ArrayList

1. Write a Program for Search an Element from ArrayList

```
import java.util.*;


public class SearchElementArrayList {
    public static void main(String[] args) {
        ArrayList<Integer> list = new ArrayList<>(Arrays.asList(10, 20, 30, 40, 50));
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a number: ");
        int num = sc.nextInt();
        if (list.contains(num)) {
            System.out.println(num + " found in the list");
        } else {
            System.out.println(num + " not found in the list");
        }
    }
}
```

Output

```
Enter a number: 30
30 found in the list
```

---

2. Write a Program for Remove Specific Element from an ArrayList

```
import java.util.*;


public class RemoveFruit {
    public static void main(String[] args) {
        ArrayList<String> fruits = new ArrayList<>(Arrays.asList("Apple", "Banana", "Mango",
"Orange", "Grapes"));
        fruits.remove("Mango");
        System.out.println(fruits);
    }
}
```

Output

```
[Apple, Banana, Orange, Grapes]
```

---

3. Write a Program for Sort Elements from ArrayList

```
import java.util.*;

public class SortArrayList {
    public static void main(String[] args) {
        ArrayList<Integer> list = new ArrayList<>(Arrays.asList(25, 10, 5, 40, 35, 15, 20));
        Collections.sort(list);
        System.out.println(list);
    }
}
```

Output

```
[5, 10, 15, 20, 25, 35, 40]
```

---

4. Write a Program for Reverse the ArrayList

```
import java.util.*;

public class ReverseArrayList {
    public static void main(String[] args) {
        ArrayList<Character> chars = new ArrayList<>(Arrays.asList('A', 'B', 'C', 'D', 'E'));
        Collections.reverse(chars);
        System.out.println(chars);
    }
}
```

Output

```
[E, D, C, B, A]
```

---

5. Write a Program for Update an Element from ArrayList

```
import java.util.*;

public class UpdateElement {
    public static void main(String[] args) {
        ArrayList<String> subjects = new ArrayList<>(Arrays.asList("Math", "Science", "English"));
        System.out.println("Before: " + subjects);
        subjects.set(subjects.indexOf("Math"), "Statistics");
        System.out.println("After: " + subjects);
    }
}
```

Output

```
Before: [Math, Science, English]
After: [Statistics, Science, English]
```

---

6. Write a Program for Remove All Elements from ArrayList

```
import java.util.*;

public class ClearArrayList {
    public static void main(String[] args) {
        ArrayList<Integer> list = new ArrayList<>(Arrays.asList(10, 20, 30, 40));
        list.clear();
        System.out.println("Size: " + list.size());
    }
}
```

Output

```
Size: 0
```

---

7. Write a Program for Iterate using Iterator

```
import java.util.*;

public class IterateCities {
    public static void main(String[] args) {
        ArrayList<String> cities = new ArrayList<>(Arrays.asList("Delhi", "Mumbai", "Bangalore",
"Chennai"));
        Iterator<String> it = cities.iterator();
        while (it.hasNext()) {
            System.out.println(it.next());
        }
    }
}
```

Output

```
Delhi
Mumbai
Bangalore
Chennai
```

---

8. Write a Program for Store Custom Objects

```
import java.util.*;

class Student {
    int id;
    String name;
    double marks;
    Student(int id, String name, double marks) {
        this.id = id; this.name = name; this.marks = marks;
    }
}

public class StudentArrayList {
    public static void main(String[] args) {
        ArrayList<Student> students = new ArrayList<>();
        students.add(new Student(1, "Amit", 85.5));
        students.add(new Student(2, "Priya", 90.2));
        students.add(new Student(3, "Raj", 78.4));
        for (Student s : students) {
            System.out.println(s.id + " " + s.name + " " + s.marks);
        }
    }
}
```

Output

```
1 Amit 85.5
2 Priya 90.2
3 Raj 78.4
```

---

9. Write a Program for Copy One ArrayList to Another

```java
import java.util.*;

public class CopyArrayList {
    public static void main(String[] args) {
        ArrayList<String> list1 = new ArrayList<>(Arrays.asList("Red", "Green", "Blue"));
        ArrayList<String> list2 = new ArrayList<>();
        list2.addAll(list1);
        System.out.println("List1: " + list1);
        System.out.println("List2: " + list2);
    }
}
```

