Problem 1:

Write a Java program that declares an array of integers and prints each element of the array using a **for** loop, **while** loop, and **enhanced for** loop.

Solution:

1. **Using a `for` loop:**

```
java
Copy code
public class ArrayLoop {
    public static void main(String[] args) {
        int[] numbers = {5, 10, 15, 20, 25}; // Declaring the array

        System.out.println("Using for loop:");
        for (int i = 0; i < numbers.length; i++) {
            System.out.println("Element at index " + i + " : " + numbers[i]);
        }
    }
}
```

2. **Using a `while` loop:**

```
java
Copy code
public class ArrayLoopWhile {
    public static void main(String[] args) {
        int[] numbers = {5, 10, 15, 20, 25}; // Declaring the array

        System.out.println("Using while loop:");
        int i = 0;
        while (i < numbers.length) {
            System.out.println("Element at index " + i + " : " + numbers[i]);
            i++;
        }
    }
}
```

3. **Using an enhanced `for` loop (for-each):**

```
java
Copy code
public class ArrayLoopEnhanced {
    public static void main(String[] args) {
        int[] numbers = {5, 10, 15, 20, 25}; // Declaring the array

        System.out.println("Using enhanced for loop:");
        for (int num : numbers) {
            System.out.println("Element: " + num);
        }
    }
}
```

Explanation:

1. **For loop:** Loops through the array by using the index (`i`).
2. **While loop:** Checks the condition (`i < numbers.length`) before each iteration and increments `i` manually.
3. **Enhanced for loop:** Directly accesses each element of the array without needing to manage the index, making it simpler when you don't need the index itself.

Problem 2:

Write a Java program to calculate the sum of all elements in an integer array using both **for** loop and **while** loop.

Solution:

1. **Using a `for` loop:**

```
java
Copy code
public class ArraySumForLoop {
    public static void main(String[] args) {
        int[] numbers = {2, 4, 6, 8, 10}; // Declaring and initializing the
array
        int sum = 0;

        for (int i = 0; i < numbers.length; i++) {
            sum += numbers[i]; // Adding each element to the sum
        }

        System.out.println("Sum of array elements using for loop: " + sum);
    }
}
```

2. **Using a `while` loop:**

```
java
Copy code
public class ArraySumWhileLoop {
    public static void main(String[] args) {
        int[] numbers = {2, 4, 6, 8, 10}; // Declaring and initializing the
array
        int sum = 0;
        int i = 0;

        while (i < numbers.length) {
            sum += numbers[i]; // Adding each element to the sum
            i++;
        }

        System.out.println("Sum of array elements using while loop: " + sum);
    }
}
```

Explanation:

- In both approaches, the sum of the array elements is calculated by iterating through the array and adding each element to a variable (`sum`).
- **For loop:** You use an index (`i`) to loop over the array and access elements at each index.
- **While loop:** You use a similar approach, but the loop continues as long as the condition (`i < numbers.length`) is true.

Problem 3:

Write a Java program to find the maximum value in an integer array using a **for** loop.

```
java
Copy code
public class ArrayMaxValue {
    public static void main(String[] args) {
        int[] numbers = {3, 9, 12, 5, 21}; // Declaring the array
        int max = numbers[0]; // Assume the first element is the largest
initially

        for (int i = 1; i < numbers.length; i++) {
            if (numbers[i] > max) {
                max = numbers[i]; // Update max if current element is greater
            }
        }

        System.out.println("Maximum value in the array is: " + max);
    }
}
```

Explanation:

- You initialize `max` to the first element of the array.
- The loop starts from the second element and compares each element with the current maximum.
- If a larger element is found, `max` is updated.

Problem 4: Find Minimum Value in an Array

Write a Java program to find the minimum value in an integer array using a **for** loop.

```
java
Copy code
public class ArrayMinValue {
    public static void main(String[] args) {
        int[] numbers = {15, 3, 25, 8, 6}; // Declaring and initializing the
array
```

```
        int min = numbers[0]; // Assume the first element is the smallest
initially

        for (int i = 1; i < numbers.length; i++) {
            if (numbers[i] < min) {
                min = numbers[i]; // Update min if a smaller element is found
            }
        }

        System.out.println("Minimum value in the array is: " + min);
    }
}
```

Explanation:

- You initialize `min` with the first element of the array and then loop through the array comparing each element to the current `min`. If a smaller element is found, `min` is updated.

Problem 5: Count Even and Odd Numbers in an Array

Write a Java program to count the number of even and odd numbers in an integer array using a **for** loop.

```
java
Copy code
public class CountEvenOdd {
    public static void main(String[] args) {
        int[] numbers = {4, 7, 12, 9, 15}; // Declaring and initializing the
array
        int evenCount = 0, oddCount = 0;

        for (int i = 0; i < numbers.length; i++) {
            if (numbers[i] % 2 == 0) {
                evenCount++; // Increment even counter if number is divisible
by 2
            } else {
                oddCount++; // Otherwise increment odd counter
            }
        }

        System.out.println("Even numbers count: " + evenCount);
        System.out.println("Odd numbers count: " + oddCount);
    }
}
```

Explanation:

- The program checks whether each number is divisible by 2. If true, it increments the `evenCount`; otherwise, it increments the `oddCount`.

