Introduction to Collections Framework

1. **Write a program to demonstrate adding and printing elements from an ArrayList.**

```
import java.util.ArrayList;

public class ArrayListExample {
  public static void main(String[] args) {
    ArrayList<String> fruits = new ArrayList<>();

    // Adding elements
    fruits.add("Apple");
    fruits.add("Banana");
    fruits.add("Mango");

    // Printing elements
    System.out.println("Fruits in the list:");
    for (String fruit : fruits) {
      System.out.println(fruit);
    }
  }
}
```

2. **Show how to use Collections.max() and Collections.min() on a list of integers.**

```
  import java.util.ArrayList;

  import java.util.Collections;


public class MaxMinExample {

  public static void main(String[] args) {

    ArrayList<Integer> numbers = new ArrayList<>();

    // Adding elements

    numbers.add(10);

    numbers.add(55);

    numbers.add(30);

    numbers.add(5);


    // Using max and min

    int max = Collections.max(numbers);

    int min = Collections.min(numbers);
```

```
                System.out.println("Numbers: " + numbers);
                System.out.println("Maximum: " + max);
                System.out.println("Minimum: " + min);
            }
        }
```

3. Demonstrate the use of Collections.sort() on a list of strings.

```
   import java.util.ArrayList;
import java.util.Collections;

public class SortStringsExample {
    public static void main(String[] args) {
        ArrayList<String> cities = new ArrayList<>();
        cities.add("Mumbai");
        cities.add("Delhi");
        cities.add("Chennai");
        cities.add("Bangalore");
        // Sorting
        Collections.sort(cities);
        System.out.println("Sorted cities:");
        for (String city : cities) {
            System.out.println(city);
        }
    }
}
```

4. You need to store a dynamic list of student names and display them in alphabetical order.

Implement this using a suitable collection.

```
import java.util.ArrayList;
import java.util.Collections;
import java.util.Scanner;



public class StudentNames {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        ArrayList<String> studentNames = new ArrayList<>();

        System.out.println("Enter student names (type 'end' to stop):");
        while (true) {
            String name = sc.nextLine();
            if (name.equalsIgnoreCase("end")) {
                break;
            }
            studentNames.add(name);
        }
        Collections.sort(studentNames);
        System.out.println("Student names in alphabetical order:");
        for (String name : studentNames) {
            System.out.println(name);
        }
    }
}
```

5. A user can input any number of integers. Your program should store them and display the sum of all elements using the Collection Framework.

```
import java.util.ArrayList;
import java.util.Scanner;
public class SumOfIntegers {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        ArrayList<Integer> numbers = new ArrayList<>();

        System.out.println("Enter integers (type -1 to stop):");

        while (true) {

            int num = sc.nextInt();

            if (num == -1) {

                break;

            }

            numbers.add(num);

        }

        int sum = 0;

        for (int number : numbers) {

            sum += number;

        }

        System.out.println("Sum of all elements: " + sum);

    }

}
```

List Interface

1. **Write a Java program to add, remove, and access elements in an ArrayList**

```
import java.util.ArrayList;

public class ArrayListExample {
    public static void main(String[] args) {
        ArrayList<String> fruits = new ArrayList<>();

        fruits.add("Apple");
        fruits.add("Banana");
        fruits.add("Mango");

        System.out.println("First fruit: " + fruits.get(0)); // Accessing elements

        fruits.remove("Banana"); // Removing an element

        System.out.println("Fruits list:");        // Display all elements

        for (String fruit : fruits) {
            System.out.println(fruit);
        }   } }
```

2. Implement a LinkedList that stores and prints employee names.

```
import java.util.LinkedList;

public class EmployeeList {
    public static void main(String[] args) {
        LinkedList<String> employees = new LinkedList<>();

        employees.add("Alice");
        employees.add("Bob");
        employees.add("Charlie");

        System.out.println("Employee Names:");
        for (String emp : employees) {
            System.out.println(emp);
        }
    }
}
```

3. Demonstrate inserting an element at a specific position in a List.

```
import java.util.ArrayList;

public class InsertAtPosition {
    public static void main(String[] args) {
        ArrayList<String> languages = new ArrayList<>();

        languages.add("Java");
        languages.add("Python");
        languages.add("C++");

        // Insert at index 1
        languages.add(1, "JavaScript");

        System.out.println("Programming Languages:");
        for (String lang : languages) {
            System.out.println(lang);
        }
    }
}
```