Output

```
List1: [Red, Green, Blue]
List2: [Red, Green, Blue]
```

LinkedList

1. Write a Program for Create and Display a LinkedList

```java
import java.util.*;

public class LinkedListColors {
    public static void main(String[] args) {
        LinkedList<String> colors = new LinkedList<>(Arrays.asList("Red", "Green", "Blue", "Yellow",
"Pink"));
        for (String color : colors) {
            System.out.println(color);
        }
    }
}
```

Output

```
Red
Green
Blue
Yellow
Pink
```

---

2. Write a Program for Add Elements at First and Last Position

```java
import java.util.*;

public class LinkedListAddFirstLast {
    public static void main(String[] args) {
        LinkedList<Integer> numbers = new LinkedList<>(Arrays.asList(20, 30, 40));
        numbers.addFirst(10);
        numbers.addLast(50);
        System.out.println(numbers);
    }
}
```

Output

```
[10, 20, 30, 40, 50]
```

---

3. Write a Program for Insert Element at Specific Position

```java
import java.util.*;

public class LinkedListInsert {
    public static void main(String[] args) {
        LinkedList<String> names = new LinkedList<>(Arrays.asList("Amit", "Raj", "Priya"));
        System.out.println("Before: " + names);
        names.add(2, "Neha");
        System.out.println("After: " + names);
    }
}
```

Output

```
Before: [Amit, Raj, Priya]
After: [Amit, Raj, Neha, Priya]
```

---

4. Write a Program for Remove Elements

```
import java.util.*;


public class LinkedListRemove {
    public static void main(String[] args) {
        LinkedList<String> animals = new LinkedList<>(Arrays.asList("Cat", "Dog", "Cow", "Horse", 
"Goat"));
        animals.removeFirst();
        System.out.println("After removing first: " + animals);
        animals.removeLast();
        System.out.println("After removing last: " + animals);
        animals.remove("Cow");
        System.out.println("After removing Cow: " + animals);
    }
}
```

Output

```
After removing first: [Dog, Cow, Horse, Goat]
After removing last: [Dog, Cow, Horse]
After removing Cow: [Dog, Horse]
```

---

5. Write a Program for Search for an Element

```
import java.util.*;


public class LinkedListSearch {
    public static void main(String[] args) {
        LinkedList<String> list = new LinkedList<>(Arrays.asList("Apple", "Banana", "Mango"));
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter fruit name: ");
        String fruit = sc.nextLine();
        if (list.contains(fruit)) {
            System.out.println(fruit + " found in the list");
        } else {
            System.out.println(fruit + " not found in the list");
        }
    }
}
```

Output

```
Enter fruit name: Mango
Mango found in the list
```

---

6. Write a Program for Iterate using ListIterator

```
import java.util.*;

public class LinkedListListIterator {
    public static void main(String[] args) {
        LinkedList<String> cities = new LinkedList<>(Arrays.asList("Delhi", "Mumbai", "Chennai"));
        ListIterator<String> it = cities.listIterator();
        while (it.hasNext()) {
            System.out.println(it.next());
        }
        while (it.hasPrevious()) {
            System.out.println(it.previous());
        }
    }
}
```

Output

```
Delhi
Mumbai
Chennai
```

```
Chennai
Mumbai
Delhi
```

---

7. Write a Program for Sort a LinkedList

```java
import java.util.*;

public class LinkedListSort {
    public static void main(String[] args) {
        LinkedList<Integer> list = new LinkedList<>(Arrays.asList(30, 10, 50, 20, 40));
        Collections.sort(list);
        System.out.println(list);
    }
}
```

Output

```
[10, 20, 30, 40, 50]
```

---

8. Write a Program for Convert LinkedList to ArrayList

```java
import java.util.*;

public class LinkedListToArrayList {
    public static void main(String[] args) {
        LinkedList<String> ll = new LinkedList<>(Arrays.asList("One", "Two", "Three"));
        ArrayList<String> al = new ArrayList<>(ll);
        System.out.println("LinkedList: " + ll);
        System.out.println("ArrayList: " + al);
    }
}
```

Output

```
LinkedList: [One, Two, Three]
ArrayList: [One, Two, Three]
```

---

9. Write a Program for Store Custom Objects

```
import java.util.*;

class Book {
    int id;
    String title;
    String author;
    Book(int id, String title, String author) {
        this.id = id; this.title = title; this.author = author;
    }
}

public class LinkedListBooks {
    public static void main(String[] args) {
        LinkedList<Book> books = new LinkedList<>();
        books.add(new Book(1, "Java Basics", "James"));
        books.add(new Book(2, "Python Guide", "Guido"));
        books.add(new Book(3, "C++ Primer", "Bjarne"));
        for (Book b : books) {
            System.out.println(b.id + " " + b.title + " " + b.author);
        }
    }
}
```

