Self Practice - Arrays and Functions

1. Imagine you're on a treasure hunt, and the map you've been given contains a series of numbers as instead of landmarks. Your task is to find the largest treasure hidden in this array of numbers.

Sample Input:

```
5, 12, 8, 29, 17, 6, 21
```

Sample Output

```
29
```

Analysis

- The function should take a list of numbers as input.
- The function should return the largest number in the list.
- The function should be named `largest_treasure`.

`Program : LargestTreasure.java`

Code:

```java
package com.self_practice;

import java.util.Scanner;

public class LargestTreasure {
    public static int largest_treasure(int[] numbers) {
        int largest = numbers[0];
        for (int i = 1; i < numbers.length; i++) {
            if (numbers[i] > largest) {
                largest = numbers[i];
            }
        }
        return largest;
    }
```

```
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter the number of elements: ");
        int n = sc.nextInt();
        System.out.println("Enter the elements: ");
        int[] numbers = new int[n];
        for (int i = 0; i < n; i++) {
            numbers[i] = sc.nextInt();
        }
        System.out.println(
                "The largest treasure is: "
                        + largest_treasure(numbers)
        );
    }
}
```

---

2. Imagine you're organizing a game night with your friends, and you've decided to play a unique game involving a series of numbers. Your task is to keep track of the number of even and odd numbers that appear in the sequence during the game.

Sample Input:

```
num = (3, 4, 6, 2, 3, 5, 7, 8, 5, 2, 9, 0, 1)
```

Sample Output:

```
Even: 7
Odd: 6
```

Analysis

- The function should take a list of numbers as input.
- The function should return the number of even and odd numbers in the list.
- The function should be named `even_odd_count`.

`Program : EvenOddCount.java`

Code:

```java
package com.self_practice;

import java.util.Scanner;

public class EvenOddCount {
    public static void even_odd_count(int[] numbers) {
        int even = 0, odd = 0;
        for (int i = 0; i < numbers.length; i++) {
            if (numbers[i] % 2 == 0) {
                even++;
            } else {
                odd++;
            }
        }
        System.out.println("Even: " + even);
        System.out.println("Odd: " + odd);
    }
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter the number of elements: ");
        int n = sc.nextInt();
        System.out.println("Enter the elements: ");
        int[] numbers = new int[n];
        for (int i = 0; i < n; i++) {
            numbers[i] = sc.nextInt();
        }
        even_odd_count(numbers);
    }
}
```

Output:

```
Enter the number of elements: 13
Enter the elements:
3 4 6 2 3 5 7 8 5 2 9 0 1
Even: 7
Odd: 6
```

3. You're working on a program that tracks student grades, and you need to find out how many times a specific grade, say 90, appears in an array of test scores (frequency).

Sample Input:

```
arr[] = {85, 90, 78, 90, 92, 90, 87, 88, 90}
```

Sample Output:

```
The grade 90 appears 4 times in the array
```

Analysis

- The function should take a list of numbers and a specific grade as input.
- The function should return the frequency of the specific grade in the list.
- The function should be named `grade_frequency`.

`Program : GradeFrequency.java`

Code:

```java
package com.self_practice;

import java.util.Scanner;

public class GradeFrequency {
    static int frequencyOfNumber(int[] arr, int num){
        int frequency = 0;
        for (int j : arr) {
            if (j == num) {
                frequency += 1;
            }
        }
        return frequency;
    }
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        System.out.print("Enter the size of array: ");
        int size = sc.nextInt();
        System.out.println("Enter the elements: ");
```

```
        int[] array = new int[size];
        for(int i=0; i<size; i++){
            array[i] = sc.nextInt();
        }
        System.out.print("Enter the element to check frequency: ");
        int frequencyNumber = sc.nextInt();
        System.out.println(frequencyOfNumber(array, frequencyNumber));
    }
}
```

Output:

```
Enter the size of array: 9
Enter the elements:
85 90 78 90 92 90 87 88 90
Enter the element to check frequency: 90
4
```

---

4. Picture yourself as an archaeologist exploring a lost civilization. You stumble upon a sequence of artifacts, each marked with a unique number within a certain range. However, one crucial artifact is missing, and it's your mission to identify which artifact is absent from the sequence. How would you go about detecting the missing artifact within the specified range based on the artifacts you've already discovered?

