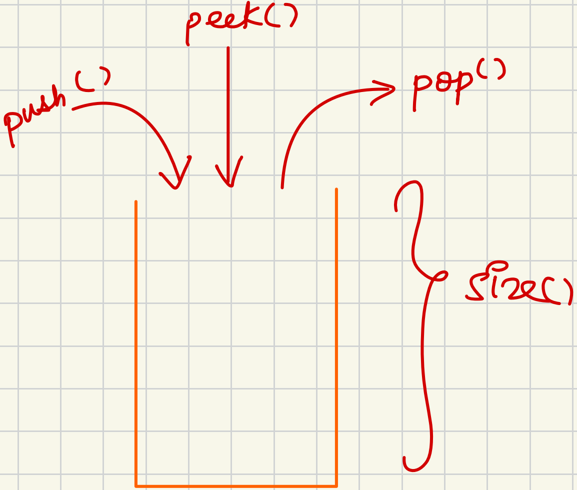
Stack.

- using standard Library
- using our own stack class



Stacks

Last in, first out
(LIFO)

Agenda

1. Extra Brackets
2. Next greater element
3. Stock Span
4. Largest area histogram

Extra Brackets.

valid exp.

string str = "(a + b)"   No Extra Bracket-

= "((a+b))"   Extra Bracket.

$$\downarrow$$

$$\boxed{(a+b}$$

NOTE: a pair of bracket is useful, when it have a new exp } inside it

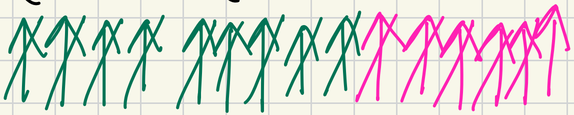
$exp = $ "$(a+b) * (d) + (m+n/(o*r))$" ✓ No Extra Bracket.

$exp = $ "$((a) + (b))$" No Extra Bracket.

$$exp = ((a+b) + d * (e + (f)) \backslash \underset{X}{(}(g)))$$

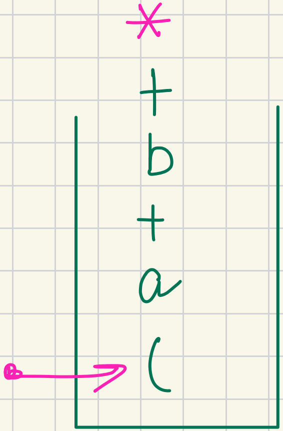
Extra Bracket Pair

str = "(a+b+(d+e)*(f))"

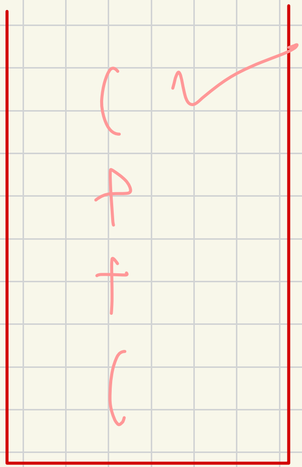


{LIFO}

- Last bracket to be opened, is the first one to be closed



$exp = $ "$((a)+(b+d)+((e+f)))$"



Extra Bracket

$O(N^2)$

```
for (int i = 0 → n)
    for (int j = 0 → n)
```

$$\begin{array}{cccccc} & i=0 & , & 1 & , \dots\dots\dots & N-1 \\ & \downarrow & & \downarrow & & \downarrow \\ = \sum & N & & N & & N = \checkmark \boxed{N^2} \end{array}$$

for (int i = 0 → n)

while ( )

$i = 0 \quad , \quad 1 \quad , \quad 2 \quad . \; . \; . \; . \; . \; . \quad n-1$

$\sum \quad a \quad b \quad c \quad \alpha$

$TC : \; a + b + c \; . \; . \; . \; . \; . \; . \; . \; . + \alpha \le N$

✓ $\boxed{TC : O(N)}$ $\boxed{SC : O(N)}$

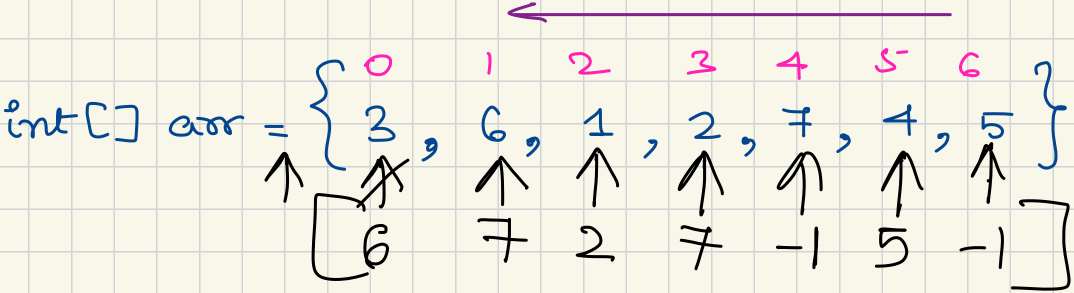
Next Greater Element On Right

$$\begin{array}{lccccccc} & 0 & 1 & 2 & 3 & 4 & 5 & 6 \\ \text{int[] arr} = \{ & 3, & 6, & 1, & 2, & 7, & 4, & 5 \; \} \end{array}$$