Output

```
1 Java Basics James
2 Python Guide Guido
3 C++ Primer Bjarne
```

---

10. Write a Program for Clone a LinkedList

```
import java.util.*;
public class LinkedListClone {
    public static void main(String[] args) {
        LinkedList<Integer> list1 = new LinkedList<>(Arrays.asList(1, 2, 3, 4));
        LinkedList<Integer> list2 = (LinkedList<Integer>) list1.clone();
        System.out.println("Original: " + list1);
        System.out.println("Cloned: " + list2);
    }
}
```

Output

```
Original: [1, 2, 3, 4]
Cloned: [1, 2, 3, 4]
```

Vector

1. Write a Program for Add 5 integers, insert at 3rd position, remove 2nd element, display using Enumeration

```
import java.util.*;

public class VectorIntegers {
    public static void main(String[] args) {
        Vector<Integer> v = new Vector<>();
        v.add(10);
        v.add(20);
        v.add(30);
        v.add(40);
        v.add(50);
        v.add(2, 25);
        v.remove(1);
        Enumeration<Integer> e = v.elements();
        while (e.hasMoreElements()) {
            System.out.println(e.nextElement());
        }
    }
}
```

Output

```
10

25

30

40

50
```

---

2. Write a Program for Vector of Strings: add names, check if exists, replace name, clear

```
import java.util.*;
public class VectorStrings {
    public static void main(String[] args) {
        Vector<String> v = new Vector<>(Arrays.asList("Amit", "Raj", "Priya", "Neha"));
        System.out.println(v.contains("Raj"));
        v.set(1, "Ravi");
        System.out.println(v);
        v.clear();
        System.out.println(v);
    }
}
```

Output

```
true
[Amit, Ravi, Priya, Neha]
[]
```

---

3. Write a Program for Copy elements from one Vector to another

```
import java.util.*;
public class VectorCopy {
    public static void main(String[] args) {
        Vector<String> v1 = new Vector<>(Arrays.asList("Red", "Green", "Blue"));
        Vector<String> v2 = new Vector<>();
        v2.addAll(v1);
        System.out.println(v1);
        System.out.println(v2);
    }
}
```

Output

```
[Red, Green, Blue]
[Red, Green, Blue]
```

---

4. Write a Program for Compare two Vectors

```
import java.util.*;

public class VectorCompare {
    public static void main(String[] args) {
        Vector<Integer> v1 = new Vector<>(Arrays.asList(1, 2, 3));
        Vector<Integer> v2 = new Vector<>(Arrays.asList(1, 2, 3));
        System.out.println(v1.equals(v2));
    }
}
```

Output

```
true
```

---

5. Write a Program for Method to return sum of Vector<Integer>

```
import java.util.*;

public class VectorSum {
    public static int sum(Vector<Integer> v) {
        int total = 0;
        for (int num : v) total += num;
        return total;
    }
    public static void main(String[] args) {
        Vector<Integer> v = new Vector<>(Arrays.asList(5, 10, 15));
        System.out.println(sum(v));
    }
}
```

Output

```
30
```

Stack

1. Write a Program for Push 5 elements, pop top, peek top, check if empty

```
import java.util.*;

public class StackOperations {
    public static void main(String[] args) {
        Stack<Integer> stack = new Stack<>();
        stack.push(10);
        stack.push(20);
        stack.push(30);
        stack.push(40);
        stack.push(50);
        stack.pop();
        System.out.println("Top: " + stack.peek());
        System.out.println("Is empty: " + stack.isEmpty());
    }
}
```

Output

```
Top: 40
Is empty: false
```

---

2. Write a Program for Reverse a string using Stack

```
import java.util.*;

public class ReverseStringStack {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a string: ");
        String str = sc.nextLine();
        Stack<Character> stack = new Stack<>();
        for (char c : str.toCharArray()) stack.push(c);
        String rev = "";
        while (!stack.isEmpty()) rev += stack.pop();
        System.out.println("Reversed: " + rev);
    }
}
```