Sample Input:

```
Items[] = [9,6,4,2,3,5,7,0,1]
```

Sample Output:

```
8
```

Analysis

- The function should take a list of numbers as input.

- The function should return the missing number within the specified range.
- The function should be named `missingArtifact`.

`Program : MissingArtifact.java`

Code:

```java
package com.self_practice;

import java.util.Scanner;

public class MissingArtifact {
    public static int missingArtifact(int[] items) {
        int n = items.length + 1;
        int total = n * (n + 1) / 2;
        int sum = 0;
        for (int item : items) {
            sum += item;
        }
        return total - sum;
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter the number of elements: ");
        int n = sc.nextInt();
        System.out.println("Enter the elements: ");
        int[] items = new int[n];
        for (int i = 0; i < n; i++) {
            items[i] = sc.nextInt();
        }
        System.out.println(
            "The missing artifact is: "
            + missingArtifact(items)
        );
    }
}
```

Output:

```
Enter the number of elements: 8
Enter the elements:
9 6 4 2 3 5 7 0 1
The missing artifact is: 8
```

5. Imagine you're managing finances for a project, and you have a list of expenses stored in an array. Your task is to calculate the total expenditure incurred so far to ensure the project stays within budget.

Sample Input:

```
200 350 120 80 150
```

Sample Output;

```
900
```

Analysis

- The function should take a list of numbers as input.
- The function should return the total expenditure.
- The function should be named `totalExpenditure`.

`Program : TotalExpenditure.java`

Code:

```java
package com.self_practice;

import java.util.Scanner;

public class TotalExpenditure {
    public static int totalExpenditure(int[] expenses) {
        int total = 0;
        for (int expense : expenses) {
            total += expense;
        }
        return total;
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter the number of expenses: ");
```

```
        int n = sc.nextInt();
        System.out.println("Enter the expenses: ");
        int[] expenses = new int[n];
        for (int i = 0; i < n; i++) {
            expenses[i] = sc.nextInt();
        }
        System.out.println(
            "The total expenditure is: "
            + totalExpenditure(expenses)
        );
    }
}
```

Output:

```
Enter the number of expenses: 5
Enter the expenses:
200 350 120 80 150
The total expenditure is: 900
```

---

6. You're participating in a coding competition where the challenge is to identify duplicate elements in an array. How would you devise a strategy to detect and remove these duplicates?

Sample Input:

```
5 8 2 5 9 2 3 8
```

Sample Output:

```
5 8 2 9 3
```

Analysis

- The function should take a list of numbers as input.
- The function should return the list of unique elements.
- The function should be named `removeDuplicates`.

Program : RemoveDuplicates.java

Code:

```java
package com.self_practice;

import java.util.Scanner;

public class RemoveDuplicates {
    public static void removeDuplicates(int[] numbers) {
        for (int i = 0; i < numbers.length; i++) {
            boolean isDuplicate = false;
            for (int j = 0; j < i; j++) {
                if (numbers[i] == numbers[j]) {
                    isDuplicate = true;
                    break;
                }
            }
            if (!isDuplicate) {
                System.out.print(numbers[i] + " ");
            }
        }
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter the number of elements: ");
        int n = sc.nextInt();
        System.out.println("Enter the elements: ");
        int[] numbers = new int[n];
        for (int i = 0; i < n; i++) {
            numbers[i] = sc.nextInt();
        }
        removeDuplicates(numbers);
    }
}
```

Output:

```
Enter the number of elements: 8
Enter the elements:
5 8 2 5 9 2 3 8
5 8 2 9 3
```

7. You're designing a thrilling game where players embark on an adventure through an array of challenges. One of the key mechanics involves rotating the elements of an array by a certain number of steps.

