Rajalakshmi Engineering College

Name: Gayathri Boopathy
Email: 240701141@rajalakshmi.edu.in
Roll no: 240701141
Phone: 9363837860
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.Scanner;
class NextGreatestElement {
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);

      // Read array size
      int n = sc.nextInt();
      int[] arr = new int[n];

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

      // Start from the rightmost element
      int maxRight = arr[n - 1];

      // Traverse from second last element to the first
      for (int i = n - 2; i >= 0; i--) {
         int temp = arr[i];
         arr[i] = maxRight;   // replace with max to the right
         if (temp > maxRight) {
            maxRight = temp; // update maxRight
         }
```

```
        }

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

        sc.close();
    }
}
```

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

2. Problem Statement:

Imagine you have an array of integer values, and you're tasked with identifying a pair of elements within the array. This pair of elements should have a sum that is the closest to zero when compared to any other pair in the array.

Your goal is to create a program that solves this problem efficiently. The program should accept an array of integers and return the pair of elements whose sum is closest to zero.

Input Format

The first line of the input is an integer N representing the size of the array.

The second line of the input contains N space-separated integer values.

Output Format

The output is displayed in the following format:

"Pair with the sum closest to zero: {value} and {value}"

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 5
9 10 -3 -5 -2
Output: Pair with the sum closest to zero: 9 and -5

Answer

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

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

        // Read size of array
        int N = sc.nextInt();
        int[] arr = new int[N];

        // Read array elements
        for (int i = 0; i < N; i++) {
            arr[i] = sc.nextInt();
        }

        int minSum = Integer.MAX_VALUE;
        int num1 = arr[0], num2 = arr[1];

        // Check all pairs
        for (int i = 0; i < N - 1; i++) {
            for (int j = i + 1; j < N; j++) {
                int sum = arr[i] + arr[j];
                if (Math.abs(sum) < Math.abs(minSum)) {
                    minSum = sum;
                    num1 = arr[i];
                    num2 = arr[j];
                }
            }
        }

        // Print result in required format
        System.out.println("Pair with the sum closest to zero: " + num1 + " and " +
num2);

        sc.close();
    }
```

}

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

3. 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.Scanner;

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

        // First matrix input
        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();
    }
}

// Second matrix input
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();
    }
}

// Merge type
int mergeType = sc.nextInt();

// Step 1: Transformation
int[][] transformed = new int[R][C];
for (int i = 0; i < R; i++) {
    int rowSum = 0;
    for (int j = 0; j < C; j++) {
        rowSum += matrix[i][j];
    }
    for (int j = 0; j < C; j++) {
        transformed[i][j] = rowSum;
    }
}

// Print transformed matrix
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: ");
```

```
        // Step 2: Merging
        if (mergeType == 0) {
            // Row-wise: append matrix2 below
            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 {
            // Column-wise: append matrix2 to the right
            for (int i = 0; i < R; i++) {
                for (int j = 0; j < C; j++) {
                    System.out.print(transformed[i][j] + " ");
                }
                for (int j = 0; j < MC; j++) {
                    System.out.print(matrix2[i][j] + " ");
                }
            }
        }

        sc.close();
    }
}
```

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

4. 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

```
import java.util.Scanner;

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

```
Scanner sc = new Scanner(System.in);

// Read size of square matrix
int n = sc.nextInt();
int[][] mat = new int[n][n];

// Read matrix elements
for (int i = 0; i < n; i++) {
    for (int j = 0; j < n; j++) {
        mat[i][j] = sc.nextInt();
    }
}

// Step 1: Transpose the matrix
for (int i = 0; i < n; i++) {
    for (int j = i + 1; j < n; j++) {
        int temp = mat[i][j];
        mat[i][j] = mat[j][i];
        mat[j][i] = temp;
    }
}

// Step 2: Reverse each row
for (int i = 0; i < n; i++) {
    int left = 0, right = n - 1;
    while (left < right) {
        int temp = mat[i][left];
        mat[i][left] = mat[i][right];
        mat[i][right] = temp;
        left++;
        right--;
    }
}

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

```
        sc.close();
    }
}
```

Status : Correct *Marks : 10/10*