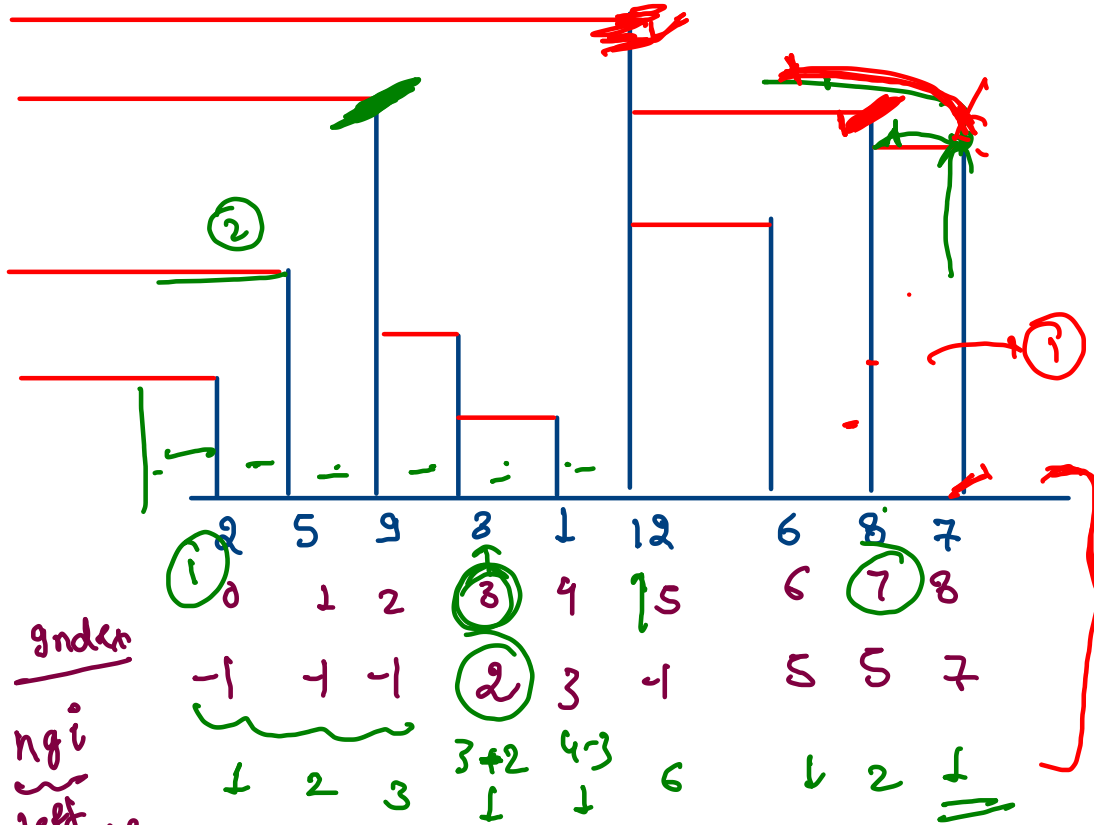
10

```
st.size()>0 &&
while(arr[i] > st.peek()){
nei[st.peek()] = i;
nge[st.peek()] = arr[i];
st.pop();
}
st.push(i);
```



Stack have indexes of element

eb = arr[st.peek()];



