ArrayList: Java arrayList class uses a dynamic array for storing the elements. It is like an array, but there is no size limit. We can add or remove elements anytime. So, it is much more flexible than the traditional array. It is found in java.util. package.

◇ It is like the vector in C++.

◇ The arrayList in java can have the duplicate element also. It implements the list interface so we can use all the methods of list interface here. The arrayList maintains the insertion order internally.

IMPORTANT POINTS:

✧ ArrayList class can contain duplicate elements.

✧ ArrayList class maintain insertion order.

✧ ArrayList class is non-synchronized.

✧ ArrayList allow random access because array works at the index basis.

✧ 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.

ArrayList class Declaration:

Public class ArrayList<dataType> extends AbstractList<dataType> implements List <dataType>, RandomAccess, Cloneable, Serializable

OR

ArrayList<dataType> list = new ArrayList<dataType>();

Difference between Array and ArrayList:

Array:

- An array has a fixed length or size.
- An array can be created using both primitive datatypes as well as non-primitive datatypes.

ArrayList:

- An arrayList can contain as many elements as you want even through an initial size is specified.
- An arrayList cannot be created using primitive datatypes. It can only be created using objects or wrapper classes.

Program:

1

```java
import java.util.ArrayList;
public class ArrayListExample {
    public static void main(String[] args) {
        ArrayList<Integer> list = new ArrayList<>(5);
        list.add(21);
        list.add(25);
        list.add(28);
        list.add(19);
        list.add(18);
        System.out.println(list);
    }
}
```

2

```java
package com.dhruv;

import java.util.ArrayList;
public class ArrayListExample {
    public static void main(String[] args) {

        ArrayList<Integer> list = new ArrayList<>(5);
        list.add(21);
        list.add(25);
        list.add(28);
        list.add(19);
        list.add(18);
        System.out.println(list);
        list.add(50); // For adding elements
        list.set(1,32); // For changing elements
        list.remove(2); // For removing elements
        System.out.println(list);
    }
}
```

Sort ArrayList: The sort() method sorts the element in an arrayList according to the specified order (ascending or descending order).

Syntax:

```
arrayList.sort(Comparator.naturalOrder())
```

Here,

arrayList- is an object of the ArrayList class.

Comparator- specifies the sort order of the arrayList.

naturalOrder() - ascending order

reverseOrder() - descending order

sort() Parameters:

The sort() method takes a single parameter.

sort() Return Values:

The sort() method does not return any value. Rather it only changes the order of elements in an arrayList.

Program:

```java
//Sort Array List
import java.util.ArrayList;
import java.util.Comparator;
public class ArraySort {
    public static void main(String[] args) {
        ArrayList<Integer> marks = new ArrayList<>();
        marks.add(98);
        marks.add(90);
        marks.add(93);
        marks.add(89);
        marks.add(40);
        System.out.println("Before Sorted = " + marks);

        marks.sort(Comparator.naturalOrder());
        System.out.println("After Natutal Order Sorted = " + marks);
        marks.sort(Comparator.reverseOrder());
        System.out.println("After Reverse Order Sorted = " + marks);
        System.out.println();
    }
}

```