Experiment Number : 04 **Date: 05-02-2026**

ARRAYS

Aim:

To understand and implement array operations in Java.

PRE LAB EXERCISE

QUESTIONS

- ✓ What is an array?

  Answer: An array is a collection of multiple values of the same data type stored in continuous memory locations and accessed using an index.

- ✓ Why are arrays used?

  Answer: Arrays are used because:

  1.To store **many values in one variable**

  2.To make programs **organized and cleaner**

  3. To allow **easy access using index numbers**

  4.To reduce **code repetition**

  5. To perform operations like **searching, sorting, and updating** easily

- ✓ What is the difference between array and variable?

  Answer:

Variable	**Array**
Stores only **one value**	Stores **multiple values**
Uses a **single memory location**	Uses **continuous memory locations**
Example: int x = 10;	Example: int[] x = {10,20,30};
Cannot hold a list of values	Can hold a list of values

IN LAB EXERCISE

Objective:

To perform array operations using simple programs.

PROGRAMS:

1. Program to Read and Print Array Elements

Code:

```
import java.util.Scanner;
public class ReadPrintArray {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int[] arr = new int[5];
        System.out.println("Enter 5 elements:");
        for(int i = 0; i < 5; i++)
            arr[i] = sc.nextInt();
        System.out.println("Array elements are:");
        for(int i = 0; i < 5; i++)
            System.out.print(arr[i] + " ");
    }
}
```

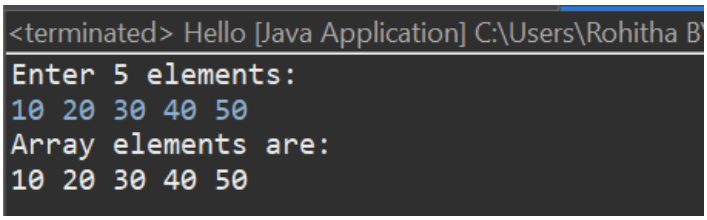
OUTPUT:

Input:
10 20 30 40 50

Output:

Array elements are:

10 20 30 40 50



2. Program to Find Sum of Array Elements

Code:

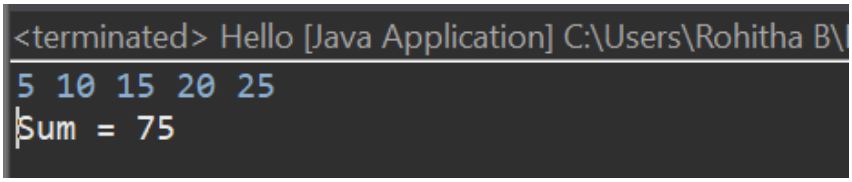
```
import java.util.Scanner;
public class SumArray {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int[] arr = new int[5];
        int sum = 0;
        System.out.println("Enter 5 elements:");
        for(int i = 0; i < 5; i++)
            arr[i] = sc.nextInt();
        for(int i = 0; i < 5; i++)
            sum += arr[i];
        System.out.println("Sum = " + sum);
    }
}
```

OUTPUT:

Input:
5 10 15 20 25

Output:

Sum = 75



3. Program to Find Largest Element in an Array

Code:

```
import java.util.Scanner;
public class LargestElement {
    public static void main(String[] args) {
```

```
        Scanner sc = new Scanner(System.in);
        int[] arr = new int[5];
        System.out.println("Enter 5 elements:");
        for(int i = 0; i < 5; i++)
            arr[i] = sc.nextInt();
        int max = arr[0];
        for(int i = 1; i < 5; i++)
            if(arr[i] > max)
                max = arr[i];
        System.out.println("Largest element = " + max);
    }
}
```

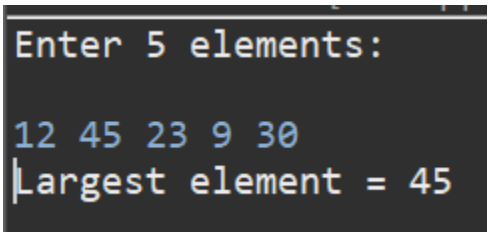
OUTPUT:

Input:

12 45 23 9 30

Output:

Largest element = 45



4. Program to Reverse an Array

Code:

```
import java.util.Scanner;
public class ReverseArray {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
```

```
        int[] arr = new int[5];
        System.out.println("Enter 5 elements:");
        for(int i = 0; i < 5; i++)
            arr[i] = sc.nextInt();
        System.out.println("Reversed array:");
        for(int i = 4; i >= 0; i--)
            System.out.print(arr[i] + " ");
    }
}
```

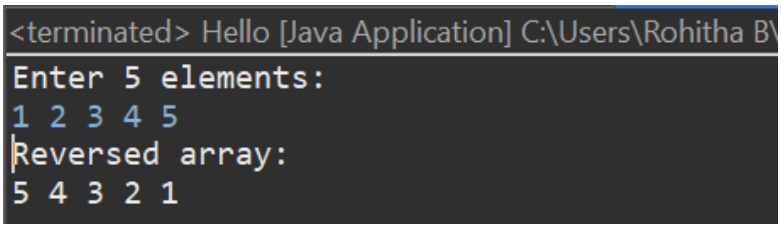
OUTPUT:

Input:
1 2 3 4 5

Output:

Reversed array:

5 4 3 2 1



5. Program to Count Even and Odd Numbers

Code:

```
import java.util.Scanner;
public class EvenOddCount {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int[] arr = new int[5];
        int even = 0, odd = 0;
        System.out.println("Enter 5 elements:");
        for(int i = 0; i < 5; i++)
```

```
            arr[i] = sc.nextInt();
        for(int i = 0; i < 5; i++) {
            if(arr[i] % 2 == 0)
                even++;
            else
                odd++;
        }

        System.out.println("Even = " + even);
        System.out.println("Odd = " + odd);
    }
}
```

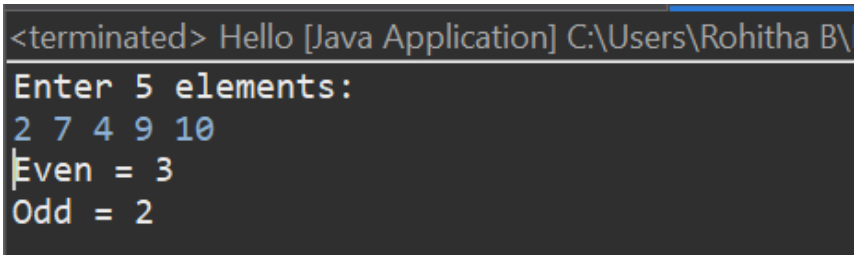
OUTPUT:

Input:
2 7 4 9 10

Output:

Even = 3

Odd = 2



6. Program to Sort Array in Ascending Order

Code:

```
import java.util.Scanner;
public class SortArray {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
```

```
        int[] arr = new int[5];
        int temp;
        System.out.println("Enter 5 elements:");
        for(int i = 0; i < 5; i++)
            arr[i] = sc.nextInt();
        for(int i = 0; i < 5; i++) {
            for(int j = i + 1; j < 5; j++) {
                if(arr[i] > arr[j]) {
                    temp = arr[i];
                    arr[i] = arr[j];
                    arr[j] = temp;
                }
            }
        }
        System.out.println("Sorted array:");
        for(int i = 0; i < 5; i++)
            System.out.print(arr[i] + " ");
    }
}
```

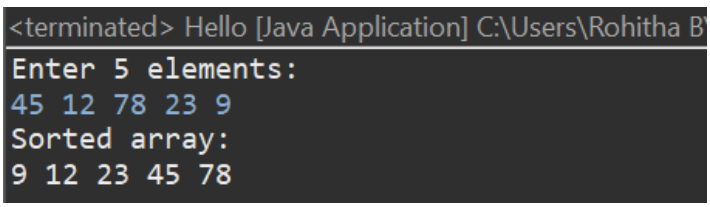
OUTPUT:

Input:
45 12 78 23 9

Output:

Sorted array:

9 12 23 45 78



7. Program to Find Second Largest Element

Code:

```
import java.util.Scanner;
public class SecondLargest {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int[] arr = new int[5];

        System.out.println("Enter 5 elements:");
        for(int i = 0; i < 5; i++)
            arr[i] = sc.nextInt();
        int largest = arr[0];
        int second = arr[0];
        for(int i = 0; i < 5; i++) {
            if(arr[i] > largest) {
                second = largest;
                largest = arr[i];
            }
        }
        System.out.println("Second largest = " + second);
    }
}
```

