**Homework 9**

Arrays

BUBBLE:

NUM COMPARISIONS: 18706067

NUM ASSIGNMENTS: 24909638

INSERTION:

NUM COMPARISIONS: 6334946

NUM ASSIGNMENTS: 6349946

SELECTION:

NUM COMPARISIONS: 12535977

NUM ASSIGNMENTS: 12550978

If we look at this data from the context of Big O and n, we know first off that n = 5000, or the size of the array.

Theoretically, Big O of Bubble Sort is O(n^2). 5000^2 is 25000000. I got numbers very close to that for my number of assignments, and a little lower for my number of comparisons. This array had a lot of smaller values at the end of the array.

The Big O of insertion sorts are O(n^2) as well. So, like the previous algorithm, the average/worst case here is 25000000 operations. I got about 6000000, which is less than half of that predicted value. This array must have been more ideal for insertion sort.

Finally, the Big O of selection sorts is also O(n^2). I see a number of operations between the bubble and insertion sorts, but definitely in the correct range of n to n^2. There isn’t a good way to see if this was an average sort or now, because the Big O, Big Sigma, and Big Theta of selection sorts are all n^2.

Linked Lists

QUICK:

NUM COMPARISIONS: 435664

NUM ASSIGNMENTS: 688942

INSERTION:

NUM COMPARISIONS: 12125324

NUM ASSIGNMENTS: 13401238

MERGE:

NUM COMPARISIONS: 21349

NUM ASSIGNMENTS: 22843

Again, n is 5000.

For the quick sort, we know that the worst case Big O is O(n^2) but the average case is O(nlogn). The number I got was in the middle, but closer to nlogn (which is about 18000).

Insertion sorts with linked lists are also O(n^2), but they have to also traverse the linked list which is a hassle and often involved O(n) time. Thus is makes sense that the number of operations I got was above what I got for arrays.

Finally, merge sorts are capable of O(nlogn) time for best, average, and worst case scenario. This was on the worst case side as the number of operations was a bit above 5000 \* log(5000).