Rajalakshmi Engineering College

Name: K.A.Auxin Sam
Email: 241901012@rajalakshmi.edu.in
Roll no: 241901012
Phone: 8939811197
Branch: REC
Department: CSE (CS) - Section 1
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 2_CY

Attempt : 1
Total Mark : 40
Marks Obtained : 40

Section 1 : Coding

1. Problem Statement

Joe has a favourite number, let's call it X. He wants to check if X is divisible by the sum of its digits. If it is, he considers it a lucky number. If not, he wants to find the closest smaller number, that is divisible by the sum of digits of X. Joe has challenged his friends to solve this puzzle at his birthday party.

Example

Input:

157

Output:

157 is not divisible by the sum of its digits.

The closest smaller number that is divisible: 156

Explanation:

The sum of the digits of X is 1+5+7=13. Since 157 is not divisible by 13, we need to find the closest smaller number that is divisible by 13. 156 is divisible by 13, it is the closest smaller number that meets the requirement.

Input Format

The input consists of an integer X, representing Joe's favourite number.

Output Format

If X is a lucky number, then the output must be in the format: "X is divisible by the sum of its digits."

If not, then the output must be in the format:

"X is not divisible by the sum of its digits.

The closest smaller number that is divisible: Y",

where X is the entered number and Y is the closest number.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 120
Output: 120 is divisible by the sum of its digits.

Answer

```
import java.util.Scanner;

public class Main {
   public static void main(String[] args) {
      Scanner scanner = new Scanner(System.in);
      int X = scanner.nextInt();
      int originalX = X;
      int sum = 0;
```

```
        while (X > 0) {
            sum += X % 10;
            X /= 10;
        }

        X = originalX;

        if (sum == 0) {
            System.out.println(X + " is divisible by the sum of its digits.");
            return;
        }

        if (X % sum == 0) {
            System.out.println(X + " is divisible by the sum of its digits.");
        } else {
            int closestNumber = X - (X % sum);
            System.out.println(X + " is not divisible by the sum of its digits.");
            System.out.println("The closest smaller number that is divisible: " +
closestNumber);
        }
    }
}
```

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

2. Problem Statement

Ram wants to evaluate the time required to break even on an investment based on initial costs, monthly profits, and monthly expenses. Write a program to calculate the break-even point in months and categorize the return on investment.

Compute the break-even point by using the formula: initial cost / (monthly profit - monthly expenses)Based on the break-even point, classify the return on investment into one of the following categories:Quick Return: If the break-even point is 3 months or fewer.Average Return: If the break-even point is between 4 and 12 months, inclusive.Long-term Return: If the break-even point exceeds 12 months.

Ram is new to programming, so he seeks your assistance in creating the program.

Note: monthly profit is always greater than monthly expenses.

Input Format

The first line of input consists of a double value representing the initial cost.

The second line consists of a double value representing the monthly profit.

The third line consists of a double value representing the monthly expenses.

Output Format

The first line prints "Break-even Point:", followed by the break-even point as a decimal number (of double datatype), formatted to two decimal places.

The second line prints "Category: ",followed by the investment return as a String, which can be one of:

- "Quick Return" if break-even point ≤ 3
- "Average Return" if break-even point ≤ 12
- "Long-term Return" if break-even point > 12

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 10000.50
5000.75
1000.10
Output: Break-even Point: 2.50
Category: Quick Return

Answer

```
import java.util.Scanner;

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

```
        double initialCost = scanner.nextDouble();
        double monthlyProfit = scanner.nextDouble();
        double monthlyExpenses = scanner.nextDouble();


        double netMonthlyProfit = monthlyProfit - monthlyExpenses;


        double breakEvenPoint = initialCost / netMonthlyProfit;


        String category;
        if (breakEvenPoint <= 3.0) {
            category = "Quick Return";
        } else if (breakEvenPoint <= 12.0) {
            category = "Average Return";
        } else {
            category = "Long-term Return";
        }


