Sort Algorithms

A common mistake for a beginner to Java is to try and write some more or less complex method for sorting an array when in fact it can be done with just one line of code.

Textbooks cover both the slow, O(n^2), sorts (eg, selection, insertion, and bubble sorts) and fast, O(n log n), sorts (quick sort, heap sort, etc). These are interesting problems, give students a lot of practice with arrays, bring up important tradeoffs, and are generally quite educational.

However, the Java library already has sort methods that are more efficient, more general, and more correct than anything you are likely to write yourself. And they are already written. Professional programmers use one of the java.util.Arrays.sort() methods; they do not write their own, except perhaps in unusual cases.

The Arrays class has a static method called sort which exists in many overloaded versions so you can pass in an array of any primitive type, an array of Strings or other Objects.

Since it is the reference of the array that is passed to the sort method nothing needs to be returned from it. **The array is altered** through the reference.

Other data structures. There are sort methods in the java.util.Collections class which can be used to sort other kinds of data structures. You will use them to sort in reverse order, for example.

Use this import:

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

Define and initialize your array of integers:

```
     int[] arrayToSort = new int[] {48, 5, 89, 80, 81, 23, 45, 16, 2};
```

Sort your array of integers:

```
     Arrays.sort ( arrayToSort );
```

Verify (print) your sorted array of integers:

```
     for (int i = 0; i < arrayToSort.length; i++)
           System.out.println ( arrayToSort[i] );
```

Some results:

```
// Integer sort
int[] intArray = new int[] {4, 1, 3, -23};
Arrays.sort(intArray);
// intArray now contains [-23, 1, 3, 4]

// Reverse-order Integer sort
int[] intArray = new int[] {4, 1, 3, -23};
Arrays.sort(intArray, Collections.reverseOrder());
// intArray now contains [4, 3, 1 -23]
```