(09-11-2024) TECHNICAL TRAINING DSA - CODING PRACTICE PROBLEMS

QUESTION

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}

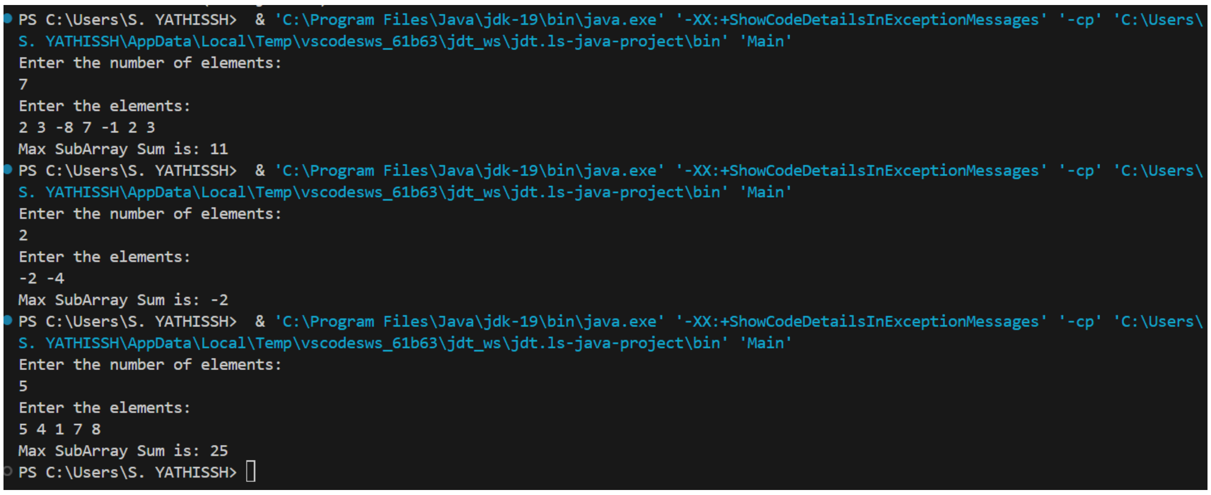
Output: 25

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

CODE:

```
import java.util.*;
public class Main {
    public static long MaxSubArraySum(int arr[],int n){
        long maxx=Long.MIN_VALUE;
        long sum=0;
        for (int i=0;i<n;i++){
            sum+=arr[i];
            if (sum > maxx) {
                maxx = sum;
            }
            if (sum < 0) {
                sum = 0;
            }
        }
        return maxx;
        }public static void main(String args[]) {
            Scanner scanner = new Scanner(System.in);

            System.out.println("Enter the number of elements:");
            int n = scanner.nextInt();

            int arr[] = new int[n];

            System.out.println("Enter the elements:");
            for (int i = 0; i < n; i++) {
                arr[i] = scanner.nextInt();
            }

            long maxsum = MaxSubArraySum(arr, n);
            System.out.println("Max SubArray Sum is: " +
maxsum);

            scanner.close();
        }
    }
```

OUTPUT:



```
PS C:\Users\S. YATHISSH>  & 'C:\Program Files\Java\jdk-19\bin\java.exe' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\
S. YATHISSH\AppData\Local\Temp\vscodesws_61b63\jdt_ws\jdt.ls-java-project\bin' 'Main'
Enter the number of elements:
7
Enter the elements:
2 3 -8 7 -1 2 3
Max SubArray Sum is: 11
PS C:\Users\S. YATHISSH>  & 'C:\Program Files\Java\jdk-19\bin\java.exe' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\
S. YATHISSH\AppData\Local\Temp\vscodesws_61b63\jdt_ws\jdt.ls-java-project\bin' 'Main'
Enter the number of elements:
2
Enter the elements:
-2 -4
Max SubArray Sum is: -2
PS C:\Users\S. YATHISSH>  & 'C:\Program Files\Java\jdk-19\bin\java.exe' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\
S. YATHISSH\AppData\Local\Temp\vscodesws_61b63\jdt_ws\jdt.ls-java-project\bin' 'Main'
Enter the number of elements:
5
Enter the elements:
5 4 1 7 8
Max SubArray Sum is: 25
PS C:\Users\S. YATHISSH>
```

Time Complexity: O(n)

Space Complexity: O(1)

QUESTION

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}

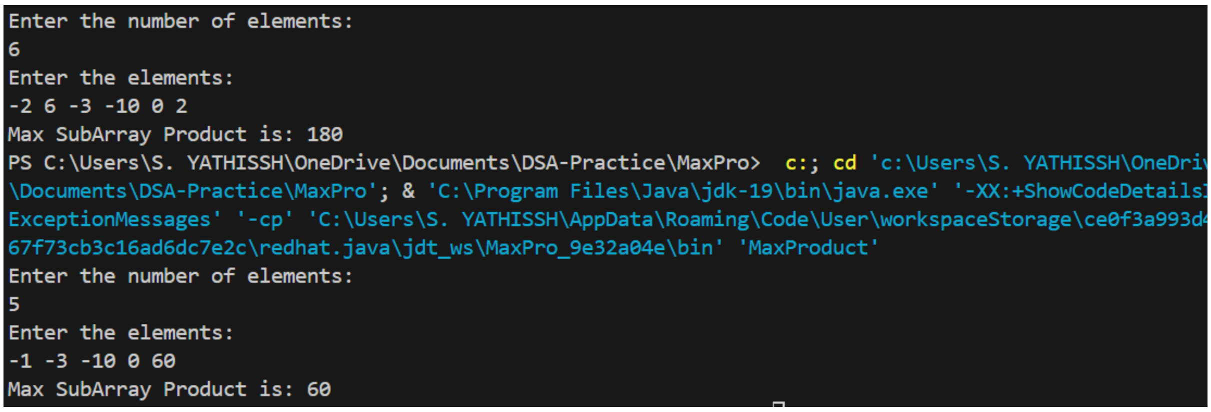
Output: 60

Explanation: The subarray with maximum product is {60}

CODE:

```
import java.util.*;

public class MaxProduct {
    public static long MaxProduct(int arr[], int n) {
        long maxx = arr[0];
        long minn = arr[0];
        long res = arr[0];

        for (int i = 1; i < n; i++) {
            if (arr[i] < 0) {
                long temp = maxx;
                maxx = minn;
                minn = temp;
            }

            maxx = Math.max(arr[i], maxx * arr[i]);
            minn = Math.min(arr[i], minn * arr[i]);
            res = Math.max(res, maxx);
        }
        return res;
    }

    public static void main(String args[]) {
        Scanner scanner = new Scanner(System.in);

        System.out.println("Enter the number of elements:");
        int n = scanner.nextInt();

        int arr[] = new int[n];

        System.out.println("Enter the elements:");
        for (int i = 0; i < n; i++) {
            arr[i] = scanner.nextInt();
        }
        long maxProduct = MaxProduct(arr, n);

        System.out.println("Max SubArray Product is: " +
maxProduct);


        scanner.close();
    }
}
```

OUTPUT:



```
Enter the number of elements:
6
Enter the elements:
-2 6 -3 -10 0 2
Max SubArray Product is: 180
PS C:\Users\S. YATHISSH\OneDrive\Documents\DSA-Practice\MaxPro>  c:; cd 'c:\Users\S. YATHISSH\OneDriv
\Documents\DSA-Practice\MaxPro'; & 'C:\Program Files\Java\jdk-19\bin\java.exe' '-XX:+ShowCodeDetailsI
ExceptionMessages' '-cp' 'C:\Users\S. YATHISSH\AppData\Roaming\Code\User\workspaceStorage\ce0f3a993d4
67f73cb3c16ad6dc7e2c\redhat.java\jdt_ws\MaxPro_9e32a04e\bin' 'MaxProduct'
Enter the number of elements:
5
Enter the elements:
-1 -3 -10 0 60
Max SubArray Product is: 60
```

Time Complexity: O(n)

Space Complexity: O(1)

QUESTION

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

Input : arr[] = { 4, 5, 6, 7, 0, 1, 2 }, key = 3

Output : -1

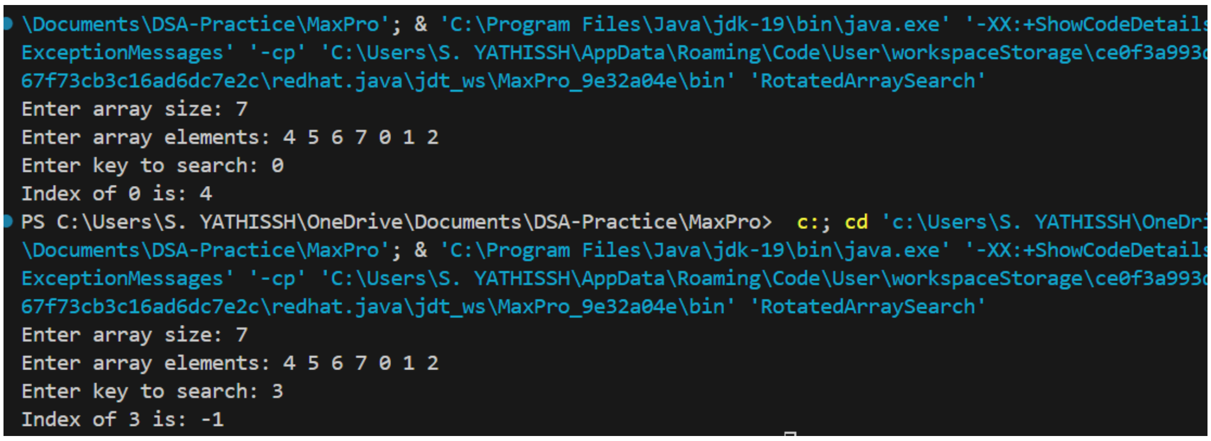
Input : arr[] = {50, 10, 20, 30, 40}, key = 10

Output : 1

CODE:

```java
import java.util.Scanner;

public class RotatedArraySearch {

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

        while (left <= right) {
            int mid = left + (right - left) / 2;

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

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

        return -1;
    }

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter array size: ");
        int n = scanner.nextInt();
        int[] arr = new int[n];

        System.out.print("Enter array elements: ");
        for (int i = 0; i < n; i++) {
            arr[i] = scanner.nextInt();
        }

        System.out.print("Enter key to search: ");
        int key = scanner.nextInt();
        int result = search(arr, key);

        System.out.println("Index of " + key + " is: " +
result);
        scanner.close();
    }
}
```

