Sorting-Bubble Sort

Lists of information are useful, provided that they are organized in some way. Imagine how difficult it would be to use a phone book if the names in it were not listed in alphabetical order!

Once data has been entered into an array, the computer can sort it. There are many different techniques, with names like the Shell sort, the select sort, and the comb sort. One of the easiest to use is the bubble sort. Unfortunately, it is also one of the slowest sorting methods.

Assume that we have an array of integer numbers to be put into ascending order. An algorithm (plan) for the bubble sort is this:

- Go through the array. Start with the first number in the list and compare each number with the number that was after it.
- If the two numbers are out of order (current number larger than the next number), swap them.
- Continue until we have compared the last number with the second last number in the array.
- Go through the list again and again until it is in order.

Let's look at an example. Here is a list of eight numbers to be put into increasing order :

```
5   7   11   6   17   19   16   2
```

Start with the 5. Nothing happens when 5 is compared with 7 and again when 7 is compared with 11. The 11 and the 6, however, are out of order and must be swapped. The list then looks like this:

```
5   7   6   11   17   19   16   2
```

Nothing happens when 11 is compared with 17.

Nothing happens when 17 is compared with 19, but the 19 and 16 must be swapped. Here is the list again:

```
5   7   6   11   17   16   19   2
```

As we continue through the list, the number 19 rises up to the end of the list, like a bubble in water. After the last number has been compared with the second last, the list is:

```
5   7   6   11   17   16   2   19
```

This completes our first time through the array. As you can see, the numbers are not in order, but the biggest number is now at the end of the list. Repeat the process listed above! If we go back to the beginning and go through the array again, the list is:

```
5   7   6   11   17   16   2   19      (after first pass)
```

As we keep repeating the process, the list looks like :

```
5   6   7   11  16   2    17   19 (after second pass)
5   6   7   11  2    16   17   19 (after third pass)
5   6   7   2   11   16   17   19 (after fourth pass)
5   6   2   7   11   16   17   19 (after fifth pass)
5   2   6   7   11   16   17   19 (after sixth pass)
2   5   6   7   11   16   17   19 (after seventh pass)
```

Sorting-Bubble Sort

We measure the effectiveness of bubble sort by:

1. The number of swaps it performed
2. The number of comparisons

The worst case scenario is when the smallest number was at the end of the array. It took seven repetitions to sort the eight numbers. In general, if there are N numbers in the array, the process must be repeated N-1 times to make sure that the numbers are in the correct order.

Algorithm for the bubble sort with N elements in an array:

Start a loop to be repeated N-1 times

- Go through the array, comparing each number with the number after it.
- If the two numbers are out of order, swap them.
- Continue until we have compared the second last with the last number in the array.

Repeat the loop (reduce the size of the array by 1 after every repetition)

```
// sort the elements in an array from smallest to largest
public static void bubbleSort (int[] a)
{
    int temp;

    // outer loop that makes (a.length – 1) passes in the array
    for (int j = 0 ; j < a.length - 1 ; j++)
    {
        // go through all the elements and swap as needed
        for (int i = 0 ; i < a.length - 1 ; i++)
        {
            // if next element is smaller than current, swap elements
            if (a [i] > a [i + 1])
            {
                temp = a [i];
                a [i] = a [i + 1];
                a [i + 1] = temp;
            }
        }
    }
}
```

Look at this web sites for a visual demo of the insertion sort.

http://math.hws.edu/eck/js/sorting/xSortLab.html

http://www.youtube.com/watch?v=MtcrEhrt_K0

Sorting-Bubble Sort

Exercises

1. Show the comparisons and exchanges that would take place in using a bubble sort to put the following data in ascending order.

   3 8 3 2 7 5

2. What changes would have to be made to the bubbleSort method in order to make it sort values in descending order?

3. Make a program that generates 20 random integer numbers in the range 100 to 250. Print the array elements horizontally, then sorted from lowest to highest, and then print the array elements horizontally again.

4. Write a program to input the height (in cm) for 8 people. The program then calculates and outputs the average height for the group followed by all of the heights from highest to lowest order.

5. Write a program that inputs 20 words from the keyboard. The program will then sort the words and print them out in ascending order. To compare words, you can use the `compareTo()` method.

   ie.

```
String x = "hello";
String y = "there";

if (x.compareTo(y) > 0)
{
        System.out.println("First variable x appears earlier in the alphabet than y");
}
else if(x.compareTo(y) < 0)
{
         System.out.println("First variable x appears later in the alphabet than y");

}
else
{
         System.out.println("The variables have equal values");
}
```

6. Write a program that generates 10 **UNIQUE (no duplicates)** random numbers between 1 and 20 and stores them in an array. The program then sorts the numbers and displays them in ascending order. (make sure that the numbers are unique)