        System.out.printf("Break-even Point: %.2f%n", breakEvenPoint);
        System.out.println("Category: " + category);
    }
}
```

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

3. Problem Statement

Noah is analyzing numbers within a given range [A, B] and wants to calculate a special sum. For each number in the range, he calculates the product of its odd digits (ignoring even digits). If the number contains no odd digits, it is skipped. The sum of these products for all numbers in the range is the result.

Write a program to compute this sum.

Example

Input:

10 12

Output:

3

Explanation:

For 10, odd digits = 1, product = 1.

For 11, odd digits = 1, 1, product = 1 * 1 = 1.

For 12, odd digits = 1, product = 1.

Total sum = 1 + 1 + 1 = 3

Input Format

The input consists of two space-separated integers A and B, representing the inclusive range boundaries.

Output Format

The output prints a single integer representing the sum of the products of odd digits for all numbers in the range.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 10 12

Output: 3

Answer

```
import java.util.Scanner;

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

```
        int B = scanner.nextInt();
        int totalSum = 0;

        for (int num = A; num <= B; num++) {
            int currentNum = num;
            int product = 1;
            boolean hasOddDigits = false;

            while (currentNum > 0) {
                int digit = currentNum % 10;
                if (digit % 2 != 0) {
                    product *= digit;
                    hasOddDigits = true;
                }
                currentNum /= 10;
            }

            if (hasOddDigits) {
                totalSum += product;
            }
        }

        System.out.println(totalSum);
    }
}
```

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

4. Problem Statement

Samantha is a diligent math student who is exploring the world of programming. She is learning Java and has recently studied conditional statements. One day, her teacher gives her an interesting problem to solve, which takes a number as input and checks whether it is a multiple of 5 or 7.

Help her complete the task.

Input Format

The input consists of a single integer N, representing the number to be checked.

Output Format

If the number is a multiple of 5 but not 7, the output prints "N is a multiple of 5".

If the number is a multiple of 7, the output prints "N is a multiple of 7".

Otherwise the output prints "N is neither multiple of 5 nor 7" where N is an entered integer.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 10
Output: 10 is a multiple of 5

Answer

```
import java.util.Scanner;

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

        if (N % 5 == 0 && N % 7 != 0) {
            System.out.println(N + " is a multiple of 5");
        } else if (N % 7 == 0) {
            System.out.println(N + " is a multiple of 7");
        } else {
            System.out.println(N + " is neither multiple of 5 nor 7");
        }
    }
}
```

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

Rajalakshmi Engineering College

Name: K.A.Auxin Sam
Email: 241901012@rajalakshmi.edu.in
Roll no: 241901012
Phone: 8939811197
Branch: REC
Department: CSE (CS) - Section 1
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 3_Q1

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Rosh is intrigued by numerical patterns. Today, she stumbled upon a puzzle while working with arrays. She wants to compute the sum of the third-largest and second-smallest elements from a list of integers. She seeks your help to implement a program that solves this for her efficiently.

Input Format

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

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

Output Format

The output displays a single integer representing the sum of the third-largest and second-smallest elements in the array.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 10
10 20 30 40 50 60 70 80 90 100
Output: 100

Answer

```
import java.util.*;


public 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 result = findThirdLargestAndSecondSmallestSum(arr);


        System.out.println(result);
        scanner.close();

    }
    public static int findThirdLargestAndSecondSmallestSum(int[]arr){
        Set<Integer>uniqueElements = new TreeSet<>();
        for(int num : arr){
            uniqueElements.add(num);

        }
        Integer[]sortedUnique = uniqueElements.toArray(new Integer[0]);
```

```
        int thirdLargest = sortedUnique[sortedUnique.length - 3];
        int secondSmallest = sortedUnique[1];
        return thirdLargest + secondSmallest;
    }
}
```

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

Rajalakshmi Engineering College

Name: K.A.Auxin Sam
Email: 241901012@rajalakshmi.edu.in
Roll no: 241901012
Phone: 8939811197
Branch: REC
Department: CSE (CS) - Section 1
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 3_Q2

Attempt : 1
Total Mark : 10
Marks Obtained : 0

Section 1 : Coding

1. Problem Statement

Monica is interested in finding a treasure but the key to opening is to get the sum of the main diagonal elements and secondary diagonal elements.

Write a program to help Monica find the diagonal sum of a square 2D array.

Note: The main diagonal of the array consists of the elements traversing from the top-left corner to the bottom-right corner. The secondary diagonal includes elements from the top-right corner to the bottom-left corner.

Input Format

The first line of input consists of an integer N, representing the number of rows and columns.

