Aarav Dev
axd220001

These are the results I got from all the sorts:

Bubble Sort: 12497500 Element Comparisons and 6267659 Swaps
Selection Sort: 12497500 Element Comparisons and 4991 Swaps
Insertion Sort: 6270104 Element Comparisons and 6265114 Swaps
Quick Sort: 66178 Element Comparisons and 36651 Swaps
Merge Sort: 55273 Element Comparisons

$O(n^2)$ algorithms:
Bubble sort and selection sort both had the same number of comparisons because they both have nested loops used to iterate over the array and compare the appropriate elements. This results in both bubble and selection sort to always have n(n-1)/2 comparisons, with n being the size of the array. I did not expect there to be a difference between these two sorts in regards to comparisons because both of their average cases are $O(n^2)$ and the best case of O(n) in bubble sort is unlikely to occur due to the randomness of the initial array.
Insertion sort had fewer comparisons than bubble and selection sort for my example. This is because despite the nested loop, the inner loop is not performed as many times as in bubble and selection sort if the current element from the outer loop is already sorted relative to the elements previous to it at that moment in the loop. Thus, although insertion sort in the worst case will have n(n-1)/2 comparisons (same as bubble and selection sort), it is unlikely again due to the randomness of the initial array, and therefore has fewer comparisons. This is not what I expected initially, as I thought the number of comparisons would be the same due to the average case of bubble, selection, and insertion sort being $O(n^2)$. However, big O notation is not exact as it leaves out constants, and thus caused me to make an inaccurate prediction.
Bubble sort and insertion sort had many more swaps than selection sort. This is because in selection sort, elements are moved to their final position immediately as opposed to bubble sort and insertion sort swapping the same element up to several times before placing it in its final position. I did expect the big difference because knowing how selection sort works, I knew that it

would perform at maximum n-1 or 4999 swaps, and insertion and bubble sort would perform many more due to their nature of swapping up to several times per iteration.
Compared to each other, bubble sort and insertion sort had a relatively small difference in the number of swaps, with bubble sort having slightly more. I expected this because bubble sort can only swap elements which are adjacent with each other, whereas insertion sort allows for each element being compared to be swapped with an element which was initially far from it in terms of index value. I expected the difference to be bigger relatively, but I was wrong because although the numbers look similar, there is still a difference of over 2500 swaps between the two, so there is still a notable difference in swaps.

O(n log n) algorithms:
Quick sort had more comparisons than merge sort. This is because when performing a quicksort, if not the best pivot is chosen, extra comparisons have to be made in the algorithm because the partitions created will differ by a margin large enough to require more comparisons. I expected this because the worst case for quicksort is $O(n^2)$, and though it usually is not that high if the pivot is chosen correctly, even being just slightly off the best choice for a pivot will increase the number of comparisons. Also, merge sort will perform a relatively consistent number of comparisons due to its nature of splitting and rejoining the numbers in the algorithm.
Quick sort and merge sort cannot be compared with regard to number of swaps because merge sort does not perform any swaps.

Insertion sort vs O(n log n) algorithms:
Insertion sort had many more comparisons than both quick sort and merge sort and also had many more swaps than quick sort (merge sort does not have any swaps). This is because insertion sort uses nested loops as compared to quick sort and merge sort which use repeated splits or partitions for each element in the array. I expected there to be a difference in these numbers because insertion sort is $O(n^2)$ whereas quick sort and merge sort are O(n log n).