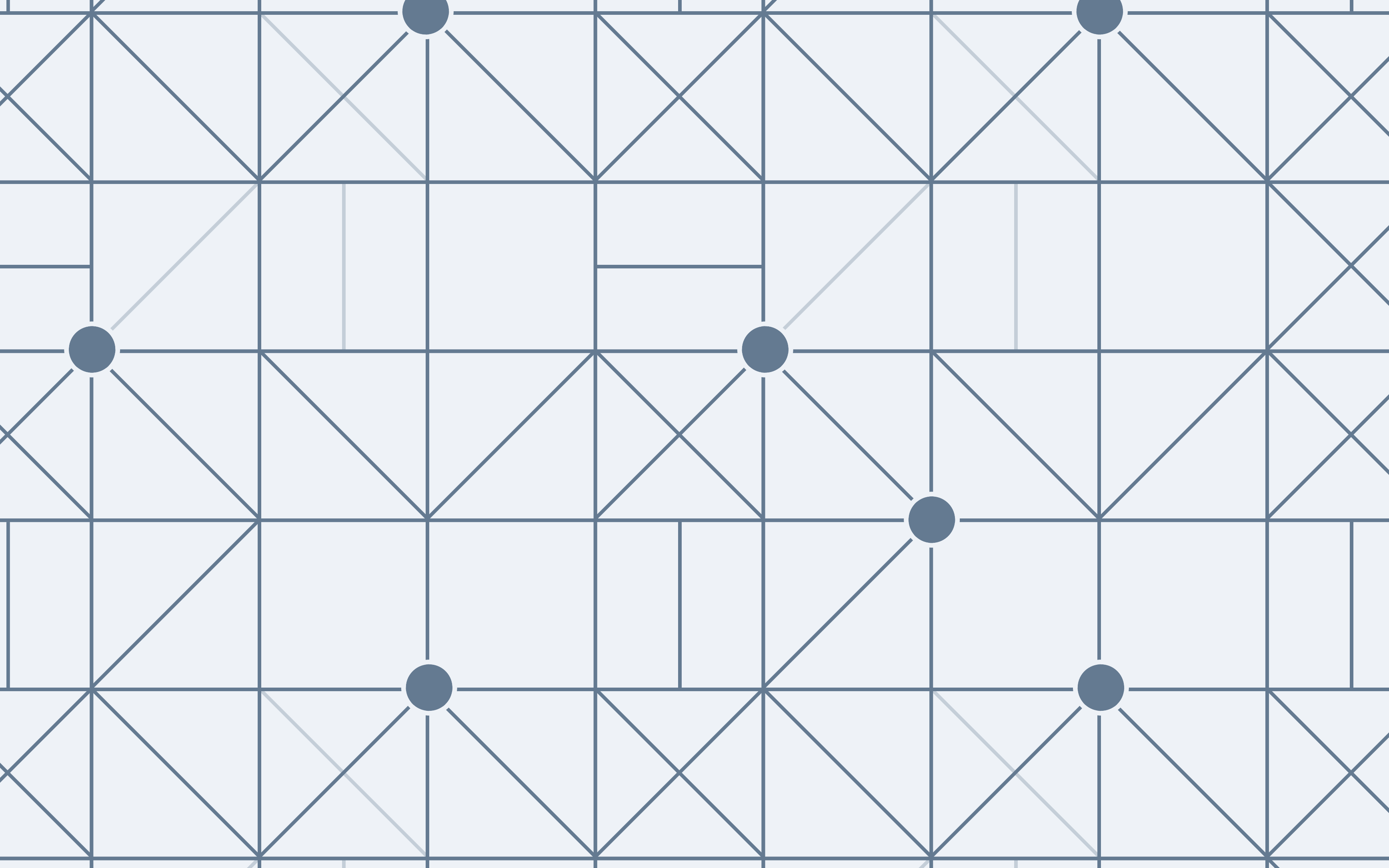
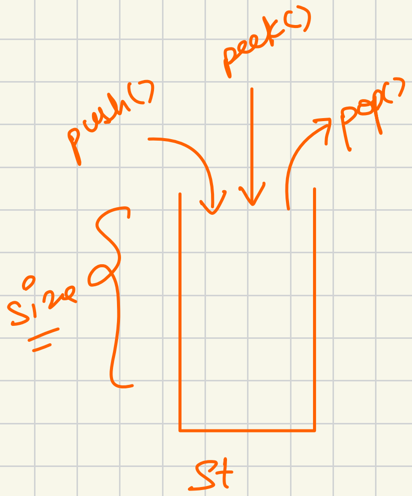


create a stack → using standard library / using our own Stack class



linear DS.
follows LIFO

each method TC : O(1)

