**Collection Framework-Assignment**

**Question 1: -What is the Collection framework in Java?**

**Ans: -**Collection framework is a combination of classes and interface ,which is used to store and manipulate the data in the form of objects.It provide classes such as ArrayList,Vector,and LinkedList ,Hashset,PriorityQueue etc and interfaces such as List ,Queue,Set etc.

**Question 2: -What is the difference between ArrayList and LinkedList?**

**Ans: -**This is the major defference .

| **ArrayList** | **LinkedList** |
| --- | --- |
| 1) ArrayList internally uses a dynamic array to store the elements. | LinkedList internally uses a doubly linked list to store the elements. |
| 2) 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. |
| 3) ArrayList is better for storing and accessing data. | LinkedList is better for manipulating data. |
| 4) The memory location for the elements of an ArrayList is contiguous. | The location for the elements of a linked list is not contagious. |
| 5) 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. |

**Question 3: -What is the difference between Iterator and ListIterator?**

**Ans: -**This is the major defference .

| **Iterator** | **ListIterator** |
| --- | --- |
| Can traverse elements present in Collection only in the forward direction. | Can traverse elements present in Collection both in forward and backward directions. |
| Helps to traverse Map, List and Set. | Can only traverse List and not the other two. |
| Indexes cannot be obtained by using Iterator. | It has methods like nextIndex() and previousIndex() to obtain indexes of elements at any time while traversing List. |
| Cannot modify or replace elements present in Collection | We can modify or replace elements with the help of set(E e) |
| Certain methods of Iterator are next(), remove() and hasNext(). | Certain methods of ListIterator are next(), previous(), hasNext(), hasPrevious(), add(E e). |

**Question 4: -What is the difference between Iterator and Enumeration?**

**Ans: -**This is the major defference .

## **Iterator: -**

* ->It is a universal cursor.
* ->It can be applied to all collection of classes.
* ->It contains the ‘remove’ method.
* ->It is not a legacy interface.Iterator can be used for the traversal of HashMap, LinkedList, ArrayList, HashSet, TreeMap, TreeSet .
* ->It can be used to traverse over HashMap, LinkedList, ArrayList, HashSet, TreeMap, and TreeSet .
* ->It can perform modifications to perform operations on the collection while traversing through it.

## **Enumeration: -**

* ->It is not a universal cursor.
* ->It is applied only to legacy classes.
* ->It doesn’t contain the ‘remove’ method.
* ->It is a legacy interface.which is used for traversing Vector, Hashtable.
* ->This interface acts like a read-only interface.
* ->Hence, no modifications can be performed on a collection while traversing over the elements in a Collection.
* ->It can be used to traverse over Vectors, and Hashtable.

**Question 5: -What is the difference between List and Set?**

**Ans: -**This is the major defference .

| **List** | **Set** |
| --- | --- |
| The List is an indexed sequence. | The Set is a non-indexed sequence. |
| List allows duplicate elements | Set doesn’t allow duplicate elements. |
| Elements by their position can be accessed. | Position access to elements is not allowed. |
| Multiple null elements can be stored. | Null elements can store only once. |
| List implementations By ArrayList, LinkedList, Vector, Stack | Set implementations By HashSet, LinkedHashSet. |

**Question 6: -What is the difference between HashSet and TreeSet?**

**Ans: -**This is the major defference.

| **Parameters** | **HashSet** | **TreeSet** |
| --- | --- | --- |
| **Ordering or Sorting** | It does not provide a guarantee to sort the data. | It provides a guarantee to sort the data. The sorting depends on the supplied Comparator. |
| **Null Objects** | In HashSet,only an element can be null. | It does not allow null elements. |
| **Comparison** | It uses **hashCode()** or **equals()** method for comparison. | It uses **compare()** or **compareTo()** method for comparison. |
| **Performance** | It is faster than TreeSet.HashSet has constant-time complexity O(1) for apping, removing, anp testing the existence of  an element. | It is slower in comparison to HashSet.TreeSet has a logarithmic-time complexity O(log n) for these operations due to the self-balancing property. |
| **Implementation** | Internally it uses **HashMap** to store its elements. | Internally it uses **TreeMap** to store its elements. |
| **Data Structure** | HashSet is backed up by a hash table. | TreeSet is backed up by a Red-black Tree. |
| **Values Stored** | It allows only **heterogeneous** value. | It allows only **homogeneous** values. |

**Question 7: -What is the difference between Array and ArrayList?**

**Ans: -**This is the major defference.

| **Base** | **Array** | **ArrayList** |
| --- | --- | --- |
| **Dimensionality** | It can be single-dimensional or multidimensional | It can only be single-dimensional |
| **Length** | **length** keyword can give the total size of the array. | **size()** method is used to compute the size of ArrayList. |
| **Size** | It is static and of fixed length | It is dynamic and can be increased or decreased in size when required. |
| **Static/ Dynamic** | Array is **static** in size. | ArrayList is **dynamic** in size. |
| **Data Type Storage** | Primitive data types can be stored directly unlikely objects | Primitive data types are not directly added to unlikely arrays, they are added indirectly with help of autoboxing and unboxing. |
| **Traversing Elements** | For and for each generally is used for iterating over arrays | Here iterator is used to traverse riverArrayList |
| **Data Type Storage** | Primitive data types can be stored directly unlikely objects | Primitive data types are not directly added to unlikely arrays, they are added indirectly with help of autoboxing and unboxing. |
| **Traversing Elements** | For and for each generally is used for iterating over arrays | Here iterator is used to traverse riverArrayList |