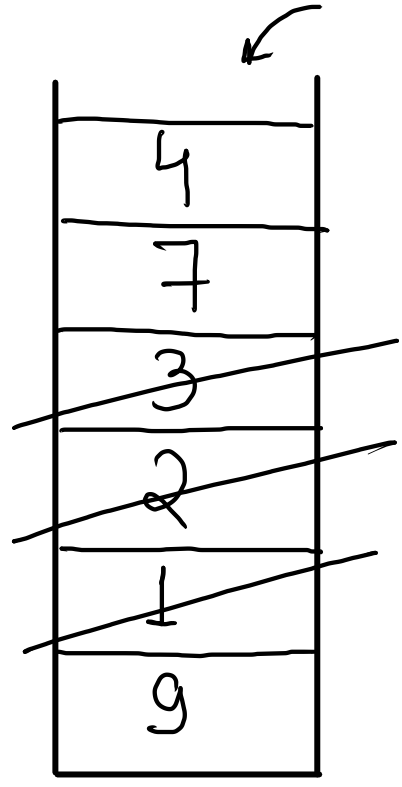
HW_Next greater element on left 1

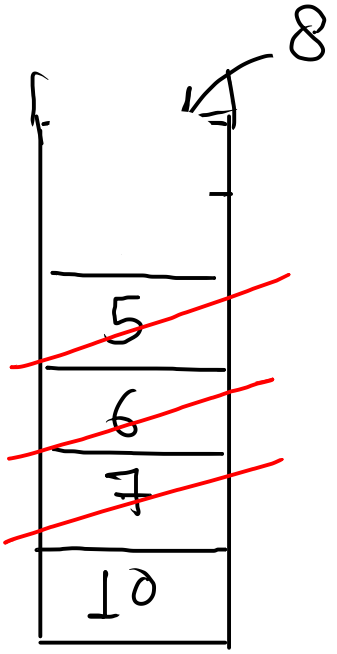
arr = [9 1 2 3 7 4]
0 1 2 3 4 5

ans = [-1 , 9 , 9 , 9 , 9 , 7]



arr = [10 , 7 , 6 , 5 , 8 , 3]
(0) (1) (2) (3) (4) (5)

ans = [-1 , 10 , 7 , 6 , 10 , 8]



psudo code

1) traverse in array

[1.1) loop until peek <= curr
pop

1.2) ans[i] = peek();

Code

```java
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    int n = scn.nextInt();
    int[] arr = new int[n];
    for (int i = 0; i < n; i++) {
        arr[i] = scn.nextInt();
    }
    int[] ans = nextGreaterOnLeft(arr, n);
    for (int i : ans) {
        System.out.print(i + " ");
    }
}
public static int[] nextGreaterOnLeft(int[] arr, int n) {
    Stack<Integer> st = new Stack<>();
    int[] ans = new int[n];
    for (int i = 0; i < n; i++) {
        while ( st.size() > 0 && st.peek() <= arr[i] ) {
            st.pop();
        }
        if (st.size() == 0) {
            ans[i] = -1;
        } else {
            ans[i] = st.peek();
        }
        st.push(arr[i]);
    }
    return ans;
}
```

T.C = O(n)

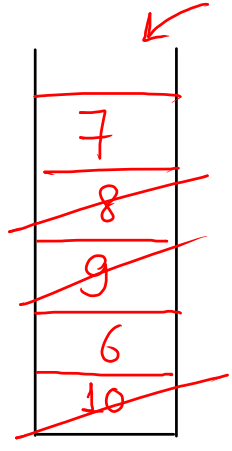
S.C = O(n)

Next Smaller Element To The Right

```java
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    int n = scn.nextInt();
    int[] arr = new int[n];
    for (int i = 0; i < n; i++) {
        arr[i] = scn.nextInt();
    }
    int[] ans = nextGreaterOnLeft(arr, n);
    for (int i : ans) {
        System.out.print(i + " ");
    }
}
public static int[] nextGreaterOnLeft(int[] arr, int n) {
    Stack<Integer> st = new Stack<>();
    int[] ans = new int[n];
    for (int i = n - 1; i >= 0; i--) {
        while ( st.size() > 0 && st.peek() >= arr[i] ) {
            st.pop();
        }
        if (st.size() == 0) {
            ans[i] = -1;
        } else {
            ans[i] = st.peek();
        }
        st.push(arr[i]);
    }
    return ans;
}
```

[6 , 6 , 6 , -1 , -1]
0 1 2 3 4

[7 , 8 , 9 , 6 , 10]



Note:-

1) next greater on left side :- normal

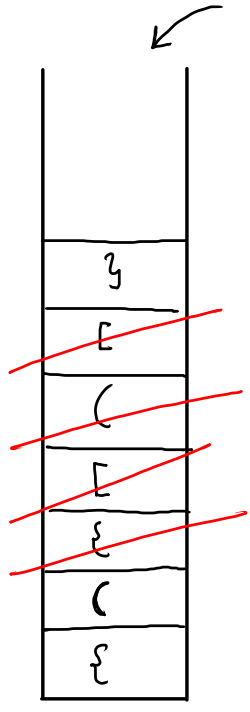
2) next greater on right side :- reverse for loop

3) next smaller on left side :- change sign in while loop

4) next smaller on right side :- do both

Valid Parentheses 4

str = "{ ({ [ ] } ([ ]) }" ;



condition

if curr ==] && peek == [

pop

if curr == } && peek == {

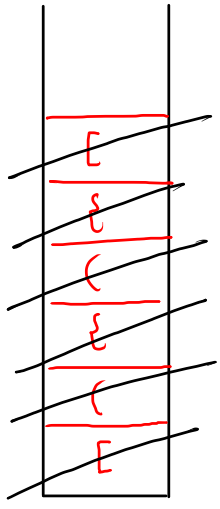
pop

if curr ==) && peek == (

pop

Ex:- str = "[({ ( ) { } } [ ])] ";

st.size() == 0



condition

if curr ==] && peek == [

pop

if curr == } && peek == {

pop

if curr ==) && peek == (

pop

Code

```java
public static boolean validParanthesis(String str) {
    Stack<Character> st = new Stack<>();
    for (int i = 0; i < str.length(); i++) {
        char curr = str.charAt(i);
        if ( curr == '(' || curr == '{' || curr == '[' ) {
            st.push(curr);
        } else {
            if (st.size() == 0) {
                return false;
            }
            if ( curr == ')' && st.peek() == '(' ) {
                st.pop();
            } else if ( curr == ']' && st.peek() == '[' ) {
                st.pop();
            } else if ( curr == '}' && st.peek() == '{' ) {
                st.pop();
            }
        }
    }
    if (st.size() == 0) {
        return true;
    } else {
        return false;
    }
}
```