Agenda :

1. Extra Brackets
2. Next greater element
3. stock span
4. Largest area histogram

Extra Brackets.

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

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

NOTE: a pair of bracket is useful, when you have new expression inside it.

= "(a+b)*(d+e+(f*h)/(i))" Extra Bracket.

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

• Balanced Pair

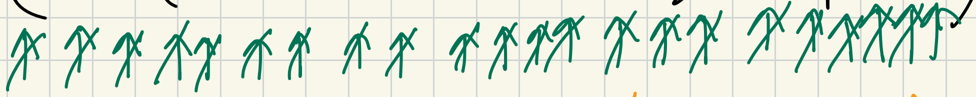
( ) ✓

) (✗

((✗

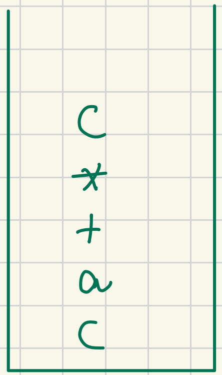
)) ✗

str = "(a + (b * d - f + (m) + o) * (p) ( ))"



(last opened bracket, is first to be closed)

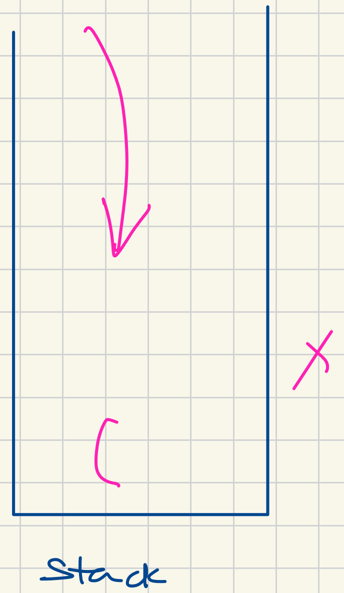
- you should have a unique expression betw your open and closed bracket. }



$\boxed{\text{NO}}$ ✓

↳ Extra Pair of Brackets.

string str = "(((a+b)+(d)))"



Extra Bracket Pair!

Time Complexity

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

$$\begin{array}{ccccc} i = 0, & 1, & 2, & \dots & N-1 \\ \downarrow & \downarrow & \downarrow & & \downarrow \\ \alpha & \beta & \gamma & - - - - - - - & \phi \\ \sum O(N) + & O(N) + & O(N) + & & + \; O(N) \end{array}$$

$$= O(N) \times N = \underline{\underline{O(N^2)}}$$

```
for (int i = 0 ⟶ N)
{
    $\boxed{\text{while ( )}}$ ⟶ in life time O(N) time.
}
```

At max
N – push instead
N → pop at max

$i = 0, 1, 2, \ldots \ldots (N-1)$

$$\begin{tikzcd} 0 \arrow[d] & 1 \arrow[d] & 2 \arrow[d] & (N-1) \arrow[d] \\ \alpha & \beta & \gamma & \phi \end{tikzcd}$$

$\sum \alpha + \beta + \gamma \ldots \phi$

$\sum$ 1 5 9

TC : $O(\alpha + \beta + \gamma + \ldots \ldots \phi) = N$

$\Rightarrow \underline{\underline{O(N)}}$ ✓

Next Greater Element on Right

$$\begin{array}{cccccccccc} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ \end{array}$$