OUTPUT:



```
\Documents\DSA-Practice\MaxPro'; & 'C:\Program Files\Java\jdk-19\bin\java.exe' '-XX:+ShowCodeDetail
ExceptionMessages' '-cp' 'C:\Users\S. YATHISSH\AppData\Roaming\Code\User\workspaceStorage\ce0f3a993
67f73cb3c16ad6dc7e2c\redhat.java\jdt_ws\MaxPro_9e32a04e\bin' 'RotatedArraySearch'
Enter array size: 7
Enter array elements: 4 5 6 7 0 1 2
Enter key to search: 0
Index of 0 is: 4
PS C:\Users\S. YATHISSH\OneDrive\Documents\DSA-Practice\MaxPro>  c:; cd 'c:\Users\S. YATHISSH\OneDr
\Documents\DSA-Practice\MaxPro'; & 'C:\Program Files\Java\jdk-19\bin\java.exe' '-XX:+ShowCodeDetail
ExceptionMessages' '-cp' 'C:\Users\S. YATHISSH\AppData\Roaming\Code\User\workspaceStorage\ce0f3a993
67f73cb3c16ad6dc7e2c\redhat.java\jdt_ws\MaxPro_9e32a04e\bin' 'RotatedArraySearch'
Enter array size: 7
Enter array elements: 4 5 6 7 0 1 2
Enter key to search: 3
Index of 3 is: -1
```

Time Complexity: O(logn)

Space Complexity: O(1)

QUESTION

4. Container with Most WaterInput: 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

Explanation:

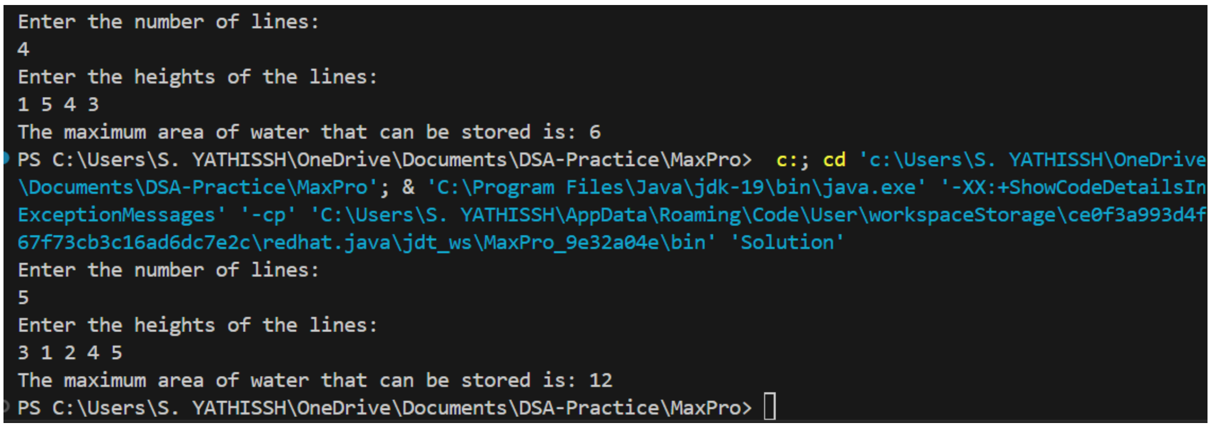
5 and 3 are distance 4 apart. So the size of the base = 4.

Height of container = min(5, 3) = 3. So total area = 4 * 3 = 12

CODE:

```
import java.util.*;
public class Solution {
    public int maxArea(int[] height) {
        int l = 0;
        int r = height.length - 1;
        int maxx = 0;

        while (l < r) {
            int w = r - l;
            int ch = Math.min(height[l], height[r]);
            int area = ch * w;
            maxx = Math.max(maxx, area);

            if (height[l] < height[r]) {
                l++;
            } else {
                r--;
            }
        }
        return maxx;
    }

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        Solution solution = new Solution();

        System.out.println("Enter the number of lines:");
        int n = scanner.nextInt();
        int[] height = new int[n];

        System.out.println("Enter the heights of the lines:");
        for (int i = 0; i < n; i++) {
            height[i] = scanner.nextInt();
        }
        int maxArea = solution.maxArea(height);
        System.out.println("The maximum area of water that can be stored is: " + maxArea);

        scanner.close();
    }
}
```

OUTPUT:



```
Enter the number of lines:
4
Enter the heights of the lines:
1 5 4 3
The maximum area of water that can be stored is: 6
PS C:\Users\S. YATHISSH\OneDrive\Documents\DSA-Practice\MaxPro>  c:; cd 'c:\Users\S. YATHISSH\OneDrive
\Documents\DSA-Practice\MaxPro'; & 'C:\Program Files\Java\jdk-19\bin\java.exe' '-XX:+ShowCodeDetailsIn
ExceptionMessages' '-cp' 'C:\Users\S. YATHISSH\AppData\Roaming\Code\User\workspaceStorage\ce0f3a993d4f
67f73cb3c16ad6dc7e2c\redhat.java\jdt_ws\MaxPro_9e32a04e\bin' 'Solution'
Enter the number of lines:
5
Enter the heights of the lines:
3 1 2 4 5
The maximum area of water that can be stored is: 12
PS C:\Users\S. YATHISSH\OneDrive\Documents\DSA-Practice\MaxPro>
```

Time Complexity: O(n)

Space Complexity: O(1)

QUESTION

5. Find the Factorial of a large number

Input: 100

Output:

9332621544394415268169923885626670049071596826438162146859296389521759999322 99

15608941463976156518286253697920827223758251185210916864000000000000000000000 000

00

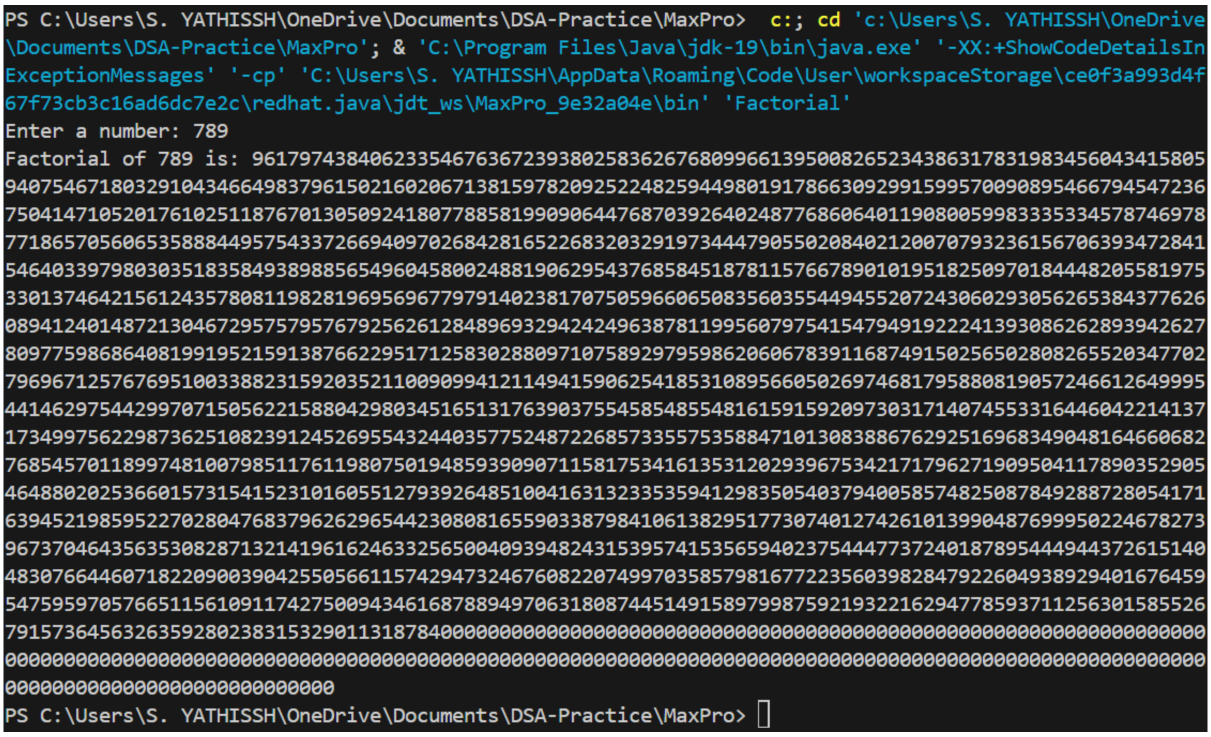
Input: 50

Output: 30414093201713378043612608166064768844377641568960512000000000000

CODE:

```
import java.math.BigInteger;
import java.util.*;

public class Factorial {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a number: ");
        int number = scanner.nextInt();

        BigInteger fact = BigInteger.ONE;
        for (int i = 1; i <= number; i++) {
            fact= fact.multiply(BigInteger.valueOf(i));
        }

        System.out.println("Factorial of " + number + " is: " +
fact);
        scanner.close();
    }
}
```

OUTPUT:



```
PS C:\Users\S. YATHISSH\OneDrive\Documents\DSA-Practice\MaxPro>  c:; cd 'c:\Users\S. YATHISSH\OneDrive
\Documents\DSA-Practice\MaxPro'; & 'C:\Program Files\Java\jdk-19\bin\java.exe' '-XX:+ShowCodeDetailsIn
ExceptionMessages' '-cp' 'C:\Users\S. YATHISSH\AppData\Roaming\Code\User\workspaceStorage\ce0f3a993d4f
67f73cb3c16ad6dc7e2c\redhat.java\jdt_ws\MaxPro_9e32a04e\bin' 'Factorial'
Enter a number: 789
Factorial of 789 is: 961797438406233546763672393802583626768099661395008265234386317831983456043415805
940754671803291043466498379615021602067138159782092522482594498019178663092991599570090895466794547236
750414710520176102511876701305092418077885819909064476870392640248776860640119080059983335334578746978
771865705606535888449575433726694097026842816522683203291973444790550208402120070793236156706393472841
546403397980303518358493898856549604580024881906295437685845187811576678901019518250970184448205581975
330137464215612435780811982819695696779791402381707505966065083560355449455207243060293056265384377626
089412401487213046729575795767925626128489693294242496387811995607975415479491922241393086262893942627
809775986864081991952159138766229517125830288097107589297959862060678391168749150256502808265520347702
796967125767695100338823159203521100909941211494159062541853108956605026974681795880819057246612649995
441462975442997071505622158804298034516513176390375545854855481615915920973031714074553316446042214137
173499756229873625108239124526955432440357752487226857335575358847101308388676292516968349048164660682
768545701189974810079851176119807501948593909071158175341613531202939675342171796271909504117890352905
464880202536601573154152310160551279392648510041631323353594129835054037940058574825087849288728054171
639452198595227028047683796262965442308081655903387984106138295177307401274261013990487699950224678273
967370464356353082871321419616246332565004093948243153957415356594023754447737240187895444944372615140
483076644607182209003904255056611574294732467608220749970358579816772235603982847922604938929401676459
547595970576651156109117427500943461687889497063180874451491589799875921932216294778593711256301585526
791573645632635928023831532901131878400000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000
PS C:\Users\S. YATHISSH\OneDrive\Documents\DSA-Practice\MaxPro> 
```