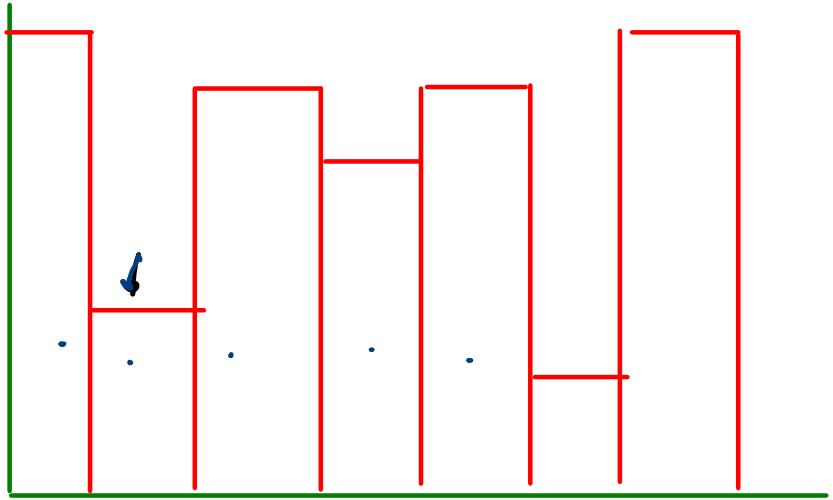
1. You are given a number n, representing the size of array a.
2. You are given n numbers, representing the prices of a share on n days.
3. You are required to find the stock span for n days.
4. Stock span is defined as the number of days passed between the current day and the first day before today when price was higher than today.

e.g.
for the array [2 5 9 3 1 12 6 8 7]
span for 2 is 1
span for 5 is 2
span for 9 is 3
span for 3 is 1
span for 1 is 1
span for 12 is 6
span for 6 is 1
span for 8 is 2
span for 7 is 1

8-7 Stock Span →

// 1 day

```
if (left greater == -1)
    lgi[i] = i+1;
```



Largest Area ??

maxArea = 0;

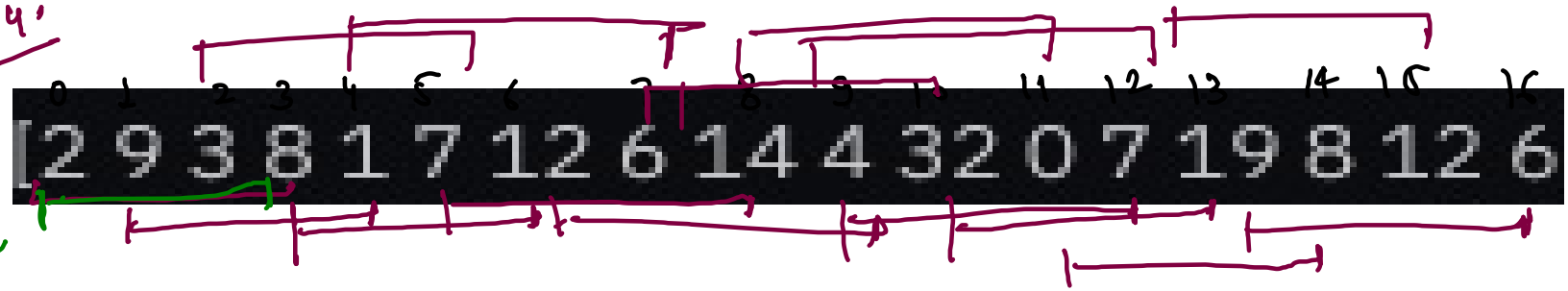
maxArea = Math.max (maxArea, (rsi[i] - lsi[i] - 1) * height[i]);
arr[i]);

Index	0	1	2	3	4	5	6	(7)
	6	2	5	4	5	1	6	
left Smaller →	-1	-1	1	1	3	-1	5	
right Smaller →	1	5	3	5	5	7	7	
width →	1	5	1	3	1	7	1	
Area =	6	10	5	12	5	7	6	

largest = (12)

width[i] = Rsi[i] - lsi[i] - 1;

