3. Set Interface

🔻 Direct:

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

```
import java.util.HashSet;
import java.util.Set;

public class UniqueStudentRollNumbers {

    public static void main(String[] args) {
        // Create a Set to store unique roll numbers
        Set<Integer> rollNumbers = new HashSet<>();

        // Add roll numbers (some are duplicates)
        rollNumbers.add(101);
        rollNumbers.add(102);
        rollNumbers.add(103);
        rollNumbers.add(101);  // Duplicate
        rollNumbers.add(104);
        rollNumbers.add(102);  // Duplicate

        // Print the unique roll numbers
        System.out.println("Unique Student Roll Numbers:");
        for (Integer roll : rollNumbers) {
            System.out.println(roll);
        }
    }
}
```

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

```
import java.util.Set;
import java.util.TreeSet;

public class SortedSetExample {

    public static void main(String[] args) {
        // Create a TreeSet to store and automatically sort elements
        Set<String> cities = new TreeSet<>();

        // Add city names (in random order)
        cities.add("Mumbai");
        cities.add("Delhi");
        cities.add("Bangalore");
        cities.add("Chennai");
```

```
        cities.add("Kolkata");

        // Print the sorted set
        System.out.println("Cities in Sorted Order:");
        for (String city : cities) {
            System.out.println(city);
        }
    }
}
```

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

```
import java.util.LinkedHashSet;
import java.util.Set;

public class LinkedHashSetExample {

    public static void main(String[] args) {
        // Create a LinkedHashSet of fruits
        Set<String> fruits = new LinkedHashSet<>();

        // Add elements (with duplicates)
        fruits.add("Apple");
        fruits.add("Banana");
        fruits.add("Orange");
        fruits.add("Apple");   // Duplicate
        fruits.add("Grapes");

        // Print the set
        System.out.println("Fruits in insertion order (no duplicates):");
        for (String fruit : fruits) {
            System.out.println(fruit);
        }
    }
}
```

Scenario-Based:

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;
import java.util.Set;

public class EmailRegistry {
```

```
    public static void main(String[] args) {
        // Set to store unique email IDs
        Set<String> emailSet = new HashSet<>();

        Scanner scanner = new Scanner(System.in);

        System.out.println("=== User Email Registration ===");
        System.out.print("Enter the number of users to register: ");
        int userCount = scanner.nextInt();
        scanner.nextLine(); // consume newline

        for (int i = 1; i <= userCount; i++) {
            System.out.print("Enter email ID for user " + i + ": ");
            String email = scanner.nextLine();

            if (emailSet.contains(email)) {
                System.out.println("❌ Email already registered. Skipping.");
            } else {
                emailSet.add(email);
                System.out.println("✅ Email registered successfully.");
            }
        }

        scanner.close();

        // Display all unique registered emails
        System.out.println("\n=== Registered Email IDs ===");
        for (String email : emailSet) {
            System.out.println(email);
        }
    }
}
```

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.ArrayList;
import java.util.HashSet;
import java.util.List;
import java.util.Scanner;
import java.util.Set;

public class UniqueCities {

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
```

```
        List<String> cityList = new ArrayList<>();
        System.out.print("How many city names will you enter? ");
        int n = scanner.nextInt();
        scanner.nextLine(); // consume newline

        System.out.println("Enter " + n + " city names:");
        for (int i = 0; i < n; i++) {
            String city = scanner.nextLine().trim();
            cityList.add(city);
        }

        // Use a HashSet to eliminate duplicates
Set<String> uniqueCities = new HashSet<>(cityList);

System.out.println("\nUnique city names:");
for (String city : uniqueCities) {
System.out.println(city);
}

scanner.close();
}
}
```