Output

```
Enter a string: hello
Reversed: olleh
```

---

3. Write a Program for Check for balanced parentheses

```
import java.util.*;

public class BalancedParentheses {
    public static void main(String[] args) {
        String expr = "(a+b)*(c-d)";
        Stack<Character> stack = new Stack<>();
        boolean valid = true;
        for (char c : expr.toCharArray()) {
            if (c == '(') stack.push(c);
            else if (c == ')') {
                if (stack.isEmpty()) { valid = false; break; }
                stack.pop();
            }
        }
        if (!stack.isEmpty()) valid = false;
        System.out.println(valid ? "Valid" : "Invalid");
    }
```

```java
}
```

Output

```
Valid
```

---

4. Write a Program for Decimal to binary using Stack

```java
import java.util.*;

public class DecimalToBinary {
    public static void main(String[] args) {
        int num = 13;
        Stack<Integer> stack = new Stack<>();
        while (num > 0) {
            stack.push(num % 2);
            num /= 2;
        }
        while (!stack.isEmpty()) System.out.print(stack.pop());
    }
}
```

Output

```
1101
```

HashSet

1. Write a Program for Create HashSet of Strings, add cities, try duplicate, iterate

```java
import java.util.*;

public class HashSetCities {
    public static void main(String[] args) {
        HashSet<String> cities = new HashSet<>();
        cities.add("Delhi");
        cities.add("Mumbai");
        cities.add("Chennai");
        cities.add("Bangalore");
        cities.add("Kolkata");
        cities.add("Mumbai");
        Iterator<String> it = cities.iterator();
        while (it.hasNext()) {
            System.out.println(it.next());
        }
    }
}
```

Output

```
Bangalore
Kolkata
Delhi
Chennai
Mumbai
```

---

2. Write a Program for Remove element, check if exists, clear

```
import java.util.*;

public class HashSetOperations {
    public static void main(String[] args) {
        HashSet<String> set = new HashSet<>(Arrays.asList("Delhi", "Mumbai", "Chennai"));
        set.remove("Mumbai");
        System.out.println(set.contains("Delhi"));
        set.clear();
        System.out.println(set);
    }
}
```

Output

```
true
[]
```

---

3. Write a Program for Method to return max element

```
import java.util.*;

public class HashSetMax {
    public static int getMax(HashSet<Integer> set) {
        return Collections.max(set);
    }
    public static void main(String[] args) {
        HashSet<Integer> nums = new HashSet<>(Arrays.asList(5, 10, 15, 20));
        System.out.println(getMax(nums));
    }
}
```

Output

```
20
```

LinkedHashSet

1. Write a Program for Add numbers and observe order

```
import java.util.*;

public class LinkedHashSetOrder {
    public static void main(String[] args) {
        LinkedHashSet<Integer> set = new LinkedHashSet<>();
        set.add(10);
        set.add(5);
        set.add(20);
        set.add(15);
        set.add(5);
        System.out.println(set);
    }
}
```

Output

```
[10, 5, 20, 15]
```

---

2. Write a Program for Custom objects with equals() and hashCode()

```
import java.util.*;

class Student {
    int id;
    String name;
    Student(int id, String name) {
        this.id = id; this.name = name;
    }
    public boolean equals(Object o) {
        if (this == o) return true;
        if (!(o instanceof Student)) return false;
        Student s = (Student) o;
        return id == s.id && name.equals(s.name);
    }
    public int hashCode() {
        return Objects.hash(id, name);
    }
}

public class LinkedHashSetStudents {
    public static void main(String[] args) {
        LinkedHashSet<Student> set = new LinkedHashSet<>();
        set.add(new Student(1, "Amit"));
        set.add(new Student(2, "Raj"));
        set.add(new Student(3, "Priya"));
        set.add(new Student(1, "Amit"));
        for (Student s : set) {
            System.out.println(s.id + " " + s.name);
        }
    }
}
```

Output

```
1 Amit
2 Raj
3 Priya
```

---

3. Write a Program for Merge two LinkedHashSets

```
import java.util.*;

public class LinkedHashSetMerge {
    public static void main(String[] args) {
        LinkedHashSet<String> set1 = new LinkedHashSet<>(Arrays.asList("A", "B", "C"));
        LinkedHashSet<String> set2 = new LinkedHashSet<>(Arrays.asList("D", "E", "C"));
        set1.addAll(set2);
        System.out.println(set1);
    }
}
```

