1] Hash set

import java.util.HashSet;

public class HashSetExample{

public static void main(String[] args){

HashSet<String> set=new HashSet<>();

set.add("Apple");

set.add("Banana");

set.add("Cherry");

set.add("Apple");

System.out.println("HashSet: "+set);

System.out.println("Apple: "+set.contains("Apple"));

set.remove("Banana");

System.out.println("after removing Banana: "+set);

set.clear();

System.out.println("after clearing: "+set);

}

}

Output:

HashSet: [Apple, Cherry, Banana]

Apple: true

after removing Banana: [Apple, Cherry]

after clearing: []

=== Code Execution Successful ===

2] linked hash set

import java.util.LinkedHashSet;

public class LinkedHashSetExample {

public static void main(String[] args) {

LinkedHashSet<String> set = new LinkedHashSet<>();

set.add("Apple");

set.add("Banana");

set.add("Cherry");

set.add("Apple");

System.out.println("LinkedHashSet: " + set);

System.out.println("Apple: " + set.contains("Apple"));

set.remove("Banana");

System.out.println("after removing Banana: "+set);

set.clear();

System.out.println("after clearing: "+set);

}

}

Output:

LinkedHashSet: [Apple, Banana, Cherry]

Apple: true

after removing Banana: [Apple, Cherry]

after clearing: []

=== Code Execution Successful ===

3] tree set

import java.util.TreeSet;

public class TreeSetExample{

public static void main(String[] args){

TreeSet<String> set2=new TreeSet<>();

set2.add("One");

set2.add("Two");

set2.add("Three");

set2.add("Four");

set2.add("Five");

System.out.println("TreeSet2: "+set2);

System.out.println("Is TreeSet2 empty: " + set2.isEmpty());

System.out.println("Size of TreeSet2: " + set2.size());

set2.clear();

System.out.println("after clearing: "+set2);

}

}

Output:

TreeSet2: [Five, Four, One, Three, Two]

Is TreeSet2 empty: false

Size of TreeSet2: 5

after clearing: []

=== Code Execution Successful ===

4] Create list using Linked List

class Node{

int data;

Node next;

public Node(int data){

this.data=data;

this.next=null;

}

}

public class MyLinkedList {

Node head;

public void addNode(int data){

Node newNode=new Node(data);

if (head==null){

head=newNode;

} else{

Node temp=head;

while(temp.next!=null){

temp=temp.next;

}

temp.next=newNode;

}

}

public void printList(){

Node temp=head;

while (temp!=null){

System.out.print(temp.data+" ");

temp=temp.next;

}

}

public static void main(String[] args){

MyLinkedList list=new MyLinkedList();

list.addNode(10);

list.addNode(20);

list.addNode(30);

list.addNode(40);

list.addNode(50);

System.out.println("Linked List:");

list.printList();

}

}

Output:

Linked List:

10 20 30 40 50

=== Code Execution Successful ===

5] Create list using Stack

import java.util.Stack;

public class StackList{

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);

System.out.println("Stack List:");

printStack(stack);

}

public static void printStack(Stack<Integer> stack){

while (!stack.isEmpty()) {

System.out.print(stack.pop()+" ");

}

}

}

Output:

Stack List:

50 40 30 20 10

=== Code Execution Successful ===

6] Create list using Vector

import java.util.Vector;

public class VectorList{

public static void main(String[] args){

Vector<Integer> vector=new Vector<>();

vector.addElement(10);

vector.addElement(20);

vector.addElement(30);

vector.addElement(40);

vector.addElement(50);

System.out.println("Vector List:");

System.out.print(vector);

}

}

Output:

Vector List:

[10, 20, 30, 40, 50]

=== Code Execution Successful ===