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

$\uparrow$

int [] nger = { 6 , 7 , 2 , 7 , -1 , 4 , 5 , 2 , 5 , -1 }

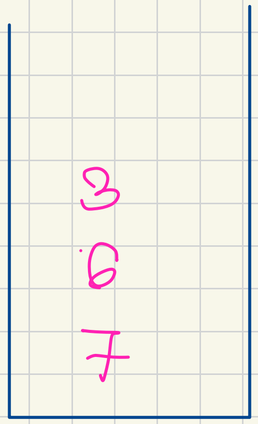
Brute force

```
for (int i = 0 → n)
{
    for (int j = i+1 → n)
        if (arr[j] > arr[i]) nger[i] = arr[j];
                             break.
}
```

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

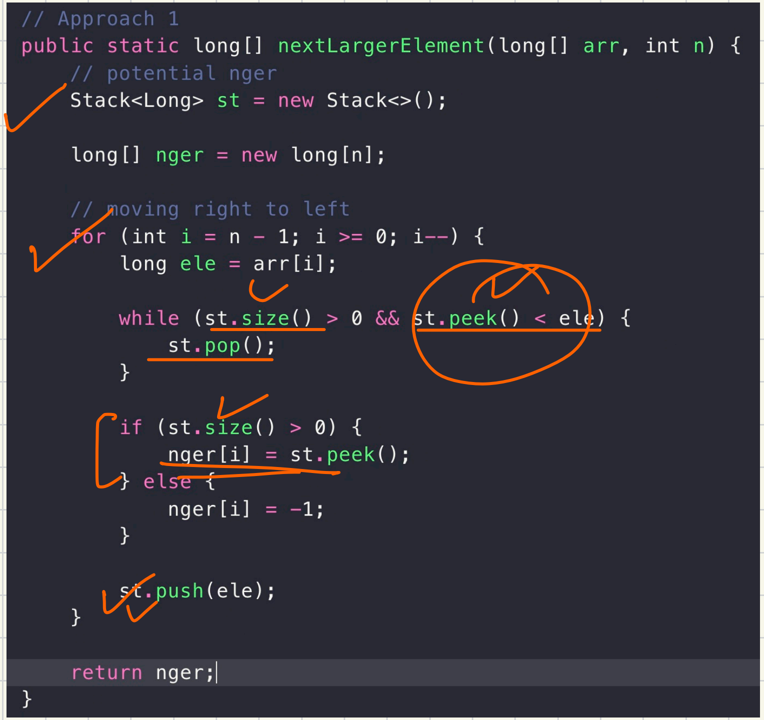
$$\begin{array}{lcccccccccc} & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ \text{int[] arr} = \{ & 3, & 6, & 1, & 2, & 7, & 3, & 4, & 1, & 2, & 5 \,\} \\ \uparrow & \cancel{\uparrow} & \cancel{\uparrow} & \cancel{\uparrow} & \cancel{\uparrow} & \cancel{\uparrow} & \cancel{\uparrow} & \cancel{\uparrow} & \cancel{\uparrow} & \cancel{\uparrow} & \cancel{\uparrow} \\ \{ & 6, & 7, & 2, & 7, & -1, & 4, & 5, & 2, & 5, & -1 \,\} \end{array}$$

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



stack {potential nger}

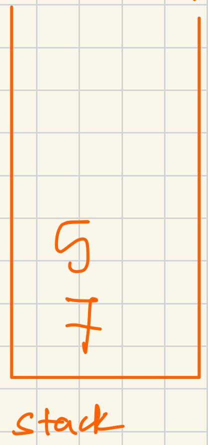
TC : O(N) , SC : O(N) ✓



```
// Approach 1
public static long[] nextLargerElement(long[] arr, int n) {
    // potential nger
    Stack<Long> st = new Stack<>();

    long[] nger = new long[n];

    // moving 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);
    }

    return nger;
}
```

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

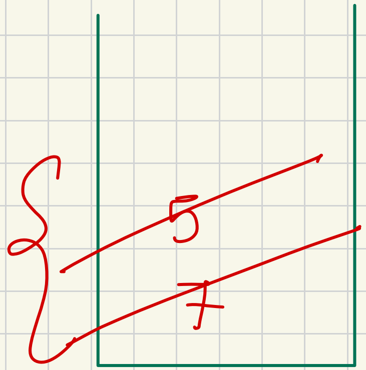
{7, 7, -1, 5, 5, -1}



Approach 2 :

$$\begin{array}{r|cccccccccc}
 & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\
\text{int[] arr} = & \{3, & 6, & 1, & 2, & 7, & 3, & 4, & 1, & 2, & 5\} \\
 & \cancel{\uparrow} & \cancel{\uparrow} & \cancel{\uparrow} & \cancel{\uparrow} & \cancel{\uparrow} & \cancel{\uparrow} & \cancel{\uparrow} & \cancel{\uparrow} & \cancel{\uparrow} & \cancel{\uparrow} \quad \uparrow \\
 & \{6, & 7, & 2, & 7, & -1, & 4, & 5, & 2, & 5, & -1\}
\end{array}$$

