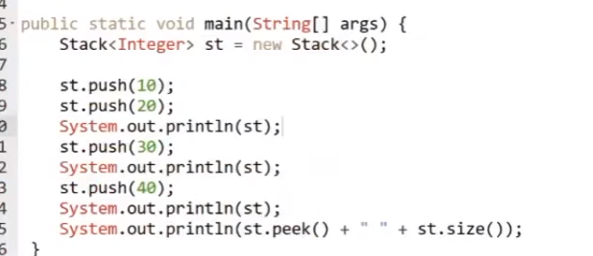
# ***Introduction to Stack & Queue IN JAVA***

1. In Stack we add the value at top and if we want to remove the value then we can remove using from TOP
2. Function in Stack is
   1. Push()
   2. Pop()
   3. Peek() – to check the top of the element
   4. Size()

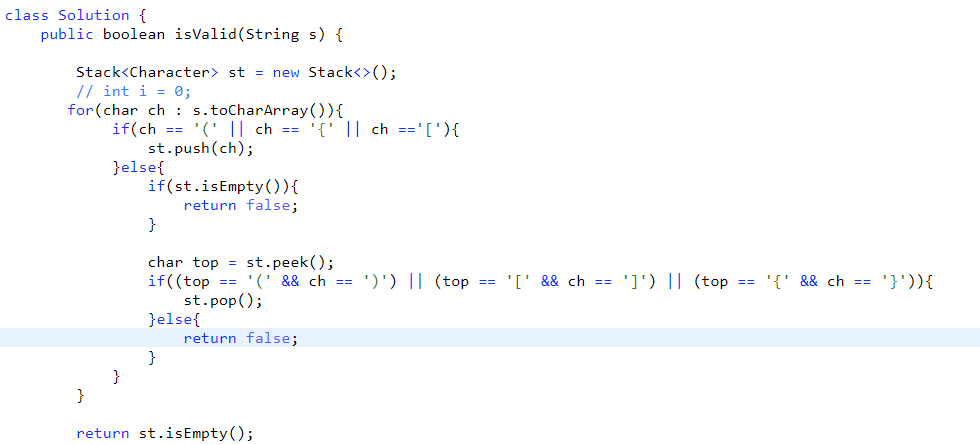
Stack<Integer> st;



**Question**: **Valid Parentheses**

**Input:** s = "()"

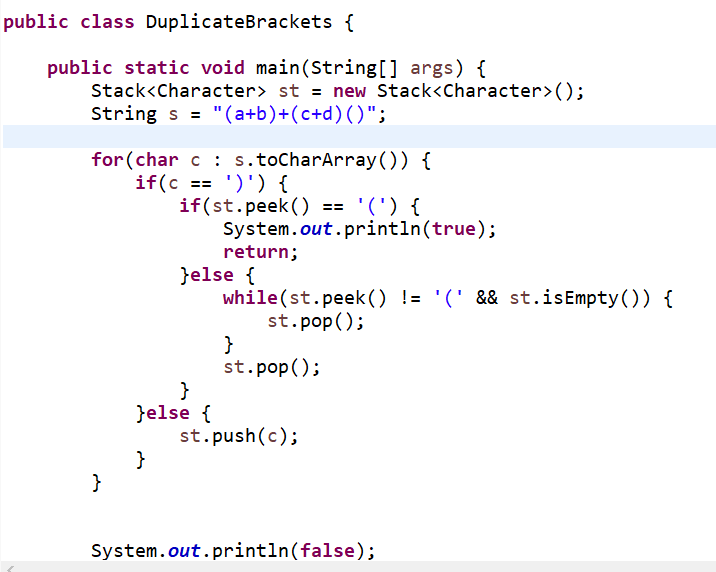
**Output:** true

**Solutions**:   


**Question** : Duplicate bracket in a String :

**String str = “(a+b)+(c+d))”**

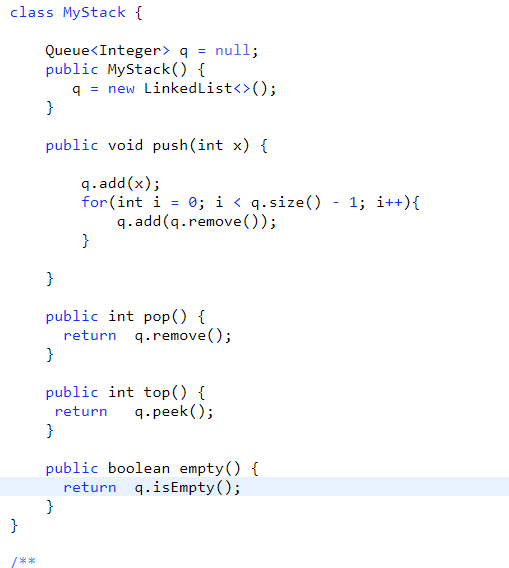
**Output : False**



**Question:** Balanced Bracket Solution:

Question Implement Stack using 2 queue

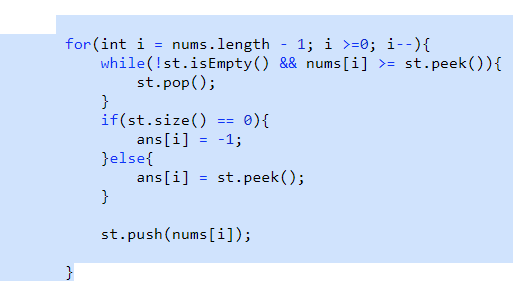
Solution:   
  
Instead of taking 2 queue we have add into the queue remove the same and add it again



Question: Next Greater Element from an array:

We will do three things to solve this problem

1. From the last we will start the iteration of the loop and for the last value we will find the greater element as -1
2. We will remove all the small value from the stack till we will able to find the greater value for that I^th value we will put greater value as result.
3. After finding the greater value push that into the stack !!



