DAY 6 DSA PRACTICE SET 1

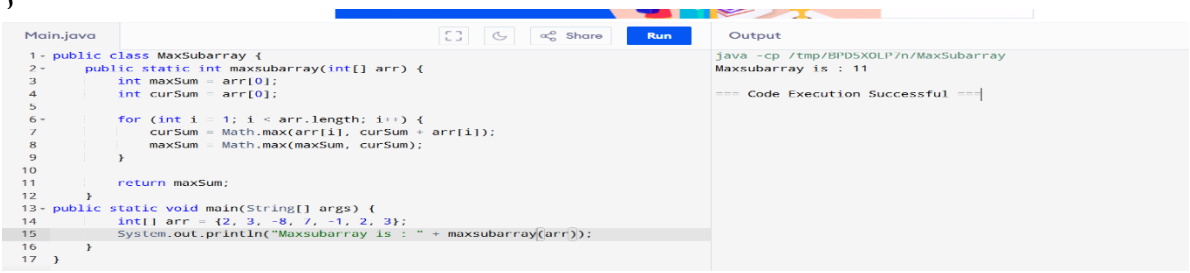
9/11/2024

1. **MAX SUB ARRAY:**

```
public class MaxSubarray {
   public static int maxSubArraySum(int[] arr) {
      int maxSum = arr[0];
      int curSum = arr[0];

      for (int i = 1; i < arr.length; i++) {
         curSum = Math.max(arr[i], curSum + arr[i]);
         maxSum = Math.max(maxSum, curSum);
      }

      return maxSum;
   }
public static void main(String[] args) {
      int[] arr = {2, 3, -8, 7, -1, 2, 3};
      System.out.println("Maximum Subarray Sum: " +
maxSubArraySum(arr));
   }
}
```



TIME COMPLEXITY: O(N)

2.

```
public class MaxProductSubarray {
   public static int maxProduct(int[] arr) {
      if (arr.length == 0) return 0;

      int maxProd = arr[0];
      int minProd = arr[0];
```

```
        int res= arr[0];

        for (int i = 1; i < arr.length; i++) {
            int cur = arr[i];

            if (cur < 0) {
                int temp = maxProd;
                maxProd = minProd;
                minProd = temp;
            }

            maxProd = Math.max(cur, maxProd * cur);
            minProd = Math.min(cur, minProd* cur);

            res= Math.max(res, maxProd);
        }

        return res;
    }

    public static void main(String[] args) {
        int[] arr1 = {-2, 6, -3, -10, 0, 2};
        System.out.println("Max prod: " + maxProduct(arr1));

    }
}
```

```
C:\Users\Admin\Desktop\day 6 coding>javac MaxProductSubarray.java

C:\Users\Admin\Desktop\day 6 coding>java MaxProductSubarray.java
Max prod: 180
```

TIME COMPLEXITY: O(n)

3.

```
import java.math.BigInteger;
public class Factorial {
    public static BigInteger factorial(int n) {
        BigInteger res = BigInteger.ONE;
```

```
        for (int i = 1; i <= n; i++) {
            res = res.multiply(BigInteger.valueOf(i));
        }
        return res;
    }

    public static void main(String[] args) {
        int n = 100;
        BigInteger fact = factorial(n);
        System.out.println(fact);
    }
}
```

```
C:\Users\Admin\Desktop\day 6 coding>javac Factorial.java

C:\Users\Admin\Desktop\day 6 coding>java Factorial.java
93326215443944152681699238856266700490715968264381621468592963895217599993229915608941463976156518286253697920827223758251185210916864000000000000000000000000
```

TIME COMPLEXITY: O(n)

4.
```
public class trappingwater{
    public static int trap(int[] arr){
        int n =arr.length;
        if(n==0) return 0;

        int lmax[]=new int[n];
        int rmax[]=new int[n];

        lmax[0]=arr[0];
        for(int i=1;i<n;i++){
            lmax[i]=Math.max(lmax[i-1],arr[i]);
        }
        rmax[n-1]=arr[n-1];
        for(int i=n-2;i>=0;i--){
            rmax[i]=Math.max(rmax[i+1],arr[i]);

        }
        int trapped=0;
        for(int i=0;i<n;i++){
```

```
        trapped+=Math.min(lmax[i],rmax[i])-arr[i];
      }
      return trapped;
    }
    public static void main(String[] args){
        int arr[]={3,0,1,0,4,0,2};
        System.out.println(trap(arr));
    }
}
```

```
C:\Users\Admin\Desktop\day 6 coding>javac trappingwater.java

C:\Users\Admin\Desktop\day 6 coding>java trappingwater.java
10

C:\Users\Admin\Desktop\day 6 coding>
```

