JAVA.UTIL.ARRAYS.SORT METHOD

Description

The **java.util.Arrays.sort***T*[]*a, Comparator* < *?superT* > *c* method sorts the specified array of objects according to the order induced by the specified comparator. All elements in the array must be mutually comparable by the specified comparator *thatis, c. compare*(*e*1, *e*2 must not throw a ClassCastException for any elements e1 and e2 in the array).

This sort is guaranteed to be stable: equal elements will not be reordered as a result of the sort.

The sorting algorithm is a modified mergesort *inwhichthemergeisomittedifthehighestelementinthelowsublistislessthanthelowestelementinthehighsublist*. This algorithm offers guaranteed n*log*n* performance.

Declaration

Following is the declaration for **java.util.Arrays.sort** method

```
public static <T> void sort(T[] a, Comparator<? super T> c)
```

Parameters

- **a** -- This is the array to be sorted.
- **c** -- The comparator to determine the order of the array. A null value indicates that the elements' natural ordering should be used.

Return Value

This method does not return any value.

Exception

- **ClassCastException** -- If the array contains elements that are not mutually comparable using the specified comparator.

Example

The following example shows the usage of java.util.Arrays.sort method.

```
package com.tutorialspoint;

import java.util.Arrays;
import java.util.Collections;
import java.util.Comparator;

public class ArrayDemo {

   public static void main(String[] args) {

   // initializing unsorted short array
   Short sArr[] = new Short[]{3, 13, 1, 9, 21};

   // let us print all the elements available in list
   for (short number : sArr) {
   System.out.println("Number = " + number);
   }

   // create a comparator
   Comparator<Short> comp = Collections.reverseOrder();

   // sorting array with reverse order using comparator
```

```
   Arrays.sort(sArr, comp);

   // let us print all the elements available in list
   System.out.println("short array with some sorted values(1 to 4) is:");
   for (short number : sArr) {
   System.out.println("Number = " + number);
   }
   }
}
```

Let us compile and run the above program, this will produce the following result:

```
Number = 3
Number = 13
Number = 1
Number = 9
Number = 21
short array with some sorted values(1 to 4) is:
Number = 21
Number = 13
Number = 9
Number = 3
Number = 1
```