Time Complexity: O(n)

Space Complexity: O(1)

QUESTION

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}

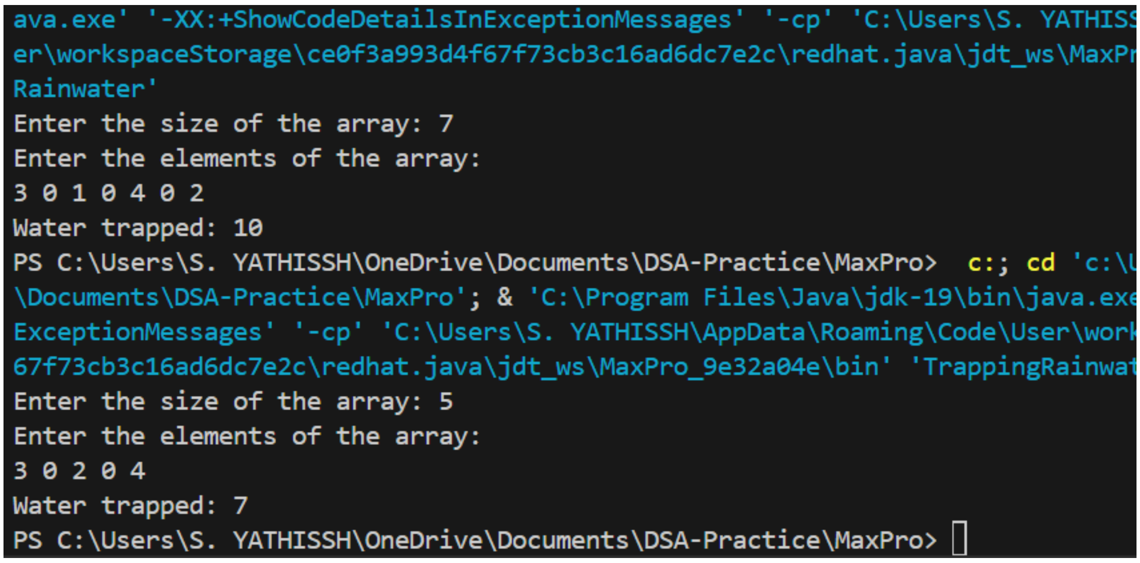
Output: 5

Explanation : we trap 0 + 0 + 5 + 0 = 5

CODE:

```
import java.util.Scanner;

public class TrappingRainwater {

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

        int[] left_max = new int[n];
        int[] right_max = new int[n];
        int waterTrapped = 0;

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

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

        for (int i = 0; i < n; i++) {
            waterTrapped += Math.min(left_max[i], right_max[i])
- arr[i];
        }
        return waterTrapped;
    }

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter the size of the array: ");
        int n = scanner.nextInt();

        int[] arr = new int[n];
        System.out.println("Enter the elements of the array: ");
        for (int i = 0; i < n; i++) {
            arr[i] = scanner.nextInt();
        }

        System.out.println("Water trapped: " + trap(arr));
        scanner.close();
    }
}
```

OUTPUT:



```
ava.exe' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\S. YATHISS
er\workspaceStorage\ce0f3a993d4f67f73cb3c16ad6dc7e2c\redhat.java\jdt_ws\MaxPr
Rainwater'
Enter the size of the array: 7
Enter the elements of the array:
3 0 1 0 4 0 2
Water trapped: 10
PS C:\Users\S. YATHISSH\OneDrive\Documents\DSA-Practice\MaxPro>  c:; cd 'c:\U
\Documents\DSA-Practice\MaxPro'; & 'C:\Program Files\Java\jdk-19\bin\java.exe
ExceptionMessages' '-cp' 'C:\Users\S. YATHISSH\AppData\Roaming\Code\User\work
67f73cb3c16ad6dc7e2c\redhat.java\jdt_ws\MaxPro_9e32a04e\bin' 'TrappingRainwat
Enter the size of the array: 5
Enter the elements of the array:
3 0 2 0 4
Water trapped: 7
PS C:\Users\S. YATHISSH\OneDrive\Documents\DSA-Practice\MaxPro>
```

Time Complexity: O(n)

Space Complexity: O(n)

QUESTION

7. Chocolate Distribution Problem

Given an array arr[] of n integers where arr[i] represents the number of chocolates in ith 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

Explanation: If we distribute chocolate packets {3, 2, 4, 9, 7}, we will get the minimum

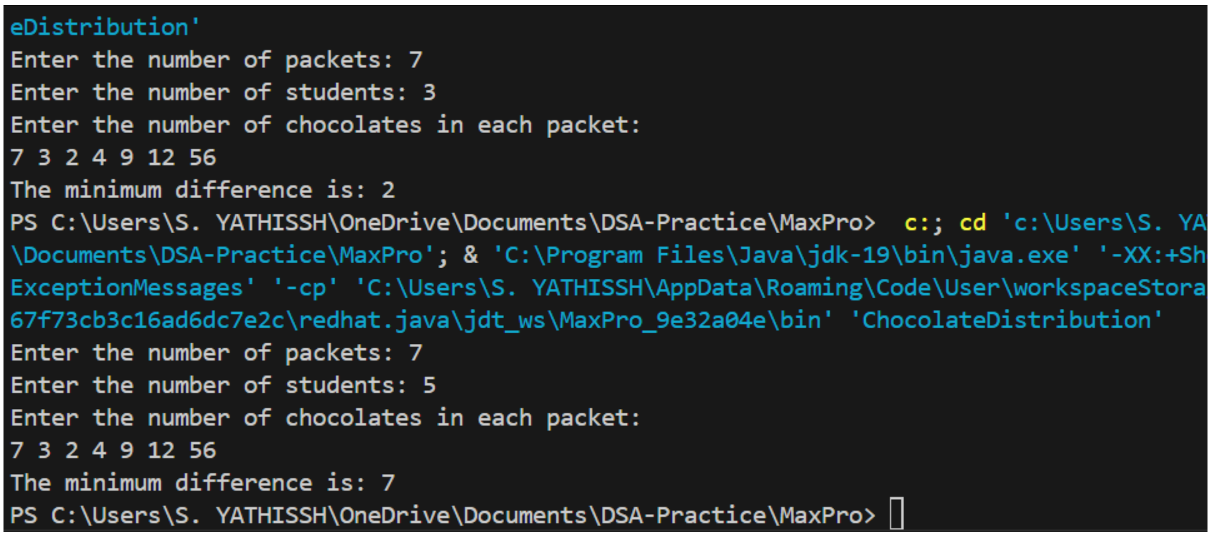
difference, that is 9 – 2 = 7.

CODE:

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

public class ChocolateDistribution {

    public static int distributeChocolate(int[] arr, int m) {
        int n = arr.length;
```

```java
        if (n < m) return -1;

        Arrays.sort(arr);
        int mind = Integer.MAX_VALUE;

        for (int i = 0; i <= n - m; i++) {
            int diff = arr[i + m - 1] - arr[i];
            mind = Math.min(mind, diff);
        }

        return mind;
    }

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter the number of packets: ");
        int n = scanner.nextInt();

        System.out.print("Enter the number of students: ");
        int m = scanner.nextInt();

        int[] arr = new int[n];

        System.out.println("Enter the number of chocolates in
each packet:");
        for (int i = 0; i < n; i++) {
            arr[i] = scanner.nextInt();
        }

        int result = distributeChocolate(arr, m);
        System.out.println("The minimum difference is: " +
result);


        scanner.close();
    }
}
```

OUTPUT:



```
eDistribution'
Enter the number of packets: 7
Enter the number of students: 3
Enter the number of chocolates in each packet:
7 3 2 4 9 12 56
The minimum difference is: 2
PS C:\Users\S. YATHISSH\OneDrive\Documents\DSA-Practice\MaxPro>  c:; cd 'c:\Users\S. YA
\Documents\DSA-Practice\MaxPro'; & 'C:\Program Files\Java\jdk-19\bin\java.exe' '-XX:+Sh
ExceptionMessages' '-cp' 'C:\Users\S. YATHISSH\AppData\Roaming\Code\User\workspaceStora
67f73cb3c16ad6dc7e2c\redhat.java\jdt_ws\MaxPro_9e32a04e\bin' 'ChocolateDistribution'
Enter the number of packets: 7
Enter the number of students: 5
Enter the number of chocolates in each packet:
7 3 2 4 9 12 56
The minimum difference is: 7
PS C:\Users\S. YATHISSH\OneDrive\Documents\DSA-Practice\MaxPro>
```

Time Complexity: O(n log n)

Space Complexity: O(1)

QUESTION

8. Merge Overlapping Intervals

Given an array of time intervals where arr[i] = [starti, endi], 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]]

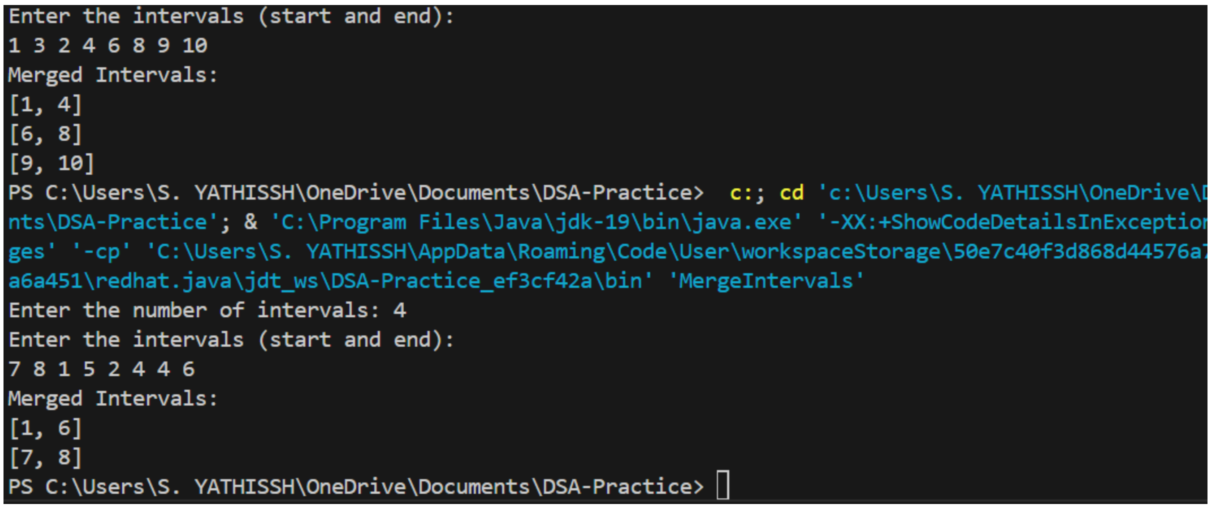
Explanation: We will merge the overlapping intervals [[1, 5], [2, 4], [4, 6]] into a single interval

[1, 6].

CODE:

```
import java.util.ArrayList;

import java.util.Arrays;

import java.util.List;

import java.util.Scanner;


public class MergeIntervals {


    public static int[][] mergeIntervals(int[][] intervals) {

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

        List<int[]> mergedList = new ArrayList<>();

        int[] currentInterval = intervals[0];

        mergedList.add(currentInterval);


        for (int[] interval : intervals) {

            int currentEnd = currentInterval[1];

            int nextStart = interval[0];

            int nextEnd = interval[1];


            if (nextStart <= currentEnd) {

                currentInterval[1] = Math.max(currentEnd, 
nextEnd);

            } else {

                currentInterval = interval;

                mergedList.add(currentInterval);
            }
        }

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

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the number of intervals: ");
        int n = scanner.nextInt();
        int[][] intervals = new int[n][2];

        System.out.println("Enter the intervals (start and
end):");
        for (int i = 0; i < n; i++) {
            intervals[i][0] = scanner.nextInt();
            intervals[i][1] = scanner.nextInt();
        }

        int[][] result = mergeIntervals(intervals);
        System.out.println("Merged Intervals:");
        for (int[] interval : result) {
            System.out.println(Arrays.toString(interval));
        }

        scanner.close();
    }
}
```

OUTPUT:



```
Enter the intervals (start and end):
1 3 2 4 6 8 9 10
Merged Intervals:
[1, 4]
[6, 8]
[9, 10]
PS C:\Users\S. YATHISSH\OneDrive\Documents\DSA-Practice>  c:; cd 'c:\Users\S. YATHISSH\OneDrive\D
nts\DSA-Practice'; & 'C:\Program Files\Java\jdk-19\bin\java.exe' '-XX:+ShowCodeDetailsInExceptio
ges' '-cp' 'C:\Users\S. YATHISSH\AppData\Roaming\Code\User\workspaceStorage\50e7c40f3d868d44576a
a6a451\redhat.java\jdt_ws\DSA-Practice_ef3cf42a\bin' 'MergeIntervals'
Enter the number of intervals: 4
Enter the intervals (start and end):
7 8 1 5 2 4 4 6
Merged Intervals:
[1, 6]
[7, 8]
PS C:\Users\S. YATHISSH\OneDrive\Documents\DSA-Practice> 
```

Time Complexity: O(n log n)

Space Complexity: O(n)

QUESTION

9. A Boolean Matrix Question

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

1 (or true) then make all the cells of ith row and jth 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}}

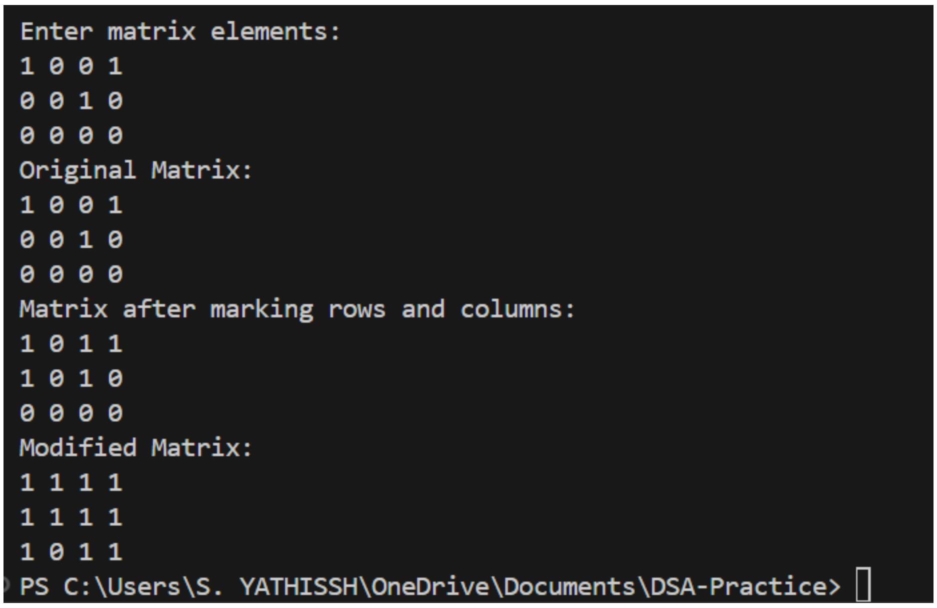
CODE:

```
import java.util.Scanner;

public class OptMatrix {

    public static void modify(int[][] mat, int m, int n) {
        boolean fr = false, fc = false;

        for (int j = 0; j < n; j++) if (mat[0][j] == 1) fr =
true;
        for (int i = 0; i < m; i++) if (mat[i][0] == 1) fc =
true;

        for (int i = 1; i < m; i++) {
            for (int j = 1; j < n; j++) {
                if (mat[i][j] == 1) {
                    mat[i][0] = 1;
                    mat[0][j] = 1;
                }
            }
        }

        System.out.println("Matrix after marking rows and
columns:");
        print(mat, m, n);

        for (int i = 1; i < m; i++) {
            for (int j = 1; j < n; j++) {
                if (mat[i][0] == 1 || mat[0][j] == 1) mat[i][j]
= 1;
```

```
            }
        }

        if (fr) for (int j = 0; j < n; j++) mat[0][j] = 1;
        if (fc) for (int i = 0; i < m; i++) mat[i][0] = 1;

        System.out.println("Modified Matrix:");
        print(mat, m, n);
    }

    public static void print(int[][] mat, int m, int n) {
        for (int i = 0; i < m; i++) {
            for (int j = 0; j < n; j++) {
                System.out.print(mat[i][j] + " ");
            }
            System.out.println();
        }
    }

    public static void main(String[] args) {
        Scanner s = new Scanner(System.in);
        System.out.print("Enter number of rows: ");
        int m = s.nextInt();
        System.out.print("Enter number of columns: ");
        int n = s.nextInt();
        int[][] mat = new int[m][n];

        System.out.println("Enter matrix elements:");
        for (int i = 0; i < m; i++)
            for (int j = 0; j < n; j++)
```

```
                mat[i][j] = s.nextInt();

        System.out.println("Original Matrix:");
        print(mat, m, n);

        modify(mat, m, n);

        s.close();
    }
}
```

OUTPUT:



```
Enter matrix elements:
1 0 0 1
0 0 1 0
0 0 0 0
Original Matrix:
1 0 0 1
0 0 1 0
0 0 0 0
Matrix after marking rows and columns:
1 0 1 1
1 0 1 0
0 0 0 0
Modified Matrix:
1 1 1 1
1 1 1 1
1 0 1 1
PS C:\Users\S. YATHISSH\OneDrive\Documents\DSA-Practice>
```

Time Complexity: O(m * n)

Space Complexity: O(1)

QUESTION

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

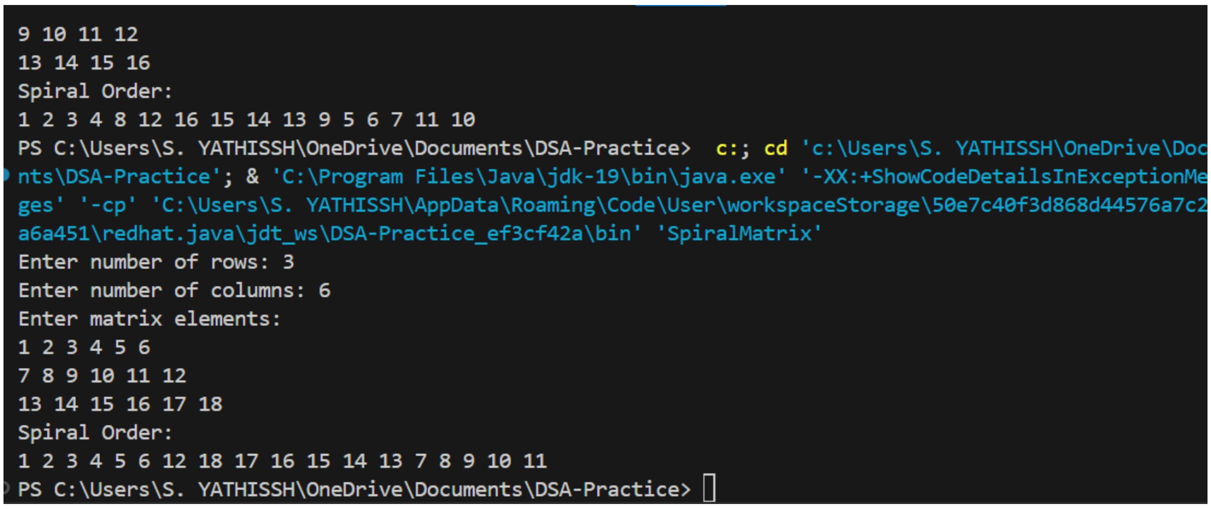
Explanation: The output is matrix in spiral format.

CODE:

```java
import java.util.Scanner;

public class SpiralMatrix {

    public static void printSpiral(int[][] mat, int m, int n) {
        int top = 0, bottom = m - 1, left = 0, right = n - 1;

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

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

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

    public static void main(String[] args) {
        Scanner s = new Scanner(System.in);
        System.out.print("Enter number of rows: ");
        int m = s.nextInt();
        System.out.print("Enter number of columns: ");
        int n = s.nextInt();
        int[][] mat = new int[m][n];

        System.out.println("Enter matrix elements:");
        for (int i = 0; i < m; i++) {
            for (int j = 0; j < n; j++) {
                mat[i][j] = s.nextInt();
            }
        }
```

```
            System.out.println("Spiral Order:");
            printSpiral(mat, m, n);

            s.close();
        }
    }
```

OUTPUT:



```
9 10 11 12
13 14 15 16
Spiral Order:
1 2 3 4 8 12 16 15 14 13 9 5 6 7 11 10
PS C:\Users\S. YATHISSH\OneDrive\Documents\DSA-Practice>  c:; cd 'c:\Users\S. YATHISSH\OneDrive\Doc
nts\DSA-Practice'; & 'C:\Program Files\Java\jdk-19\bin\java.exe' '-XX:+ShowCodeDetailsInExceptionMe
ges' '-cp' 'C:\Users\S. YATHISSH\AppData\Roaming\Code\User\workspaceStorage\50e7c40f3d868d44576a7c2
a6a451\redhat.java\jdt_ws\DSA-Practice_ef3cf42a\bin' 'SpiralMatrix'
Enter number of rows: 3
Enter number of columns: 6
Enter matrix elements:
1 2 3 4 5 6
7 8 9 10 11 12
13 14 15 16 17 18
Spiral Order:
1 2 3 4 5 6 12 18 17 16 15 14 13 7 8 9 10 11
PS C:\Users\S. YATHISSH\OneDrive\Documents\DSA-Practice>
```

Time Complexity: O(m * n)

Space Complexity: O(1)

QUESTION

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

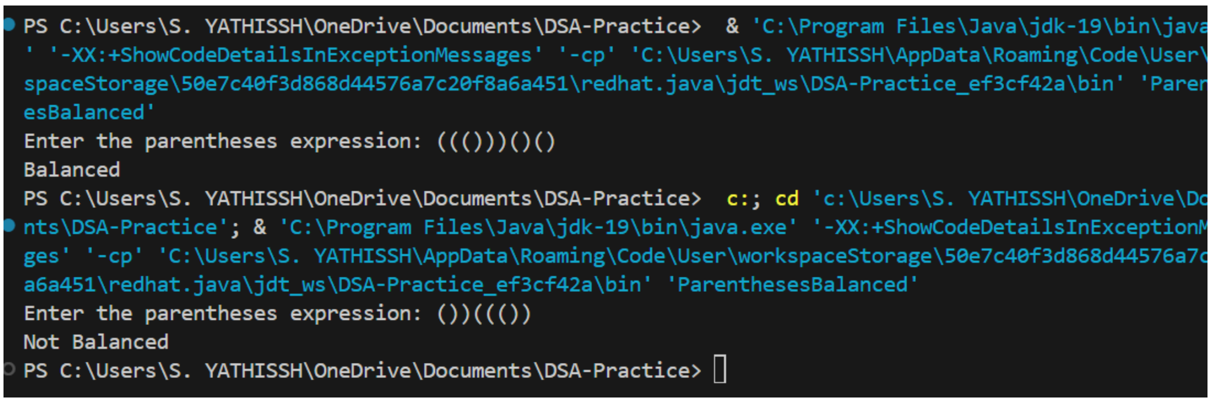
Input: str = “())((())”

Output: Not Balanced

CODE:

```java
import java.util.Scanner;

public class ParenthesesBalanced {

    public static String isBalanced(String str) {
        int count = 0;
        for (int i = 0; i < str.length(); i++) {
            if (str.charAt(i) == '(') {
                count++;
            } else if (str.charAt(i) == ')') {
                count--;
            }

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

        if (count == 0) {
            return "Balanced";
        } else {
            return "Not Balanced";
        }
    }

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the parentheses expression: ");
        String str = scanner.nextLine();

        System.out.println(isBalanced(str));

        scanner.close();
    }
}
```

OUTPUT:



```
PS C:\Users\S. YATHISSH\OneDrive\Documents\DSA-Practice>  & 'C:\Program Files\Java\jdk-19\bin\java
' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\S. YATHISSH\AppData\Roaming\Code\User\
spaceStorage\50e7c40f3d868d44576a7c20f8a6a451\redhat.java\jdt_ws\DSA-Practice_ef3cf42a\bin' 'Paren
esBalanced'
Enter the parentheses expression: ((()))()()
Balanced
PS C:\Users\S. YATHISSH\OneDrive\Documents\DSA-Practice>  c:; cd 'c:\Users\S. YATHISSH\OneDrive\Do
nts\DSA-Practice'; & 'C:\Program Files\Java\jdk-19\bin\java.exe' '-XX:+ShowCodeDetailsInExceptionM
ges' '-cp' 'C:\Users\S. YATHISSH\AppData\Roaming\Code\User\workspaceStorage\50e7c40f3d868d44576a7c
a6a451\redhat.java\jdt_ws\DSA-Practice_ef3cf42a\bin' 'ParenthesesBalanced'
Enter the parentheses expression: ())((())
Not Balanced
PS C:\Users\S. YATHISSH\OneDrive\Documents\DSA-Practice> 
```

Time Complexity: O(n)

Space Complexity: O(1)

QUESTION:

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”

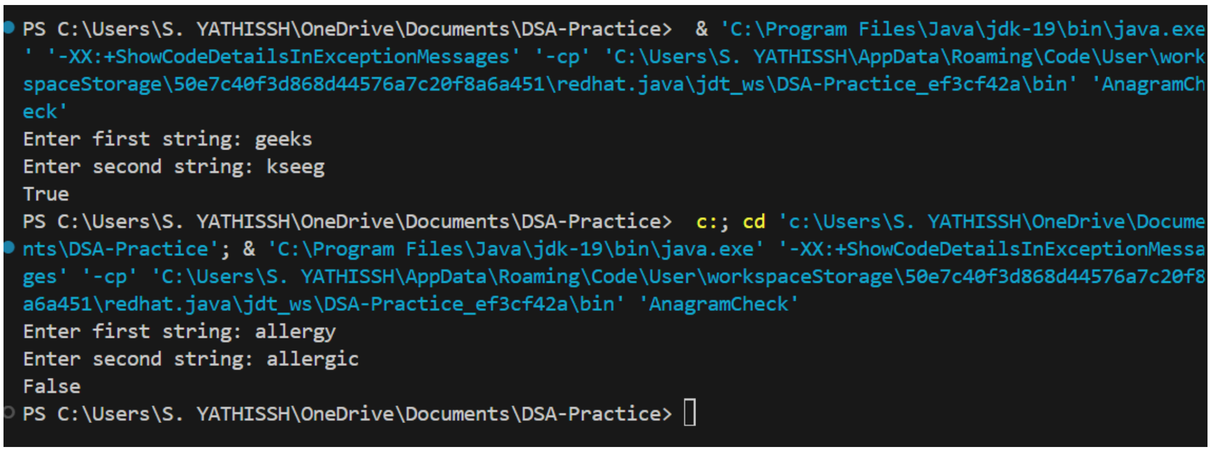
Output: true

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

CODE:

```
import java.util.Scanner;

public class AnagramCheck {

    public static boolean areAnagrams(String s1, String s2) {
        if (s1.length() != s2.length()) {
            return false;
        }

        int[] charCount = new int[26];

        for (int i = 0; i < s1.length(); i++) {
            charCount[s1.charAt(i) - 'a']++;
            charCount[s2.charAt(i) - 'a']--;
        }

        for (int count : charCount) {
            if (count != 0) {
                return false;
            }
        }

        return true;
    }

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter first string: ");
        String s1 = scanner.nextLine();

        System.out.print("Enter second string: ");
        String s2 = scanner.nextLine();

        if (areAnagrams(s1, s2)) {
            System.out.println("True");
        } else {
            System.out.println("False");
        }

        scanner.close();
    }
}
```

OUTPUT:



```
PS C:\Users\S. YATHISSH\OneDrive\Documents\DSA-Practice>  & 'C:\Program Files\Java\jdk-19\bin\java.exe
' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\S. YATHISSH\AppData\Roaming\Code\User\work
spaceStorage\50e7c40f3d868d44576a7c20f8a6a451\redhat.java\jdt_ws\DSA-Practice_ef3cf42a\bin' 'AnagramCh
eck'
Enter first string: geeks
Enter second string: kseeg
True
PS C:\Users\S. YATHISSH\OneDrive\Documents\DSA-Practice>  c:; cd 'c:\Users\S. YATHISSH\OneDrive\Docume
nts\DSA-Practice'; & 'C:\Program Files\Java\jdk-19\bin\java.exe' '-XX:+ShowCodeDetailsInExceptionMessa
ges' '-cp' 'C:\Users\S. YATHISSH\AppData\Roaming\Code\User\workspaceStorage\50e7c40f3d868d44576a7c20f8
a6a451\redhat.java\jdt_ws\DSA-Practice_ef3cf42a\bin' 'AnagramCheck'
Enter first string: allergy
Enter second string: allergic
False
PS C:\Users\S. YATHISSH\OneDrive\Documents\DSA-Practice> 
```

Time Complexity: O(n)

Space Complexity: O(1)

QUESTION

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”

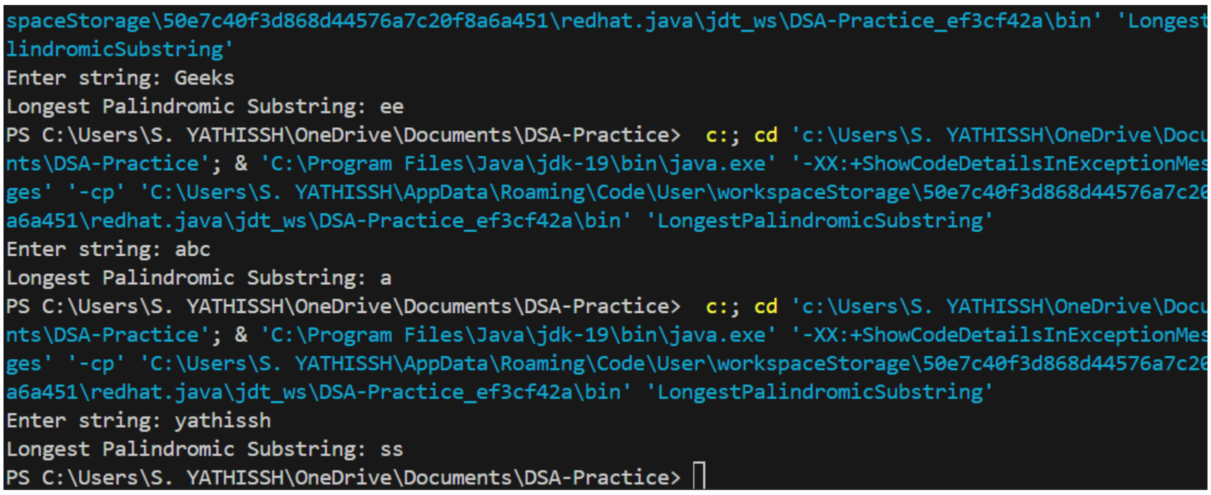
Input: str = “”

Output: “”

CODE:

```
import java.util.Scanner;

public class LongestPalindromicSubstring {

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

        int start = 0, maxLen = 1;

        for (int i = 0; i < s.length(); i++) {
            int len1 = expand(s, i, i);
            int len2 = expand(s, i, i + 1);
            int len = Math.max(len1, len2);
            if (len > maxLen) {
                maxLen = len;
                start = i - (len - 1) / 2;
            }
        }

        return s.substring(start, start + maxLen);
    }

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

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter string: ");
        String s = sc.nextLine();
        System.out.println("Longest Palindromic Substring: " +
lps(s));
        sc.close();
    }
}
```

OUTPUT:



```
spaceStorage\50e7c40f3d868d44576a7c20f8a6a451\redhat.java\jdt_ws\DSA-Practice_ef3cf42a\bin' 'Longest
lindromicSubstring'
Enter string: Geeks
Longest Palindromic Substring: ee
PS C:\Users\S. YATHISSH\OneDrive\Documents\DSA-Practice>  c:; cd 'c:\Users\S. YATHISSH\OneDrive\Docu
nts\DSA-Practice'; & 'C:\Program Files\Java\jdk-19\bin\java.exe' '-XX:+ShowCodeDetailsInExceptionMes
ges' '-cp' 'C:\Users\S. YATHISSH\AppData\Roaming\Code\User\workspaceStorage\50e7c40f3d868d44576a7c20
a6a451\redhat.java\jdt_ws\DSA-Practice_ef3cf42a\bin' 'LongestPalindromicSubstring'
Enter string: abc
Longest Palindromic Substring: a
PS C:\Users\S. YATHISSH\OneDrive\Documents\DSA-Practice>  c:; cd 'c:\Users\S. YATHISSH\OneDrive\Docu
nts\DSA-Practice'; & 'C:\Program Files\Java\jdk-19\bin\java.exe' '-XX:+ShowCodeDetailsInExceptionMes
ges' '-cp' 'C:\Users\S. YATHISSH\AppData\Roaming\Code\User\workspaceStorage\50e7c40f3d868d44576a7c20
a6a451\redhat.java\jdt_ws\DSA-Practice_ef3cf42a\bin' 'LongestPalindromicSubstring'
Enter string: yathissh
Longest Palindromic Substring: ss
PS C:\Users\S. YATHISSH\OneDrive\Documents\DSA-Practice> 
```

Time Complexity : O(n^2)

Space Complexity : O(1)

QUESTION

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”]

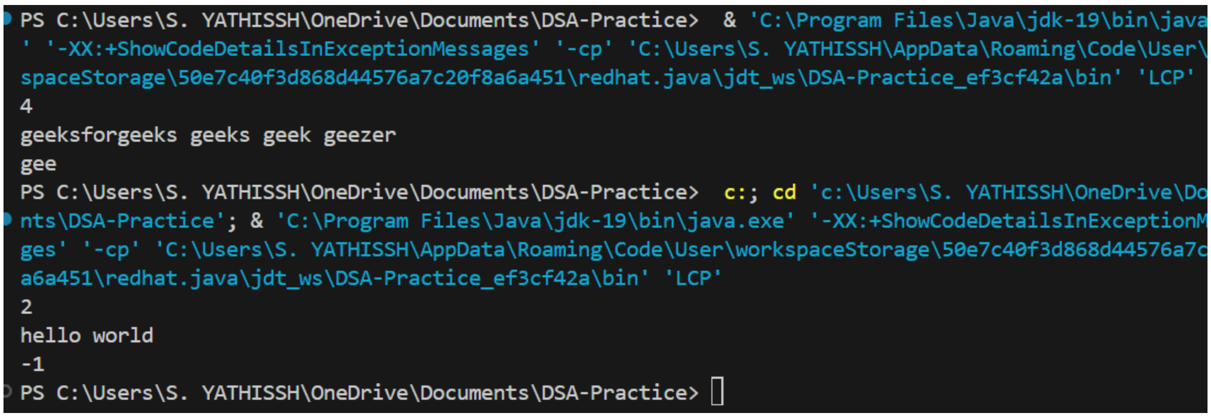
Output: -1

Explanation: There‟s no common prefix in the given string

CODE:

```
import java.util.Arrays;

import java.util.Scanner;


public class LCP {


    public static String lcp(String[] arr) {

        if (arr.length == 0) return "-1";

        Arrays.sort(arr);

        String f = arr[0], l = arr[arr.length - 1];

        int i = 0;

        while (i < f.length() && i < l.length() && f.charAt(i)
== l.charAt(i)) i++;

        return i == 0 ? "-1" : f.substring(0, i);

    }


    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        int n = sc.nextInt();

        String[] arr = new String[n];

        for (int i = 0; i < n; i++) arr[i] = sc.next();

        System.out.println(lcp(arr));

        sc.close();

    }
}
```

OUTPUT:



```
PS C:\Users\S. YATHISSH\OneDrive\Documents\DSA-Practice>  & 'C:\Program Files\Java\jdk-19\bin\java
' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\S. YATHISSH\AppData\Roaming\Code\User\
spaceStorage\50e7c40f3d868d44576a7c20f8a6a451\redhat.java\jdt_ws\DSA-Practice_ef3cf42a\bin' 'LCP'
4
geeksforgeeks geeks geek geezer
gee
PS C:\Users\S. YATHISSH\OneDrive\Documents\DSA-Practice>  c:; cd 'c:\Users\S. YATHISSH\OneDrive\Do
nts\DSA-Practice'; & 'C:\Program Files\Java\jdk-19\bin\java.exe' '-XX:+ShowCodeDetailsInExceptionM
ges' '-cp' 'C:\Users\S. YATHISSH\AppData\Roaming\Code\User\workspaceStorage\50e7c40f3d868d44576a7c
a6a451\redhat.java\jdt_ws\DSA-Practice_ef3cf42a\bin' 'LCP'
2
hello world
-1
PS C:\Users\S. YATHISSH\OneDrive\Documents\DSA-Practice>
```

Time Complexity : O(n log n + k)

Space Complexity : O(1)

QUESTION

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]

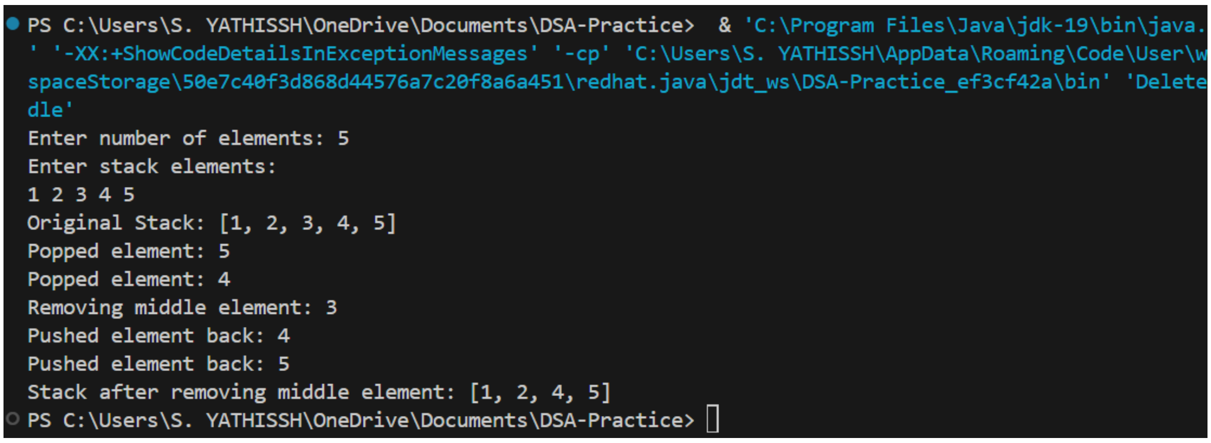
Input : Stack[] = [1, 2, 3, 4, 5, 6]

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

CODE:

```
import java.util.Stack;

import java.util.Scanner;


public class DeleteMiddle {


    public static void deleteMiddle(Stack<Integer> stack, int
middle) {
        if (middle == 0) {
            System.out.println("Removing middle element: " +
stack.peek());
            stack.pop();
            return;
        }

        int top = stack.pop();
        System.out.println("Popped element: " + top);

        deleteMiddle(stack, middle - 1);

        stack.push(top);
        System.out.println("Pushed element back: " + top);
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        Stack<Integer> stack = new Stack<>();
        System.out.print("Enter number of elements: ");
        int n = sc.nextInt();

        System.out.println("Enter stack elements:");
        for (int i = 0; i < n; i++) {
            int element = sc.nextInt();
            stack.push(element);
        }

        System.out.println("Original Stack: " + stack);
        int middle = n / 2;
        deleteMiddle(stack, middle);


        System.out.println("Stack after removing middle element:
" + stack);


        sc.close();
    }
}
```

OUTPUT:



```
PS C:\Users\S. YATHISSH\OneDrive\Documents\DSA-Practice>  & 'C:\Program Files\Java\jdk-19\bin\java.
' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\S. YATHISSH\AppData\Roaming\Code\User\w
spaceStorage\50e7c40f3d868d44576a7c20f8a6a451\redhat.java\jdt_ws\DSA-Practice_ef3cf42a\bin' 'Delete
dle'
Enter number of elements: 5
Enter stack elements:
1 2 3 4 5
Original Stack: [1, 2, 3, 4, 5]
Popped element: 5
Popped element: 4
Removing middle element: 3
Pushed element back: 4
Pushed element back: 5
Stack after removing middle element: [1, 2, 4, 5]
PS C:\Users\S. YATHISSH\OneDrive\Documents\DSA-Practice>
```

Time Complexity : O(n)

Space Complexity : O(n)

QUESTION

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

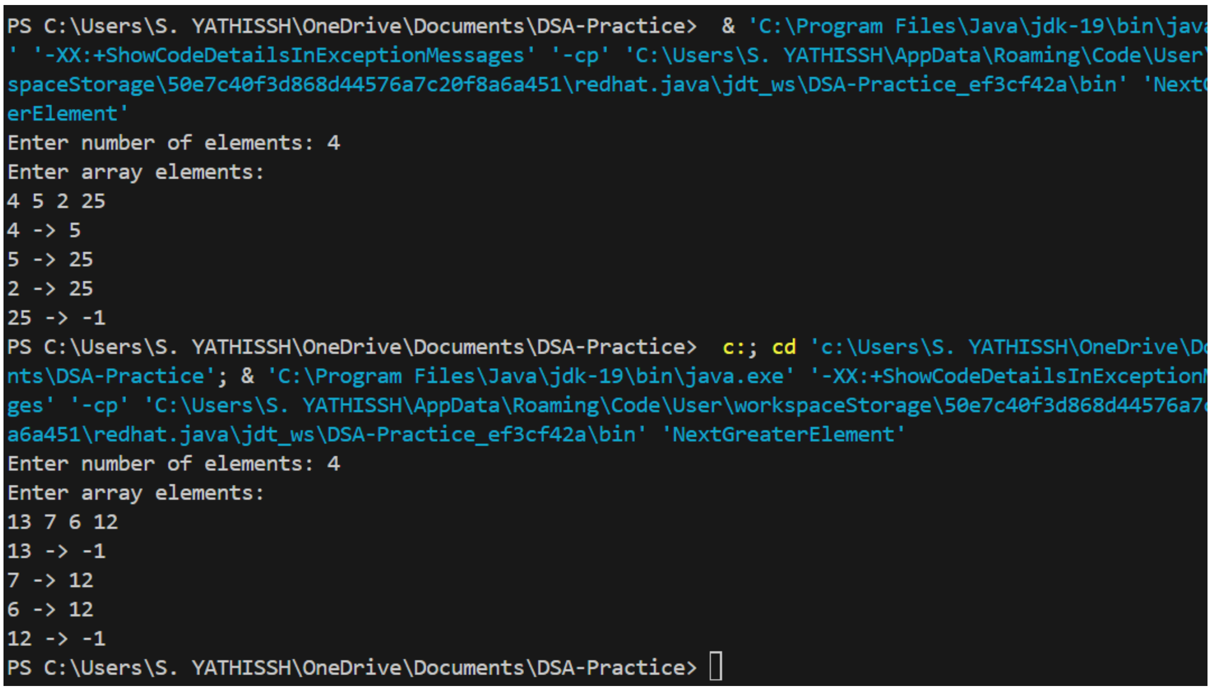
12 –> -1

Explanation: 13 and 12 don"t have any element greater than them present on the right side

CODE:

```
import java.util.Scanner;

import java.util.Stack;


public class NextGreaterElement {


    public static void printNGE(int[] arr) {

        int n = arr.length;

        int[] nge = new int[n];

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


        for (int i = n - 1; i >= 0; i--) {

            while (!stack.isEmpty() && stack.peek() <= arr[i]) {

                stack.pop();

            }


            nge[i] = stack.isEmpty() ? -1 : stack.peek();

            stack.push(arr[i]);

        }
        for (int i = 0; i < n; i++) {
            System.out.println(arr[i] + " -> " + nge[i]);
        }
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        System.out.print("Enter number of elements: ");
        int n = sc.nextInt();
        int[] arr = new int[n];

        System.out.println("Enter array elements:");
        for (int i = 0; i < n; i++) {
            arr[i] = sc.nextInt();
        }

        printNGE(arr);

        sc.close();
    }
}
```

OUTPUT:



```
PS C:\Users\S. YATHISSH\OneDrive\Documents\DSA-Practice>  & 'C:\Program Files\Java\jdk-19\bin\java
' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\S. YATHISSH\AppData\Roaming\Code\User
spaceStorage\50e7c40f3d868d44576a7c20f8a6a451\redhat.java\jdt_ws\DSA-Practice_ef3cf42a\bin' 'Next
erElement'
Enter number of elements: 4
Enter array elements:
4 5 2 25
4 -> 5
5 -> 25
2 -> 25
25 -> -1
PS C:\Users\S. YATHISSH\OneDrive\Documents\DSA-Practice>  c:; cd 'c:\Users\S. YATHISSH\OneDrive\Do
nts\DSA-Practice'; & 'C:\Program Files\Java\jdk-19\bin\java.exe' '-XX:+ShowCodeDetailsInException
ges' '-cp' 'C:\Users\S. YATHISSH\AppData\Roaming\Code\User\workspaceStorage\50e7c40f3d868d44576a7
a6a451\redhat.java\jdt_ws\DSA-Practice_ef3cf42a\bin' 'NextGreaterElement'
Enter number of elements: 4
Enter array elements:
13 7 6 12
13 -> -1
7 -> 12
6 -> 12
12 -> -1
PS C:\Users\S. YATHISSH\OneDrive\Documents\DSA-Practice> 
```

Time Complexity : O(n)

Space Complexity : O(n)

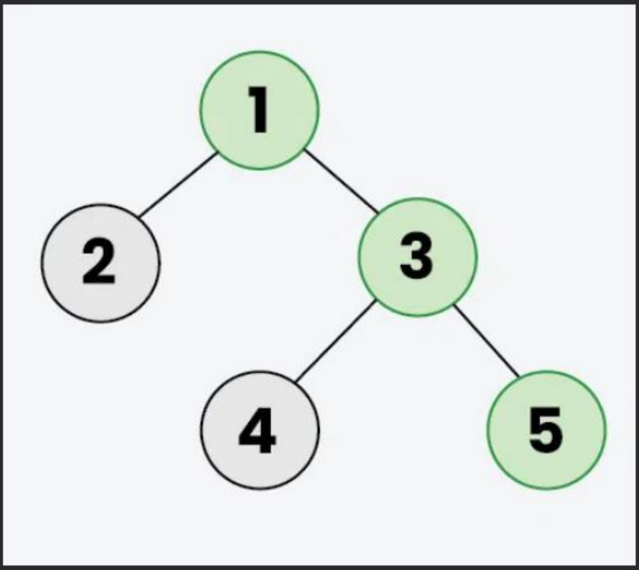
QUESTION

19. Print Right View of a Binary Tree

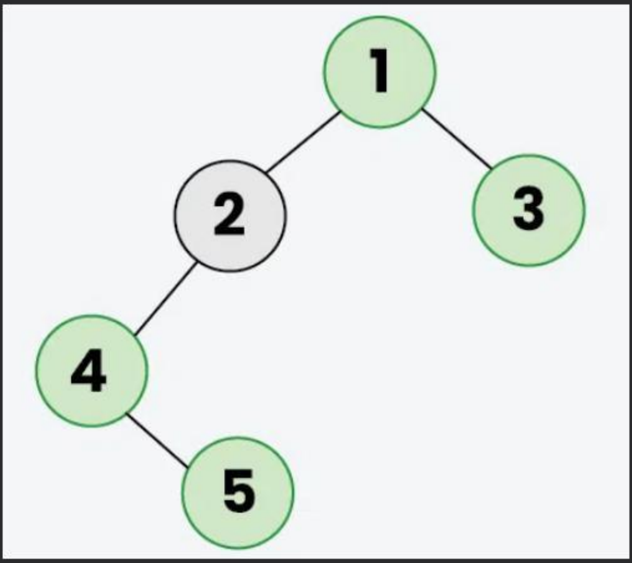
Given a Binary Tree, the task is to print the Right view of it. The right view of a Binary Tree is a

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.*



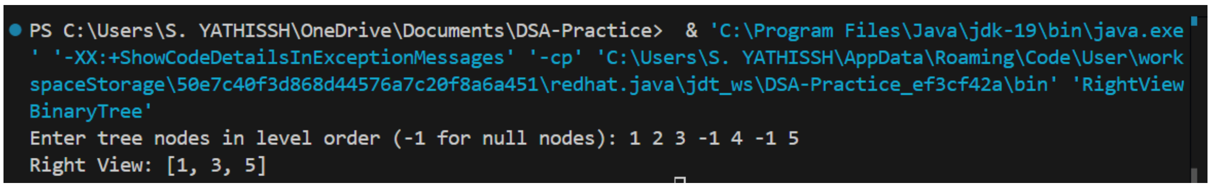
*Example 2: The **Green** colored nodes (1, 3, 4, 5) represents the Right view in the below Binary tree.*



CODE:

```java
import java.util.*;

class TreeNode {
    int val;
    TreeNode left, right;

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

public class RightViewBinaryTree {

    public static TreeNode buildTree(List<Integer> nodes) {
        if (nodes.isEmpty() || nodes.get(0) == -1) return null;

        TreeNode root = new TreeNode(nodes.get(0));
        Queue<TreeNode> queue = new LinkedList<>();
        queue.offer(root);

        int i = 1;
        while (i < nodes.size()) {
            TreeNode current = queue.poll();

            if (nodes.get(i) != -1) {
                current.left = new TreeNode(nodes.get(i));
                queue.offer(current.left);
            }
            i++;

            if (i < nodes.size() && nodes.get(i) != -1) {
                current.right = new TreeNode(nodes.get(i));
                queue.offer(current.right);
            }
            i++;
        }
        return root;
    }

    public static List<Integer> rightView(TreeNode root) {
        List<Integer> result = new ArrayList<>();
        if (root == null) return result;

        Queue<TreeNode> queue = new LinkedList<>();
        queue.offer(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.val);
                }
                if (node.left != null) queue.offer(node.left);
                if (node.right != null) queue.offer(node.right);
            }
        }
        return result;
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        System.out.print("Enter tree nodes in level order (-1
for null nodes): ");
        String[] input = sc.nextLine().split(" ");
        List<Integer> nodes = new ArrayList<>();

        for (String s : input) {
            nodes.add(Integer.parseInt(s));
        }

        TreeNode root = buildTree(nodes);

        List<Integer> rightViewNodes = rightView(root);
        System.out.println("Right View: " + rightViewNodes);

        sc.close();
    }
}
```

OUTPUT:



```
PS C:\Users\S. YATHISSH\OneDrive\Documents\DSA-Practice>  & 'C:\Program Files\Java\jdk-19\bin\java.exe
' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\S. YATHISSH\AppData\Roaming\Code\User\work
spaceStorage\50e7c40f3d868d44576a7c20f8a6a451\redhat.java\jdt_ws\DSA-Practice_ef3cf42a\bin' 'RightView
BinaryTree'
Enter tree nodes in level order (-1 for null nodes): 1 2 3 -1 4 -1 5
Right View: [1, 3, 5]
```

Time Complexity : O(n)

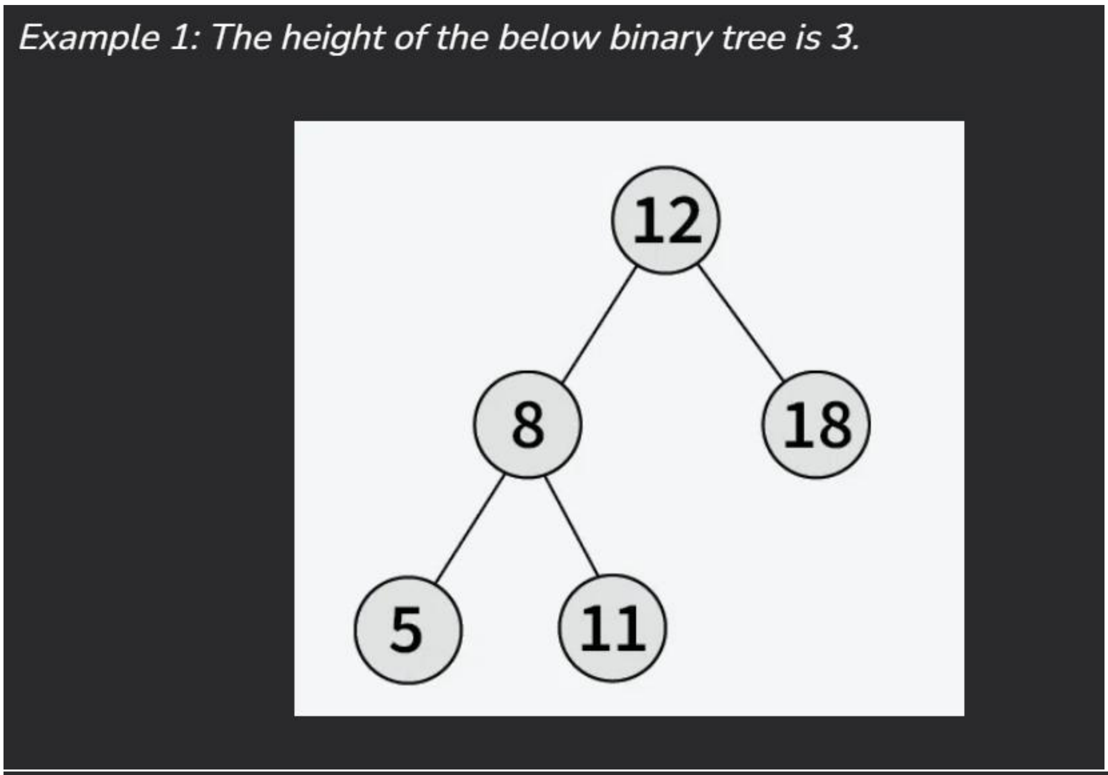
Space Complexity : O(n)

QUESTION

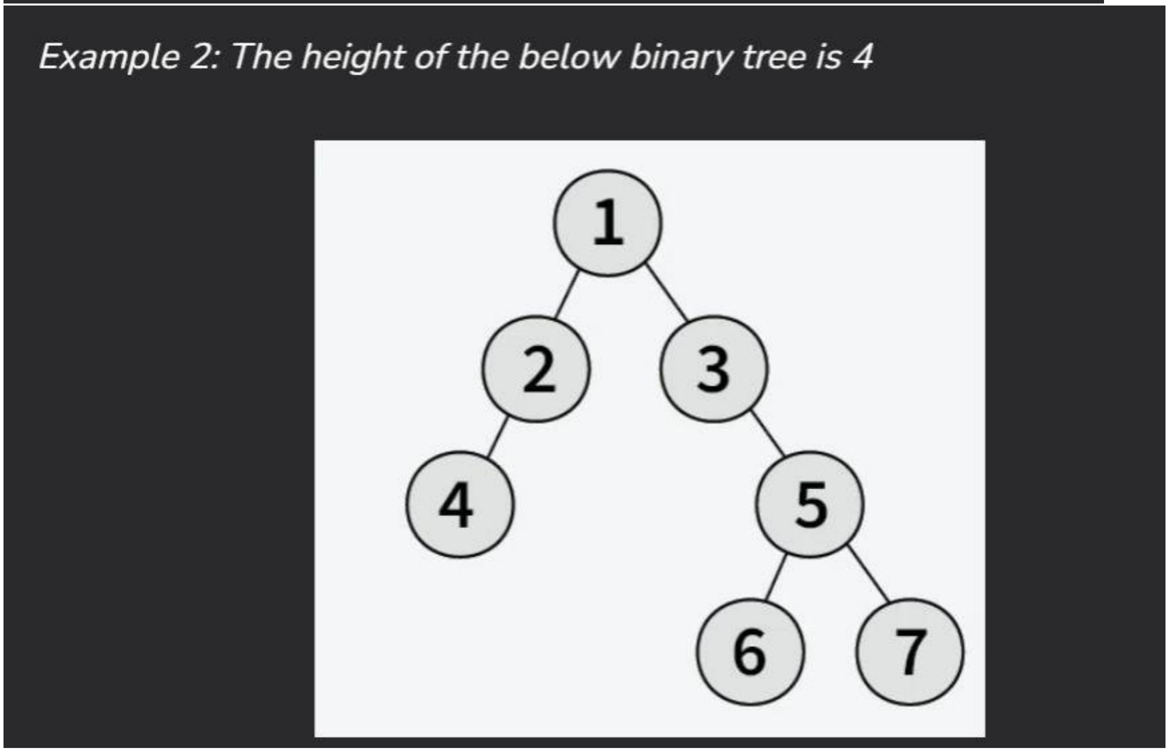
20. Maximum Depth or Height of Binary Tree

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.



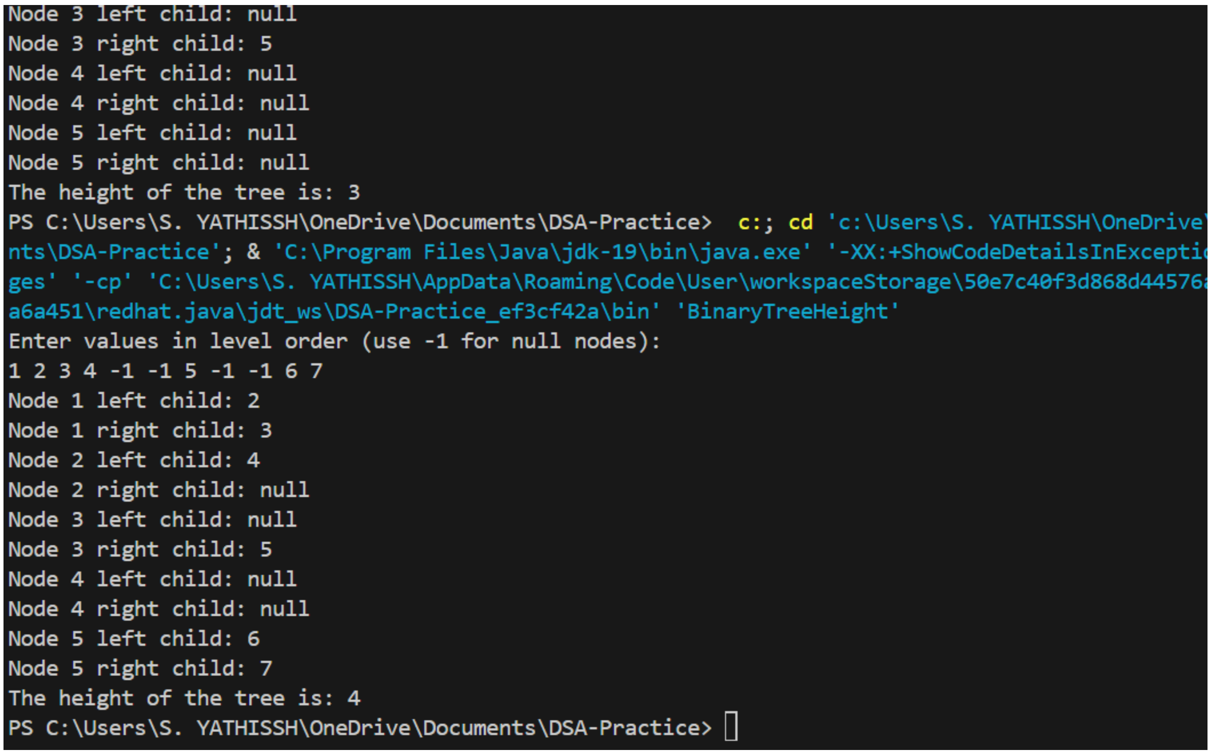
Example 2: The height of the below binary tree is 4

CODE:

```
import java.util.*;


class TreeNode {

    int val;

    TreeNode left, right;

    TreeNode(int x) { val = x; }

}


public class BinaryTreeHeight {

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        System.out.println("Enter values in level order (use -1 
for null nodes):");

        String[] values = sc.nextLine().split(" ");


        TreeNode root = buildTree(values);

        System.out.println("The height of the tree is: " + 
maxDepth(root));

    }


    public static TreeNode buildTree(String[] values) {

        if (values.length == 0 || values[0].equals("-1")) return 
null;


        TreeNode root = new 
TreeNode(Integer.parseInt(values[0]));

        Queue<TreeNode> queue = new LinkedList<>();

        queue.add(root);

        int i = 1;
        while (i < values.length) {
            TreeNode current = queue.poll();

            // Assign left child
            if (!values[i].equals("-1")) {
                current.left = new
TreeNode(Integer.parseInt(values[i]));
                queue.add(current.left);
            }
            System.out.println("Node " + current.val + " left
child: " + (current.left != null ? current.left.val : "null"));
            i++;

            // Assign right child if available
            if (i < values.length && !values[i].equals("-1")) {
                current.right = new
TreeNode(Integer.parseInt(values[i]));
                queue.add(current.right);
            }
            System.out.println("Node " + current.val + " right
child: " + (current.right != null ? current.right.val :
"null"));
            i++;
        }

        return root;
    }

    public static int maxDepth(TreeNode root) {
        if (root == null) return 0;
        int leftDepth = maxDepth(root.left);
        int rightDepth = maxDepth(root.right);
```

```java
        return Math.max(leftDepth, rightDepth) + 1;
    }
}
```

OUTPUT:



```
Node 3 left child: null
Node 3 right child: 5
Node 4 left child: null
Node 4 right child: null
Node 5 left child: null
Node 5 right child: null
The height of the tree is: 3
PS C:\Users\S. YATHISSH\OneDrive\Documents\DSA-Practice>  c:; cd 'c:\Users\S. YATHISSH\OneDrive
nts\DSA-Practice'; & 'C:\Program Files\Java\jdk-19\bin\java.exe' '-XX:+ShowCodeDetailsInExcepti
ges' '-cp' 'C:\Users\S. YATHISSH\AppData\Roaming\Code\User\workspaceStorage\50e7c40f3d868d44576
a6a451\redhat.java\jdt_ws\DSA-Practice_ef3cf42a\bin' 'BinaryTreeHeight'
Enter values in level order (use -1 for null nodes):
1 2 3 4 -1 -1 5 -1 -1 6 7
Node 1 left child: 2
Node 1 right child: 3
Node 2 left child: 4
Node 2 right child: null
Node 3 left child: null
Node 3 right child: 5
Node 4 left child: null
Node 4 right child: null
Node 5 left child: 6
Node 5 right child: 7
The height of the tree is: 4
PS C:\Users\S. YATHISSH\OneDrive\Documents\DSA-Practice>
```

Time Complexity : O(n)

Space Complexity : O(n)