Area[i] = width * Right

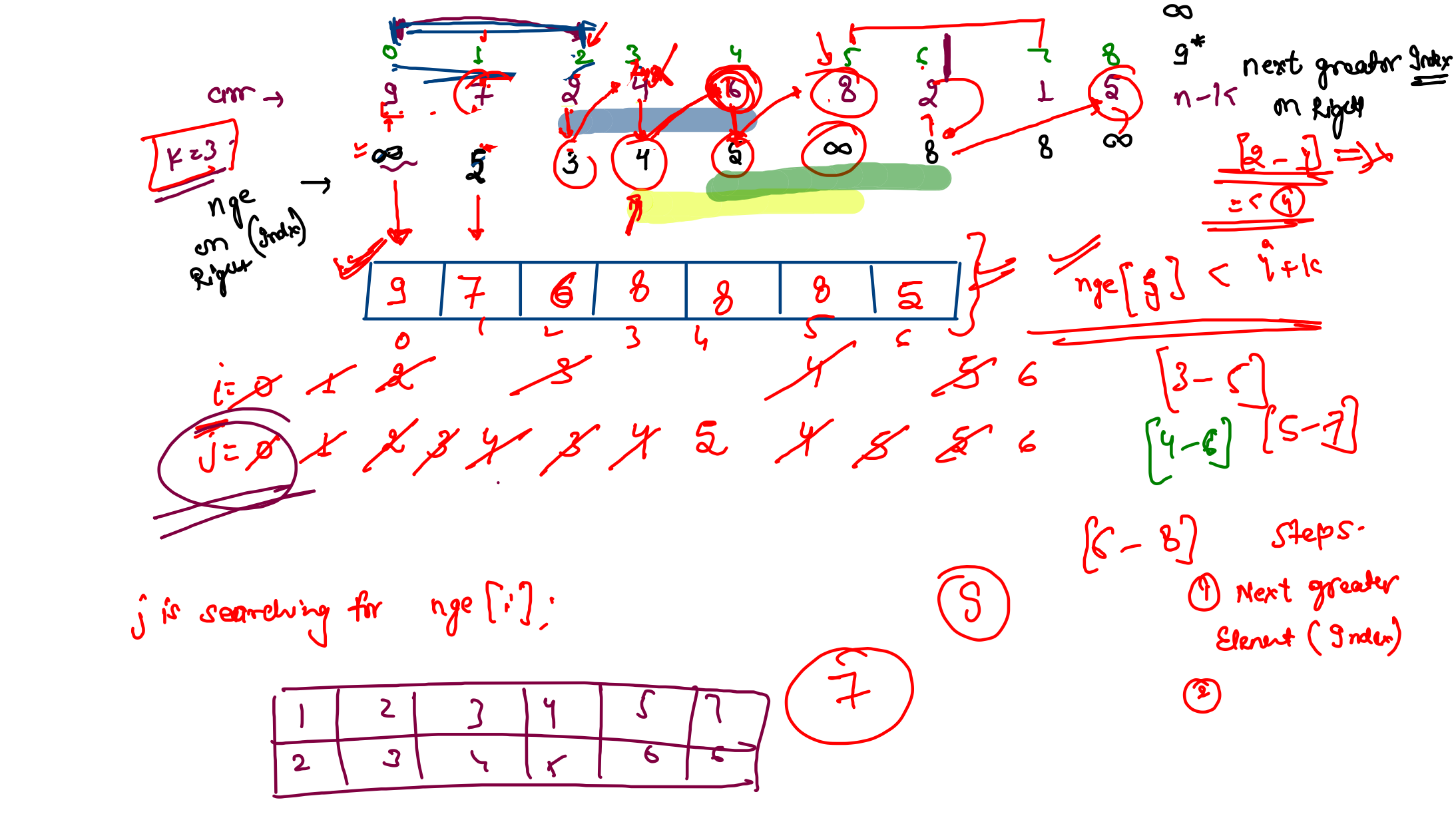


A-Sliding window

0-3 → 9
1-4 → 9
2-5 → 8
3-6 → 12
4-7 → 12
5-8 → 14
6-9 → 14
7-10 → 32
8-11 → 32
9-12 → 32
10-13 → 32
11-14 → 19
12-15 → 19
13-16 → 19

Max in k size window

9 9 8 12 12 14 14 32 32 32 32 19 19


arr →
K=3
nge on Right (index)
9 7 6 8 8 8 5
0 1 2 3 4 5 6
nge[j] < i+k
[3-5]
[4-6]
[5-7]
[6-8]
Steps.
① Next greater Element (Index)
②
j is searching for nge[i].
9
7
1 2 3 4 5 7
2 3 4 5 6 5