Sorting Part 1: Iterative Sorts

You have the great misfortune of working for a boss that just can’t make up his mind… about anything. He often has to go through various collections of data and needs those collections sorted, but he’s not sure which sort algorithm would be best. Since he can’t decide, he’s tasked you with building a sorting library that will allow him to choose which algorithm to use and when.

# Requirements

1. You must write your library in C# or Java. You are expected to follow the language conventions properly.
2. C# - Namespace = SortingLibrary  
   Java – Package name = sorting.library
3. Name your class Sorter.
4. Sorter will have the following static methods:
   1. BubbleSort  
      SelectionSort  
      InsertionSort
   2. Each method is required to implement the sorting algorithm of the same name.
   3. Use proper casing for your chosen language in naming your methods.
   4. Each method will take in a generic array as a parameter, but it must be constrained to datatypes that implement the IComparable<T> interface (C#) or Comparable<T> interface (Java).
   5. Each method returns void as each sort is expected to be “in place”.
5. Your deliverable is the DLL or JAR itself, NOT the code base.
6. Name the library LastnameF\_SortingLib, where you replace “Lastname” with your last name and “F” with your first initial. That’s last name, first initial, underscore, SortingLib. This will be the naming convention for all submissions in Algo I & II.
   1. In Java, this can be done in the file explorer as needed.
   2. In C#, you MUST set the name for the assembly in the project properties. Renaming the DLL manually in file explorer after the fact BREAKS THE DLL!!!

# Rubric

**Automatic Zero:** Your deliverable is not a library, any exception is thrown by your code, or you fail to correctly implement any of the expected sorting algorithms.

(10 points) BubbleSort works

(10 points) SelectionSort works

(10 points) InsertionSort works