

Coding Practice Problems

1)

Maximum Subarray Sum – Kadane's Algorithm:

Given an array `arr[]`, the task is to find the subarray that has the maximum sum and return its sum.

Input: `arr[] = {2, 3, -8, 7, -1, 2, 3}`

Output: 11

Explanation: The subarray `{7, -1, 2, 3}` has the largest sum 11.

Input: `arr[] = {-2, -4}`

Output: -2

Explanation: The subarray `{-2}` has the largest sum -2.

Input: `arr[] = {5, 4, 1, 7, 8}`

Output: 25

Explanation: The subarray `{5, 4, 1, 7, 8}` has the largest sum 25

Code:

```
public class Main1 {
    public int maxSum(int[] arr) {
        if (arr == null || arr.length == 0) {
            return 0;
        }

        int maxSoFar = arr[0];
        int currMax = arr[0];
        for (int i = 1; i < arr.length; i++) {
            currMax = Math.max(arr[i], currMax + arr[i]);
            maxSoFar = Math.max(maxSoFar, currMax);
        }
        return maxSoFar;
    }
    public static void main(String[] args) {
        Main1 solution = new Main1();
        int[] arr = {2, 3, -8, 7, -1, 2, 3};
        System.out.println(solution.maxSum(arr));
    }
}
```

Output:

```
PS D:\Project\New folder> & 'C:\Program Files\Java\jdk-23\bin\java.exe' '-agentlib:jdwp=transport=dt_socket,server=n,suspend=y,address=localhost:5559' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\dhoni\AppData\Roaming\Code\User\workspaceStorage\db3b031b7c28ce48d67c8dc032990ec\redhat.java\jdt_ws\New_folder_a5c0e6id\bin' 'Main'
the sum of subarray: 11
PS D:\Project\New folder>
```

Time complexity: $O(N)$

2)

Maximum Product Subarray

Given an integer array, the task is to find the maximum product of any subarray.

Input: `arr[] = {-2, 6, -3, -10, 0, 2}`

Output: 180

Explanation: The subarray with maximum product is `{6, -3, -10}` with product = $6 * (-3) * (-10)$

= 180

Input: `arr[] = {-1, -3, -10, 0, 60}`

Output: 60

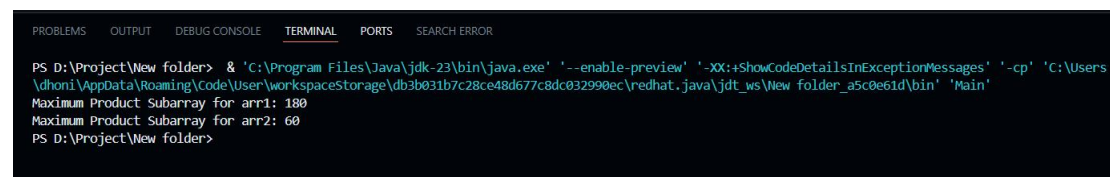
Explanation: The subarray with maximum product is {60}

Code:

```
public class Main2 {
    public int maxProd(int[] arr) {
        if (arr == null || arr.length == 0) {
            return 0;
        }

        int maxP = arr[0];
        int minP = arr[0];
        int res = arr[0];
        for (int i = 1; i < arr.length; i++) {
            int curr = arr[i];
            int tempMax = Math.max(curr, Math.max(maxP * curr, minP * curr));
            minP = Math.min(curr, Math.min(maxP * curr, minP * curr));
            maxP = tempMax;
            res = Math.max(res, maxP);
        }
        return res;
    }
    public static void main(String[] args) {
        Main2 solution = new Main2();
        int[] arr = {-2, 6, -3, -10, 0, 2};
        System.out.println("Maximum Product Subarray: " + solution.maxProd(arr));
    }
}
```

Output:



```
PS D:\Project\New folder> & 'C:\Program Files\Java\jdk-23\bin\java.exe' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\dhoni\AppData\Roaming\Code\User\workspaceStorage\db3b031b7c28ce48d677c8dc032990ec\redhat.java\jdt_ws\New folder_a5c0e61d\bin' 'Main'
Maximum Product Subarray for arr1: 180
Maximum Product Subarray for arr2: 60
PS D:\Project\New folder>
```

Time complexity: $O(N)$

3)

Search in a sorted and rotated Array

Given a sorted and rotated array `arr[]` of n distinct elements, the task is to find the index of given

key in the array. If the key is not present in the array, return -1.

Input : `arr[] = {4, 5, 6, 7, 0, 1, 2}`, `key = 0`

Output : 4

Code:

```
public class Main3 {
    public int search(int[] arr, int key) {
        int l = 0;
        int r = arr.length - 1;

        while (l <= r) {
            int m = l + (r - l) / 2;
            if (arr[m] == key) {
                return m;
            }
        }
    }
}
```

```

        if (arr[l] <= arr[m]) {
            if (key >= arr[l] && key < arr[m]) {
                r = m - 1;
            } else {
                l = m + 1;
            }
        } else {
            if (key > arr[m] && key <= arr[r]) {
                l = m + 1;
            } else {
                r = m - 1;
            }
        }
    }
    return -1;
}

public static void main(String[] args) {
    Main3 solution = new Main3();

    int[] arr1 = {4, 5, 6, 7, 0, 1, 2};
    int key1 = 0;
    System.out.println("Index of key1: " + solution.search(arr1, key1));
    int key2 = 3;
    System.out.println("Index of key2: " + solution.search(arr1, key2));
    int[] arr2 = {50, 10, 20, 30, 40};
    int key3 = 10;
    System.out.println("Index of key3: " + solution.search(arr2, key3));
}
}

