Lab 12

Sorting

Objectives of this Lab:

1. To actually write code for two sorting algorithms: bubble and selection sort, and
2. to see the difference in performance between these and quicksort provided by Java.

Description:

This lab requires you to complete the Sorting.java class, which implements four different sorting algorithms: bubble, selection, insertion, and quicksort. The TA will help you visualize the sorting algorithms, using the excellent website here. For this lab, the insertion sorting code is provided as an example, and quicksort is provided by Java in the form of the Arrays.sort() method. Arrays.sort() uses quicksort for primitives and mergesort for objects. A second class named FillArray.java provides a couple of useful methods for filling an array with random numbers and checking whether an array is sorted. Here are the two methods you must implement:

```
// Bubble sorting
public static void bubbleSort(double[] values) {
   // Put the code for bubble sort here
}

// Selection sorting
public static void selectionSort(double[] values) {
   // Put the code for selection sort here
}
```

Instructions:

1. Make an Eclipse project named R21.
2. Download Sorting.java and FillArray.java.

3. Look through the provided code, which your TA will help explain.
4. First, complete the bubbleSort method in Sorting.java.
5. Second, complete the selectionSort method in Sorting.java.
6. **HINT 1**: These algorithms are documented on the Internet.
7. **HINT 2**: Start with 10 elements and uncomment debug code in FillArray.java.
8. The provided code will verify your sorting code and measure performance.
9. When your sorting code is verified, measure performance with 50000 elements.

Testing:

Here is the approximate output of the program after all methods and test code are complete, for sorting 50000 elements. Don't worry about your output exactly matching, since your performance may vary somewhat:

COMPLETELY RANDOM:
Bubble sort of 50000 elements = 3389 milliseconds
Selection sort of 50000 elements = 684 milliseconds
Insertion sort of 50000 elements = 1203 milliseconds
Quick sort of 50000 elements = 16 milliseconds

HALF SORTED:
Bubble sort of 50000 elements = 1676 milliseconds
Selection sort of 50000 elements = 676 milliseconds
Insertion sort of 50000 elements = 597 milliseconds
Quick sort of 50000 elements = 2 milliseconds

REVERSE ORDER:
Bubble sort of 50000 elements = 855 milliseconds
Selection sort of 50000 elements = 920 milliseconds
Insertion sort of 50000 elements = 578 milliseconds
Quick sort of 50000 elements = 1 milliseconds

ALREADY SORTED:
Bubble sort of 50000 elements = 932 milliseconds
Selection sort of 50000 elements = 675 milliseconds
Insertion sort of 50000 elements = 0 milliseconds
Quick sort of 50000 elements = 1 milliseconds

Turn in and Get Credit:

Turn in your code to GitHub here: https://classroom.github.com/a/U3NWymVk