Sorting using Built-in methods in Java

Arrays.sort()

The Arrays.sort() is a built-in method in Java of Arrays class which is used to sort an array in ascending or descending or any other order specified by the user.

Syntax:

```
public static void sort(int[] arr, int from_Index, int to_Index)

arr - The array to be sorted.
from_Index - The index of the first element, inclusive, to be sorted.
to_Index - The index of the last element, exclusive, to be sorted.
```

Below are different ways of using the sort() method of Arrays class in Java to sort arrays differently.

- **A Java program to sort an array of integers in ascending order**.

Java

```java
// A sample Java program to sort an array of integers
// using Arrays.sort(). It by default sorts in
// ascending order
import java.util.Arrays;

public class SortExample
{
    public static void main(String[] args)
    {
        // Our arr contains 8 elements
        int[] arr = {13, 7, 6, 45, 21, 9, 101, 102};
```

Track Progress
6 of **106** Complete. (6%)

Dash

All

```
}
```

Articles

Output:

```
Modified arr[] : [6, 7, 9, 13, 21, 45, 101, 102]
```

Videos

- **We can also use sort() to sort a subarray of arr[].**

Problems

Java

Quiz

```
// A sample Java program to sort a subarray
// using Arrays.sort().
import java.util.Arrays;

public class SortExample
{
    public static void main(String[] args)
    {
        // Our arr contains 8 elements
        int[] arr = {13, 7, 6, 45, 21, 9, 2, 100};

        // Sort subarray from index 1 to 4, i.e.,
        // only sort subarray {7, 6, 45, 21} and
        // keep other elements as it is.
        Arrays.sort(arr, 1, 5);

        System.out.printf("Modified arr[] : %s",
                              Arrays.toString(arr));
    }
}
```

Output:

Menu

```
Modified arr[] : [13, 6, 7, 21, 45, 9, 2, 100]
```

Track Progress

6 of **106** Complete. (6%)

```
import java.util.Arrays;
import java.util.Collections;

public class SortExample
{
    public static void main(String[] args)
    {
        // Note that we have Integer here instead of
        // int[] as Collections.reverseOrder doesn't
        // work for primitive types.
        Integer[] arr = {13, 7, 6, 45, 21, 9, 2, 100};

        // Sorts arr[] in descending order
        Arrays.sort(arr, Collections.reverseOrder());

        System.out.printf("Modified arr[] : %s",
                          Arrays.toString(arr));
    }
}
```

Output:

```
Modified arr[] : [100, 45, 21, 13, 9, 7, 6, 2]
```

- **We can also sort strings in alphabetical order**

Java

```
// A sample Java program to sort an array of strings
// in ascending and descending orders using Arrays.sort().
import java.util.Arrays;
import java.util.Collections;

public class SortExample
{
    public static void main(String[] args)
```

Dash

```
                "quiz.geeksforgeeks.org",
```

All

```
        // Sorts arr[] in ascending order
        Arrays.sort(arr);
        System.out.printf("Modified arr[] : \n%s\n\n",
                          Arrays.toString(arr));

        // Sorts arr[] in descending order
        Arrays.sort(arr, Collections.reverseOrder());

        System.out.printf("Modified arr[] : \n%s\n\n",
                          Arrays.toString(arr));
    }
}
```

Articles

Videos

Problems

Quiz

Output:

```
Modified arr[] :
[code 1="practice.geeksforgeeks.org," 2="quiz.geeksforgeeks.org" langu
age=".geeksforgeeks.org,"][/code]

Modified arr[] :
[quiz.geeksforgeeks.org, practice.geeksforgeeks.org, code.geeksforgeek
s.org]
```

- **We can also sort an array according to user defined criteria**: We use Comparator interface for this purpose. Below is an example.