```

Output

```

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  SEARCH ERROR

PS D:\Project\New folder> & 'C:\Program Files\Java\jdk-23\bin\java.exe' '-agentlib:jdwp=transport=dt_socket,server=n,suspend=y,address=localhost:58800' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\dhoni\AppData\Roaming\Code\User\workspaceStorage\db3b031b7c28ce48d677c8dc032990ec\redhat.java\jdt_ws\New folder_a5c0e61d\bin' 'Main'
Index of key1: 4
Index of key2: -1
Index of key3: 1
PS D:\Project\New folder>

```

Time Complexity: $O(\log N)$

4)

Container with Most Water

Given n non-negative integers a_1, a_2, \dots, a_n where each represents a point at coordinate (i, a_i) . ' n ' vertical lines are drawn such that the two endpoints of line i is at (i, a_i) and $(i, 0)$. Find two lines, which together with x-axis forms a container, such that the container contains the most water.

The program should return an integer which corresponds to the maximum area of water that can be contained (maximum area instead of maximum volume sounds weird but this is the 2D plane we are working with for simplicity).

Note: You may not slant the container.

Input: $arr = [1, 5, 4, 3]$

Output: 6

Explanation:

5 and 3 are distance 2 apart. So the size of the base = 2.

Height of container = $\min(5, 3) = 3$. So total area = $3 * 2 = 6$

Input: arr = [3, 1, 2, 4, 5]

Output: 12

Code:

```
public class ContainerWithMostWater {
    public static int maxArea(int[] arr) {
        int left = 0;
        int right = arr.length - 1;
        int maxArea = 0;

        while (left < right) {
            int height = Math.min(arr[left], arr[right]);
            int width = right - left;
            int area = height * width;
            maxArea = Math.max(maxArea, area);

            if (arr[left] < arr[right]) {
                left++;
            } else {
                right--;
            }
        }

        return maxArea;
    }

    public static void main(String[] args) {
        int[] arr1 = {1, 5, 4, 3};
        System.out.println("Max Area for arr1: " + maxArea(arr1));
        int[] arr2 = {3, 1, 2, 4, 5};
        System.out.println("Max Area for arr2: " + maxArea(arr2));
    }
}
```

Output:



```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  SEARCH ERROR

PS D:\Project\New folder> & 'C:\Program Files\Java\jdk-23\bin\java.exe' '-agentlib:jdwp=transport=dt_socket,server=n,suspend=y,address=localhost:59355' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\dhoni\AppData\Roaming\Code\User\workspaceStorage\db3b031b7c28ce48d677c8dc032990ec\redhat.java\jdt_ws\New_folder_a5c0e61d\bin' 'ContainerWithMostWater'
Max Area for arr1: 6
Max Area for arr2: 12
PS D:\Project\New folder>
```

Time Complexity: $O(N)$

5)

Find the Factorial of a large number

Input: 100

Output:

933262154439441526816992388562667004907159682643816214685929638952175999932
299

1560894146397615651828625369792082722375825118521091686400000000000000000000
000
00
Input: 50
Output: 30414093201713378043612608166064768844377641568960512000000000000

Code:

```
import java.math.BigInteger;

public class FactorialOfLargeNumber {
    public static BigInteger factorial(int number) {
        BigInteger result = BigInteger.ONE;
        for (int i = 2; i <= number; i++) {
            result = result.multiply(BigInteger.valueOf(i));
        }
        return result;
    }
    public static void main(String[] args) {
        int number1 = 100;
        System.out.println("Factorial of " + number1 + ":");
        System.out.println(factorial(number1));
        int number2 = 50;
        System.out.println("Factorial of " + number2 + ":");
        System.out.println(factorial(number2));
    }
}
```

Output:

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS SEARCH ERROR
PS D:\Project\New folder> ^C
PS D:\Project\New folder>
PS D:\Project\New folder> d:; cd 'd:\Project\New folder'; & 'C:\Program Files\Java\jdk-23\bin\java.exe' '-agentlib:jdwp=transport=dt_socket,server=n
,suspend=y,address=localhost:59512' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\dhoni\AppData\Roaming\Code\User\work
spaceStorage\db3b031b7c28ce48d677c8dc032990ec\redhat.java\jdt_ws\New_folder_a5c0e61d\bin' 'FactorialOfLargeNumber'
Factorial of 100:
933262154439441526816992388562667004907159682643816214685929638952175999322991560894146397615651828625369792082722375825118521091686400000000000000
000000000
Factorial of 50:
30414093201713378043612608166064768844377641568960512000000000000
PS D:\Project\New folder> |
```

Time Complexity: $O(N)$

6)

. Trapping Rainwater Problem states that given an array of n non-negative integers `arr[]` representing an elevation map where the width of each bar is 1, compute how much water it can

trap after rain.

Input: `arr[] = {3, 0, 1, 0, 4, 0, 2}`

Output: 10

Explanation: The expected rainwater to be trapped is shown in the above image.

Input: `arr[] = {3, 0, 2, 0, 4}`

Output: 7

Explanation: We trap $0 + 3 + 1 + 3 + 0 = 7$ units.

Input: `arr[] = {1, 2, 3, 4}`

Output: 0

Explanation : We cannot trap water as there is no height bound on both sides

Input: `arr[] = {10, 9, 0, 5}`

Output: 5

Explanation : We trap $0 + 0 + 5 + 0 = 5$

Code:

```
import java.util.*;
class GfG {
    static int findWater(int[] arr) {
        int n = arr.length;
        int[] left = new int[n];
        int[] right = new int[n];
        int res = 0;
        left[0] = arr[0];
        for (int i = 1; i < n; i++)
            left[i] = Math.max(left[i - 1], arr[i]);
        right[n - 1] = arr[n - 1];
        for (int i = n - 2; i >= 0; i--)
            right[i] = Math.max(right[i + 1], arr[i]);
        for (int i = 1; i < n - 1; i++) {
            int minOf2 = Math.min(left[i - 1], right[i + 1]);
            if (minOf2 > arr[i]) {
                res += minOf2 - arr[i];
            }
        }
        return res;
    }
    public static void main(String[] args) {
        int[] arr = { 0, 1, 0, 2, 1, 0, 1, 3, 2, 1, 2, 1 };
        System.out.println(findWater(arr));
    }
}
```

Output:

```
PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS SEARCH ERROR
PS D:\Project\New folder> & 'C:\Program Files\Java\jdk-23\bin\java.exe' '-agentlib:jdwp=transport=dt_socket,server=n,suspend=y,address=localhost:52230' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\dhoni\AppData\Roaming\Code\User\workspaceStorage\db3b031b7c28ce48d677c8dc032990ec\redhat.java\jdk_ws\New folder_a5c0e61d\bin' 'GfG'
6
PS D:\Project\New folder>
```

Time Complexity: $O(N)$

7)

Chocolate Distribution Problem

Given an array `arr[]` of n integers where `arr[i]` represents the number of chocolates in i th packet.

Each packet can have a variable number of chocolates. There are m students, the task is to distribute chocolate packets such that:

Each student gets exactly one packet.

The difference between the maximum and minimum number of chocolates in the packets given to the students is minimized.

Input: `arr[] = {7, 3, 2, 4, 9, 12, 56}`, $m = 3$

Output: 2

Explanation: If we distribute chocolate packets `{3, 2, 4}`, we will get the minimum difference, that is 2.

Input: `arr[] = {7, 3, 2, 4, 9, 12, 56}`, $m = 5$

Output: 7

Code:

```
import java.util.Arrays;
```

```

class ChocolateDistribution {
    public static int findMinDifference(int[] packets, int students) {
        int totalPackets = packets.length;
        if (students == 0 || totalPackets == 0 || students > totalPackets) {
            return 0;
        }
        Arrays.sort(packets);
        int minDifference = Integer.MAX_VALUE;
        for (int i = 0; i <= totalPackets - students; i++) {
            int currentDifference = packets[i + students - 1] - packets[i];
            minDifference = Math.min(minDifference, currentDifference);
        }
        return minDifference;
    }

    public static void main(String[] args) {
        int[] packets = {7, 3, 2, 4, 9, 12, 56};
        int students = 3;
        System.out.println("Minimum difference: " + findMinDifference(packets, students));
    }
}

```

Output:

```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS SEARCH ERROR
PS D:\Project\New folder> ^C
PS D:\Project\New folder>
PS D:\Project\New folder> d:; cd 'd:\Project\New folder'; & 'C:\Program Files\Java\jdk-23\bin\java.exe' '-agentlib:jdwp=transport=dt_socket,server=n
,suspend=y,address=localhost:56077' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'c:\Users\dhoni\AppData\Roaming\Code\User\work
spaceStorage\db3b031b7c28ce48d677c8dc032990ec\redhat.java\jdt_ws\New folder_a5c0e61d\bin' 'ChocolateDistribution'
Minimum difference: 2
PS D:\Project\New folder>

```

Time Complexity: $O(N \log N)$

8)

Merge Overlapping Intervals

Given an array of time intervals where $arr[i] = [start_i, end_i]$, the task is to merge all the overlapping intervals into one and output the result which should have only mutually exclusive intervals.

Input: $arr[] = [[1, 3], [2, 4], [6, 8], [9, 10]]$

Output: $[[1, 4], [6, 8], [9, 10]]$

Explanation: In the given intervals, we have only two overlapping intervals $[1, 3]$ and $[2, 4]$.

Therefore, we will merge these two and return $[[1, 4], [6, 8], [9, 10]]$. Input: $arr[] = [[7, 8], [1, 5], [2, 4], [4, 6]]$

Output: $[[1, 6], [7, 8]]$

Explanation: We will merge the overlapping intervals $[[1, 5], [2, 4], [4, 6]]$ into a single interval

$[1, 6]$.

Code:

```

import java.util.ArrayList;
import java.util.Arrays;
import java.util.List;

public class MergeIntervals {
    public static int[][] merge(int[][] intervals) {
        Arrays.sort(intervals, (a, b) -> Integer.compare(a[0], b[0]));
        List<int[]> mergedIntervals = new ArrayList<>();
    }
}

```

