Page No.
Date : / /

Ch-4 - Introduction To Collection

* Explain Collection Interface.

=> The collection interface is the root interface of the Java Collection Framework.

There are three types of Collection

1) List
2) Set
3) Queue

1 List Interface:

The list interface is allows to add and remove elements in lists.

2 Set Interface:

The set interface is allows to store elements in different sets in Maths.

3 Queue :

The Queue interface is used when we want to store and access elements in FIFO Manner.

* Explain Arraylist with example.

=> Arraylist is a type of array in which we do not have to defined array Length or size.

Arraylist is also called resizable arrays.

Using array we have defined array size and this size can not change.

But Using arraylist we do not have to defined any type of size of array.

- Syntax:

```
ArrayList <data> arraylist =
          type    name

        new ArrayList();
```

SMVS
Page No.
Date: / /

For use of arraylist, first we have to defined one util arraylist package.

Arraylist can provide different method.

1) Add Elements
2) Access Elements
3) Change Elements
4) Remove Elements

Ex.

```
import java.util.ArrayList;

Class java
{
  public static void main (
        String, args[])
{
  ArrayList<String> Java =

        new ArrayList();

    Java.add("A");
    Java.add("B");
    Java.add("C");

  System.out.println (Java);
```

```
String a = Java.get(1);
System.out.println(a);
Java.set(2, "E");
System.out.println(Java);
Java String b = Java.remove(0);
System.out.println(b);
}
}
```

* Explain LinkedList with example.

=> Java Collection Framework LinkedList provides Doubly linkedlist Function on Data Structure.

Each element in Linked list is called a node.

Node contain Three Fields:

1 Prev : Stores an address of the previous element

2 Next : Stores an address of the next element.

3 Data : Actual data

For use of Linked List, First we have to defined one util Linkedlist packeges.

- Syntax :

```
LinkedList<data> linked list =
            type        name

    new LinkedList<>();
```

- LinkedList can provides different method.

1) Add element
2) Access element
3) Change element
4) Remove element.

Ex. import java.util.LinkedList;

```
class java
{
  public static void main(String,
        args[])
  {
    LinkedList<String> Java =
          new LinkedList<>();

    Java.add("A");
    Java.add("B");
    Java.add("C");

    System.out.println(Java);

    String a = Java.get(0);
    System.out.println(a);

    String b = Java.set(1, "E");
    System.out.println(b);

    String c = Java.remove(2);
    System.out.println(c);

  }

}
```

* Explain Treeset with example.

=> Javacallection Framework treeset provides Function of Tree of Data structure.

For use of treeset, First we have to declare ~~util~~ util treeset Packeges.

Treeset can also provide Different method.

1) Add element
2) Access element
3) Remove element
4) Change element

-> Syntax:

```
Treeset<data> treeset =
          Type    name

    new Treeset<>();
```

Ex.
```
import ~~util~~ java.util.Treeset;

class java
{
```

```
public static void main (String
        args[])
{
  Treeset<String> Java =

        new Treeset<>();

  Java.add("A");
  Java.add("B");
  Java.add("C");

  System.out.println(Java);

  String a = Java.get(0);
  System.out.println(a);

  String b = Java.set(2, "E");
  System.out.println(b);

  String c = Java.remove(1);
  System.out.println(c);

 }

}
```