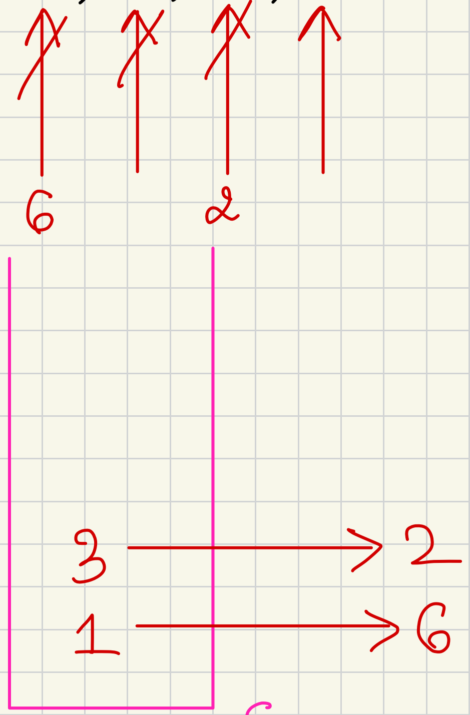
people looking for uger



stack {people looking for uger}

$$\begin{array}{cccccccccc} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ 3, & 6, & 1, & 2, & 7, & 3, & 4, & 1, & 2, & 5 \end{array}$$

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



stack

{people looking for nger}

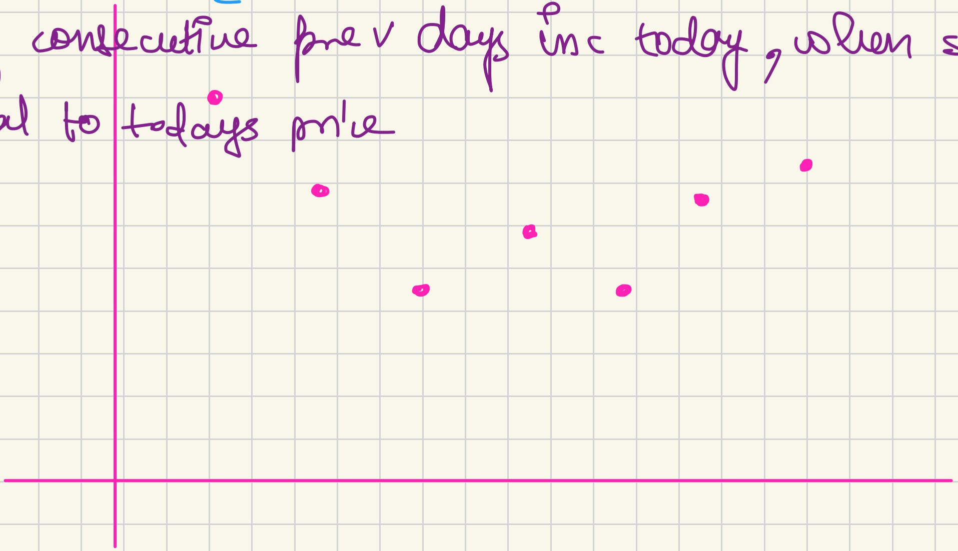
Monotonic Stack → Adv DSA

Homework
↳

Stock Span Problem.

$$\begin{array}{cccccccc} & 0 & 1 & 2 & 3 & 4 & 5 & 6 \\ \text{int[] arr} = \{ & 100, & 80, & 60, & 70, & 60, & 75, & 85 \} \\ \{ & 1, & 1, & 1, & 2, & 1, & 4, & 6 \} \end{array}$$

span: no. of consecutive prev days inc today, when stock price was less equal to todays price



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