The following N lines consist of N space-separated integers, representing the 2D array elements.

Output Format

The first line of output prints "Sum of the main diagonal: " followed by an integer, representing the sum of the main diagonal.

The second line prints "Sum of the secondary diagonal: " followed by an integer, representing the sum of the secondary diagonal.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 3
1 2 3
4 5 6
7 8 9

Output: Sum of the main diagonal: 15
Sum of the secondary diagonal: 15

Answer

```
import java.util.Scanner;
 public class Main{
    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();


       }
    }
    int mainDiagonalSum = 0;
    for (int i = 0; i < n; i++){
```

```
        mainDiagonalSum += matrix[i][i];

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

    }
    System.out.println("Sum of the main diagonal:" + mainDiagonalSum);
    System.out.println("Sum of the secondary diagonal: " +
secondaryDiagonalSum);

    scanner.close();
}
```

Status : Wrong ***Marks : 0/10***

Rajalakshmi Engineering College

Name: K.A.Auxin Sam
Email: 241901012@rajalakshmi.edu.in
Roll no: 241901012
Phone: 8939811197
Branch: REC
Department: CSE (CS) - Section 1
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 3_Q3

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

You are developing a warehouse management system for a shipping company. The system uses an integer array to represent the weights of packages in a specific order. To verify that the weight capacity is not exceeded, the program needs to calculate the sum of the weights of the first and last packages in the list.

Task:

Write a code to calculate the sum of the weights of the first and last packages in the list. The program should take an integer array as input and return the total weight of the first and last packages.

Input Format

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

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

Output Format

The output is displayed in the following format:

"Sum of the first and last elements: <<Sum>>"

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 5
10 20 30 40 50
Output: Sum of the first and last elements: 60

Answer

```
import java.util.Scanner;

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


        int n = scanner.nextInt();


        int[] weights = new int[n];
        for (int i = 0; i < n; i++) {
            weights[i] = scanner.nextInt();
        }


        int sum = weights[0] + weights[n - 1];


        System.out.println("Sum of the first and last elements: " + sum);

        scanner.close();
    }
```

```
}
```

Status : Correct *Marks : 10/10*

Rajalakshmi Engineering College

Name: K.A.Auxin Sam
Email: 241901012@rajalakshmi.edu.in
Roll no: 241901012
Phone: 8939811197
Branch: REC
Department: CSE (CS) - Section 1
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 3_Q4

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Sesha is developing a weather monitoring system for a region with multiple weather stations. Each weather station collects temperature data hourly and stores it in a 2D array.

Write a program that can add the temperature data from two different weather stations to create a combined temperature record for the region.

Input Format

The first line of input consists of two space-separated integers N and M, representing the number of rows and columns of the matrices, respectively.

The next N lines consist of M space-separated integers, representing the values of the first matrix.

The following N lines consist of M space-separated integers, representing the values of the second matrix.

Output Format

The output prints the addition of the two matrices in N rows and M columns, representing the combined temperature record.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 3 3
1 2 3
4 5 6
7 8 9
1 1 1
2 2 2
3 3 3

Output: 2 3 4
6 7 8
10 11 12

Answer

```
import java.util.Scanner;

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


        int n = scanner.nextInt();
        int m = scanner.nextInt();


        int[][] station1 = new int[n][m];
        int[][] station2 = new int[n][m];
        int[][] combined = new int[n][m];


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

```
        for (int j = 0; j < m; j++) {
            station1[i][j] = scanner.nextInt();
        }
    }

    for (int i = 0; i < n; i++) {
        for (int j = 0; j < m; j++) {
            station2[i][j] = scanner.nextInt();
        }
    }

    for (int i = 0; i < n; i++) {
        for (int j = 0; j < m; j++) {
            combined[i][j] = station1[i][j] + station2[i][j];
        }
    }

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

    scanner.close();
  }
}
```

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

Rajalakshmi Engineering College

Name: K.A.Auxin Sam
Email: 241901012@rajalakshmi.edu.in
Roll no: 241901012
Phone: 8939811197
Branch: REC
Department: CSE (CS) - Section 1
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 3_Q5

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Sharon is creating a program that finds the first repeated element in an integer array. The program should efficiently identify the first element that appears more than once in the given array. If no such element is found, it should appropriately display a message.

Help Sharon to complete the program.

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 array elements.

Output Format

If a repeated element is found, print the first element that appears more than once.

If no repeated element is found, print "No repeated element found in the array".

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 8
12 21 13 14 21 36 47 21
Output: 21

Answer