TIME COMPLEXITY: O(n)

5.
```
public class containerwithmostwater {
    public static int maxarea(int[] arr) {
        int l = 0, r = arr.length - 1;
        int maxarea = 0;
        while (l < r) {
            int h = Math.min(arr[l], arr[r]);
            int w = r - l;
            int area = h * w;
            maxarea = Math.max(maxarea, area);

            if (arr[l] < arr[r]) {
                l++;
            } else {
                r--;
            }
        }
        return maxarea;
    }

    public static void main(String[] args) {
```

```
        int[] arr = {1, 5, 4, 3};
        System.out.println(maxarea(arr));
    }
}
```

```
C:\Users\Admin\Desktop\day 6 coding>javac containerwithmostwater.java

C:\Users\Admin\Desktop\day 6 coding>java containerwithmostwater.java
6

C:\Users\Admin\Desktop\day 6 coding>
```

TIME COMPLEXITY: O(n)

6.
```
import java.util.Arrays;
public class chocolatedistribution{
    public static int findmindiff(int[] arr,int m){
        Arrays.sort(arr);
        int mindiff=Integer.MAX_VALUE;
        for(int i=0;i<=arr.length-m;i++){
            int diff=arr[i+m-1]-arr[i];
            mindiff=Math.min(mindiff,diff);
        }
        return mindiff;
    }
    public static void main(String[] args){
        int choco[]={7,5,7,8,9,12,25,57};
        int student=3;
        int res=findmindiff(choco,student);
        System.out.println(res);
    }
}
```

```
C:\Users\Admin\Desktop\day 6 coding>javac chocolatedistribution.java

C:\Users\Admin\Desktop\day 6 coding>java chocolatedistribution.java
1

C:\Users\Admin\Desktop\day 6 coding>
```

TIME COMPLEXITY: O(nlogn)

7.
```
public class searchinrotatedarray {

    public static int search(int[] arr, int key) {
        int low = 0, high = arr.length - 1;

        while (low <= high) {
            int mid = low + (high - low) / 2;

            if (arr[mid] == key) {
                return mid;
            }

            if (arr[low] <= arr[mid]) {
                if (arr[low] <= key && key < arr[mid]) {
                    high = mid - 1;
                } else {
                    low = mid + 1;
                }
            } else {
                if (arr[mid] < key && key <= arr[high]) {
                    low = mid + 1;
                } else {
                    high = mid - 1;
                }
            }
        }

        return -1;
    }

    public static void main(String[] args) {
        int[] arr = {4, 5, 6, 7, 0, 1, 2};
        System.out.println(search(arr, 0));
        System.out.println(search(arr, 3));


    }
```

}

```
C:\Users\Admin>cd C:\Users\Admin\Desktop\day 6 coding

C:\Users\Admin\Desktop\day 6 coding>javac searchinrotatedarray.java

C:\Users\Admin\Desktop\day 6 coding>java searchinrotatedarray.java
4
-1
```

TIME COMPLEXITY: O(n)

8.
```
import java.util.*;

public class MergeIntervals {

    public static int[][] merge(int[][] intervals) {
        if (intervals.length <= 1) {
            return intervals;
        }

        Arrays.sort(intervals, (a, b) -> Integer.compare(a[0], b[0]));

        List<int[]> result = new ArrayList<>();
        result.add(intervals[0]);

        for (int i = 1; i < intervals.length; i++) {
            int[] lastInt = result.get(result.size() - 1);
            int[] curInt = intervals[i];

            if (lastInt[1] >= curInt[0]) {
                lastInt[1] = Math.max(lastInt[1], curInt[1]);
            } else {
                result.add(curInt);
            }
        }

        return result.toArray(new int[result.size()][]);
    }
```

```
    public static void main(String[] args) {
        int[][] intervals1 = {{1, 3}, {2, 4}, {6, 8}, {9, 10}};
        System.out.println(Arrays.deepToString(merge(intervals1)));
       }
}
```

```
C:\Users\Admin\Desktop\day 6 coding>cd C:\Users\Admin\Desktop\day 6 coding

C:\Users\Admin\Desktop\day 6 coding>java mergeintervals.java
[[1, 4], [6, 8], [9, 10]]

C:\Users\Admin\Desktop\day 6 coding>
```