Sample Input:

```
1 2 3 4 5 6 7
Rotate : 3
```

Sample Output:

```
4 5 6 7 1 2 3
```

Analysis

- The function should take a list of numbers and the number of steps to rotate as input.
- The function should return the rotated array.
- The function should be named `rotateArray`.

`Program : RotateArray.java`

Code:

```java
package com.self_practice;

import java.util.Scanner;

public class RotateArray {
    public static void rotateArray(int[] numbers, int rotate) {
        int n = numbers.length;
        for (int i = 0; i < rotate; i++) {
            int temp = numbers[0];
            for (int j = 0; j < n - 1; j++) {
                numbers[j] = numbers[j + 1];
            }
            numbers[n - 1] = temp;
        }
        for (int number : numbers) {
```

```
            System.out.print(number + " ");
        }
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter the number of elements: ");
        int n = sc.nextInt();
        System.out.println("Enter the elements: ");
        int[] numbers = new int[n];
        for (int i = 0; i < n; i++) {
            numbers[i] = sc.nextInt();
        }
        System.out.print("Enter the number of steps to rotate: ");
        int rotate = sc.nextInt();
        rotateArray(numbers, rotate);
    }
}
```

Output:

```
Enter the number of elements: 7
Enter the elements:
1 2 3 4 5 6 7
Enter the number of steps to rotate: 3
4 5 6 7 1 2 3
```

---

8. Write a program to find the minimum and maximum element of each row and column in the given two-dimensional arrays.

Sample Input:

```
Enter row size: 3
Enter column size: 3
Enter 3 * 3 array elements are: 4
1
2
5
3
6
3
```

```
7
8
```

Sample Output:

```
Given Array is:
4 1 2
5 3 6
3 7 8
The minimum and maximum element of 1st row is: 1, 4
The minimum and maximum element of 2nd row is: 3, 6
The minimum and maximum element of 3rd row is: 3, 8
The minimum and maximum element of 1st column is: 3,
5
The minimum and maximum element of 2nd column is: 1,
7
The minimum and maximum element of 3rd column is: 2,
8
```

Analysis

- Get the row and column size from the user.
- Get the elements of the 2D array from the user.
- Print the 2D array.
- Find the minimum and maximum element of each row and column.
- Print the minimum and maximum element of each row and column.

`Program : MinMaxElements.java`

Code:

```java
package com.self_practice;

import java.util.Scanner;

public class MinMaxElements {
    public void findMinMax(int row, int col, int[][] arr){
        System.out.println("Given Array is: ");
        for(int i=0; i<row; i++){
            for(int j=0; j<col; j++){
                System.out.print(arr[i][j]+" ");
            }
            System.out.println();
```

```
        }
        for(int i=0; i<row; i++){
            int minRow = Integer.MAX_VALUE;
            int maxRow = Integer.MIN_VALUE;
            for(int j=0; j<col; j++){
                if(arr[i][j] < minRow){
                    minRow = arr[i][j];
                }
                if(arr[i][j] > maxRow){
                    maxRow = arr[i][j];
                }
            }
            System.out.println(
                "The minimum and maximum element of "
                + ( i+1 ) + " row is: "
                + minRow + ", " + maxRow
            );
        }
        for(int i=0; i<col; i++){
            int minCol = Integer.MAX_VALUE;
            int maxCol = Integer.MIN_VALUE;
            for(int j=0; j<row; j++){
                if(arr[j][i] < minCol){
                    minCol = arr[j][i];
                }
                if(arr[j][i] > maxCol){
                    maxCol = arr[j][i];
                }
            }
            System.out.println(
                "The minimum and maximum element of "
                 + ( i+1 ) + " column is: "
                 + minCol + ", " + maxCol
            );
        }
    }

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

        System.out.print("Enter row size: ");
        int row = sc.nextInt();
        System.out.print("Enter column size: ");
        int col = sc.nextInt();

        int[][] arr = new int[row][col];
        System.out.println(
```

```
            "Enter "
            + row +" * " + col
            +" array elements are: "
        );
        for(int i=0; i<row; i++){
            for(int j=0; j<col; j++){
                arr[i][j] = sc.nextInt();
            }
        }

        MinMaxElements mme = new MinMaxElements();
        mme.findMinMax(row, col, arr);

        sc.close();
    }
}
```

