DEPARTMENT OF
COMPUTER SCIENCE & ENGINEERING

Experiment-4

Student Name: Khyati Singh **UID: 22BCS16405**

Branch: CSE **Section/Group: 901/DL/A**

Semester: 6th **Date of Performance: 10/01/2025**

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

1) Aim: Develop Java programs using core concepts such as data structures, collections, and multithreading to manage and manipulate data.

2) Objective: To develop a functional application that effectively utilizes arrays to store, Collections in Java. ArrayList, LinkedList, HashMap, TreeMap, HashSet in Java. Multithreading in Java. Thread Synchronization. Thread Priority, Thread LifeCycle.

3) Implementation/Code:

CODE 1:

```
import java.util.*;
class Employee {
    int id;
    String name;
    double salary;
    public Employee(int id, String name, double salary) {
        this.id = id;
        this.name = name;
        this.salary = salary;
    }
    @Override
    public String toString() {
        return "ID: " + id + ", Name: " + name + ", Salary: " + salary;
    }
}
public class EmployeeManager {
    private static List<Employee> employees = new ArrayList<>();
    public static void addEmployee(int id, String name, double salary) {
        employees.add(new Employee(id, name, salary));
        System.out.println("Employee added successfully.");
    }
```

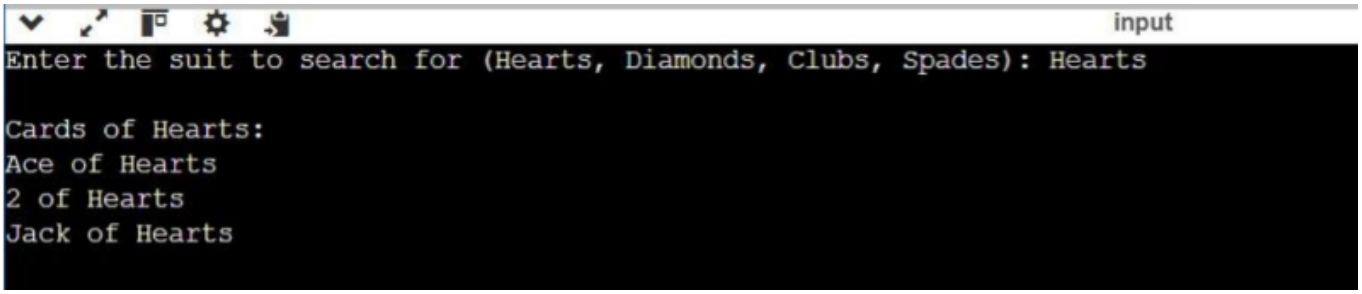
```
    public static void updateEmployee(int id, String newName, double newSalary) {
        for (Employee emp : employees) {
            if (emp.id == id) {
                emp.name = newName;
                emp.salary = newSalary;
                System.out.println("Employee updated successfully.");
                return;
            }
        }
        System.out.println("Employee not found.");
    }
    public static void removeEmployee(int id) {
        Iterator<Employee> iterator = employees.iterator();
        while (iterator.hasNext()) {
            if (iterator.next().id == id) {
                iterator.remove();
                System.out.println("Employee removed successfully.");
                return;
            }
        }
        System.out.println("Employee not found.");
    }
    public static void searchEmployee(int id) {
        for (Employee emp : employees) {
            if (emp.id == id) {
                System.out.println(emp);
                return;
            }
        }
        System.out.println("Employee not found.");
    }
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        while (true) {
            System.out.println("\n1. Add Employee\n2. Update Employee\n3. Remove
Employee\n4. Search Employee\n5. Exit");
            System.out.print("Enter choice: ");
            int choice = scanner.nextInt();
```

```
switch (choice) {
    case 1:
        System.out.print("Enter ID: ");
        int id = scanner.nextInt();
        scanner.nextLine();
        System.out.print("Enter Name: ");
        String name = scanner.nextLine();
        System.out.print("Enter Salary: ");
        double salary = scanner.nextDouble();
        addEmployee(id, name, salary);
        break;
    case 2:
        System.out.print("Enter Employee ID to update: ");
        int updateId = scanner.nextInt();
        scanner.nextLine();
        System.out.print("Enter New Name: ");
        String newName = scanner.nextLine();
        System.out.print("Enter New Salary: ");
        double newSalary = scanner.nextDouble();
        updateEmployee(updateId, newName, newSalary);
        break;
    case 3:
        System.out.print("Enter Employee ID to remove: ");
        int removeId = scanner.nextInt();
        removeEmployee(removeId);
        break;
    case 4:
        System.out.print("Enter Employee ID to search: ");
        int searchId = scanner.nextInt();
        searchEmployee(searchId);
        break;
    case 5:
        System.out.println("Exiting...");
        scanner.close();
        return;
    default:
        System.out.println("Invalid choice, please try again.");
} } } }
```

CODE 2:

```
import java.util.*;
class Card {
    private String rank;
    private String suit;
    public Card(String rank, String suit) {
        this.rank = rank;
        this.suit = suit;
    }
    public String getRank() {
        return rank;
    }
    public String getSuit() {
        return suit;
    }
    @Override
    public String toString() {
        return rank + " of " + suit;
    }
}
public class CardCollection {
    public static void main(String[] args) {
        List<Card> cards = new ArrayList<>();
        cards.add(new Card("Ace", "Hearts"));
        cards.add(new Card("2", "Hearts"));
        cards.add(new Card("King", "Diamonds"));
        cards.add(new Card("Queen", "Clubs"));
        cards.add(new Card("Jack", "Hearts"));
        cards.add(new Card("3", "Diamonds"));
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the suit to search for (Hearts, Diamonds, Clubs, Spades): ");
        String suitToSearch = scanner.nextLine();
        System.out.println("\nCards of " + suitToSearch + ":");
        for (Card card : cards) {
            if (card.getSuit().equalsIgnoreCase(suitToSearch)) {
                System.out.println(card);}
        }
        scanner.close(); } }
```

4) Output:



5) Learning Outcome:

1. Understand how to implement and manipulate ArrayList, LinkedList, HashMap, TreeMap, and HashSet in Java.
2. Gain practical knowledge of using Java collections to manage and store various types of data.
3. Learn the fundamentals of multithreading in Java and its synchronization techniques. 4. Explore thread lifecycle, thread priorities, and how they affect thread execution. 5. Develop skills to design and implement synchronized systems to avoid issues like double bookings in applications.