Experiment 01

Student Name: Deepak Kumar **UID: 22BCS10494**

Branch: BE-CSE **Section/Group: TPP-DL-902/B**

Semester: 6th **Date of Performance: 08/01/25**

Subject Name: Java Programming Lab **Subject Code: 22CSH-359**

1. **Aim:** Create an application to save the employee information using arrays.
   Given the following table containing information about employees of an organization, develop a small java application, which accepts employee id from the command prompt and displays the following details as output:
   You may assume that the array is initialized with the following details:

Emp No.	Emp Name	Join Date	Desig Code	Dept	Basic	HRA	IT
1001	Ashish	01/04/2009	e	R&D	20000	8000	3000
1002	Sushma	23/08/2012	c	PM	30000	12000	9000
1003	Rahul	12/11/2008	k	Acct	10000	8000	1000
1004	Chahat	29/01/2013	r	Front Desk	12000	6000	2000
1005	Ranjan	16/07/2005	m	Engg	50000	20000	20000
1006	Suman	1/1/2000	e	Manufacturing	23000	9000	4400
1007	Tanmay	12/06/2006	c	PM	29000	12000	10000

Objective: To learn about arrays and create a employee table and display them.

2. **Procedure/Algorithm:**

- **Initialization:** Set up the scanner object and prompt the user for the number of employees.
- **Menu Loop:** Continuously show the menu and handle user input.
- **Case 1** - Add Employee:
  Check if there's space in the employees array.
  If yes, read the employee details, create a new Employee object, and store it in the array.
  If the array is full, display an appropriate message.
- **Case 2** - Display Employees:
  If there are no employees, print a message indicating this.
  Otherwise, iterate through the employees array up to count and display each employee's details.
- **Case 3** - Exit: Print a goodbye message, close the scanner, and exit the program.
- **Default Case:** Handle invalid choices by prompting the user again.

3. **Implementation code:**

```
import java.util.Scanner;

class Employee {
    int id;
    String name;
    String department;
    double salary;

    Employee(int id, String name, String department, double salary) {
        this.id = id;
        this.name = name;
        this.department = department;
        this.salary = salary;
```

```
    }

    void displayEmployee() {
        System.out.printf("ID: %d, Name: %s, Department: %s, Salary: %.2f\n", id,
name, department, salary);
    }
}

public class EmployeeManagement {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the number of employees: ");
        int n = scanner.nextInt();
        scanner.nextLine(); // Consume newline

        Employee[] employees = new Employee[n];
        int count = 0;

        while (true) {
            System.out.println("\nMenu:");
            System.out.println("1. Add Employee");
            System.out.println("2. Display Employees");
            System.out.println("3. Exit");
            System.out.print("Choose an option: ");
            int choice = scanner.nextInt();
            scanner.nextLine(); // Consume newline

            switch (choice) {
                case 1:
                    if (count < n) {
                        System.out.print("Enter Employee ID: ");
                        int id = scanner.nextInt();
                        scanner.nextLine(); // Consume newline
                        System.out.print("Enter Employee Name: ");
                        String name = scanner.nextLine();
                        System.out.print("Enter Employee Department: ");
                        String department = scanner.nextLine();
                        System.out.print("Enter Employee Salary: ");
                        double salary = scanner.nextDouble();
```

```
                    employees[count] = new Employee(id, name, department, salary);
                    count++;
                    System.out.println("Employee added successfully!");
                } else {
                    System.out.println("Employee array is full!");
                }
                break;

            case 2:
                if (count == 0) {
                    System.out.println("No employees to display.");
                } else {
                    System.out.println("\nEmployee Details:");
                    for (int i = 0; i < count; i++) {
                        employees[i].displayEmployee();
                    }
                }
                break;

            case 3:
                System.out.println("Exiting program. Goodbye!");
                scanner.close();
                return;

            default:
                System.out.println("Invalid choice. Please try again.");
        }
    }
}
```

4. **Output:**

```
Choose an option: 1
Enter Employee ID: 1003
Enter Employee Name: Deepak
Enter Employee Department: o
Enter Employee Salary: 4589621
Employee added successfully!

Menu:
1. Add Employee
2. Display Employees
3. Exit
Choose an option: 1
Enter Employee ID: 1004
Enter Employee Name: Simmi
Enter Employee Department: i
Enter Employee Salary: 48572
Employee added successfully!

Menu:
1. Add Employee
2. Display Employees
3. Exit
Choose an option: 2

Employee Details:
ID: 1001, Name: Ashish, Department: r, Salary: 74520.00
ID: 1002, Name: Kamal, Department: q, Salary: 845621.00
ID: 1003, Name: Deepak, Department: o, Salary: 4589621.00
ID: 1004, Name: Simmi, Department: i, Salary: 48572.00
```

5. **Learning Outcomes:** Here are the learning outcomes from studying and implementing of arrays.
   a) Demonstrate: Apply key concepts to real-world scenarios to showcase understanding.
   b) Analyze: Critically evaluate information, identify patterns, and draw meaningful conclusions.
   c) Create: Develop original work, including presentations, reports, or projects, to exhibit comprehension and skills.
   d) Communicate: Convey ideas and findings effectively through oral and written communication.
   e) Collaborate: Contribute to group projects and exhibit strong teamwork capabilities in a collaborative environment.