---

9. Write a Java program to accept N numbers from console and print the sum of the elements of the array with the following condition. Condition: If the array has elements `a` and `b` in succeeding order, ignore the numbers between `a` and `b` inclusive for the calculation.

Sample Input:

```
7
10,3,6,1,2,7,9
6
7
```

Sample Output:

```
22
```

Sample Input:

```
5
7,1,2,3,6
```

```
6
7
```

Sample Output:

```
19
```

Sample Input:

```
4
1,6,7,9
6
7
```

Sample Output:

```
10
```

Analysis:

- Get the number of elements and the array of integers from the user.
- Get the elements `a` and `b` from the user.
- Initialize a variable to store the sum of the elements.
- Loop through the array and check if the element is `a`.
- If found, skip the elements until `b`.
- Print the sum of the elements.

`Program: SumOfElements.java`

Code:

```java
package com.self_practice;

import java.util.Scanner;

public class SumOfElements {
    public int findSum(int n, int[] arr, int a, int b){
        int sum = 0;
        boolean skip = false;
        for(int i=0; i<n; i++){
            if(arr[i] == a){
                skip = true;
```

```
            }else if(arr[i] == b){
                skip = false;
            }else if(!skip){
                sum += arr[i];
            }
        }
        return sum;
    }

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

        System.out.print("Enter the number of elements: ");
        int n = sc.nextInt();

        System.out.println("Enter the array elements: ");
        int[] arr = new int[n];
        for(int i=0; i<n; i++){
            arr[i] = sc.nextInt();
        }

        System.out.print("Enter the element a: ");
        int a = sc.nextInt();
        System.out.print("Enter the element b: ");
        int b = sc.nextInt();

        SumOfElements soe = new SumOfElements();
        int sum = soe.findSum(n, arr, a, b);
        System.out.println(sum);

        sc.close();
    }
}
```

Output:

```
Enter the number of elements: 7
Enter the array elements:
10 3 6 1 2 7 9
Enter the element a: 6
Enter the element b: 7
22
```

10. reate a Java program that utilizes a function to receive input from the user for the lengths of the three sides of a triangle. Then, calculate the area of the triangle using these side lengths and display the result.

Sample Input:

```
3 4 5
```

Sample Output:

```
6
```

Analysis:

- Get the lengths of the three sides of the triangle from the user.
- Calculate the semi-perimeter of the triangle.
- Calculate the area of the triangle using the semi-perimeter.
- Print the area of the triangle.

`Program: AreaOfTriangle.java`

Code:

```java
package com.self_practice;

import java.util.Scanner;

public class AreaOfTriangle {
    public double calculateArea(double a, double b, double c){
        double s = (a + b + c) / 2;
        return Math.sqrt(s * (s - a) * (s - b) * (s - c));
    }

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

        System.out.println("Enter the lengths of the three sides of the triangle:
        double a = sc.nextDouble();
        double b = sc.nextDouble();
        double c = sc.nextDouble();
```

```
        AreaOfTriangle aot = new AreaOfTriangle();
        double area = aot.calculateArea(a, b, c);
        System.out.println(area);

        sc.close();
    }
}
```

Output:

```
Enter the lengths of the three sides of the triangle:
3 4 5
6.0
```

---

11. You're tasked with designing a function that calculates the sum of two integer values, 'a' and 'b'. However, if either 'a' or 'b' is a "teen" value in the range 13 to 19 (inclusive), the function should return 19 instead of their sum.

