

PROBLEM STATEMENT 3

Name	PRN	Roll No.	Batch
Prathamesh Takalkar	1032201443	PA_36	A3
Durva Chavan	1032201697	PA_41	A3
Tejaswini Joshi	1032201734	PA_42	A3
Tanvi Moholkar	1032201961	PA_45	A3

ASSIGNMENT NO: 9**Aim:**

Write a Java program to show the use of Collection Frameworks in Java

Objectives:

1. To study the concept of Collection Framework.

Theory:

1. Collection, Collection Framework

The Java platform includes a collections framework. A collection is an object that represents a group of objects (such as the classic Vector class). A collections framework is a unified architecture for representing and manipulating collections, enabling collections to be manipulated independently of implementation details.

2. Methods of collection framework.

- addAll(): It is used to add all of the specified elements to the specified collection.
- copy (): It is used to copy all the elements from one list into another list.
- disjoint (): It returns true if the two specified collections have no elements in common.
- emptyList(): It is used to get a List that has no elements.
- emptySet(): It is used to get the set that has no elements.
- max (): It is used to get the maximum value of the given collection, according to the natural ordering of its elements.
- min (): It is used to get the minimum value of the given collection, according to the natural ordering of its elements.
- reverse (): It is used to reverse the order of the elements in the specified list.
- sort (): It is used to sort the elements present in the specified list of collection in ascending order.

Platform:

Open-Source Java Programming tool like Eclipse Editor/NetBeans.

Conclusion:

Thus, studied the concept of the collection framework.

PROBLEM STATEMENT 3

Name	PRN	Roll No.	Batch
Prathamesh Takalkar	1032201443	PA_36	A3
Durva Chavan	1032201697	PA_41	A3
Tejaswini Joshi	1032201734	PA_42	A3
Tanvi Moholkar	1032201961	PA_45	A3

FAQs:

1) What is the difference between ArrayList and LinkedList?

ArrayList	LinkedList
ArrayList internally uses a dynamic array to store the elements.	LinkedList internally uses a doubly linked list to store the elements.
Manipulation with ArrayList is slow because it internally uses an array. If any element is removed from the array, all the other elements are shifted in memory.	Manipulation with LinkedList is faster than ArrayList because it uses a doubly linked list, so no bit shifting is required in memory.
3) An ArrayList class can act as a list only because it implements List only.	LinkedList class can act as a list and queue both because it implements List and Deque interfaces.
4) ArrayList is better for storing and accessing data.	LinkedList is better for manipulating data.
5) The memory location for the elements of an ArrayList is contiguous.	The location for the elements of a linked list is not contiguous.
6) Generally, when an ArrayList is initialized, a default capacity of 10 is assigned to the ArrayList.	There is no case of default capacity in a LinkedList. In LinkedList, an empty list is created when a LinkedList is initialized.
7) To be precise, an ArrayList is a resizable array.	LinkedList implements the doubly linked list of the list interface.

2) What is the difference between hashset and treeset?

Key	Hash Set	Tree Set
Implementation	Hash set is implemented using HashTable	The tree set is implemented using a tree structure.
Null Object	HashSet allows a null object	The tree set does not allow the null object. It throws the null pointer exception.
Methods	Hash set use equals method to compare two objects	Tree set use compare method for comparing two objects.
Heterogeneous object	Hash set doesn't now allow a heterogeneous object	Tree set allows a heterogeneous object
Ordering	HashSet does not maintain any order	TreeSet maintains an object in sorted order

PROBLEM STATEMENT 3

Name	PRN	Roll No.	Batch
Prathamesh Takalkar	1032201443	PA_36	A3
Durva Chavan	1032201697	PA_41	A3
Tejaswini Joshi	1032201734	PA_42	A3
Tanvi Moholkar	1032201961	PA_45	A3

3) List down the primary interfaces provided by the java collection framework.

a. The core collection interfaces are:

- Collection: The root of the collection hierarchy. ...
- Set: A collection that cannot contain duplicate elements. ...
- List: An ordered collection (sometimes called a sequence). ...
- Queue: A collection used to hold multiple elements prior to processing. ...
- Map: An object that maps keys to values.

