Analysis:

The program does not use optimized bubble sort because it needs the algorithm to sort in the pattern of the worst case every time for a better comparison with the mergesort. As we see from the chart “analysis.csv,” as the file size gradually increased by 10k every time, time needed for the bubble sort increases significantly in a way that resembles exponential growth, starting from about 44,440,000 nanoseconds at 1k file to about 57,000,000,000 nanoseconds at 100k. According to the Big O notation of bubble sort, an n-size file needs about n^2 the time to execute the program. The graph below shows the pattern of bubble sort that resembles its Big O notation. When it comes to the merge sort, according to the csv chart, time needed for each execution gradually increases along with the 10k-increasing process, starting from about 2,200,000 at 1k file to 67,900,000 at 100k. Since the best case and worst case of the Big O notation of mergesort are the same, the program does not need to be optimize for comparison with the bubble sort. The Big O notation of mergesort for an N-sized file is O(logN\*N), which is similar to what the graph of its sorting time (relative to the graph of bubble sort) shows. The graph below shows a pattern of mergesort’s execution time relative to bubble sort, which shows that with the same increasing rate of file size, time needed for bubble sort increases more significantly than mergesort. In conclusion, according to the Big O notation as well as the graph, mergesort is a less time-consuming and efficient sorting algorithm compared to bubble sort.