ArrayList :-

ArrayList is an implementation class of List interface, present in java.util package.

→ It is used to represent group of Objects as a single entity. Each Object is called as element.

→ ArrayList class implements Serializable, Cloneable & RandomAccess marker interfaces.

→ The data structure of ArrayList is Growable or Resizable Array.

→ It allows duplicates, multiple null elements

→ It preserves insertion Order.

→ The initial capacity of ArrayList is 10. and size grows by formula

$$newCapacity = \left[currentCapacity \times \frac{3}{2} \right] + 1$$

→ ArrayList class has 3 overloaded constructors in it.

1. ArrayList()
2. ArrayList(int initialCapacity)
3. ArrayList(Collection c)

→ ArrayList is Suitable for frequent retrieval & Search Operation as it is index based & implements RandomAccess interface.

→ ArrayList is not Suitable for frequent insertion & removal of elements in between ArrayList as there is lot of Shift Operation involved.

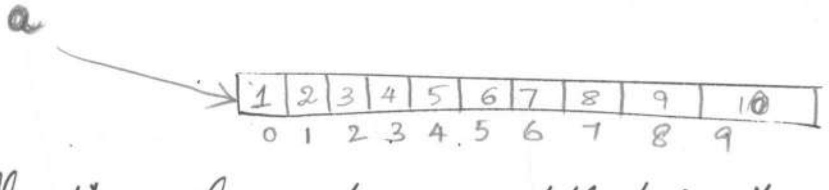
Working :-

ArrayList a = new ArrayList();

→ Creates an empty ArrayList Object with initial capacity of 10. Once the ArrayList Object is filled completely, the Size grows based on formula.

$nc = \left[cc * \frac{3}{2}\right] + 1$;

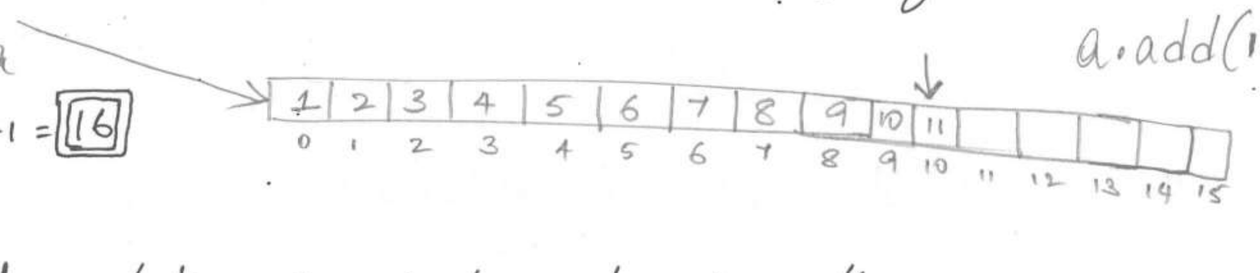
a.add(1);
a.add(2);
(3);
⋮
(10);



Once, all the elements are filled in the ArrayList, if we try to add a new element then Size grows.

a.add(11)

$\left[10 * \frac{3}{2}\right] + 1 = \boxed{16}$



NOTE :- 1. ArrayList is index based collection.
2. It Allows both homogeneous & heterogeneous type of elements.

<u>Vector</u> [java.util]

→ Vector is an implementation class of List interface
→ Vector is used to represent a group of Objects as a Single entity, each Object in it is called as element
→ Vector implements Serializable, Cloneable & RandomAccess Marker interfaces.
→ Vector allows multiple null values & duplicates.
→ It is index based and preserves insertion order.
→ The data Structure is Growable/Resizable Array.
→ The initial Capacity of Vector is <u>10</u>.
→ The incremental capacity is twice (2x) the the Current capacity.

There are <u>4</u> Overloaded Constructors in Vector class

1. Vector()
2. Vector(int initialCapacity)
3. Vector(int initialCapacity, int incrementalCapacity)
4. Vector(Collection c)

→ It is suitable for frequent search & retrieval Operation as it is index based & implementing

java.util.RandomAccess Marker interface.