4) What do you understand by Iterator in the Java Collection Framework?

- a. Iterators in Java are used in the Collection framework to retrieve elements one by one. It is a universal iterator as we can apply it to any Collection object. By using Iterator, we can perform both read and remove operations. It is an improved version of Enumeration with the additional functionality of removing an element.
- b. Iterator must be used whenever we want to enumerate elements in all Collection framework implemented interfaces like Set, List, Queue, Deque, and all implemented classes of Map interface. Iterator is the only cursor available for the entire collection framework.
- c. Iterator object can be created by calling iterator () method present in Collection interface.

PROBLEM STATEMENT 3

Name	PRN	Roll No.	Batch
Prathamesh Takalkar	1032201443	PA_36	A3
Durva Chavan	1032201697	PA_41	A3
Tejaswini Joshi	1032201734	PA_42	A3
Tanvi Moholkar	1032201961	PA_45	A3

CODE:

```
package tri_6;

/* Problem statement 3:
Write a Java program:
1. to create a new tree set
2. add some colors(string) and print out the tree set.
3. add all the elements of a specified tree set to another tree set.
4. get the number of elements in a tree set
5. find the numbers less than 7 in a tree set.
6. remove a given element from a tree set

Group members:
PA36_Prathamesh Takalkar
PA41_Durva Chavan
PA42_Tejaswini Joshi
PA45_Tanvi Moholkar
*/

import java.util.*;

public class PS3_Active_Learning {
    public static void main(String[] args) {
        // Creation of a treeSet
        TreeSet<String> treeSet1 = new TreeSet<>();

        // Adding colors to the string
        treeSet1.add("Pink");
        treeSet1.add("Black");
        treeSet1.add("White");
        treeSet1.add("Green");
        treeSet1.add("Blue");
        treeSet1.add("Yellow");

        // Printing the treeSet
```

PROBLEM STATEMENT 3

Name	PRN	Roll No.	Batch
Prathamesh Takalkar	1032201443	PA_36	A3
Durva Chavan	1032201697	PA_41	A3
Tejaswini Joshi	1032201734	PA_42	A3
Tanvi Moholkar	1032201961	PA_45	A3

```
Iterator<String> iterator = treeSet1.iterator();
while (iterator.hasNext()) {
    System.out.println(iterator.next());
}

// Creating another TreeSet
TreeSet<String> treeSet2 = new TreeSet<>();

// Adding elements to a tree set
treeSet2.add("Cat");
treeSet2.add("Dog");
treeSet2.add("Tiger");
treeSet2.add("Lion");
treeSet2.add("Wolf");

// Adding TreeSet2 to TreeSet1
System.out.println("TreeSet1: " + treeSet1 + "\nTreeSet2: " + treeSet2);
System.out.println(treeSet1.addAll(treeSet2));
System.out.println("TreeSet1 after adding elements of TreeSet2 to it: " + treeSet1);

// Number of elements in the treeSet
System.out.println("Number of elements in treeSet1: " + treeSet1.size());
System.out.println("Number of elements in treeSet2: " + treeSet2.size());

// Creating a treeSet with numbers
TreeSet<Integer> treeSet3 = new TreeSet<>();
TreeSet<Integer> treeheadset = new TreeSet<>();

// Adding numbers in treeSet3
treeSet3.add(32);
treeSet3.add(2);
treeSet3.add(98);
treeSet3.add(0);
treeSet3.add(231);
treeSet3.add(4);
```

PROBLEM STATEMENT 3

Name	PRN	Roll No.	Batch
Prathamesh Takalkar	1032201443	PA_36	A3
Durva Chavan	1032201697	PA_41	A3
Tejaswini Joshi	1032201734	PA_42	A3
Tanvi Moholkar	1032201961	PA_45	A3

