***\*\** Collection *\*\****

1. **Collection :**
2. The **Collection** is a framework that provides architecture to store and manipulate the group of objects.
3. Collections can achieve all the operations that you perform on a data such as Searching, Sorting, Insertion, Manipulation, And Deletion.
4. Java Collection framework provides many interfaces (Set, List, Queue, Deque) and classes ([ArrayList](https://www.javatpoint.com/java-arraylist), Vector, [LinkedList](https://www.javatpoint.com/java-linkedlist), [PriorityQueue](https://www.javatpoint.com/java-priorityqueue), HashSet, LinkedHashSet, TreeSet).

#### What is Collection in Java ?

Ans : Any group of individual objects which are represented as a single unit is known as a collection.

#### What is a framework in Java

Ans : A framework is a set of classes and interfaces which provide a ready-made architecture.

#### What is Collection framework

Ans : The Collection framework represents a unified architecture for storing and manipulating a group of objects. It has:

1. Interfaces and its implementations, i.e., classes
2. Algorithm

**The java.util package contains all the**[**classes**](https://www.javatpoint.com/object-and-class-in-java)**and**[**interfaces**](https://www.javatpoint.com/interface-in-java)**for the Collection framework.**

The Collection interface (java.util.Collection) and Map interface (java.util.Map) are the two main “root” interfaces of Java collection classes.



### Iterator interface

Iterator interface provides the facility of iterating the elements in a forward direction only.

#### Methods of Iterator interface

There are only three methods in the Iterator interface. They are:

|  |  |  |
| --- | --- | --- |
| No. | Method | Description |
| 1 | public boolean hasNext() | It returns true if the iterator has more elements otherwise it returns false. |
| 2 | public Object next() | It returns the element and moves the cursor pointer to the next element. |
| 3 | public void remove() | It removes the last elements returned by the iterator. It is less used. |

## Iterable Interface

1. The Iterable interface is the root interface for all the collection classes.
2. The Collection interface extends the Iterable interface and therefore all the subclasses of Collection interface also implement the Iterable interface.
3. It contains only one abstract method. i.e.,

Iterator<T> iterator()

## List Interface :

1. List Interface is called interface of collection interface.
2. It extends Collection interface.
3. It can have duplicate values.
4. In list insertion order is prepared.
5. In list interface we can store heterogeneous type of data.
6. In list we can search data on base of index.

**List interface is implemented by the classes**

1. ArrayList
2. LinkedList
3. Vector
4. Stack.

To instantiate the List interface, we must use :

List <data-type> list1= new ArrayList();

List <data-type> list2 = new LinkedList();

List <data-type> list3 = new Vector();

List <data-type> list4 = new Stack();

## ArrayList

* 1. It implements list interface and extends Abstract list Class.
  2. It is grow-able in nature.
  3. It allows to Store duplicate data.
  4. We can store heterogeneous type of data.
  5. It is not synchronized .
  6. It is not thread safe.
  7. Null value can also be inserted.
  8. In Array List insertion order is preserved.
  9. In array insertion and deletion operation are complex.
  10. We prefer to use Array list when we need to perform search

operation.

**Q1. When ArrayList is full this formula is used to increase capacity of**

**ArrayList**

**New capacity = ( current capacity \* 2 +1 ) / 2**

**Array List Have three Constructors**

* 1. Zero Parameter.
  2. One int Parameter.
  3. One Collection Parameter.

What is benefit of parameterized Constructor.

There are many methods declared in the Collection interface. They are as follows:

|  |  |  |
| --- | --- | --- |
| No. | Method | Description |
| 1 | public boolean add(E e) | It is used to insert an element in this collection. |
| 2 | public boolean addAll(Collection<? extends E> c) | It is used to insert the specified collection elements in the invoking collection. |
| 3 | public boolean remove(Object element) | It is used to delete an element from the collection. |
| 4 | public boolean removeAll(Collection<?> c) | It is used to delete all the elements of the specified collection from the invoking collection. |
| 5 | default boolean removeIf(Predicate<? super E> filter) | It is used to delete all the elements of the collection that satisfy the specified predicate. |
| 6 | public boolean retainAll(Collection<?> c) | It is used to delete all the elements of invoking collection except the specified collection. |
| 7 | public int size() | It returns the total number of elements in the collection. |
| 8 | public void clear() | It removes the total number of elements from the collection. |
| 9 | public boolean contains(Object element) | It is used to search an element. |
| 10 | public boolean containsAll(Collection<?> c) | It is used to search the specified collection in the collection. |
| 11 | public Iterator iterator() | It returns an iterator. |
| 12 | public Object[] toArray() | It converts collection into array. |
| 13 | public <T> T[] toArray(T[] a) | It converts collection into array. Here, the runtime type of the returned array is that of the specified array. |
| 14 | public boolean isEmpty() | It checks if collection is empty. |
| 15 | default Stream<E> parallelStream() | It returns a possibly parallel Stream with the collection as its source. |
| 16 | default Stream<E> stream() | It returns a sequential Stream with the collection as its source. |
| 17 | default Spliterator<E> spliterator() | It generates a Spliterator over the specified elements in the collection. |
| 18 | public boolean equals(Object element) | It matches two collections. |
| 19 | public int hashCode() | It returns the hash code number of the collection. |