Name______________________________________ EID___________

University of Texas, Austin
EE 312
Recitation 8 (Exercise 7)

For this exercise, you will be evaluating the speed of two sorting algorithms. Turn in this sheet when done.

1. Here is the pseudo-code to sort an array in increasing order using insertion sort. Write a void return C function to implement it. The starter code is provided in sort_functions.c.

```
for i = 1 to length(A)
        j ← i
        while j > 0 and A[j-1] > A[j]
                swap A[j] and A[j-1]
                j ← j - 1
        end while
end for
```

2.

a) Make a large array that is unsorted (or sorted in reverse, or sorted in decreasing order). You may use the function provided to you.
b) Sort this array using your insertion sort algorithm. You may test your algorithm using the check_sorted() function provided to you.
c) Instrument your code to measure the time taken for the sort, using the clock function in clock.h, as in the example provided in the starter code.
d) Repeat a-c for a total of 3 more different array sizes. Note the times taken for each repetition. You can start with, say, 10,000 ints and try 20,000, 30,000 and 40,000 ints, for example.
e) Divide the sorting time by the (known) BigO function of insertion sort for worst case data, and confirm that it is more or less independent of the array size.
f) Do steps a-d for the mystery sort function called fun1 provided to you.
g) Find the BigO complexity of the mystery sort function:
    i. Divide the sorting time by N, $N\log_2 N$, $N^{3/2}$ and N^2. You may use the provided Excel spreadsheet if you like, or copy the formulas to a Google spreadsheet, for example. Is any of the four division results nearly independent of size?
    ii. Estimate the complexity function of the mystery sort by plotting time vs. the four functions above, or simply by picking the one that from (i) that is nearly independent of size.

Big-O of Insertion Sort (worst case) is ______________________

Experiment #	Array Size	Sort Time	Time/N^2
1			
2			
3			
4			

Big-O of fun1 sort is ______________________

Experiment #	Array Size	Sort Time (T)	T/N	T/$N\log_2 N$	T/$N^{3/2}$	T/ N^2
1						
2						
3						
4						