```
import java.util.Scanner;
import java.util.HashSet;

public 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();
      }


      HashSet<Integer> seen = new HashSet<>();
      boolean found = false;

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

         if (seen.contains(arr[i])) {
            System.out.println(arr[i]);
```

```
                found = true;
                break;
            }

            seen.add(arr[i]);
        }


        if (!found) {
            System.out.println("No repeated element found in the array");
        }

        scanner.close();
    }
}
```

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

Rajalakshmi Engineering College

Name: K.A.Auxin Sam
Email: 241901012@rajalakshmi.edu.in
Roll no: 241901012
Phone: 8939811197
Branch: REC
Department: CSE (CS) - Section 1
Batch: 2028
Degree: B.E - CSE (CS)

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 3_Q5

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Sharon is creating a program that finds the first repeated element in an integer array. The program should efficiently identify the first element that appears more than once in the given array. If no such element is found, it should appropriately display a message.

Help Sharon to complete the program.

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 array elements.

Output Format

If a repeated element is found, print the first element that appears more than once.

If no repeated element is found, print "No repeated element found in the array".

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 8
12 21 13 14 21 36 47 21
Output: 21

Answer

```
import java.util.Scanner;
import java.util.HashSet;

public 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();
      }


      HashSet<Integer> seen = new HashSet<>();
      boolean found = false;

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

         if (seen.contains(arr[i])) {
            System.out.println(arr[i]);
```

```
                found = true;
                break;
            }

            seen.add(arr[i]);
        }


        if (!found) {
            System.out.println("No repeated element found in the array");
        }

        scanner.close();
    }
}
```

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

Rajalakshmi Engineering College

Name: K.A.Auxin Sam
Email: 241901012@rajalakshmi.edu.in
Roll no: 241901012
Phone: 8939811197
Branch: REC
Department: CSE (CS) - Section 1
Batch: 2028
Degree: B.E - CSE (CS)

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 : 20

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 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 bestSum = Integer.MAX_VALUE;
      int element1 = 0, element2 = 0;


      for (int i = 0; i < n - 1; i++) {
         for (int j = i + 1; j < n; j++) {
            int currentSum = arr[i] + arr[j];
            int absSum = Math.abs(currentSum);
```

```
            if (absSum < Math.abs(bestSum)) {
                bestSum = currentSum;
                element1 = arr[i];
                element2 = arr[j];
            }
        }
    }

    System.out.println("Pair with the sum closest to zero: " + element1 + " and "
+ element2);

    scanner.close();
  }
}
```

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

2. 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;

public class Main {


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

        for (int layer = 0; layer < n / 2; layer++) {
            int first = layer;
            int last = n - 1 - layer;


            for (int i = first; i < last; i++) {
                int offset = i - first;
```

```
            int top = matrix[first][i];

            matrix[first][i] = matrix[last - offset][first];

            matrix[last - offset][first] = matrix[last][last - offset];

            matrix[last][last - offset] = matrix[i][last];

            matrix[i][last] = top;
        }
    }
}

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();
    }
}
```

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

3. 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

```
def process_grid_operations():

    rows, cols = map(int, input().split())

    matrix = []
```

```
    for i in range(rows):
        row = list(map(int, input().split()))
        matrix.append(row)

    insert_type, insert_index = map(int, input().split())

    if insert_type == 0:
        new_row = list(map(int, input().split()))
        matrix.insert(insert_index, new_row)
    else:
        new_column = list(map(int, input().split()))
        for i in range(len(matrix)):
            matrix[i].insert(insert_index, new_column[i])

    print("After insertion")
    for row in matrix:
        print(' '.join(map(str, row)) + ' ')

    delete_type, delete_index = map(int, input().split())

    if delete_type == 0:
        matrix.pop(delete_index)
    else:
        for row in matrix:
            row.pop(delete_index)

    print("After deletion")
    for row in matrix:
        print(' '.join(map(str, row)) + ' ')

process_grid_operations()
```

Status : Wrong **Marks : 0/10**

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

-

Status : -

Marks : 0/10