Java

```
// Java program to demonstrate working of Comparator
// interface
import java.util.*;

// A class to represent a student.
class Point
{
    int x, y;
```

Menu

Track Progress

6 of **106** Complete. (6%)

```
class MySort implements Comparator<Point>

    public int compare(Point a, Point b)
    {
        return a.x - b.x;
    }
}

// Driver class
class Main
{
    public static void main (String[] args)
    {
        Point [] arr = {new Point(10, 20), new Point(3, 12), new Poi
        Arrays.sort(arr, new MySort());
        for (int i=0; i<arr.length; i++)
            System.out.println(arr[i].x + " " + arr[i].y);
    }
}
```

Output:

```
3 12
5 7
10 20
```

Collections.sort()

The **Collections.sort()** method is present in Collections class. It is used to sort the elements present in the specified list of Collection in ascending order.

It works similar to the Arrays.sort() method but it is better as it can sort the elements of Array as well as any collection interfaces like a linked list, queue and many more.

```
public static void sort(List myList)
```

```
This method doesn't return anything
```

Example:

```
Let us suppose that our list contains
{"Geeks For Geeks", "Friends", "Dear", "Is", "Superb"}

After using Collection.sort(), we obtain a sorted list as
{"Dear", "Friends", "Geeks For Geeks", "Is", "Superb"}
```

Below are some ways of using the Collections.sort() method in Java:

- **Sorting an ArrayList in ascending order**

JAVA

```
// Java program to demonstrate working of Collections.sort()
import java.util.*;

public class Collectionsorting
{
    public static void main(String[] args)
    {
        // Create a list of strings
        ArrayList<String> al = new ArrayList<String>();
        al.add("Geeks For Geeks");
        al.add("Friends");
        al.add("Dear");
        al.add("Is");
        al.add("Superb");
        Collections.sort(al);
```

```
                                 " Collection.sort() :\n" + al);
    }
}
```

Output:

```
List after the use of Collection.sort() :
[Dear, Friends, Geeks For Geeks, Is, Superb]
```

- **Sorting an ArrayList in descending order**

JAVA

```
// Java program to demonstrate working of Collections.sort()
// to descending order.
import java.util.*;

public class Collectionsorting
{
    public static void main(String[] args)
    {
        // Create a list of strings
        ArrayList<String> al = new ArrayList<String>();
        al.add("Geeks For Geeks");
        al.add("Friends");
        al.add("Dear");
        al.add("Is");
        al.add("Superb");

        /* Collections.sort method is sorting the
        elements of ArrayList in ascending order. */
        Collections.sort(al, Collections.reverseOrder());

        // Let us print the sorted list
        System.out.println("List after the use of" +
                                 " Collection.sort() :\n" + al);
```

Output:

- **Sorting an ArrayList according to user defined criteria:** We can use Comparator Interface for this purpose.

Java

```
// Java program to demonstrate working of Comparator
// interface and Collections.sort() to sort according
// to user defined criteria.
import java.util.*;
import java.lang.*;
import java.io.*;

// A class to represent a student.
class Student
{
    int rollno;
    String name, address;

    // Constructor
    public Student(int rollno, String name,
                               String address)
    {
        this.rollno = rollno;
        this.name = name;
        this.address = address;
    }

    // Used to print student details in main()
    public String toString()
    {
        return this.rollno + " " + this.name +
                           " " + this.address;
    }
}
```

Track Progress

6 of **106** Complete. (6%)

```
    // Used for sorting in ascending order of
```

```
        return a.rollno - b.rollno;
    }
}

// Driver class
class Main
{
    public static void main (String[] args)
    {
        ArrayList<Student> ar = new ArrayList<Student>();
        ar.add(new Student(111, "bbbb", "london"));
        ar.add(new Student(131, "aaaa", "nyc"));
        ar.add(new Student(121, "cccc", "jaipur"));

        System.out.println("Unsorted");
        for (int i=0; i<ar.size(); i++)
            System.out.println(ar.get(i));

        Collections.sort(ar, new Sortbyroll());

        System.out.println("\nSorted by rollno");
        for (int i=0; i<ar.size(); i++)
            System.out.println(ar.get(i));
    }
}
```

Output :

```
Unsorted
111 bbbb london
131 aaaa nyc
121 cccc jaipur

Sorted by rollno
111 bbbb london
121 cccc jaipur
131 aaaa nyc
```

Dash

All

Marked as Read

Articles

Report An Issue

If you are facing any issue on this page. Please let us know.

Videos

Problems

Quiz

Menu

Track Progress

6 of **106** Complete. (6%)