Rajalakshmi Engineering College

Name: HARI PRASANNA S
Email: 241501064@rajalakshmi.edu.in
Roll no: 241501064
Phone: 9042038178
Branch: REC
Department: AI & ML - Section 2
Batch: 2028
Degree: B.E - AI & ML

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:

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.Scanner;
public class Main{
    public static void main(String args[]){
        Scanner sc=new Scanner(System.in);
        int n=sc.nextInt();
        int[] a=new int[n];
        for(int i=0;i<n;i++){
            a[i]=sc.nextInt();
        }
        int sum;
        int k=a[0],l=a[1];
        int m=a[0]+a[1];
        for(int i=0;i<n-1;i++){
            for(int j=i+1;j<n;j++){
                sum=a[i]+a[j];
                if(Math.abs(sum)<Math.abs(m)){
                    m=Math.abs(sum);
                    k=a[i];
                    l=a[j];
                }
            }
        }
        System.out.printf("Pair with the sum closest to zero: %d and %d",k,l);
    }
}
```

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

```
import java.util.Scanner;
public class Main{
    public static void main(String args[]){
        Scanner sc=new Scanner(System.in);
        int n=sc.nextInt();
        int m=sc.nextInt();
        int sum=0;
        int[][] a=new int[n][m];
        for(int i=0;i<n;i++){
            for(int j=0;j<m;j++){
                a[i][j]=sc.nextInt();
                sum+=a[i][j];
            }
            for(int k=0;k<m;k++){
                a[i][k]=sum;
            }
            sum=0;
        }
        System.out.println("Transformed matrix:");
        for(int i=0;i<n;i++){
            for(int j=0;j<m;j++){
                System.out.print(a[i][j]+" ");
            }
            System.out.println();
        }
        System.out.println("Final merged matrix:");
        int h=sc.nextInt();
        int p=sc.nextInt();
        int[][] b=new int[h][p];
        for(int i=0;i<h;i++){
            for(int j=0;j<p;j++){
                b[i][j]=sc.nextInt();
            }
        }
        int k=sc.nextInt();
        int l=0,x=0;
        if(k==0){
            for(int i=0;i<n;i++){
                for(int j=0;j<m;j++){

                        System.out.print(a[i][j]+ " ");

                    }
                System.out.println();
            }
            for(int i=0;i<h;i++){
                for(int j=0;j<p;j++){
                    System.out.print(b[i][j]+" ");
                }
                System.out.println();
            }
        }
        else{
            for(int i=0;i<n;i++){
                for(int j=0;j<m;j++){
                 System.out.print(a[i][j]+" ");
                }
        for(int j=0;j<p;j++){
          System.out.print(b[i][j]+" ");
        }
        System.out.println();
      }
    }
  }
}
```

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

3. Problem Statement

Alex is a treasure hunter who collects valuable items during their quests. Each item has a specific point value, and Alex wants to maximize their score by strategically removing items one at a time.

The rule is simple: Alex removes the item with the highest point value in each step until no items are left, summing the values of the removed items to calculate the maximum score.

Help Alex to complete his task.

Input Format

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

The second line of input consists of N space-separated integers, representing the point values of the items.

Output Format

The output prints "Maximum Sum: " followed by the calculated maximum score after removing all items.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 14
7 14 21 28 35 42 49 56 63 70 77 84 91 98
Output: Maximum Sum: 735

Answer

```
import java.util.Scanner;
public class Main{
   public static void main(String args[]){
      Scanner sc=new Scanner(System.in);
      int n=sc.nextInt();
      int[] a=new int[n];
      int sum=0;
      for(int i=0;i<n;i++){
         a[i]=sc.nextInt();
         sum+=a[i];
      }
      System.out.println("Maximum Sum: "+sum);
      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.*;
public class Main{
   public static void main(String args[]){
      Scanner sc=new Scanner(System.in);
      int n=sc.nextInt();
      int[][] mat=new int[n][n];
      for(int i=0;i<n;i++){
         for(int j=0;j<n;j++){
            mat[i][j]=sc.nextInt();
         }
      }
      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;
            }
        }
        for(int i=0;i<n;i++){
            for(int j=0;j<n/2;j++){
                int temp=mat[i][j];
                mat[i][j]=mat[i][n-1-j];
                mat[i][n-1-j]=temp;
            }
        }
        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***