Lab 6

Selection Sort		
List Size	**Comparisons**	**Time (seconds)**
1,000 (observed)	499500	0.07170605659484863
2,000 (observed)	1999000	0.24486017227172852
4,000 (observed)	7998000	1.080225944519043
8,000 (observed)	31996000	4.569947957992554
16,000 (observed)	127992000	17.587136030197144
32,000 (observed)	511984000	68.59300208091736
100,000 (estimated)	4999950000	215
500,000 (estimated)	124999750000	1075
1,000,000 (estimated)	499999500000	2150
10,000,000 (estimated)	49999995000000	21500

Insertion Sort		
List Size	**Comparisons**	**Time (seconds)**
1,000 (observed)	247986	0.07004213333129883
2,000 (observed)	1018717	0.3148181438446045
4,000 (observed)	3995264	1.2329838275909424
8,000 (observed)	16112194	4.996768951416016
16,000 (observed)	64667449	21.37734293937683
32,000 (observed)	257507119	78.58646726608276
100,000 (estimated)	2499975000	245
500,000 (estimated)	62499875000	1225
1,000,000 (estimated)	249999750000	2450
10,000,000 (estimated)	24999997500000	24500

Lab 6

1. Which sort do you think is better? Why?

   I think that selection sort is better based on the sole fact that it takes less time. Although there may be more comparisons, all I really care about is how long it takes for my process to complete. Therefore I believe selection sort is the better option.

2. Which sort is better when sorting a list that is already sorted (or mostly sorted)? Why?

   When sorting a list that is already sorted, insertion sort will preform much faster. This is because insertion sort doesn't have to check every number with all the previously sorted numbers. If the list is sorted it will only take n-1 comparisons making it much faster than selection sorts n(n-1)/2 comparisons.

3. You probably found that insertion sort had about half as many comparisons as selection sort. Why? Why are the times for insertion sort not half what they are for selection sort? (For part of the answer, think about what insertion sort has to do more of compared to selection sort.

   The reason why insertion sort has about half as many comparisons as selection sort is due to the fact that in the average case, insertion sort can stop comparing the value once it gets half way through the list while selection sort must keep comparing. Insertion sort has to make much more swaps than selection sort. The most swaps that selection sort has to make on the worst case is n-1 compared to n(n-1)/4 for worst case insertion sort. In the average case selection sort preforms far less swaps than insertion sort has shifts.