

Experiment Number : 04

Date: 04-02-26

P. Theeran

24BCS298

CSE-A1

ARRAYS

Aim:

To understand and implement array operations in Java.

PRE LAB EXERCISE

QUESTIONS

1.What is an array?

- An array is a collection of elements of the same data type stored in consecutive memory locations.

2.Why are arrays used?

- Arrays are used to store multiple values using a single name and to easily access and process large amounts of data.

3.What is the difference between array and variable?

- A **variable** stores only one value at a time.
- An **array** stores multiple values of the same data type under one name.

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] + " ");

    }

}
```

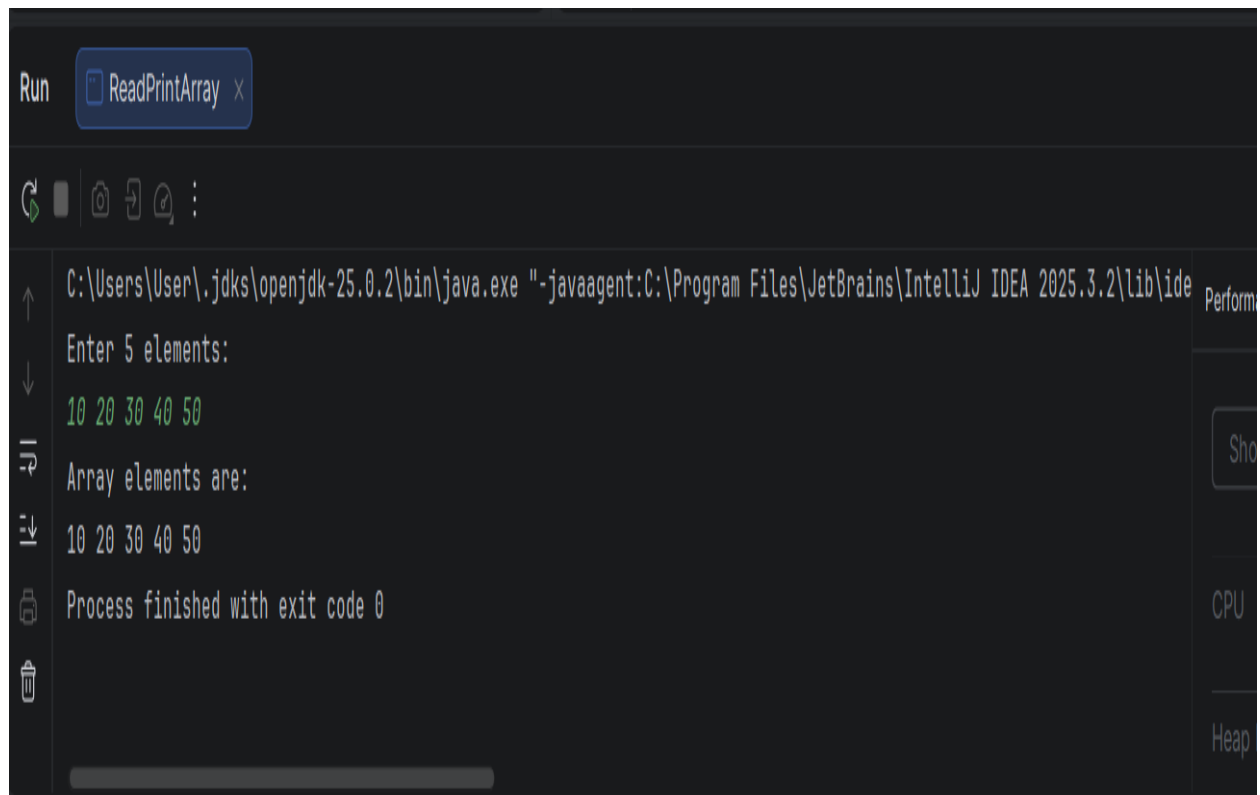
OUTPUT:**Input:**

10 20 30 40 50

Output:

Array elements are:

10 20 30 40 50