Problem 6: Find the Average of Array Elements

Write a Java program to calculate the average value of elements in an integer array using a **for** loop.

```
java
Copy code
public class ArrayAverage {
    public static void main(String[] args) {
        int[] numbers = {10, 20, 30, 40, 50}; // Declaring and initializing
the array
        int sum = 0;

        for (int i = 0; i < numbers.length; i++) {
            sum += numbers[i]; // Summing all elements
        }

        double average = (double) sum / numbers.length; // Calculating the
average

        System.out.println("Average value of array elements is: " + average);
    }
}
```

Explanation:

- You loop through the array, adding each element to `sum`. Afterward, divide the sum by the total number of elements (`numbers.length`) to calculate the average.

Problem 7: Reverse an Array

Write a Java program to reverse an integer array using a **for** loop.

```
java
Copy code
public class ReverseArray {
    public static void main(String[] args) {
        int[] numbers = {2, 4, 6, 8, 10}; // Declaring and initializing the
array
        int[] reversed = new int[numbers.length]; // Array to hold reversed
elements

        for (int i = 0; i < numbers.length; i++) {
            reversed[i] = numbers[numbers.length - 1 - i]; // Reversing the
array
        }

        System.out.print("Reversed array: ");
        for (int num : reversed) {
            System.out.print(num + " "); // Printing reversed array
        }
    }
}
```

Explanation:

- The program uses a `for` loop to traverse the array from the last element to the first and copies the values into a new array.

Problem 8: Sum of Array Elements (For-each)

Write a Java program that calculates the sum of an integer array using a **for-each** loop.

```
java
Copy code
public class SumArrayForEach {
    public static void main(String[] args) {
        int[] numbers = {10, 20, 30, 40, 50};
        int sum = 0;

        for (int num : numbers) {
            sum += num;  // Adding each element to sum
        }

        System.out.println("Sum of array elements: " + sum);
    }
}
```

Explanation:

- The **for-each** loop iterates through each element in the array, accumulating the sum.

Problem 9: Print Array Elements (For-each)

Write a Java program to print all the elements in a string array using a **for-each** loop.

```
java
Copy code
public class PrintArrayForEach {
    public static void main(String[] args) {
        String[] fruits = {"Apple", "Banana", "Cherry", "Date"};

        for (String fruit : fruits) {
            System.out.println(fruit);  // Printing each element
        }
    }
}
```

Explanation:

- The **for-each** loop is used to access each element in the array and print it.

Problem 10: Find Maximum Value (For-each)

Write a Java program that finds the maximum value in an integer array using a **for-each** loop.

```
java
Copy code
public class MaxValueForEach {
    public static void main(String[] args) {
        int[] numbers = {12, 35, 7, 56, 22};
        int max = numbers[0];  // Assume the first element is the max
initially

        for (int num : numbers) {
            if (num > max) {
                max = num;  // Update max if a larger element is found
            }
        }

        System.out.println("Maximum value in the array: " + max);
    }
}
```

Explanation:

- The **for-each** loop compares each element to the current max and updates it if a larger value is found.

Problem 11: Check if All Elements Are Positive (For-each)

Write a Java program to check if all the elements in an array are positive numbers using a **for-each** loop.

```
java
Copy code
public class CheckPositiveForEach {
    public static void main(String[] args) {
        int[] numbers = {1, 2, 3, 4, -5};
        boolean allPositive = true;

        for (int num : numbers) {
            if (num <= 0) {
                allPositive = false;  // Set to false if any non-positive
number is found
                break;
            }
        }

        if (allPositive) {
            System.out.println("All elements are positive.");
        } else {
            System.out.println("There are non-positive elements in the
array.");
        }
    }
}
```

Explanation:

- The **for-each** loop checks each element in the array to see if all are positive numbers, and the loop stops as soon as a non-positive number is found.

Problem 12: Concatenate String Array Elements (For-each)

Write a Java program that concatenates all the elements in a string array using a **for-each** loop.

```
java
Copy code
public class ConcatenateStringsForEach {
    public static void main(String[] args) {
        String[] words = {"Hello", "World", "Java"};
        StringBuilder result = new StringBuilder();

        for (String word : words) {
            result.append(word).append(" ");  // Concatenating each element
with a space
        }

        System.out.println("Concatenated String: " +
result.toString().trim());
    }
}
```

Explanation:

- The **for-each** loop is used to concatenate all elements of the string array into a single string.

Problem 13: Count Positive and Negative Numbers (For-each)

Write a Java program to count how many positive and negative numbers are present in an integer array using a **for-each** loop.

```
java
Copy code
public class CountPosNegForEach {
    public static void main(String[] args) {
        int[] numbers = {-10, 23, -4, 5, 12, -7};
        int positiveCount = 0, negativeCount = 0;

        for (int num : numbers) {
            if (num > 0) {
                positiveCount++;  // Increment positive count
            } else if (num < 0) {
                negativeCount++;  // Increment negative count
            }
        }

        System.out.println("Positive numbers: " + positiveCount);
        System.out.println("Negative numbers: " + negativeCount);
    }
}
```

Explanation:

- The **for-each** loop is used to iterate over each number, counting how many are positive and negative.