**COLLECTION**

-Collection is a framework that provides an architecture to store and manipulate the group of objects into a single unit.

-Strictly speaking, if you want to represent a group of objects into a single unit then we should go for a collection interface.

-**A Collection represents a single unit of objects, i.e., a group**.



-Iterable (I)-This is the root interface of collection framework.

-It provide the feature to iterate the data into forward direction only.

***-It has three methods***

1) Public Boolean hasNext();

-it return true is elements available

2) Public Object next();

-it return the next elements

3) Public void remove();

-it remove the last elements

-Except from this we have

***-2 sorting***

1) Comparable vs Comparator

***-3 Iterators***

1) Iterator

2) Enumeration

3) List Iterator

***-2 utility classes***

1) Collections

2) Arrays

1) Collection (I)-

-All required method are here

-Add,addAll,remove,removeAll,retainAll,clear,contains,

containsAll,isEmpty,size,toArray

2) List (I)-

-If duplicates are allowed and insertion order must be preserved and null insertion is allowed all Heterogeneous object allowed then use list.

-Methods(most done with index no)

Add,addAll,get,set,remove,indexOf,lastIndexOf,ListIterator

3) ArrayList(C):

-The underlying Ds is Growable Array and resizable array.

-ArrayList is NON-Synchronized class

-Except treeset and treemap all collection class has Heterogeneous object are allowed.

It has three constructor

1)AL al=new AL();

-create empty array list of size 10

-Nc =(CC\*3/2)+1

2) AL al=new AL(int initialCapacity);

3) AL al=new AL(Collection obj);

-we use collection to hold multiple object into single unit and transfer that object from one location to another location with the help of networks that’s why every collection **implement Serializable and Cloneable interface which are marker interface.**

**-But AL and Vector are also implements the RandomAccess interface which is also marker interface which does not contain any method but by implementing our object get special ability.**

**-Because of this we can very fastly find the elements from the arraylist and vector**

-If your Frequent operation is retrieval then array list is best choice

-but if your frequent operation is insertion or deletion in middle then array is worst choice.

* In ArrayList, manipulation is little bit slower than the LinkedList in Java because a lot of shifting needs to occur if any element is removed from the array list.

2) LinkedList (C):

-The underlying data structure of LL is doubly linked list.

-LL is linear data structure in which element are not stored in contagious memory location .every node has one value and address of next node.

-we use ll for developing stack and queue.

Advantage of LL

-fast for insertion and deletion in middle

-Dynamic in size

**Drawbacks:**   
**1)** Random access is not allowed. We have to access elements sequentially starting from the first node. So we cannot do binary search with linked lists efficiently with its default implementation.   
**2)** Extra memory space for a pointer is required with each element of the list.

-Methods for stack and queue

AddFirst,addLast,removeFirst,removeLast,getFirst,getLast

HOW TO CREATE AND LINKED THE NODE WITH EACH OTHER