Sample Input1:

```
3 4
```

Sample Output1:

```
7
```

Sample Input2:

```
13 8
```

Sample Output2:

```
19
```

Analysis:

- Get the two integer values 'a' and 'b' from the user.
- Check if either 'a' or 'b' is a "teen" value.
- If found, return 19.

`Program: TeenSum.java`

Code:

```java
package com.self_practice;

import java.util.Scanner;

public class TeenSum {
    public int calculateSum(int a, int b){
        if((a >= 13 && a <= 19) || (b >= 13 && b <= 19)){
            return 19;
        }
        return a + b;
    }

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

        System.out.print("Enter two integer values: ");
        int a = sc.nextInt();
        int b = sc.nextInt();

        TeenSum ts = new TeenSum();
        int sum = ts.calculateSum(a, b);
        System.out.println(sum);

        sc.close();
    }
}
```

Output:

```
Enter two integer values: 13 8
19
```

---

12. A communication system converts data into binary number format for transmission purpose. User enters data in numeric form. Write a method to convert decimal number into binary numbers & print it.

Sample Input:

```
25
```

Sample Output:

```
11001
```

Analysis:

- Get the decimal number from the user.
- Convert the decimal number into binary number.
- Print the binary number.

`Program: DecimalToBinary.java`

Code:

```java
package com.self_practice;

import java.util.Scanner;

public class DecimalToBinary {
    public void convertToBinary(int n){
        int[] binary = new int[1000];
        int i = 0;
        while(n > 0){
            binary[i] = n % 2;
            n = n / 2;
            i++;
        }
        for(int j=i-1; j>=0; j--){
            System.out.print(binary[j]);
        }
    }

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

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

        System.out.print("Enter a decimal number: ");
        int n = sc.nextInt();

        DecimalToBinary dtb = new DecimalToBinary();
        dtb.convertToBinary(n);

        sc.close();
    }
}
```

Output:

```
Enter a decimal number: 25
11001
```

---

13. Implement the function calculate Net Salary. User must input Basic Salary and Output should be net salary calculated based on following allowances: Allowances:

```
DA = 70% of Basic Salary
HRA = 7% of Basic Salary
MA = 2% of Basic Salary
TA = 4% of Basic Salary
```

Deduction:

```
PF = 12% of Basic Salary
Income/professional tax = User Input (e.g., 500)
Net Salary = Basic Salary + Allowances - Deduction
```

Sample Input :

```
Enter the Basic Salary:
50000
Enter the Income/Professional Tax:
1500
```

Sample Output:

```
Net Salary: 59300.0
```

Analysis:

- Get the basic salary and income tax from the user.
- Calculate the allowances and deductions.
- Calculate the net salary.

`Program: NetSalary.java`

Code:

```java
package com.hands_on;

import java.util.Scanner;

public class NetSalary {
    public double calculateNetSalary(double basicSalary, double incomeTax){
        double da = 0.7 * basicSalary;
        double hra = 0.07 * basicSalary;
        double ma = 0.02 * basicSalary;
        double ta = 0.04 * basicSalary;
        double pf = 0.12 * basicSalary;
        return basicSalary + da + hra + ma + ta - pf - incomeTax;
    }

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

        System.out.print("Enter the Basic Salary: ");
        double basicSalary = sc.nextDouble();
        System.out.print("Enter the Income/Professional Tax: ");
        double incomeTax = sc.nextDouble();

        NetSalary ns = new NetSalary();
        double netSalary = ns.calculateNetSalary(basicSalary, incomeTax);
        System.out.println("Net Salary: " + netSalary);

        sc.close();
    }
}
```

Output

```
Enter the Basic Salary: 50000
Enter the Income/Professional Tax: 1500
Net Salary: 59300.0
```