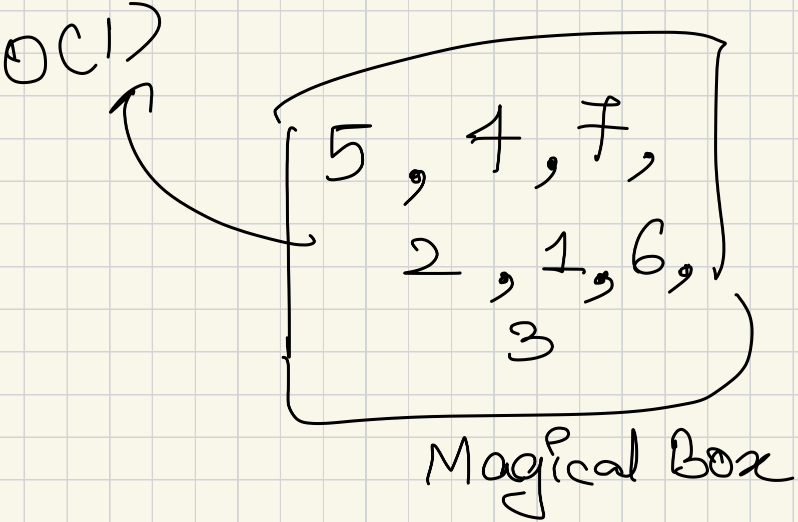
$$\text{nger[]} = \{ 6, 7, 2, 7, -1, 5, -1 \}$$

Brute force

$$\left.\begin{array}{l} TC : O(N^2) \\ SC : O(1) \end{array}\right\}$$



```
int[] arr = {3, 6, 1, 2, 7, 4, 5}
```

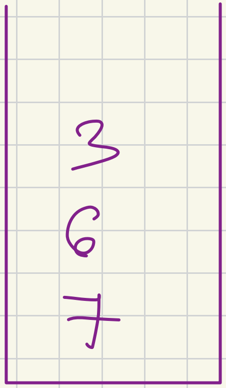


0 1 2 3 4 5 6

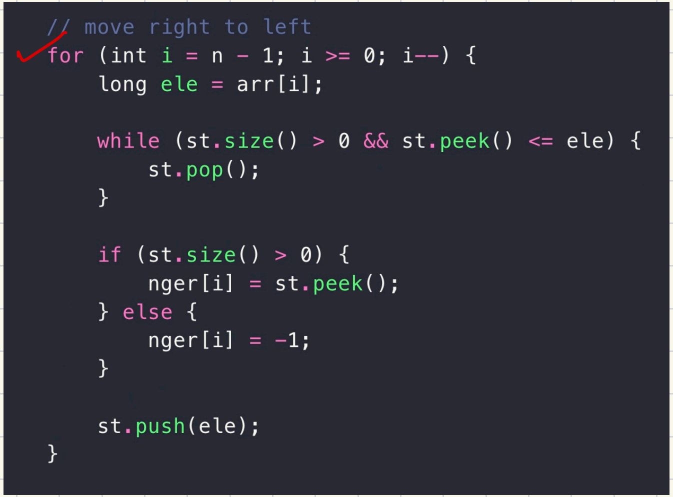
int[] arr = {3, 6, 1, 2, 7, 4, 5}

{6 7 2 7 -1 5 -1}

O(N)



potential Nger



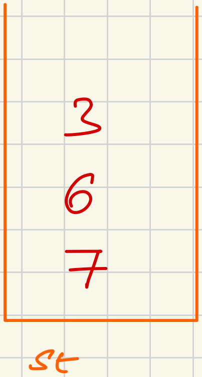
```
// move right to left
for (int i = n - 1; i >= 0; i--) {
    long ele = arr[i];

    while (st.size() > 0 && st.peek() <= ele) {
        st.pop();
    }

    if (st.size() > 0) {
        nger[i] = st.peek();
    } else {
        nger[i] = -1;
    }

    st.push(ele);
}
```

	0	1	2	3	4	5	6
int[] arr =	3	6	1	2	7	4	5
	6	7	2	7	-1	5	-1

↓

TC : O(N)

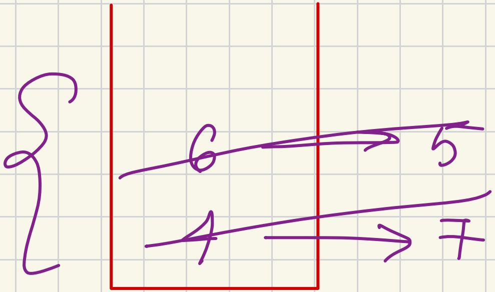
SC : O(N)



Approach_2 .

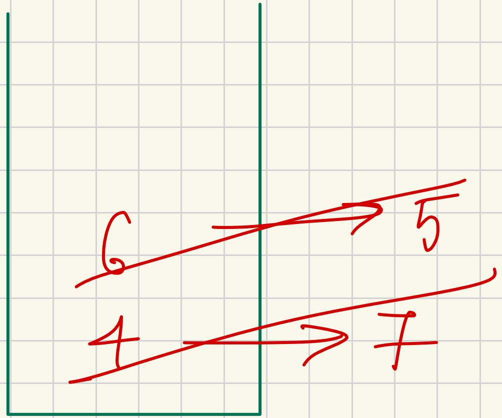
int[] arr = {3, 6, 1, 2, 7, 4, 5}

0	1	2	3	4	5	6
3	6	1	2	7	4	5
6	7	2	7	-1	5	-1



stack { people looking for nger }

0	1	2	3	4	5	6
{3,	6,	4,	2,	7,	1,	5}
6	7	7	7	-1	5	-1



st {people Looking uger}

H.W.

1. Balanced Brackets
2. previous greater element
3. Dry Run Both Approaches discussed today.