Background:

This homework covers the lectures of Sept. 25 and 27: writing static methods and using arrays.

Bubble sort is a sorting algorithm that starts by checking to see if elements 0 and 1 are in the right order, and if not, flipping them. Then, it checks whether elements 1 and 2 are in order, flipping if necessary, and so on. After it checks the second to last and last elements, it starts over with elements 0 and 1. It repeats this until a full pass through the elements results in no flips, indicating the elements are in order. For example:

Before first pass through:  { 0, 4, 2, 7, 1 }  
After first pass:  { 0, 2, 4, 1, 7 }  
After second pass:  { 0, 2, 1, 4, 7 }  
After third pass:  { 0, 1, 2, 4, 7 }  
After fourth pass:  no changes, so we’re done.

Problems:

1. Create a class named IntSorter with a method:
   ```java
   public static int bubbleSort(int[] a) {
       ...
   }
   ```
   It should modify a in place so that when it returns, a is sorted in ascending order. It should return the number of elements that were compared.

2. Write a main function that tries out the bubbleSort method for several arrays. If the array is already sorted and has length N, it should take N − 1 comparisons to determine that. How many comparisons does it take to sort an array of length 5 that is reverse-sorted to begin with? How about an array of length 1000? (Please don't hard-code a list of length 1000 in your code.)

You should submit this assignment as two files, submit/hw4/IntSorter.java and submit/hw4/hw4.txt.