TIME COMPLEXITY: O(nlogn)

9.
```
public class BooleanMatrix {

    public static void modifyMatrix(int[][] matrix) {
        int rows = matrix.length;
        int cols = matrix[0].length;

        boolean[] rowFlags = new boolean[rows];
        boolean[] colFlags = new boolean[cols];

        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < cols; j++) {
                if (matrix[i][j] == 1) {
                    rowFlags[i] = true;
                    colFlags[j] = true;
                }
            }
        }

        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < cols; j++) {
                if (rowFlags[i] || colFlags[j]) {
                    matrix[i][j] = 1;
                }
            }
        }
```

```
    }

    public static void printMatrix(int[][] matrix) {
        for (int i = 0; i < matrix.length; i++) {
            for (int j = 0; j < matrix[0].length; j++) {
                System.out.print(matrix[i][j] + " ");
            }
            System.out.println();
        }
    }

    public static void main(String[] args) {
        int[][] matrix1 = {{1, 0}, {0, 0}};
        modifyMatrix(matrix1);
        printMatrix(matrix1);

        int[][] matrix2 = {{0, 0, 0}, {0, 0, 1}};
        modifyMatrix(matrix2);
        printMatrix(matrix2);
            }
}
```

```
C:\Users\Admin\Desktop\day 6 coding>cd C:\Users\Admin\Desktop\day 6 coding

C:\Users\Admin\Desktop\day 6 coding>java booleanmatrix.java
1 1
1 0
0 0 1
1 1 1

C:\Users\Admin\Desktop\day 6 coding>
```

TIME COMPLEXITY: O(m*n)

10. public class Spiralmatrix{

```
    public static void printSpiral(int[][] matrix) {
        int top = 0, bottom = matrix.length - 1, left = 0, right = matrix[0].length -
1;
```

```
    while (top <= bottom && left <= right) {
        //  top row
        for (int i = left; i <= right; i++) {
            System.out.print(matrix[top][i] + " ");
        }
        top++;

        // right column
        for (int i = top; i <= bottom; i++) {
            System.out.print(matrix[i][right] + " ");
        }
        right--;

        //  bottom row
        if (top <= bottom) {
            for (int i = right; i >= left; i--) {
                System.out.print(matrix[bottom][i] + " ");
            }
            bottom--;
        }

        // left column
        if (left <= right) {
            for (int i = bottom; i >= top; i--) {
                System.out.print(matrix[i][left] + " ");
            }
            left++;
        }
    }
}

public static void main(String[] args) {
    int[][] matrix1 = {{1, 2, 3, 4},
                       {5, 6, 7, 8},
                       {9, 10, 11, 12},
                       {13, 14, 15, 16}};
    printSpiral(matrix1);
    System.out.println();
```

```
    }
}
```

```
C:\Users\Admin\Desktop\day 6 coding>cd C:\Users\Admin\Desktop\day 6 coding

C:\Users\Admin\Desktop\day 6 coding>java spiralmatrix.java
1 2 3 4 8 12 16 15 14 13 9 5 6 7 11 10

C:\Users\Admin\Desktop\day 6 coding>
```

TIME COMPLEXITY: O(m*n)

11.
```
public class parenthesesbalance {
    public static String isBalanced(String input) {
        int cnt = 0;

        for (int i = 0; i < input.length(); i++) {
            char ch = input.charAt(i);

            if (ch == '(') {
                cnt++;
            } else if (ch == ')') {
                cnt--;
            }

            if (cnt < 0) {
                return "Not Balanced";
            }
        }

        return cnt == 0 ? "Balanced" : "Not Balanced";
    }

    public static void main(String[] args) {
        String str1 = "((()))()()";
        String str2 = "())((())";

        System.out.println(isBalanced(str1));
        System.out.println(isBalanced(str2));
```

```
    }
}
```

```
C:\Users\Admin\Desktop\day 6 coding>javac parenthesesbalance.java

C:\Users\Admin\Desktop\day 6 coding>java parenthesesbalance.java
Balanced
Not Balanced
```

