Intro to Algorithms

Homework

SortComparatorSubclass

The purpose of this assignment is for you to become familiar with the *Comparator* (in contrast to *Comparable*) JDK interface, in the context of Sorting.

There is another purpose of this assignment which we'll discuss later.

The Assignment

Item

Implement an Item class with (at least) this API

```
public Item(final String description, final double price)
public String getDescription()
public double getPrice()
```

Color

Cut-and-paste this code into `Color.java` in the same package:

```
public enum Color {
  RED, ORANGE, YELLOW, GREEN, BLUE, INDIGO, VIOLET;
}
```

ColoredItem

Implement an ColoredItem class which **extends** Item. It must have (at least) this API:

```
public ColoredItem(final String description, final double price, final Color color)
public Color getColor()
```

SortDriver

Implement a SortDriver class with this API:

Sort API

The result of invoking the `sort()` method, passing it `Item[]`, is that the Array is sorted in lexicographic order on the value of `getDescription()`, followed by ascending order on `getPrice()`.

```
  static public void sort(Item[] a)
```

Sort By Price API

The result of invoking the `sortByPrice()` method, passing an `Item[]` is that the Array is sorted in ascending order on `getPrice()`.

```
static public void sortByPrice(Item[] a)
```

Sort API and ColoredItem

Your implementation `sort()` API must allow the client to invoke it, passing an Array of ColoredItem instances. The semantics of the sort are: the Array is sorted in lexicographic order on the value of `getDescription()`, followed by ascending order on `getPrice()`, followed by the value returned by the `color` field's *compareTo* implementation (by the JDK).

Object: The Cosmic Superclass

(I attribute the phrase to Horstmann.) All sorts of mistakes can be made when overriding the following cosmic methods:

- equals
- hashCode

Even experienced programmers can get this wrong, especially when dealing with sub-classes. In addition to testing the "sort" API, I plan to test your implementation of these methods.

Making Your Life Easier

You may use the `Arrays.sort()` utility methods for this assignment: i.e., this assignment is **not** about implementing your own sort routines or Sedgewick's.

Packaging

Assume that you've installed your YU Git repo in a directory named `GIT`.

- Your homework assignments for this course must be rooted in `GIT/IntroToAlgorithms/homework`. I'll refer to this directory as `ROOT`.
- Your code will reside in a package named `edu.yc.oats.algs`.
- Your code for this assignment will be rooted in: `ROOT/SortComparatorSubclass`. I'll refer to this directory as `DIR`.
- Your class must be named **SortDriver.java**, and it must be reside `DIR/src/main/java/edu/yc/oats/algs`.
- Your assignment may not use any external libraries: just the SDK and your code in this package.

Grading

- If your code cannot be compiled & run (either because it doesn't follow the packaging conventions above, or because of a compilation bug) -- **automatic 0** for the assignment
- If your code runs, but doesn't pass my tests, you'll get a *maximum* of 8. The actual grade will depend on how close your code was to passing the tests.
- If your code runs, passes the tests, but is "really ugly", you get a 9.
- Maximum grade is 10.