Rajalakshmi Engineering College

Name: Jai Shuriya J
Email: 240701200@rajalakshmi.edu.in
Roll no: 240701200
Phone: 8754381941
Branch: REC
Department: CSE - Section 10
Batch: 2028
Degree: B.E - CSE

Scan to verify results

2024_28_III_OOPS Using Java Lab

REC_2028_OOPS using Java_Week 3_CY

Attempt : 1
Total Mark : 40
Marks Obtained : 40

Section 1 : Coding

1. Problem Statement:

Mason is participating in a coding challenge where he must manipulate an integer array. His task is to replace every element in the array with the next greatest element to its right. The last element of the array remains unchanged, as there is no element to its right.

Your job is to help Mason write a program that performs this transformation and outputs the modified array.

Input Format

The first line of input contains an integer n representing the number of elements in the array.

The second line of input contains n space-separated integers representing the elements of the array.

Output Format

The output prints the modified array of n integers, where each element (except the last one) is replaced by the maximum element to its right, and the last element remains unchanged.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 6
12 3 91 15 12 14

Output: 91 91 15 14 14 14

Answer

```
// You are using Java
import java.util.*;

public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        int[] arr = new int[n];

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

        int max = arr[n - 1];
        for (int i = n - 2; i >= 0; i--) {
            int temp = arr[i];
            arr[i] = max;
            if (temp > max) {
                max = temp;
            }
        }

        for (int i = 0; i < n; i++) {
            System.out.print(arr[i] + " ");
        }
```

```
    }
}
```

Status : Correct **Marks : 10/10**

2. Problem Statement

Rina is managing the inventory for a library, where each row of a 2D matrix represents the number of different genres of books available on each shelf.

She wants to perform the following operations:

Transformation: Replace each element in a row with the sum of all elements in that row.Merging: After transformation, Rina will provide one additional matrix, and specify whether to merge the transformed matrix with this new matrix row-wise or column-wise.

Input Format

The first line contains two integers R and C, representing the number of rows and columns of the initial matrix.

The next R lines contain C space-separated integers, representing the book counts in the library.

The next line contains two integers MR and MC, representing the dimensions of the second matrix (to be merged).

The next MR lines contain MC space-separated integers, representing the second matrix.

The last line contains an integer mergeType:

- 0 Row-wise merging (append the second matrix below the transformed matrix).
- 1 Column-wise merging (append the second matrix to the right of the transformed matrix).

Output Format

The output prints "Transformed matrix: "followed by the transformed 2D matrix where each element in a row is replaced with the sum of the elements in that row.

The output prints "Final merged matrix: ", followed by the merging based on mergeType.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 3 4
8 2 4 9
4 5 6 1
7 8 9 3
2 4
3 5 7 2
6 1 4 9
0
Output: Transformed matrix:
23 23 23 23
16 16 16 16
27 27 27 27
Final merged matrix:
23 23 23 23
16 16 16 16
27 27 27 27
3 5 7 2
6 1 4 9

Answer

```
// You are using Java
import java.util.*;

public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int R = sc.nextInt();
        int C = sc.nextInt();
        int[][] matrix1 = new int[R][C];
        for (int i = 0; i < R; i++) {
            for (int j = 0; j < C; j++) {
                matrix1[i][j] = sc.nextInt();
```

```
    }
}

int[][] transformed = new int[R][C];
for (int i = 0; i < R; i++) {
    int sum = 0;
    for (int j = 0; j < C; j++) {
        sum += matrix1[i][j];
    }
    Arrays.fill(transformed[i], sum);
}

int MR = sc.nextInt();
int MC = sc.nextInt();
int[][] matrix2 = new int[MR][MC];
for (int i = 0; i < MR; i++) {
    for (int j = 0; j < MC; j++) {
        matrix2[i][j] = sc.nextInt();
    }
}

int mergeType = sc.nextInt();

System.out.print("Transformed matrix: ");
for (int i = 0; i < R; i++) {
    for (int j = 0; j < C; j++) {
        System.out.print(transformed[i][j] + " ");
    }
}

System.out.print("Final merged matrix: ");
if (mergeType == 0) {
    for (int i = 0; i < R; i++) {
        for (int j = 0; j < C; j++) {
            System.out.print(transformed[i][j] + " ");
        }
    }
    for (int i = 0; i < MR; i++) {
        for (int j = 0; j < MC; j++) {
            System.out.print(matrix2[i][j] + " ");
        }
    }
```

```
        } else {
            int maxRows = Math.max(R, MR);
            for (int i = 0; i < maxRows; i++) {
                if (i < R) {
                    for (int j = 0; j < C; j++) {
                        System.out.print(transformed[i][j] + " ");
                    }
                } else {
                    for (int j = 0; j < C; j++) {
                        System.out.print("0 ");
                    }
                }
                if (i < MR) {
                    for (int j = 0; j < MC; j++) {
                        System.out.print(matrix2[i][j] + " ");
                    }
                } else {
                    for (int j = 0; j < MC; j++) {
                        System.out.print("0 ");
                    }
                }
            }
        }
    }
}
```

