Insertion Sort:

Input Size	Execution Time
10	-0.97 seconds
50	-0.96 seconds
500	-0.94 seconds
5000	0.54 seconds
10000	3.69 seconds
15000	9.26 seconds
30000	38.49 seconds
60000	166.44 seconds or 2 minutes 46 seconds

Merge Sort:

Input Size	Execution Time
10	-0.96 seconds
50	-0.95 seconds
500	-0.92 seconds
5000	-0.42 seconds
10000	-0.23 seconds
15000	0.23 seconds
30000	1.59 seconds
60000	3.73 seconds

For this experiment, I was exploring two sort algorithms to see which one was faster. I began at a small input size and increased it rapidly to see which algorithm would sort the data faster. I had a random number generator insert numbers into an array. I constrained the random numbers to be between 1 and 500,000. From my experiment, I can see that when the input size is small there is little to no difference between insertion sort and merge sort. For example, at input size 10 the time of execution is only separated by .01 of a second. That remains the case for input size up to 500. After jumping to input size 5000 I begin to see the difference between the two algorithms. At input size 5000 the difference is much greater than .01 and as the input size increases beyond 5000, I see that the merge sort is the more efficient algorithm. At input size of 60000, the insertion sort algorithm took 2 minutes and 46 seconds to sort the data, while on the other hand, the merge sort algorithm took a mere 3.73 seconds to sort the same input size. To conclude, if the input size is below about 5000 then using either algorithm will produce around the same results. As you increase your input size beyond 5000 data members the merge sort algorithm quickly proves that it is superior at sorting larger data sets.