# Rajalakshmi Engineering College

Name: Adithya B
Email: 240701018@rajalakshmi.edu.in
Roll no: 240701018
Phone: 9444117405
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

Emma is a data analyst working with a grid-based system where each cell contains important numerical data. The grid represents spatial data, inventory records, or structured reports that require periodic updates.

Due to system updates and new requirements, Emma needs to modify the grid in the following ways:

She wants to insert either a new row or a new column at a given position.Later, she needs to delete either a row or a column from the modified matrix.

***Input Format***

The first line contains two integers rows and cols (the dimensions of the matrix).

The next rows lines contain cols space-separated integers representing the initial matrix.

The next line contains two integers insertType and insertIndex:

- insertType = 0 for row insertion, 1 for column insertion.
- insertIndex is the position where the new row/column should be added.

If inserting a row, the next cols integers represent the new row or If inserting a column, the next rows integers represent the new column.

The next line contains two integers deleteType and deleteIndex:

- deleteType = 0 for row deletion, 1 for column deletion.
- deleteIndex is the position to be deleted.

***Output Format***

The first line of output prints the string "After insertion" followed by the modified matrix with the inserted row or column.

Each row of the matrix is printed on a new line with space-separated integers.

The next line prints the string "After deletion" followed by the final matrix after the specified deletion operation.

Each row of the resulting matrix is printed on a new line with space-separated integers.

Refer to the sample output for formatting specifications.

***Sample Test Case***

Input: 3 3
1 2 3
4 5 6
7 8 9
0 1
10 11 12
1 2
Output: After insertion

1 2 3
10 11 12
4 5 6
7 8 9
After deletion
1 2
10 11
4 5
7 8

***Answer***

```
import java.util.Scanner;

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


        int rows = scanner.nextInt();
        int cols = scanner.nextInt();
        int[][] matrix = new int[rows][cols];
        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < cols; j++) {
                matrix[i][j] = scanner.nextInt();
            }
        }

        // Read insertion type and index
        int insertType = scanner.nextInt(); // 0 for row, 1 for column
        int insertIndex = scanner.nextInt();

        // Perform row or column insertion
        int[][] newMatrix;
        if (insertType == 0) { // Insert row
            newMatrix = new int[rows + 1][cols];
            for (int i = 0, ni = 0; i <= rows; i++, ni++) {
                if (i == insertIndex) {
                    for (int j = 0; j < cols; j++) {
                        newMatrix[i][j] = scanner.nextInt();
                    }
                    ni--;
                } else {
```

```
            for (int j = 0; j < cols; j++) {
                newMatrix[i][j] = matrix[ni][j];
            }
        }
    }
    rows++;
} else { // Insert column
    newMatrix = new int[rows][cols + 1];
    for (int i = 0; i < rows; i++) {
        for (int j = 0, nj = 0; j <= cols; j++, nj++) {
            if (j == insertIndex) {
                newMatrix[i][j] = scanner.nextInt();
                nj--;
            } else {
                newMatrix[i][j] = matrix[i][nj];
            }
        }
    }
    cols++;
}

// Print matrix after insertion
System.out.println("After insertion");
for (int i = 0; i < rows; i++) {
    for (int j = 0; j < cols; j++) {
        System.out.print(newMatrix[i][j] + " ");
    }
    System.out.println();
}

// Read deletion type and index
int deleteType = scanner.nextInt(); // 0 for row, 1 for column
int deleteIndex = scanner.nextInt();

// Perform row or column deletion
int[][] finalMatrix;
if (deleteType == 0) { // Delete row
    finalMatrix = new int[rows - 1][cols];
    for (int i = 0, ni = 0; i < rows; i++) {
        if (i == deleteIndex) {
            continue;
        }
```

```
            for (int j = 0; j < cols; j++) {
                finalMatrix[ni][j] = newMatrix[i][j];
            }
            ni++;
        }
        rows--;
    } else { // Delete column
        finalMatrix = new int[rows][cols - 1];
        for (int i = 0; i < rows; i++) {
            for (int j = 0, nj = 0; j < cols; j++) {
                if (j == deleteIndex) {
                    continue;
                }
                finalMatrix[i][nj++] = newMatrix[i][j];
            }
        }
        cols--;
    }

    // Print matrix after deletion
    System.out.println("After deletion");
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            System.out.print(finalMatrix[i][j] + " ");
        }
        System.out.println();
    }
    scanner.close();
  }
}
```

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

2. 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***

```
import java.util.Scanner;

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

      int rows = scanner.nextInt();
      int cols = scanner.nextInt();
      int[][] matrix = new int[rows][cols];

      for (int i = 0; i < rows; i++) {
         for (int j = 0; j < cols; j++) {
            matrix[i][j] = scanner.nextInt();
```

```
        }
        }

        int maxSumRowIndex = 0;
        int maxRowSum = Integer.MIN_VALUE;

        for (int i = 0; i < rows; i++) {
            int rowSum = 0;
            for (int j = 0; j < cols; j++) {
                rowSum += matrix[i][j];
            }
            if (rowSum > maxRowSum) {
                maxRowSum = rowSum;
                maxSumRowIndex = i;
            }
        }

        int[][] newMatrix = new int[rows - 1][cols];
        int newRowIndex = 0;

        for (int i = 0; i < rows; i++) {
            if (i == maxSumRowIndex) {
                continue;
            }
            for (int j = 0; j < cols; j++) {
                newMatrix[newRowIndex][j] = matrix[i][j];
            }
            newRowIndex++;
        }

        for (int i = 0; i < rows - 1; i++) {
            for (int j = 0; j < cols; j++) {
                System.out.print(newMatrix[i][j] + " ");
            }
            System.out.println();
        }

        scanner.close();
    }
}
```

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

3. 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***

```
import java.util.*;

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

      // Read array size
      int n = scanner.nextInt();
```

```
        // Read array elements
        int[] arr = new int[n];
        for (int i = 0; i < n; i++) {
            arr[i] = scanner.nextInt();
        }
        findClosestSumPair(arr, n);

        scanner.close();
    }

    public static void findClosestSumPair(int[] arr, int n) {
        int minSum = Integer.MAX_VALUE;
        int first = 0, second = 0;

        // Check all possible pairs
        for (int i = 0; i < n - 1; i++) {
            for (int j = i + 1; j < n; j++) {
                int sum = arr[i] + arr[j];

                // Check if this sum is closer to zero
                if (Math.abs(sum) < Math.abs(minSum)) {
                    minSum = sum;
                    first = arr[i];
                    second = arr[j];
                }
            }
        }

        // Output the result
        System.out.println("Pair with the sum closest to zero: " + first + " and " +
second);
    }
}
```

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

      int n = scanner.nextInt();

      int[][] matrix = new int[n][n];

      for (int i = 0; i < n; i++) {
         for (int j = 0; j < n; j++) {

            matrix[i][j] = scanner.nextInt();
         }
      }

      rotateMatrix(matrix);

      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();
}

      scanner.close();
   }

   public static void rotateMatrix(int[][] matrix) {
      int n = matrix.length;

      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++) {
            for (int j = 0; j < n / 2; j++) {

                int temp = matrix[i][j];
                matrix[i][j] = matrix[i][n - j - 1];
                matrix[i][n - j - 1] = temp;
            }
        }
    }
}
```

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