Output

```
[A, B, C, D, E]
```

TreeSet

1. Write a Program for Add countries in random order and print sorted

```
import java.util.*;

public class TreeSetCountries {
    public static void main(String[] args) {
        TreeSet<String> countries = new TreeSet<>();
        countries.add("India");
        countries.add("USA");
        countries.add("Brazil");
        countries.add("Australia");
        countries.add("Canada");
        System.out.println(countries);
    }
}
```

Output

```
[Australia, Brazil, Canada, India, USA]
```

---

2. Write a Program for First, last, lower, higher elements

```
import java.util.*;

public class TreeSetNumbers {
    public static void main(String[] args) {
        TreeSet<Integer> nums = new TreeSet<>(Arrays.asList(10, 20, 30, 40, 50));
        System.out.println("First: " + nums.first());
        System.out.println("Last: " + nums.last());
        System.out.println("Lower than 30: " + nums.lower(30));
        System.out.println("Higher than 30: " + nums.higher(30));
    }
}
```

Output

```
First: 10
Last: 50
Lower than 30: 20
Higher than 30: 40
```

---

3. Write a Program for Custom comparator for reverse alphabetical order

```
import java.util.*;

public class TreeSetReverse {
    public static void main(String[] args) {
        TreeSet<String> set = new TreeSet<>(Collections.reverseOrder());
        set.add("Banana");
        set.add("Apple");
        set.add("Mango");
        System.out.println(set);
    }
}
```

Output

```
[Mango, Banana, Apple]
```

Queue

1. Write a Program for Bank Queue Simulation

```
import java.util.*;

public class BankQueue {
    public static void main(String[] args) {
        Queue<String> queue = new LinkedList<>();
        queue.add("Customer1");
        queue.add("Customer2");
        queue.add("Customer3");
        queue.add("Customer4");
        queue.add("Customer5");
        while (!queue.isEmpty()) {
            System.out.println("Serving: " + queue.poll());
            System.out.println("Queue: " + queue);
        }
    }
}
```

Output

```
Serving: Customer1
Queue: [Customer2, Customer3, Customer4, Customer5]
Serving: Customer2
```

```
Queue: [Customer3, Customer4, Customer5]
Serving: Customer3
Queue: [Customer4, Customer5]
Serving: Customer4
Queue: [Customer5]
Serving: Customer5
Queue: []
```

---

2. Write a Program for Task Manager

```java
import java.util.*;

public class TaskManager {
    public static void main(String[] args) {
        Queue<String> tasks = new LinkedList<>();
        tasks.add("Task1");
        tasks.add("Task2");
        tasks.add("Task3");
        System.out.println("Next task: " + tasks.peek());
        tasks.poll();
        System.out.println("Queue after removing: " + tasks);
    }
}
```

Output

```
Next task: Task1
Queue after removing: [Task2, Task3]
```

---

3. Write a Program for Method to get even numbers from a queue

```java
import java.util.*;

public class QueueEvenNumbers {
    public static List<Integer> getEvenNumbers(Queue<Integer> q) {
        List<Integer> evens = new ArrayList<>();
        for (int num : q) {
            if (num % 2 == 0) evens.add(num);
        }
        return evens;
    }
    public static void main(String[] args) {
        Queue<Integer> q = new LinkedList<>(Arrays.asList(1, 2, 3, 4, 5, 6));
        System.out.println(getEvenNumbers(q));
    }
}
```

Output

```
[2, 4, 6]
```

Write a Programs for PriorityQueue

1. Write a Program for Hospital Emergency Queue (Highest priority first)

```
import java.util.*;

class Patient {
    String name;
    int priority; // Higher number = more urgent
    Patient(String name, int priority) {
        this.name = name;
        this.priority = priority;
    }
    public String toString() {
        return name + " (Priority: " + priority + ")";
    }
}

public class HospitalQueue {
    public static void main(String[] args) {
        PriorityQueue<Patient> pq = new PriorityQueue<>((a, b) -> b.priority - a.priority);
        pq.add(new Patient("John", 2));
        pq.add(new Patient("Mary", 5));
        pq.add(new Patient("Alex", 3));

        while (!pq.isEmpty()) {
            System.out.println("Treating: " + pq.poll());
        }
    }
}
```

Output

```
Treating: Mary (Priority: 5)
Treating: Alex (Priority: 3)
Treating: John (Priority: 2)
```

