CSC 530/730 – Programming and Data Structure
Homework 1
(100 points)

Problems

Practice basic sorting algorithms by revising the given JAVA example program, `Example04BasicSorting`, as follows:

1. Revise function `bubbleSort()` so that each round of the algorithm moves the smallest element in a subarray to the beginning spot of the subarray by necessary swapping. For example, given an array with five elements 40 30 10 50 20, the algorithm should work as follows:

   Given:
   40 30 10 50 20 (Look at 40 30 10 50 20 and start the 1st round)

   1st round:
   40 30 10 20 50
   40 30 10 20 50
   40 10 30 20 50
   10 40 30 20 50 (Now look at 40 30 20 50 and start the 2nd round)

   2nd round:
   10 40 30 20 50
   10 40 20 30 50
   10 **20** 40 30 50 (Then look at 40 30 50 and start the 3rd round)

   3rd round:
   10 20 40 30 50
   10 20 **30** 40 50 (Finally look at 40 50 and start the 4th round)

   4th round:
   10. 20 30 **40** 50

2. Revise function `selectionSort()` so that each round of the algorithm finds the largest element in a subarray and swaps it with the last element in that subarray. For example, given an array with five elements 40 30 10 50 20, the algorithm should work as follows:

   Given:
   40 30 10 50 20 (Look at 40 30 10 50 20 and start the 1st round)

   1st round:
   40 30 10 20 **50** (Now look at 40 30 10 20 and start the 2nd round)

   2nd round:
   20 30 10 **40** 50 (Then look at 20 30 10 and start the 3rd round)

3rd round:
20 10 **30** 40 50 (Finally look at 20 10 and start the 4th round)

4th round:
10 **20** 30 40 50

3. Revise function insertionSort() so that each round of the algorithm results in a sorted subarray at the end of the array by doing appropriate insertions. For example, given an array with five elements 40 30 10 50 20, the algorithm should work as follows:

Given:
40 30 10 50 20

1st round:
40 30 10 **20** **50** (20 50 become sorted)

2nd round:
40 30 **10** **20** **50** (10 20 50 become sorted)

3rd round:
40 **10** **20** **30** **50** (10 20 30 50 become sorted)

4th round:
10 **20** **30** **40** **50** (10 20 30 40 50 become sorted)

4. Declare and implement a new function `oddevenSort()` by repeatedly making two passes through the array. On the first pass you look at all the pairs of items, a[j] and a[j+1], where j is odd (j = 1, 3, 5, …). If their key values are out of order, you swap them. On the second pass you do the same for all the even values (j = 0, 2, 4, 6, …). You do these two passes repeatedly until the array is sorted.

Given:
40 30 10 50 20

1st round:
40 10 30 20 50 (j = 1, 3)

2nd round:
10 40 20 30 50 (j = 0, 2)

3rd round:
10 20 40 30 50 (j = 1, 3)

4th round:
10 20 30 40 50 (j = 0, 2)

5th round:
10 20 30 40 50 (j = 1, 3) (no swap!)

6th round:

10 20 30 40 50 (j = 0, 2) (no swap!)

Submission

Compress the JAVA **project folder** into a *.zip* file and submit it on Blackboard.