**What is Java Collection Framework?**

Java Collection Framework is a framework which provides some predefined classes and interfaces to store and manipulate the group of objects. Using Java collection framework, you can store the objects as a List or as a Set or as a Queue or as a Map and perform basic operations like adding, removing, updating, sorting, searching etc.. with ease.

**Why Java Collection Framework?**

Earlier, arrays are used to store the group of objects. But, arrays are of fixed size. You can’t change the size of an array once it is defined. It causes lots of difficulties while handling the group of objects. To overcome this drawback of arrays, Java Collection Framework is introduced from JDK 1.2.

**Java Collections Hierarchy :**

All the classes and interfaces related to Java collections are kept in java.util package. List, Set, Queue and Map are four top level interfaces of Java collection framework. All these interfaces (except Map) are the sub interfaces of java.util.Collection interface.

**List :**

List is a sequential collection of objects.

Elements are positioned using zero-based index.

Elements can be inserted or removed or retrieved from any arbitrary position using an integer index.

**Popular Implementations :**

ArrayList, Vector And LinkedList

**Internal Structure :**

**ArrayList :** Internally uses re-sizable array which grows or shrinks as we add or delete elements.

**Vector :** Same as ArrayList but it is synchronized.

**LinkedList :** Elements are stored as Nodes where each node consists of three parts – Reference To Previous Element, Value Of The Element and Reference To Next Element.

**Queue :**

Queue is a data structure where elements are added from one end called tail of the queue and elements are removed from another end called head of the queue.

Queue is typically FIFO (First-In-First-Out) type of data structure.

**Popular Implementations :**

PriorityQueue, ArrayDeque and LinkedList (implements List also)

**Internal Structure :**

**PriorityQueue :** It internally uses re-sizable array to store the elements and a Comparator to place the elements in some specific order.

**ArrayDeque :** It internally uses re-sizable array to store the elements.

**Set :**

Set is a linear collection of objects with no duplicates.

Set interface does not have it’s own methods. All it’s methods are inherited from Collection interface. It just applies restriction on methods so that duplicate elements are always avoided.

**Popular Implementations :**

HashSet, LinkedHashSet and TreeSet

**Internal Structure :**

**HashSet :** Internally uses HashMap to store the elements.

**LinkedHashSet :** Internally uses LinkedHashMap to store the elements.

**TreeSet :** Internally uses TreeMap to store the elements.

**Map :**

Map stores the data in the form of key-value pairs where each key is associated with a value.

Map interface is part of Java collection framework but it doesn’t inherit Collection interface.

**Popular Implementations :**

HashMap, LinkedHashMap And TreeMap

**Internal Structure :**

**HashMap** : It internally uses an array of buckets where each bucket internally uses linked list to hold the elements.

**LinkedHashMap :** Same as HashMap but it additionally uses a doubly linked list to maintain insertion order of elements.

**TreeMap :** It internally uses Red-Black tree.