```
Run ReadPrintArray x
C:\Users\User\.jdk\openjdk-25.0.2\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2025.3.2\lib\ide
Enter 5 elements:
10 20 30 40 50
Array elements are:
10 20 30 40 50
Process finished with exit code 0
```

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



```
Run ReadPrint x
C:\Users\User\.jdk\openjdk-25.0.2\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2025.3.2\lib\idea_rt.jar=53158" -Dfile.encoding=UTF-8 -Dsun.stdout.encoding=UTF-8
Enter 5 elements:
5 10 15 20 25
Sum = 75
Process finished with exit code 0
```

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);  
    }  
}
```

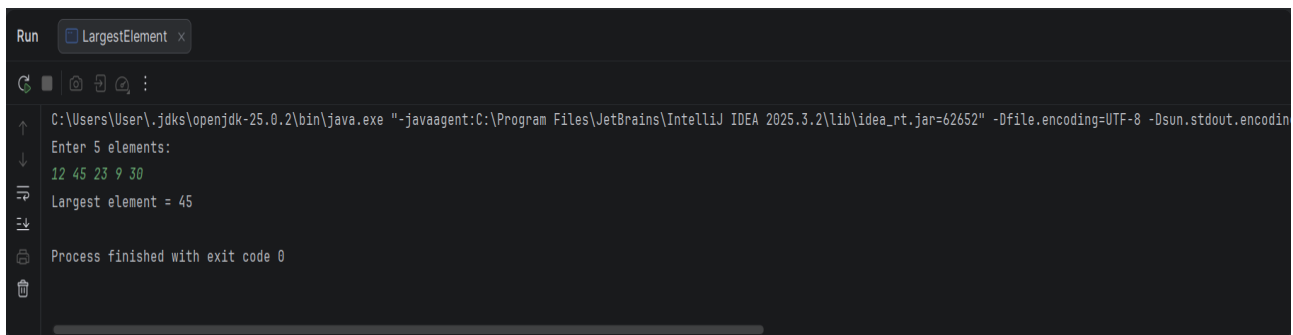
OUTPUT:

Input:

12 45 23 9 30

Output:

Largest element = 45



The screenshot shows a 'Run' window titled 'LargestElement'. It displays the command prompt output for a Java program. The command line shows the Java executable path and various JVM options. The input 'Enter 5 elements:' is followed by the user input '12 45 23 9 30'. The output is 'Largest element = 45'. The window concludes with 'Process finished with exit code 0'.

```
Run LargestElement x  
C:\Users\User\jdk\openjdk-25.0.2\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2025.3.2\lib\idea_rt.jar=62652" -Dfile.encoding=UTF-8 -Dsun.stdout.encoding=UTF-8  
Enter 5 elements:  
12 45 23 9 30  
Largest element = 45  
Process finished with exit code 0
```

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:

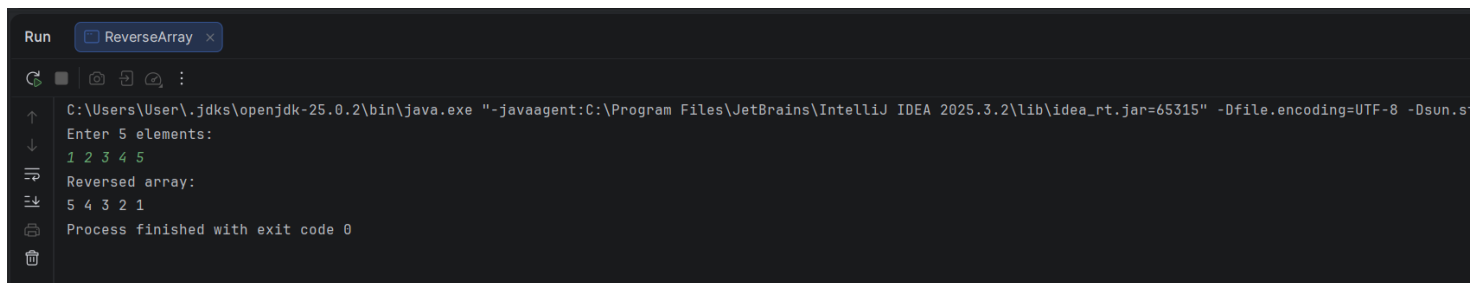
Input:

1 2 3 4 5

Output:

Reversed array:

5 4 3 2 1



The screenshot shows a terminal window titled 'Run ReverseArray'. It displays the command to run a Java program, the input '1 2 3 4 5', the output 'Reversed array: 5 4 3 2 1', and a message indicating the process finished with exit code 0.

```
Run ReverseArray x
C:\Users\User\.jdk\openjdk-25.0.2\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2025.3.2\lib\idea_rt.jar=65315" -Dfile.encoding=UTF-8 -Dsun.s
Enter 5 elements:
1 2 3 4 5
Reversed array:
5 4 3 2 1
Process finished with exit code 0
```

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:

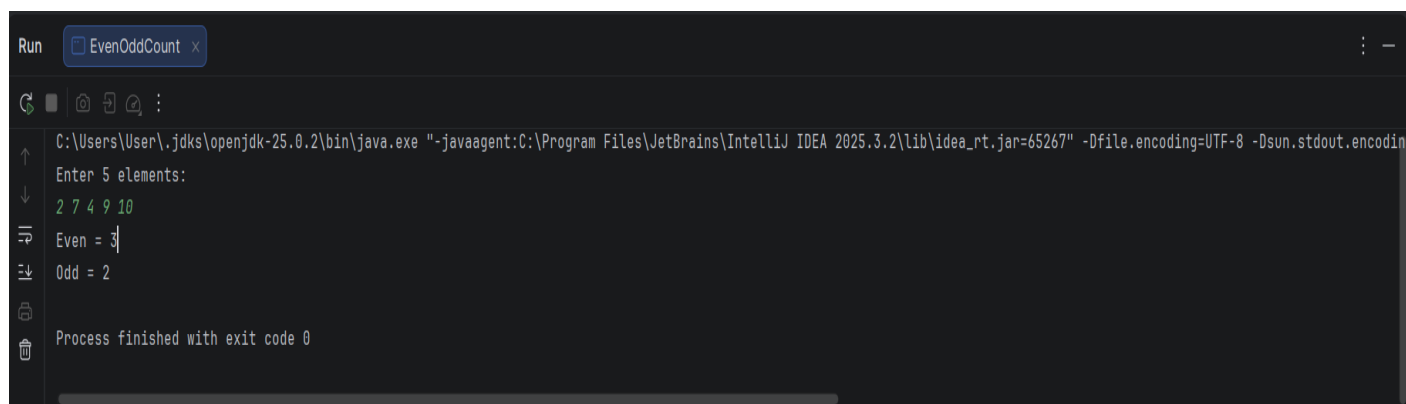
Input:

2 7 4 9 10

Output:

Even = 3

Odd = 2



The screenshot shows a 'Run' window titled 'EvenOddCount'. The command line at the top shows the Java command being executed. The output area displays the program's execution: it prompts 'Enter 5 elements:', followed by the input '2 7 4 9 10' (highlighted in green). The output then shows 'Even = 3' and 'Odd = 2'. At the bottom, it states 'Process finished with exit code 0'.

```
Run EvenOddCount x  
C:\Users\User\.jdk\openjdk-25.0.2\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2025.3.2\lib\idea_rt.jar=65267" -Dfile.encoding=UTF-8 -Dsun.stdout.encoding=UTF-8  
Enter 5 elements:  
2 7 4 9 10  
Even = 3  
Odd = 2  
Process finished with exit code 0
```

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:

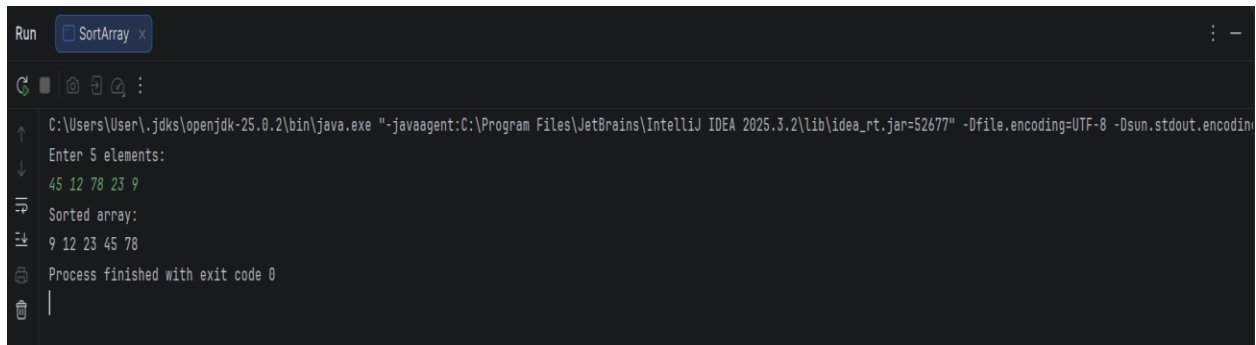
Input:

45 12 78 23 9

Output:

Sorted array:

9 12 23 45 78



```
Run SortArray x
C:\Users\User\jdk\openjdk-25.0.2\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2025.3.2\lib\idea_rt.jar=52677" -Dfile.encoding=UTF-8 -Dsun.stdout.encoding=UTF-8
Enter 5 elements:
45 12 78 23 9
Sorted array:
9 12 23 45 78
Process finished with exit code 0
```

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

```

Run SecondLargest x
C:\Users\User\.jdk\openjdk-25.0.2\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2025.3.2\lib\idea_rt.jar=58040" -Dfile.encoding=UTF-8 -Dsun.stdout.encoding=UTF-8
Enter 5 elements:
10 45 23 89 67
Second largest = 45
Process finished with exit code 0

```

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:

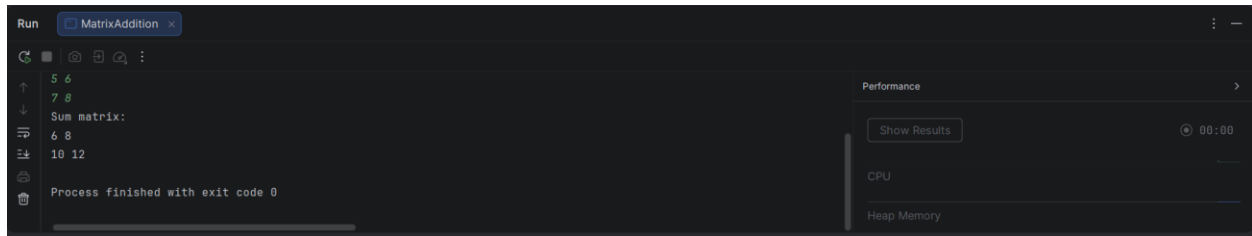
5 6

7 8

Sum matrix:

6 8

10 12



```
Run MatrixAddition x
5 6
7 8
Sum matrix:
6 8
10 12
Process finished with exit code 0

Performance
Show Results 00:00
CPU
Heap Memory
```

POST LAB EXERCISE

1. Why is array indexing usually started from zero instead of one?

- Because the index represents the offset from the starting memory address, and the first element has offset 0.

2. What happens if we try to access an array element outside its declared size?

- It causes an error or undefined behavior and may crash the program or give garbage values.

3. How does memory allocation differ for static arrays and dynamic arrays?

- **Static arrays:** Memory is allocated at compile time and size is fixed.
- **Dynamic arrays:** Memory is allocated at runtime and size can be changed.

4. Why is searching faster in arrays compared to linked lists?

- Because arrays allow direct access using index, while linked lists require sequential traversal.

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

- **Contiguous memory:** Elements are stored in continuous memory locations (arrays).
- **Non-contiguous memory:** Elements are stored at different memory locations linked together (linked lists).

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		