Status : Correct **Marks : 10/10**

3. Problem Statement:

Emma, a budding computer vision enthusiast, is working on a challenging image processing project. She has a square image represented as a 2D matrix of integers. As part of a special filter operation, she needs to rotate the image by 90 degrees clockwise, but there's a twist — she must perform the rotation in-place, using no extra space.

This means Emma has to rotate the matrix without creating a new one. Your task is to help her implement a Java program that takes this square matrix as input and rotates it within the same structure.

Can you help Emma efficiently rotate the image so that her project can move to the next stage?

Input Format

The first line of input contains a single integer n, representing the number of rows and columns of the square matrix (i.e., the matrix is of size n x n).

The next n lines each contain n space-separated integers, representing the elements of each row of the 2D array.

Output Format

The first line of output prints "Rotated 2D Array:"

The next n lines of output print the rotated matrix.

Each line contains n space-separated integers representing a row of the rotated matrix.

Refer to the sample output for format specification.

Sample Test Case

Input: 3
1 2 3
4 5 6
7 8 9
Output: Rotated 2D Array:
7 4 1
8 5 2
9 6 3

Answer

```
// You are using Java
import java.util.*;

public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
```

```
        int[][] matrix = new int[n][n];
        for (int i = 0; i < n; i++) {
            for (int j = 0; j < n; j++) {
                matrix[i][j] = sc.nextInt();
            }
        }

        for (int i = 0; i < n; i++) {
            for (int j = i + 1; j < n; j++) {
                int temp = matrix[i][j];
                matrix[i][j] = matrix[j][i];
                matrix[j][i] = temp;
            }
        }

        for (int i = 0; i < n; i++) {
            int left = 0, right = n - 1;
            while (left < right) {
                int temp = matrix[i][left];
                matrix[i][left] = matrix[i][right];
                matrix[i][right] = temp;
                left++;
                right--;
            }
        }

        System.out.println("Rotated 2D Array:");
        for (int i = 0; i < n; i++) {
            for (int j = 0; j < n; j++) {
                System.out.print(matrix[i][j] + " ");
            }
            System.out.println();
        }
    }
}
```

Status : Correct **Marks : 10/10**

4. Problem Statement

Nikila is working as an intern in a software firm and is practicing with a

matrix where each row represents a set of numerical values. Her task is to identify the row with the highest sum of its elements and remove that row from the matrix. After removing the row with the highest sum, Nikila needs to print the updated matrix.

Your task is to help Nikila in implementing the same. If there are two or more rows that have same the highest sum, the firstly encountered row is deleted.

Input Format

The first line of the input consists of two space-separated integers, R and C, representing the number of rows and columns in the matrix, respectively.

The following R lines each contain, C space-separated integers representing the matrix elements.

Output Format

The output prints the matrix after removing the row with the highest sum. Each row should be printed on a new line, with elements separated by a space.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 2 2
1 2
3 4

Output: 1 2

Answer

```
// You are using Java
import java.util.*;

public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int R = sc.nextInt();
        int C = sc.nextInt();
```

```
        int[][] matrix = new int[R][C];
        for (int i = 0; i < R; i++) {
            for (int j = 0; j < C; j++) {
                matrix[i][j] = sc.nextInt();
            }
        }

        int maxSum = Integer.MIN_VALUE;
        int maxRowIndex = -1;
        for (int i = 0; i < R; i++) {
            int rowSum = 0;
            for (int j = 0; j < C; j++) {
                rowSum += matrix[i][j];
            }
            if (rowSum > maxSum) {
                maxSum = rowSum;
                maxRowIndex = i;
            }
        }

        for (int i = 0; i < R; i++) {
            if (i == maxRowIndex) continue;
            for (int j = 0; j < C; j++) {
                System.out.print(matrix[i][j] + " ");
            }
            System.out.println();
        }
    }
}
```

Status : Correct **Marks : 10/10**