This will work for the non circular ARRAY

If we have circular array then we need to add each element into the stack and for every element we need to check the next greater element into the array

int [] ans = new int[nums.length];

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

for(int i = nums.length - 1; i >=0; i--){

st.push(nums[i]);

}

Question: Stock span problems

**Input**

["StockSpanner", "next", "next", "next", "next", "next", "next", "next"]

[[], [100], [80], [60], [70], [60], [75], [85]]

**Output**

[null, 1, 1, 1, 2, 1, 4, 6]

**Explanation**

StockSpanner stockSpanner = new StockSpanner();

stockSpanner.next(100); // return 1

stockSpanner.next(80); // return 1

stockSpanner.next(60); // return 1

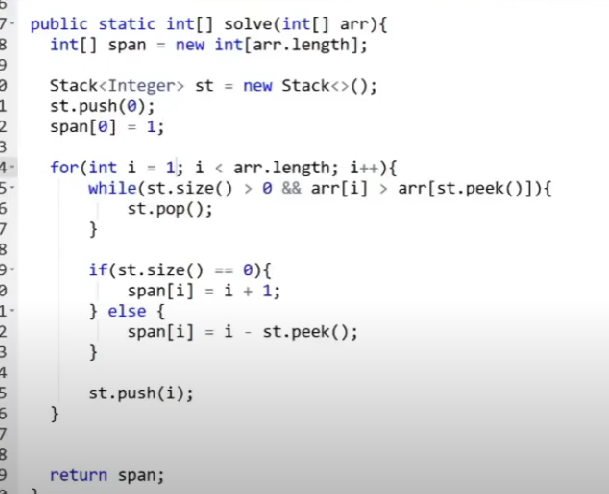
stockSpanner.next(70); // return 2

stockSpanner.next(60); // return 1

stockSpanner.next(75); // return 4, because the last 4 prices (including today's price of 75) were less than or equal to today's price.

stockSpanner.next(85); // return 6

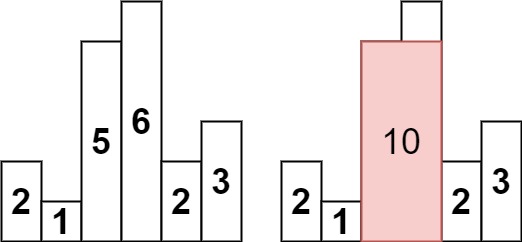
We need to check the span of the stock from right to left which is similar to as Next greater element we just have to modify the approach and find the span from the current ( idx -> NGE )



Question **Largest Rectangle in Histogram**

Given an array of integers heights representing the histogram's bar height where the width of each bar is 1, return *the area of the largest rectangle in the histogram*.

**Example 1:**



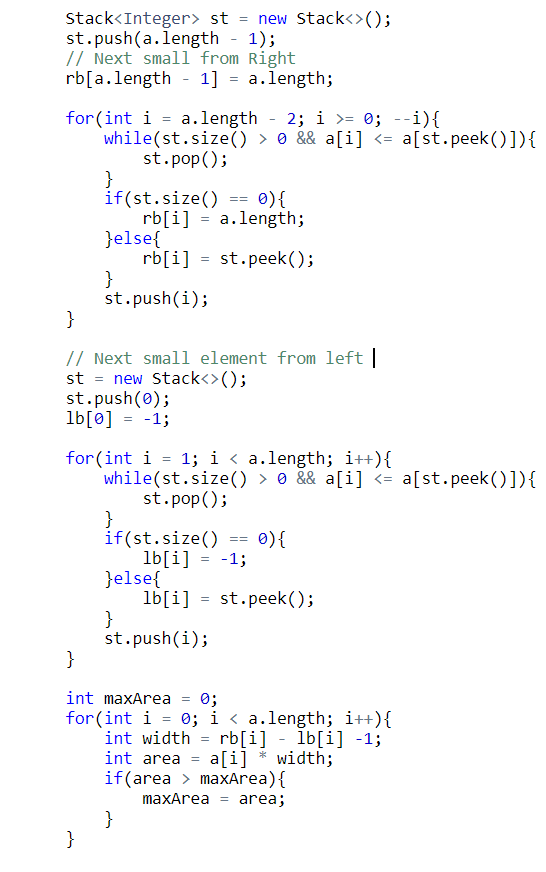
**Input:** heights = [2,1,5,6,2,3]

**Output:** 10

**Explanation:** The above is a histogram where width of each bar is 1.

The largest rectangle is shown in the red area, which has an area = 10 units.

Solution :



Approach is to solve this problem is :  
  
Step1 : We need to find the area at max where we can reach the dead end so we need to find the smallest value from both the side from left and from right side

Step2: Find the next smaller element from left and right from the stack if u found the greater element remove that if u found the element smaller then a[i] stop and update for the idx value that we have found the smallest for this element push the same into the stack !!

Run one loop from I -> 0 to n -1

Width = lb[i] –rb[i] -1 ;

Area = a[i] \* width;

maxArea = Math.max(maxArea,area);

return maxArea;

Question: Infix Evaluation in STACK !!

* ***Infix Notation:*** Operators are written between the operands they operate on, e.g. 3 + 4.
* ***Prefix Notation:***Operators are written before the operands, e.g + 3 4
* ***Postfix Notation:*** Operators are written after operands.
* s consists of integers and operators ('+', '-', '\*', '/') separated by some number of spaces.
* s represents **a valid expression**.
* All the integers in the expression are non-negative integers in the range [0, 231 - 1].
* The answer is **guaranteed** to fit in a **32-bit integer**.

If same priority we will solve the value from left only!!