TIME COMPLEXITY: O(n2)

12.
```
import java.util.HashMap;

public class anagramcheck {
    public static boolean checkAnagrams(String str1, String str2) {
        if (str1.length() != str2.length()) {
            return false;
        }

        HashMap<Character, Integer> charCount = new HashMap<>();

        for (char ch : str1.toCharArray()) {
            charCount.put(ch, charCount.getOrDefault(ch, 0) + 1);
        }

        for (char ch : str2.toCharArray()) {
            if (!charCount.containsKey(ch)) {
                return false;
            }
            charCount.put(ch, charCount.get(ch) - 1);
            if (charCount.get(ch) == 0) {
                charCount.remove(ch);
            }
        }

        return charCount.isEmpty();
    }

    public static void main(String[] args) {
```

```
        String str1 = "geeks";
        String str2 = "kseeg";

        System.out.println(checkAnagrams(str1, str2)); // Output: true

        str1 = "allergy";
        str2 = "allergic";
        System.out.println(checkAnagrams(str1, str2)); // Output: false
    }
}
```

```
C:\Users\Admin\Desktop\day 6 coding>javac anagramcheck.java

C:\Users\Admin\Desktop\day 6 coding>java anagramcheck.java
true
false

C:\Users\Admin\Desktop\day 6 coding>
```

13.
```
public class longpalindromesubstring {

    public static String longPalin(String s) {
        if (s == null || s.length() < 1) {
            return "";
        }

        String res = "";

        for (int i = 0; i < s.length(); i++) {
            String odd = expand(s, i, i);
            if (odd.length() > res.length()) {
                res = odd;
            }

            String even = expand(s, i, i + 1);
            if (even.length() > res.length()) {
                res = even;
            }
        }
```

```
        return res;
    }

    private static String expand(String s, int l, int r) {
        while (l >= 0 && r < s.length() && s.charAt(l) == s.charAt(r)) {
            l--;
            r++;
        }
        return s.substring(l + 1, r);
    }

    public static void main(String[] args) {
        String a = "forgeeksskeegfor";
        System.out.println(longPalin(a));

        String b = "abc";
        System.out.println(longPalin(b));

        String c = "";
        System.out.println(longPalin(c));
    }
}
```

```
C:\Users\Admin\Desktop\day 6 coding>javac longpalindromesubstring.java

C:\Users\Admin\Desktop\day 6 coding>java longpalindromesubstring.java
geeksskeeg
a


C:\Users\Admin\Desktop\day 6 coding>
```

13. import java.util.Arrays;

```
public class LongestCommonPrefix {

    public static String lcp(String[] a) {
        if (a == null || a.length == 0) {
            return "-1";
        }
```

```
        Arrays.sort(a);

        String x = a[0];
        String y = a[a.length - 1];

        int i = 0;
        while (i < x.length() && i < y.length() && x.charAt(i) == y.charAt(i)) {
            i++;
        }

        String z = x.substring(0, i);
        return z.isEmpty() ? "-1" : z;
    }

    public static void main(String[] args) {
        String[] b = {"geeksforgeeks", "geeks", "geek", "geezer"};
        System.out.println(lcp(b));
        String[] c = {"hello", "world"};
        System.out.println(lcp(c));    }
}
```

```
C:\Users\Admin\Desktop\day 6 coding>java LongestCommonPrefix.java
gee
-1

C:\Users\Admin\Desktop\day 6 coding>
```