```

        for (int[] interval : intervals) {
            if (mergedIntervals.isEmpty() || mergedIntervals.get(mergedIntervals.size() - 1)[1]
< interval[0]) {
                mergedIntervals.add(interval);
            } else {
                mergedIntervals.get(mergedIntervals.size() - 1)[1] =
                    Math.max(mergedIntervals.get(mergedIntervals.size() - 1)[1], interval[1]);
            }
        }
        return mergedIntervals.toArray(new int[mergedIntervals.size()][2]);
    }

    public static void main(String[] args) {
        int[][] intervals1 = {{1, 3}, {2, 4}, {6, 8}, {9, 10}};
        int[][] result1 = merge(intervals1);
        System.out.println("Merged intervals: " + Arrays.deepToString(result1));
        int[][] intervals2 = {{7, 8}, {1, 5}, {2, 4}, {4, 6}};
        int[][] result2 = merge(intervals2);
        System.out.println("Merged intervals: " + Arrays.deepToString(result2));
    }
}

```

Output:

```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS SEARCH ERROR
PS D:\Project\New folder> & 'C:\Program Files\Java\jdk-23\bin\java.exe' '-enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\
\dhoni\AppData\Roaming\Code\User\workspaceStorage\db3b031b7c28ce48d677c8dc032990ec\redhat.java\jdt_ws\New folder_a5c0e61d\bin' 'MergeIntervals'
Merged intervals: [[1, 4], [6, 8], [9, 10]]
Merged intervals: [[1, 6], [7, 8]]
PS D:\Project\New folder>

```

Time Complexity: $O(N \log N)$

9)

. A Boolean Matrix Question

Given a boolean matrix $mat[M][N]$ of size $M \times N$, modify it such that if a matrix cell $mat[i][j]$ is

1 (or true) then make all the cells of i th row and j th column as 1.

Input: $\{\{1, 0\},$

$\{0, 0\}\}$

Output: $\{\{1, 1\}$

$\{1, 0\}\}$

Input: $\{\{0, 0, 0\},$

$\{0, 0, 1\}\}$

Output: $\{\{0, 0, 1\},$

$\{1, 1, 1\}\}$

Input: $\{\{1, 0, 0, 1\},$

$\{0, 0, 1, 0\},$

$\{0, 0, 0, 0\}\}$

Output: $\{\{1, 1, 1, 1\},$

$\{1, 1, 1, 1\},$

$\{1, 0, 1, 1\}\}$

Code:

```

import java.util.Arrays;

public class BooleanMatrix {
    public static void modifyMatrix(int[][] mat) {
        int m = mat.length;
        int n = mat[0].length;
        boolean[] rowFlags = new boolean[m];
    }
}

```



```

        boolean[] colFlags = new boolean[n];
        for (int i = 0; i < m; i++) {
            for (int j = 0; j < n; j++) {
                if (mat[i][j] == 1) {
                    rowFlags[i] = true;
                    colFlags[j] = true;
                }
            }
        }
        for (int i = 0; i < m; i++) {
            if (rowFlags[i]) {
                Arrays.fill(mat[i], 1);
            }
        }
        for (int j = 0; j < n; j++) {
            if (colFlags[j]) {
                for (int i = 0; i < m; i++) {
                    mat[i][j] = 1;
                }
            }
        }
    }

    public static void printMatrix(int[][] mat) {
        for (int[] row : mat) {
            for (int cell : row) {
                System.out.print(cell + " ");
            }
            System.out.println();
        }
    }

    public static void main(String[] args) {
        int[][] mat1 = {{1, 0}, {0, 0}};
        modifyMatrix(mat1);
        System.out.println("Modified Matrix 1:");
        printMatrix(mat1);
        int[][] mat2 = {{0, 0, 0}, {0, 0, 1}};
        modifyMatrix(mat2);
        System.out.println("\nModified Matrix 2:");
        printMatrix(mat2);
        int[][] mat3 = {{1, 0, 0, 1}, {0, 0, 1, 0}, {0, 0, 0, 0}};
        modifyMatrix(mat3);
        System.out.println("\nModified Matrix 3:");
        printMatrix(mat3);
    }
}

```

Output:

```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS SEARCH ERROR

PS D:\Project\New folder> & 'C:\Program Files\Java\jdk-23\bin\java.exe' '-agentlib:jdwp=transport=dt_socket,server=n,suspend=y,address=localhost:57127' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\dhoni\AppData\Roaming\Code\User\workspaceStorage\db3b031b7c28ce48d677c8dc032990ec\redhat.java\jdt_ws\New folder_a5c0e61d\bin' 'BooleanMatrix'
Modified Matrix 1:
1 1
1 0

Modified Matrix 2:
0 0 1
1 1 1

Modified Matrix 3:
1 1 1 1
1 1 1 1
1 0 1 1
PS D:\Project\New folder>

```

Time Complexity: $O(M*N)$

10)

Print a given matrix in spiral form

Given an m x n matrix, the task is to print all elements of the matrix in spiral form.

Input: matrix = {{1, 2, 3, 4},

{5, 6, 7, 8},

{9, 10, 11, 12},

{13, 14, 15, 16}}

Output: 1 2 3 4 8 12 16 15 14 13 9 5 6 7 11 10

Input: matrix = { {1, 2, 3, 4, 5, 6},

{7, 8, 9, 10, 11, 12},

{13, 14, 15, 16, 17, 18}}

Output: 1 2 3 4 5 6 12 18 17 16 15 14 13 7 8 9 10 11

Explanation: The output is matrix in spiral format.

Code:

```
public class SpiralMatrix {

    public static void printSpiral(int[][] matrix) {
        int m = matrix.length;
        int n = matrix[0].length;
        int top = 0, bottom = m - 1;
        int left = 0, right = n - 1;
        while (top <= bottom && left <= right) {
            for (int i = left; i <= right; i++) {
                System.out.print(matrix[top][i] + " ");
            }
            top++;
            for (int i = top; i <= bottom; i++) {
                System.out.print(matrix[i][right] + " ");
            }
            right--;
            if (top <= bottom) {
                for (int i = right; i >= left; i--) {
                    System.out.print(matrix[bottom][i] + " ");
                }
                bottom--;
            }
            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}
        };
        System.out.println("Spiral Order of Matrix 1:");
        printSpiral(matrix1);
        System.out.println("\n");
        int[][] matrix2 = {
            {1, 2, 3, 4, 5, 6},
            {7, 8, 9, 10, 11, 12},
        };
    }
}
```

```

        {13, 14, 15, 16, 17, 18}
    };
    System.out.println("Spiral Order of Matrix 2:");
    printSpiral(matrix2);
}
}

```

Output:

```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS SEARCH ERROR
PS D:\Project\New folder> & 'C:\Program Files\Java\jdk-23\bin\java.exe' '-agentlib:jdwp=transport=dt_socket,server=n,suspend=y,address=localhost:57189' '-enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\dhoni\AppData\Roaming\Code\User\workspaceStorage\db3b031b7c28ce48d677c8dc032990ec\redhat.java\jdt_ws\New folder_a5c0e61d\bin' 'SpiralMatrix'
Spiral Order of Matrix 1:
1 2 3 4 8 12 16 15 14 13 9 5 6 7 11 10

Spiral Order of Matrix 2:
1 2 3 4 5 6 12 18 17 16 15 14 13 7 8 9 10 11
PS D:\Project\New folder>

```

Time Complexity: $O(M*N)$

13)

Check if given Parentheses expression is balanced or not

Given a string str of length N, consisting of „(, „), and „,„ only, the task is to check whether it is balanced or not. Input: str = “((()))()()”

Output: Balanced

Input: str = “())(()”

Output: Not Balanced

Code:

```

import java.util.Stack;
public class BalancedParentheses {
    public static boolean isBalanced(String str) {
        Stack<Character> stack = new Stack<>();
        for (char ch : str.toCharArray()) {
            if (ch == '(') {
                stack.push(ch);
            } else if (ch == ')') {
                if (stack.isEmpty()) {
                    return false;
                }
                stack.pop();
            }
        }
        return stack.isEmpty();
    }
    public static void main(String[] args) {
        String str1 = "((( )))()()";
        System.out.println("Input: " + str1);
        System.out.println("Output: " + (isBalanced(str1) ? "Balanced" : "Not Balanced"));
        String str2 = "() )(( )";
        System.out.println("Input: " + str2);
        System.out.println("Output: " + (isBalanced(str2) ? "Balanced" : "Not Balanced"));
    }
}

```

Output:

```
PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS SEARCH ERROR
PS D:\Project\New folder> & 'C:\Program Files\Java\jdk-23\bin\java.exe' '-agentlib:jdwp=transport=dt_socket,server=n,suspend=y,address=localhost:57263' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\dhoni\AppData\Roaming\Code\User\workspaceStorage\db3b031b7c28ce48d677c8dc032990ec\redhat.java\jdt_ws\New folder_a5c0e61d\bin' 'BalancedParentheses'
Input: (((()))())
Output: Balanced
Input: ())((())
Output: Not Balanced
PS D:\Project\New folder>
```

Time Complexity : $O(N)$

14)

Check if two Strings are Anagrams of each other

Given two strings s1 and s2 consisting of lowercase characters, the task is to check whether the

two given strings are anagrams of each other or not. An anagram of a string is another string that

contains the same characters, only the order of characters can be different.

Input: s1 = "geeks" s2 = "kseeg"

Output: true

Explanation: Both the string have same characters with same frequency. So, they are anagrams.

Input: s1 = "allergy" s2 = "allergic"

Output: false

Explanation: Characters in both the strings are not same. s1 has extra character „y“ and s2 has extra characters „i“ and „c“, so they are not anagrams.

Input: s1 = "g", s2 = "g"

Output: true

Explanation: Characters in both the strings are same, so they are anagrams.

Code:

```
public class AnagramCheck {
    public static boolean areAnagrams(String s1, String s2) {
        if (s1.length() != s2.length()) {
            return false;
        }
        int[] charCount = new int[26];
        for (char c : s1.toCharArray()) {
            charCount[c - 'a']++;
        }
        for (char c : s2.toCharArray()) {
            charCount[c - 'a']--;
        }
        for (int count : charCount) {
            if (count != 0) {
                return false;
            }
        }
        return true;
    }

    public static void main(String[] args) {
        String s1 = "geeks";
        String s2 = "kseeg";
        System.out.println("Are '" + s1 + "' and '" + s2 + "' anagrams? " + areAnagrams(s1, s2));

        String s3 = "allergy";
        String s4 = "allergic";
```

```

        System.out.println("Are '" + s3 + "' and '" + s4 + "' anagrams? " + areAnagrams(s3,
s4));
    }
}

```

Output:

```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS SEARCH ERROR
PS D:\Project\New folder> & 'C:\Program Files\Java\jdk-23\bin\java.exe' '-agentlib:jdwp=transport=dt_socket,server=n,suspend=y,address=localhost:57263' '-enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\dhoni\AppData\Roaming\Code\User\workspaceStorage\db3b031b7c28ce48d677c8dc032990ec\redhat.java\jdt_ws\New folder_a5c0e61d\bin' 'BalancedParentheses'
Input: (((()))())
Output: Balanced
Input: ()()()()
Output: Not Balanced
PS D:\Project\New folder>

```

Time Complexity: $O(N \log N)$

15)

Longest Palindromic Substring

Given a string str, the task is to find the longest substring which is a palindrome. If there are multiple answers, then return the first appearing substring.

Input: str = “forgeeksskeegfor”

Output: “geeksskeeg”

Explanation: There are several possible palindromic substrings like “kssk”, “ss”, “eeksskee” etc.

But the substring “geeksskeeg” is the longest among all.

Input: str = “Geeks”

Output: “ee”

Input: str = “abc”

Output: “a”

Input: str = “”

Output: “”

Code:

```

public class GfG {
    static boolean checkPal(String s, int low, int high) {
        while (low < high) {
            if (s.charAt(low) != s.charAt(high))
                return false;
            low++;
            high--;
        }
        return true;
    }
    static String longestPalSubstr(String s) {
        int n = s.length();
        int maxLen = 1, start = 0;
        for (int i = 0; i < n; i++) {
            for (int j = i; j < n; j++) {
                if (checkPal(s, i, j) && (j - i + 1) > maxLen) {
                    start = i;
                    maxLen = j - i + 1;
                }
            }
        }
        return s.substring(start, start + maxLen);
    }
    public static void main(String[] args) {
        String s = "forgeeksskeegfor";
        System.out.println(longestPalSubstr(s));
    }
}

```

```

    }
}

```

Output:

```

PROBLEMS 2 OUTPUT DEBUG CONSOLE TERMINAL PORTS SEARCH ERROR

PS D:\Project\New folder> & 'C:\Program Files\Java\jdk-23\bin\java.exe' '-agentlib:jdwp=transport=dt_socket,server=n,suspend=y,address=localhost:55849' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\dhoni\AppData\Roaming\Code\User\workspaceStorage\db3b031b7c28ce48d677c8dc032990ec\redhat.java\jdt_ws\New folder_a5c0e61d\bin' 'Gfg'
6
PS D:\Project\New folder> ^C
PS D:\Project\New folder>
PS D:\Project\New folder> d;; cd 'd:\Project\New folder'; & 'C:\Program Files\Java\jdk-23\bin\java.exe' '-agentlib:jdwp=transport=dt_socket,server=n,suspend=y,address=localhost:57537' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\dhoni\AppData\Roaming\Code\User\workspaceStorage\db3b031b7c28ce48d677c8dc032990ec\redhat.java\jdt_ws\New folder_a5c0e61d\bin' 'Gfg'
6
PS D:\Project\New folder>

```

Time Complexity: $O(N)^3$

16)

Longest Common Prefix using Sorting

Given an array of strings `arr[]`. The task is to return the longest common prefix among each and

every strings present in the array. If there's no prefix common in all the strings, return "-1".

Input: `arr[] = ["geeksforgeeks", "geeks", "geek", "geezer"]`

Output: `gee`

Explanation: "gee" is the longest common prefix in all the given strings. Input: `arr[] = ["hello", "world"]`

Output: `-1`

Explanation: There's no common prefix in the given strings.

Code:

```

import java.util.Arrays;
public class LongestCommonPrefix {
    public static String findLongestCommonPrefix(String[] arr) {
        if (arr == null || arr.length == 0) {
            return "-1";
        }
        Arrays.sort(arr);
        String first = arr[0];
        String last = arr[arr.length - 1];
        int commonPrefixLength = 0;
        while (commonPrefixLength < first.length() && commonPrefixLength < last.length()
            && first.charAt(commonPrefixLength) == last.charAt(commonPrefixLength)) {
            commonPrefixLength++;
        }
        String commonPrefix = first.substring(0, commonPrefixLength);
        return commonPrefix.isEmpty() ? "-1" : commonPrefix;
    }
    public static void main(String[] args) {
        String[] arr1 = { "geeksforgeeks", "geeks", "geek", "geezer" };
        System.out.println("Longest Common Prefix: " + findLongestCommonPrefix(arr1));
        String[] arr2 = { "hello", "world" };
        System.out.println("Longest Common Prefix: " + findLongestCommonPrefix(arr2));
    }
}

```

Output:

```
PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS SEARCH ERROR
PS D:\Project\New folder> & 'C:\Program Files\Java\jdk-23\bin\java.exe' '-agentlib:jdwp=transport=dt_socket,server=n,suspend=y,address=localhost:57719' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\dhoni\AppData\Roaming\Code\User\workspaceStorage\db3b031b7c28ce48d677c8dc032990ec\redhat.java\jdt_ws\New folder_a5c0e61d\bin' 'LongestCommonPrefix'
Longest Common Prefix: gee
Longest Common Prefix: -1
PS D:\Project\New folder>
```

Time Complexity: $O(N \log N + M)$

17)

. Delete middle element of a stack

Given a stack with push(), pop(), and empty() operations, The task is to delete the middle element

of it without using any additional data structure.

Input : Stack[] = [1, 2, 3, 4, 5]

Output : Stack[] = [1, 2, 4, 5]

Input : Stack[] = [1, 2, 3, 4, 5, 6]

Output : Stack[] = [1, 2, 4, 5, 6]

Code:

```
import java.util.Stack;
public class DeleteMiddleElement {
    public static void deleteMiddle(Stack<Integer> stack) {
        int middleIndex = stack.size() / 2;
        deleteMiddleHelper(stack, middleIndex);
    }
    private static void deleteMiddleHelper(Stack<Integer> stack, int currentIndex) {
        if (currentIndex == 0) {
            stack.pop();
            return;
        }
        int topElement = stack.pop();
        deleteMiddleHelper(stack, currentIndex - 1);
        stack.push(topElement);
    }
    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);
        System.out.println("Original Stack: " + stack1);
        deleteMiddle(stack1);
        System.out.println("After Deleting Middle Element: " + stack1);
        Stack<Integer> stack2 = new Stack<>();
        stack2.push(1);
        stack2.push(2);
        stack2.push(3);
        stack2.push(4);
        stack2.push(5);
        stack2.push(6);
        System.out.println("Original Stack: " + stack2);
        deleteMiddle(stack2);
        System.out.println("After Deleting Middle Element: " + stack2);
    }
}
```

Output:

```
PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS SEARCH ERROR
PS D:\Project\New folder> & 'C:\Program Files\Java\jdk-23\bin\java.exe' '-agentlib:jdwp=transport=dt_socket,server=n,suspend=y,address=localhost:57780' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\dhoni\AppData\Roaming\Code\User\workspaceStorage\db3b031b7c28ce48d677c8dc032990ec\redhat.java\jdt_ws\New_folder_a5c0e61d\bin' 'DeleteMiddleElement'
Original Stack: [1, 2, 3, 4, 5]
After Deleting Middle Element: [1, 2, 4, 5]
Original Stack: [1, 2, 3, 4, 5, 6]
After Deleting Middle Element: [1, 2, 4, 5, 6]
PS D:\Project\New folder>
```

Time Complexity: $O(N)$

18)

Next Greater Element (NGE) for every element in given Array

Given an array, print the Next Greater Element (NGE) for every element.

Note: The Next greater Element for an element x is the first greater element on the right side of x

in the array. Elements for which no greater element exist, consider the next greater element as -1 .

Input: `arr[] = [4 , 5 , 2 , 25]`

Output: `4 -> 5`

`5 -> 25`

`2 -> 25`

`25 -> -1`

Explanation: Except 25 every element has an element greater than them present on the right side

Input: `arr[] = [13 , 7, 6 , 12]`

Output: `13 -> -1`

`7 -> 12`

`6 -> 12`

`12 -> -1`

Explanation: 13 and 12 don't have any element greater than them present on the right side

Code:

```
import java.util.Stack;
import java.util.HashMap;
public class NextGreaterElement {
    public static HashMap<Integer, Integer> findNextGreaterElements(int[] arr) {
        HashMap<Integer, Integer> ngeMap = new HashMap<>();
        Stack<Integer> stack = new Stack<>();
        for (int i = 0; i < arr.length; i++) {
            int current = arr[i];
            while (!stack.isEmpty() && stack.peek() < current) {
                ngeMap.put(stack.pop(), current);
            }
            stack.push(current);
        }
        while (!stack.isEmpty()) {
            ngeMap.put(stack.pop(), -1);
        }
        return ngeMap;
    }
    public static void main(String[] args) {
        int[] arr1 = {4, 5, 2, 25};
        System.out.println("Next Greater Elements:");
        HashMap<Integer, Integer> result1 = findNextGreaterElements(arr1);
        for (int num : arr1) {
            System.out.println(num + " -> " + result1.get(num));
        }
        int[] arr2 = {13, 7, 6, 12};
```



```

        System.out.println("\nNext Greater Elements:");
        HashMap<Integer, Integer> result2 = findNextGreaterElements(arr2);
        for (int num : arr2) {
            System.out.println(num + " -> " + result2.get(num));
        }
    }
}

```

Output:

```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS SEARCH ERROR
PS D:\Project\New folder> & 'C:\Program Files\Java\jdk-23\bin\java.exe' '-agentlib:jdwp=transport=dt_socket,server=n,suspend=y,address=localhost:57816' '-enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\dhoni\AppData\Roaming\Code\User\workspacestorage\db3b031b7c28ce48d677c8dc032990ec\redhat.java\jdt_ws\New folder_a5c0e61d\bin' 'NextGreaterElement'
Next Greater Elements:
4 -> 5
5 -> 25
2 -> 25
25 -> -1

Next Greater Elements:
13 -> -1
7 -> 12
6 -> 12
12 -> -1
PS D:\Project\New folder>

```

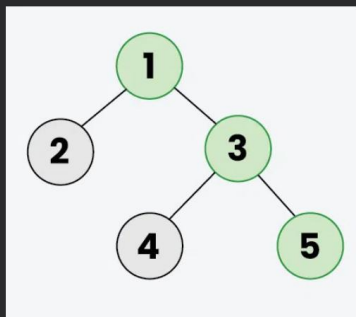
Time Complexity: $O(N)$

19)

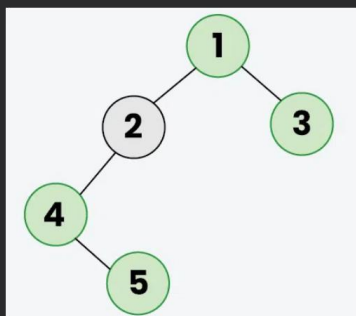
. Print Right View of a Binary Tree

Given a Binary Tree, the task is to print the Right view of it. The right view of a Binary Tree is a set of rightmost nodes for every level.

Example 1: The **Green** colored nodes (1, 3, 5) represents the Right view in the below Binary tree.



Example 2: The **Green** colored nodes (1, 3, 4, 5) represents the Right view in the below Binary tree.



Code:

```

import java.util.ArrayList;
import java.util.LinkedList;

```

```

import java.util.List;
import java.util.Queue;

class TreeNode {
    int value;
    TreeNode left, right;
    TreeNode(int value) {
        this.value = value;
        left = right = null;
    }
}

public class BinaryTreeRightView {
    public List<Integer> rightView(TreeNode root) {
        List<Integer> result = new ArrayList<>();

        if (root == null) {
            return result;
        }
        Queue<TreeNode> queue = new LinkedList<>();
        queue.add(root);
        while (!queue.isEmpty()) {
            int levelSize = queue.size();
            for (int i = 0; i < levelSize; i++) {
                TreeNode node = queue.poll();
                if (i == levelSize - 1) {
                    result.add(node.value);
                }
                if (node.left != null) {
                    queue.add(node.left);
                }
                if (node.right != null) {
                    queue.add(node.right);
                }
            }
        }
        return result;
    }

    public static void main(String[] args) {
        BinaryTreeRightView tree = new BinaryTreeRightView();

        TreeNode root = new TreeNode(1);
        root.left = new TreeNode(2);
        root.right = new TreeNode(3);
        root.left.right = new TreeNode(4);
        root.right.right = new TreeNode(5);

        List<Integer> rightView = tree.rightView(root);

        System.out.println("Right view of the binary tree: " + rightView);
    }
}

```

Output:

```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS SEARCH ERROR
PS D:\Project\New folder> & 'C:\Program Files\Java\jdk-23\bin\java.exe' '-agentlib:jdwp=transport=dt_socket,server=n,suspend=y,address=localhost:58332' '-enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\dhoni\AppData\Roaming\Code\User\workspaceStorage\db3b031b7c28ce48d677c8dc032990ec\redhat.java\jdt_ws\New folder_a5c0e61d\bin' 'BinaryTreeRightView'
Right view of the binary tree: [1, 3, 5]
PS D:\Project\New folder>

```

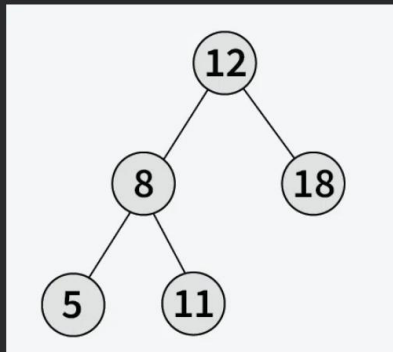
Time Complexity: $O(N)$

20)

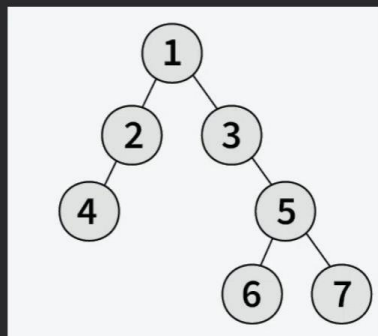
Maximum Depth or Height of Binary Tree

Given a binary tree, the task is to find the maximum depth or height of the tree. The height of the tree is the number of vertices in the tree from the root to the deepest node.

Example 1: The height of the below binary tree is 3.



Example 2: The height of the below binary tree is 4



Code:

```
class TreeNode {
    int value;
    TreeNode left, right;

    TreeNode(int value) {
        this.value = value;
        left = right = null;
    }
}

public class BinaryTreeHeight {
    public int maxDepth(TreeNode root) {
        if (root == null) {
            return 0;
        }

        int leftHeight = maxDepth(root.left);
        int rightHeight = maxDepth(root.right);

        return 1 + Math.max(leftHeight, rightHeight);
    }
}
```

```

public static void main(String[] args) {
    BinaryTreeHeight tree = new BinaryTreeHeight();
    TreeNode root1 = new TreeNode(12);
    root1.left = new TreeNode(8);
    root1.right = new TreeNode(18);
    root1.left.left = new TreeNode(5);
    root1.left.right = new TreeNode(11);
    System.out.println("Height of Example 1 tree: " + tree.maxDepth(root1));
    TreeNode root2 = new TreeNode(1);
    root2.left = new TreeNode(2);
    root2.right = new TreeNode(3);
    root2.left.left = new TreeNode(4);
    root2.right.left = new TreeNode(5);
    root2.right.left.left = new TreeNode(6);
    root2.right.left.right = new TreeNode(7);
    System.out.println("Height of Example 2 tree: " + tree.maxDepth(root2));
}
}

```

Output:

```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS SEARCH ERROR
PS D:\Project\New folder> & 'C:\Program Files\Java\jdk-23\bin\java.exe' '-agentlib:jdwp=transport=dt_socket,server=n,suspend=y,address=localhost:58407' '--enable-preview' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\dhoni\AppData\Roaming\Code\User\workspaceStorage\db3b031b7c28ce48d677c8dc032990ec\redhat.java\jdt_ws\New folder_a5c0e61d\bin' 'BinaryTreeHeight'
Height of Example 1 tree: 3
Height of Example 2 tree: 4
PS D:\Project\New folder>

```

Time Complexity: $O(N)$