OUTPUT:

Input:
10 45 23 89 67

Output:

Second largest = 67

```
<terminated> Hello [Java Application] C:\Users\Rohitha B\
Enter 5 elements:
10 45 23 89 67
Second largest = 67
```

8. Program for Matrix Addition (2D Array)

Code:

```
import java.util.Scanner;
public class MatrixAddition {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int[][] a = new int[2][2];
        int[][] b = new int[2][2];
        int[][] sum = new int[2][2];
        System.out.println("Enter elements of matrix A:");
        for(int i = 0; i < 2; i++)
            for(int j = 0; j < 2; j++)
                a[i][j] = sc.nextInt();
        System.out.println("Enter elements of matrix B:");
        for(int i = 0; i < 2; i++)
            for(int j = 0; j < 2; j++)
                b[i][j] = sc.nextInt();
        for(int i = 0; i < 2; i++)
            for(int j = 0; j < 2; j++)
                sum[i][j] = a[i][j] + b[i][j];
        System.out.println("Sum matrix:");
        for(int i = 0; i < 2; i++) {
            for(int j = 0; j < 2; j++)
                System.out.print(sum[i][j] + " ");
            System.out.println();
```

```
        }
    }
}
```

OUTPUT:

Matrix A:

1 2

3 4

Matrix B:

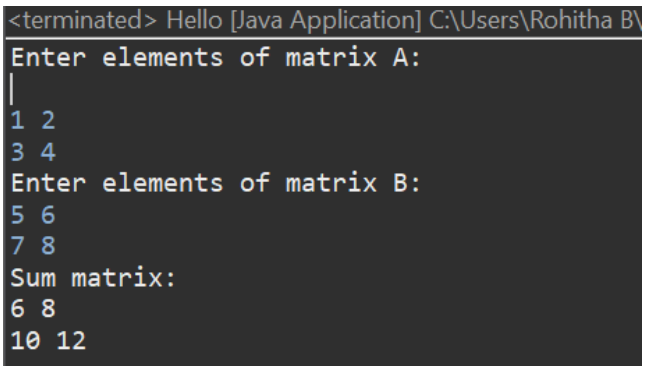
5 6

7 8

Sum matrix:

6 8

10 12



POST LAB EXERCISE

- ✓ Why is array indexing usually started from zero instead of one?

  Answer: Because array indexing represents the **offset from the starting memory address**.

- Index 0 means **first memory location**
- Index 1 means **next memory location**

  Starting from zero makes **address calculation faster and simpler** for the computer.

- ✓ What happens if we try to access an array element outside its declared size?

  Answer: It causes a **runtime error**.

  EX: ArrayIndexOutOfBoundsException

  int[] a = new int[5];

System.out.println(a[5]); // ERROR

- ✓ How does memory allocation differ for static arrays and dynamic arrays?

  Answer:

Static Array	Dynamic Array
Size fixed at compile time	Size decided at runtime
Cannot be changed	Can grow or shrink
Memory allocated once	Memory allocated as needed
Example: int a[5];	Example: ArrayList in Java

- ✓ Why is searching faster in arrays compared to linked lists?

  Answer: Because arrays use **contiguous memory**, allowing:

  ✓ Direct access using index

  ✓ Random access in O(1) time

  Linked lists must traverse nodes one by one, which is **slower (O(n))**.

- ✓ What is the difference between contiguous and non-contiguous memory allocation?

  Answer:

Contiguous	Non-Contiguous
Memory stored continuously	Memory stored at different locations
Faster access	Slower access
Used by arrays	Used by linked lists
Example: Array	Example: Linked List

Result:

Thus the array operations were executed successfully.

ASSESSMENT

Description	Max Marks	Marks Awarded
Pre Lab Exercise	**5**	
In Lab Exercise	**10**	
Post Lab Exercise	**5**	
Viva	**10**	
Total	**30**	
	Faculty Signature	

ROHITHA B

24BCS229