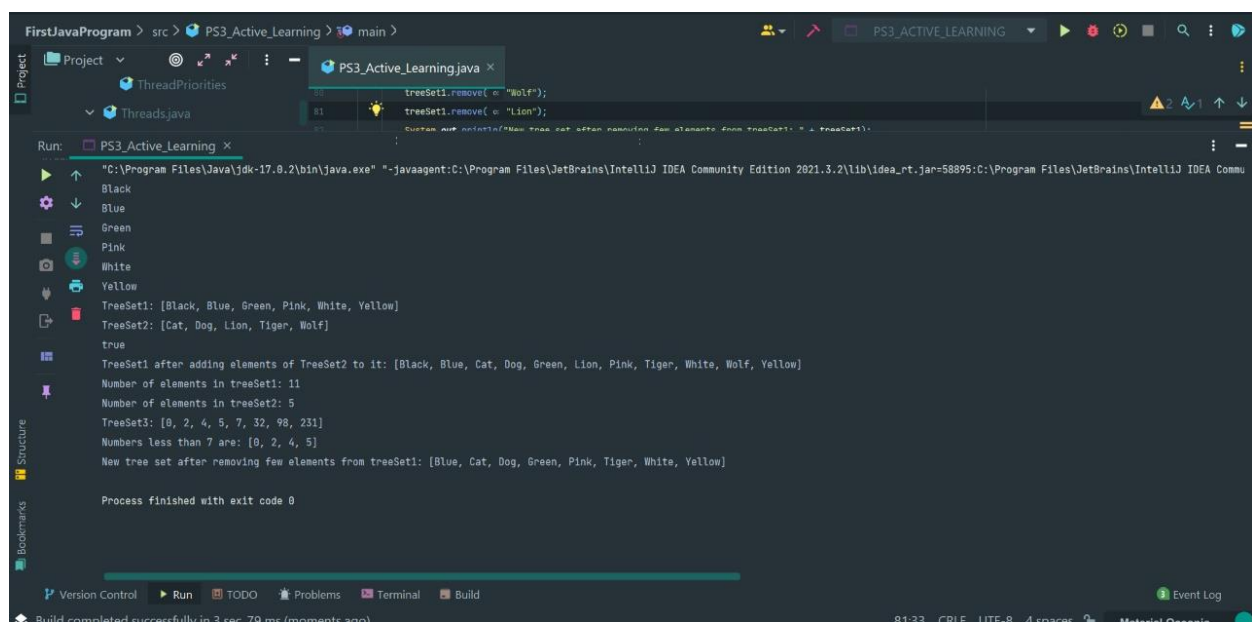
```
treeSet3.add(5);
treeSet3.add(7);

// Displaying treeSet3
System.out.println("TreeSet3: " + treeSet3);

// Finding numbers less than 7
treeheadset = (TreeSet<Integer>) treeSet3.headSet(7);
System.out.println("Numbers less than 7 are: " + treeheadset);

// Removing an element from the treeSet
treeSet1.remove("Black");
treeSet1.remove("Wolf");
treeSet1.remove("Lion");
System.out.println("New tree set after removing few elements from treeSet1: " + treeSet1);
}
}
```

OUTPUT:



```
Run: PS3_Active_Learning x
"C:\Program Files\Java\jdk-17.0.2\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2021.3.2\lib\idea_rt.jar=58895:C:\Program Files\JetBrains\IntelliJ IDEA Commu
Black
Blue
Green
Pink
White
Yellow
TreeSet1: [Black, Blue, Green, Pink, White, Yellow]
TreeSet2: [Cat, Dog, Lion, Tiger, Wolf]
true
TreeSet1 after adding elements of TreeSet2 to it: [Black, Blue, Cat, Dog, Green, Lion, Pink, Tiger, White, Wolf, Yellow]
Number of elements in treeSet1: 11
Number of elements in treeSet2: 5
TreeSet3: [0, 2, 4, 5, 7, 32, 98, 231]
Numbers less than 7 are: [0, 2, 4, 5]
New tree set after removing few elements from treeSet1: [Blue, Cat, Dog, Green, Pink, Tiger, White, Yellow]

Process finished with exit code 0
```