4. You're building a to-do list manager. Use ArrayList to add tasks, remove completed ones, and display pending tasks.

```
  import java.util.ArrayList;
import java.util.Scanner;

public class ToDoListManager {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        ArrayList<String> tasks = new ArrayList<>();
        while (true) {
            System.out.println("\n1. Add Task\n2. Remove Task\n3. View Tasks\n4. Exit");
            int choice = sc.nextInt();
            sc.nextLine(); // Consume newline

            switch (choice) {
                case 1:
                    System.out.print("Enter task: ");
                    String task = sc.nextLine();
                    tasks.add(task);
                    break;
                case 2:
                    System.out.print("Enter task to remove: ");
                    String removeTask = sc.nextLine();
                    tasks.remove(removeTask);
                    break;
                case 3:
                    System.out.println("Pending Tasks:");
                    for (String t : tasks) {
                        System.out.println("- " + t);
                    }
                    break;
                case 4:
                    System.out.println("Exiting To-Do List Manager.");
                    return;
                default:
                    System.out.println("Invalid choice.");
            }
        }
    }
}
```

5. Create a simple shopping cart system where users can add/remove products using a List

```
import java.util.ArrayList;

import java.util.Scanner;


public class ShoppingCart {

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        ArrayList<String> cart = new ArrayList<>();

        while (true) {

            System.out.println("\n1. Add Product\n2. Remove Product\n3. View Cart\n4. Exit");

            int choice = sc.nextInt();
            sc.nextLine(); // Consume newline

            switch (choice) {
                case 1:
                    System.out.print("Enter product name to add: ");
                    String product = sc.nextLine();
                    cart.add(product);
                    break;
                case 2:
                    System.out.print("Enter product name to remove: ");
                    String remove = sc.nextLine();
                    cart.remove(remove);
                    break;
                case 3:
                    System.out.println("Products in cart:");
                    for (String item : cart) {
                        System.out.println("- " + item);
                    }
                    break;
                case 4:
                    System.out.println("Exiting Shopping Cart.");
                    return;
                default:
                    System.out.println("Invalid choice.");
            }
        }
    }
}
```

Set Interface

1. **Write a program using HashSet to store unique student roll numbers.**

```
import java.util.HashSet;

public class UniqueRollNumbers {
   public static void main(String[] args) {
      HashSet<Integer> rollNumbers = new HashSet<>();

      // Adding roll numbers
      rollNumbers.add(101);
      rollNumbers.add(102);
      rollNumbers.add(103);
      rollNumbers.add(101); // Duplicate, will be ignored

      System.out.println("Unique Student Roll Numbers:");
      for (int roll : rollNumbers) {
         System.out.println(roll);
      }
   }
}
```

2. **Demonstrate how to use TreeSet to automatically sort elements.**

```
import java.util.TreeSet;

public class SortedNames {
   public static void main(String[] args) {
      TreeSet<String> names = new TreeSet<>();

      names.add("Zara");
      names.add("Adam");
      names.add("Eve");
      names.add("Bob");

      System.out.println("Sorted Names:");
      for (String name : names) {
         System.out.println(name);
      }
   }
}
```

3. **Use LinkedHashSet to maintain insertion order and prevent duplicates.**

```
import java.util.LinkedHashSet;

public class OrderedSubjects {
    public static void main(String[] args) {
        LinkedHashSet<String> subjects = new LinkedHashSet<>();

        subjects.add("Math");
        subjects.add("Science");
        subjects.add("English");
        subjects.add("Math"); // Duplicate, will not be added

        System.out.println("Subjects (in insertion order):");
        for (String sub : subjects) {
            System.out.println(sub);
        }
    }
}
```

4. **Design a program to store registered email IDs of users such that no duplicates are allowed.**

```
import java.util.HashSet;
import java.util.Scanner;

public class EmailRegistry {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        HashSet<String> emailSet = new HashSet<>();

        while (true) {
            System.out.print("Enter email ID (or type 'exit' to stop): ");
            String email = sc.nextLine();
            if (email.equalsIgnoreCase("exit")) break;

            if (emailSet.add(email)) {
                System.out.println("Email registered.");
            } else {
                System.out.println("Duplicate email! Not allowed.");
            }
        }

        System.out.println("\nRegistered Email IDs:");
        for (String e : emailSet) {
            System.out.println(e);
        }
    }
}
```

5. Create a program where a Set is used to eliminate duplicate entries from a list of city names entered by users.

```
import java.util.HashSet;
import java.util.Scanner;

public class UniqueCities {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        HashSet<String> cities = new HashSet<>();

        System.out.println("Enter city names (type 'done' to finish):");
        while (true) {
            String city = sc.nextLine();
            if (city.equalsIgnoreCase("done")) break;
            cities.add(city); // Set will automatically ignore duplicates
        }

        System.out.println("Unique cities entered:");
        for (String c : cities) {
            System.out.println(c);
        }
    }
}
```