→ It is not suitable for frequent insertion/removal of element in between as there is a lot of shift operations involved.

Additional imp. points.

→ It is legacy class [old class, introduced in 1.0 version of JDK].

→ It is Synchronized, thread safe [All the methods in vector is declared with Keyword synchronized]

→ Vector class has some Sub class Specific methods in it.

like : ex : → public Object firstElement()
→ public Object lastElement()
→ public boolean removeElement(Object o)
→ insertElementAt
→ setElementAt

Difference between HashMap aud Hashtable

HashMap	Hashtable
→ HashMap was introduced in 1.2	Hashtable was introduced in 1.0
→ It is not legacy class	It is a legacy class
→ It is non-Synchronized	It is Synchronized
→ It is not thread safe	It is thread safe
→ HashMap initialCapacity is 16.	Hashtable initial capacity is 11.
→ HashMap extends AbstractMap class	Hashtable extends Dictionary class
→ HashMap will allow one null key & multiple null values.	Hashtable will not allow null key nor null value.
→ HashMap performance is good compared to Hashtable	Hashtable performance is less compared to HashMap

Difference between List and Set.

List	Set
* List will allow duplicate elements.	Set will not allow duplicate elements.
* List will allow multiple null values	Set will allow max. of one null value.
* List is index based Collection.	Set is hashing based Collection.
* List will preserve insertion order	Set will not preserve insertion Order.
* List has some interface Specific Methods declared in it. for ex: listIterator, get and many more.	Set has no interface Specific methods. All the methods are inherited from Collection interface & is Modified according to set interface.

HashMap

HashMap is an implementation class of Map interface. It was introduced in JDK 1.2 and is present in java.util package

→ It is used to represent data in the form of key & value pair. Each key-value pair in Map is called as Entry

→ HashMap allows one null key & multiple null values.

→ HashMap will not preserve any insertion/sorting order.

→ HashMap cannot hold duplicate keys, but can hold duplicate values.

→ It is hash table based. Map implementation class.

X → It is non-Synchronized

→ The initial capacity of HashMap is 16 & load factor is 0.75.

It has mainly 4 constructors in it.

1. HashMap()
2. HashMap(int initialCapacity)
3. HashMap (int initialCapacity, float loadFactor)
4. HashMap (Map<k,v> m);

LinkedList [java.util]

→ It is an implementation class of List interface

→ It is used to represent a group of Objects as a Single entity. each Object is Considered as an element.

→ It implements Serializable & Cloneable Marker interfaces.

→ The underlying data structure is Doubly Linked List.

→ It allows duplicates, multiple null values.

→ It is index based & preserves insertion Order.

→ The Initial Capacity of Linked List is 0.

→ It has two Overloaded Constructors in it.

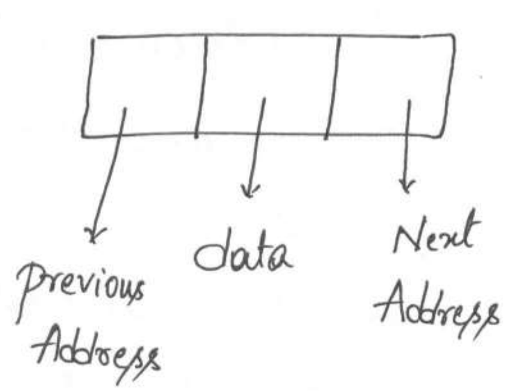
1. LinkedList()
2. LinkedList(Collection c)

Additional important points

→ LinkedList is suitable for frequent insertion & removal Operation as the data Structure is doubly linked list there is no Shifting Operations like Arraylist.

→ LinkedList is not suitable for frequent Search or retrieval Operations, as LinkedList is not implementing RandomAccess Marker interface.

double ended node



→ In LinkedList the first node previous address and Last node next address will be null.

→ It allows both homogeneous & heterogeneous Objects.