14.
```
import java.util.Stack;

public class DeleteMiddleElement {

    public static void deleteMiddle(Stack<Integer> stack, int size, int current) {
        if (stack.isEmpty() || current == size / 2) {
            stack.pop();
            return;
        }

        int temp = stack.pop();
```

```
        deleteMiddle(stack, size, current + 1);
        stack.push(temp);
    }

    public static void deleteMiddleElement(Stack<Integer> stack) {
        int size = stack.size();
        if (size == 0) {
            return;
        }
        deleteMiddle(stack, size, 0);
    }

    public static void main(String[] args) {
        Stack<Integer> stack1 = new Stack<>();
        stack1.push(1);
        stack1.push(2);
        stack1.push(3);
        stack1.push(4);
        stack1.push(5);

        deleteMiddleElement(stack1);
        System.out.println(stack1);
        Stack<Integer> stack2 = new Stack<>();
        stack2.push(1);
        stack2.push(2);
        stack2.push(3);
        stack2.push(4);
        stack2.push(5);
        stack2.push(6);

        deleteMiddleElement(stack2);
        System.out.println(stack2);     }
}
```

```
C:\Users\Admin\Desktop\day 6 coding>java DeleteMiddleElement.java
[1, 2, 4, 5]
[1, 2, 4, 5, 6]

C:\Users\Admin\Desktop\day 6 coding>
```

15.
```
import java.util.Stack;

public class nextgreaterelement {

    public static void printNGE(int[] arr) {
        Stack<Integer> stack = new Stack<>();
        int n = arr.length;

        for (int i = 0; i < n; i++) {
            while (!stack.isEmpty() && arr[stack.peek()] < arr[i]) {
                int index = stack.pop();
                System.out.println(arr[index] + " --> " + arr[i]);
            }
            stack.push(i);
        }

        while (!stack.isEmpty()) {
            int index = stack.pop();
            System.out.println(arr[index] + " --> -1");
        }
    }

    public static void main(String[] args) {
        int[] arr1 = {4, 5, 2, 25};
        printNGE(arr1);

        int[] arr2 = {13, 7, 6, 12};
        printNGE(arr2);
    }
```

}

```
C:\Users\Admin\Desktop\day 6 coding>java nextgreaterelement.java
4 --> 5
2 --> 25
5 --> 25
25 --> -1
6 --> 12
7 --> 12
12 --> -1
13 --> -1

C:\Users\Admin\Desktop\day 6 coding>
```

16. import java.util.*;

```
class Node {
    int data;
    Node left, right;

    public Node(int data) {
        this.data = data;
        left = right = null;
    }
}

public class BinaryTreeRightView {

    public static void rightView(Node root) {
        if (root == null) {
            return;
        }

        Queue<Node> queue = new LinkedList<>();
        queue.add(root);

        while (!queue.isEmpty()) {
            int nodeCount = queue.size();

            for (int i = 1; i <= nodeCount; i++) {
```

```
            Node node = queue.poll();


            if (i == nodeCount) {
                System.out.print(node.data + " ");
            }


            if (node.left != null) {
                queue.add(node.left);
            }
            if (node.right != null) {
                queue.add(node.right);
            }
        }
    }
}

public static void main(String[] args) {
    Node root = new Node(1);
    root.left = new Node(2);
    root.right = new Node(3);
    root.left.left = new Node(4);
    root.left.right = new Node(5);
    root.right.right = new Node(6);
    root.left.left.left = new Node(7);

    System.out.print("Right View: ");
    rightView(root);
}
}
```

```
C:\Users\Admin\Desktop\day 6 coding>javac BinaryTreeRightView.java

C:\Users\Admin\Desktop\day 6 coding>java BinaryTreeRightView
Right View: 1 3 6 7
C:\Users\Admin\Desktop\day 6 coding>
```

17.
```
class Node {
   int data;
   Node left, right;

   public Node(int data) {
      this.data = data;
      left = right = null;
   }
}

public class BinaryTreeHeight {

   public static int maxDep(Node root) {
      if (root == null) {
         return 0;
      }
      int ld = maxDep(root.left);
      int rd = maxDep(root.right);
      return Math.max(ld, rd) + 1;
   }

   public static void main(String[] args) {
      Node root = new Node(1);
      root.left = new Node(2);
      root.right = new Node(3);
      root.left.left = new Node(4);
      root.left.right = new Node(5);
      root.left.left.left = new Node(6);

      System.out.println("Maximum Depth or Height of Binary Tree: " +
maxDep(root));
   }
}
```

```
C:\Users\Admin\Desktop\day 6 coding>java BinaryTreeHeight
Maximum Depth or Height of Binary Tree: 4

C:\Users\Admin\Desktop\day 6 coding>
```