---

2. Write a Program for Print Job Priorities

```
import java.util.*;

public class PrintQueue {
    public static void main(String[] args) {
        PriorityQueue<Integer> printJobs = new PriorityQueue<>(Collections.reverseOrder());
        printJobs.add(5); // High priority
        printJobs.add(1); // Low priority
        printJobs.add(3);

        while (!printJobs.isEmpty()) {
            System.out.println("Printing job with priority: " + printJobs.poll());
        }
    }
}
```

Output

```
Printing job with priority: 5
Printing job with priority: 3
Printing job with priority: 1
```

---

3. Write a Program for Merging Two PriorityQueues

```
import java.util.*;

public class MergePQ {
    public static void main(String[] args) {
        PriorityQueue<Integer> pq1 = new PriorityQueue<>(Collections.reverseOrder());
        pq1.addAll(Arrays.asList(5, 1, 3));

        PriorityQueue<Integer> pq2 = new PriorityQueue<>(Collections.reverseOrder());
        pq2.addAll(Arrays.asList(4, 2, 6));

        pq1.addAll(pq2);
        System.out.println("Merged PriorityQueue: " + pq1);
    }
}
```

Output

```
Merged PriorityQueue: [6, 5, 4, 1, 2, 3]
```

Map

1. Write a Program for Student Database using HashMap

```
import java.util.*;

public class StudentDatabase {
    public static void main(String[] args) {
        Map<Integer, String> students = new HashMap<>();
        students.put(101, "Alice");
        students.put(102, "Bob");
        students.put(103, "Charlie");
        for (Map.Entry<Integer, String> entry : students.entrySet()) {
            System.out.println("Roll No: " + entry.getKey() + ", Name: " + entry.getValue());
        }
    }
}
```

Output

```
Roll No: 101, Name: Alice
Roll No: 102, Name: Bob
Roll No: 103, Name: Charlie
```

---

2. Write a Program for Word Frequency Counter using LinkedHashMap

```
import java.util.*;

public class WordFrequency {
    public static void main(String[] args) {
        String text = "apple banana apple orange banana apple";
        String[] words = text.split(" ");

        Map<String, Integer> freq = new LinkedHashMap<>();
        for (String word : words) {
            freq.put(word, freq.getOrDefault(word, 0) + 1);
        }

        System.out.println(freq);
    }
}
```

Output

```
{apple=3, banana=2, orange=1}
```

---

3. Write a Program for Sorting Map by Keys using TreeMap

```
import java.util.*;
```

```
public class SortMapByKeys {
    public static void main(String[] args) {
        Map<String, Integer> scores = new TreeMap<>();
        scores.put("John", 85);
        scores.put("Alice", 92);
        scores.put("Bob", 78);

        for (Map.Entry<String, Integer> entry : scores.entrySet()) {
            System.out.println(entry.getKey() + " => " + entry.getValue());
        }
    }
}
```

Output

```
Alice => 92
Bob => 78
John => 85
```

Set

1. Write a Program for Removing Duplicate Numbers using HashSet

```
import java.util.*;

public class RemoveDuplicates {
    public static void main(String[] args) {
        List<Integer> numbers = Arrays.asList(1, 2, 2, 3, 4, 4, 5);

        Set<Integer> uniqueNumbers = new HashSet<>(numbers);

        System.out.println("Unique Numbers: " + uniqueNumbers);
    }
}
```

Output

```
Unique Numbers: [1, 2, 3, 4, 5]
```

(Order is not guaranteed in a HashSet)

---

2. Write a Program for Maintaining Insertion Order using LinkedHashSet

```java
import java.util.*;

public class OrderedSetExample {
    public static void main(String[] args) {
        Set<String> cities = new LinkedHashSet<>();
        cities.add("Bengaluru");
        cities.add("Mysuru");
        cities.add("Hubballi");
        cities.add("Mysuru"); // Duplicate, ignored

        System.out.println("Cities: " + cities);
    }
}
```

Output

```
Cities: [Bengaluru, Mysuru, Hubballi]
```

---

3. Write a Program for Sorted Unique Words using TreeSet

```java
import java.util.*;

public class SortedUniqueWords {
    public static void main(String[] args) {
        String text = "banana apple orange mango apple banana";
        String[] words = text.split(" ");

        Set<String> sortedWords = new TreeSet<>(Arrays.asList(words));

        System.out.println("Sorted Unique Words: " + sortedWords);
    }
}
```

Output

```
Sorted Unique Words: [apple, banana, mango, orange]
```