

# Rajalakshmi Engineering College

Name: RANNESH KHUMAR B R  
Email: 240701422@rajalakshmi.edu.in  
Roll no: 2116240701  
Phone: 9042350670  
Branch: REC  
Department: CSE - Section 8  
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**

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***

```
// You are using Java
import java.util.*;
class main{
    public static void main(String[] args){
        Scanner scan=new Scanner(System.in);
        int n=scan.nextInt();
        int arr[]=new int[n];

        int sum=0;
        for(int i=0;i<n;i++){
            arr[i]=scan.nextInt();
            sum+=arr[i];
        }
        System.out.print("Maximum Sum: "+sum);

    }
}
```

**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.*;
class main{
    public static void main(String args[]){
        Scanner scan= new Scanner(System.in);
        int n=scan.nextInt();
        int m=scan.nextInt();

        int m1[][]=new int[n][m];
        int res[][]=new int[n][m];

        System.out.println("Transformed matrix:");

        int sum=0;

        for(int i=0;i<n;i++){
            for(int j=0;j<m;j++){
                m1[i][j]=scan.nextInt();
                sum=sum+m1[i][j];
            }
        }

        for(int i=0;i<n;i++){
            for(int j=0;j<m;j++){
                res[i][j]=(m1[i][j]*sum)/n;
            }
        }

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

```
    }

    for(int k=0;k<m;k++){
        res[i][k]=sum;
        System.out.print(res[i][k]+" ");
    }
    sum=0;
    System.out.println();
}

int a=scan.nextInt();
int b=scan.nextInt();

int m2[][]= new int[a][b];
for(int i=0;i<a;i++){
    for(int j=0;j<b;j++){
        m2[i][j]=scan.nextInt();
    }
}

int choice=scan.nextInt();
System.out.println("Final merged matrix:");
if(choice==0){
    for(int i=0;i<n;i++){
        for(int j=0;j<m;j++){
            System.out.print(res[i][j]+" ");
        }
        System.out.println();
    }
    for(int i=0;i<a;i++){
        for(int j=0;j<b;j++){
            System.out.print(m2[i][j]+" ");
        }
        System.out.println();
    }
}

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

```
        for(int k=0;k<b;k++){
            System.out.print(m2[i][k]+" ");
        }
        System.out.println();
    }
}
}
```

**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

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

class main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        int n = scanner.nextInt();
        int[] arr = new int[n];
        for (int i = 0; i < n; i++) {
            arr[i] = scanner.nextInt();
        }

        int minAbsSum = Integer.MAX_VALUE;
        int first = 0, second = 0;

        for (int i = 0; i < n; i++) {
            for (int j = i + 1; j < n; j++) {
                int sum = arr[i] + arr[j];
                int absSum = Math.abs(sum);
                if (absSum < minAbsSum) {
                    minAbsSum = absSum;
                    first = arr[i];
                    second = arr[j];
                }
            }
        }

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

Status : Correct

Marks : 10/10

### 4. Problem Statement

Robin is a tech-savvy teenager who is diving into programming.

He is working on a project to find special elements in an array called 'leaders.' Leaders are those exceptional elements that are greater than the sum of all the elements to their right.

Assist Robin in writing this program.

### Example

Input:

6

16 28 74 19 25 11

Output:

74 25 11

Explanation:

The element 16 is not greater than the sum of elements to its right ( $28 + 74 + 19 + 25 + 11 = 157$ )

The element 28 is not greater than the sum of elements to its right ( $74 + 19 + 25 + 11 = 129$ )

The element 74 is greater than the sum of elements to its right ( $19 + 25 + 11 = 55$ )

The element 19 is not greater than the sum of elements to its right ( $25 + 11 = 36$ )

The element 25 is greater than the sum of elements to its right (11)

The last element 11 is always a leader since there are no elements to its right.

So, the output is {74, 25, 11}.

### *Input Format*

The first line of input consists of an integer N, representing the number of elements in the array.

The second line consists of N space-separated integers, representing the elements of the array.

### **Output Format**

The output prints the special elements in the given array, that are greater than the sum of all the elements to their right.

Refer to the sample output for formatting specifications.

### **Sample Test Case**

Input: 5

3 4 2 5 1

Output: 5 1

### **Answer**

```
import java.util.Scanner;

class main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        int n = scanner.nextInt();
        int[] arr = new int[n];
        for (int i = 0; i < n; i++) {
            arr[i] = scanner.nextInt();
        }

        boolean[] isLeader = new boolean[n];
        int sumRight = 0;
        int count = 0;

        for (int i = n - 1; i >= 0; i--) {
            if (arr[i] > sumRight) {
                isLeader[i] = true;
                count++;
            }
            sumRight += arr[i];
        }
    }
}
```

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

**Status :** Correct

**Marks :** 10/10