**Project 1 Sorting Report**

Kuan Wang (kw300)

Yuhang Liu (yl655)

In this project, we implemented five algorithms for sorting an array. The five algorithms are Selection Sort, Insertion Sort, Bubble Sort, Merge Sort and Quick Sort, respectively. The implementation of the five algorithms proves to be correct, because the results successfully passed all the testcases.

Apart from the correct implementation, we also analyzed the performance of each algorithm with different size of input arrays based on their runtimes. Basically, a good algorithm has smaller sorting runtime with the same input runtime. From *Figure-1* below, Quick Sort is the best algorithm and Merge Sort comes the next with slightly longer runtime. Selection Sort and Insertion Sort ranks the third and the fourth. Bubble Sort is the worst algorithm because it takes the longest runtime.

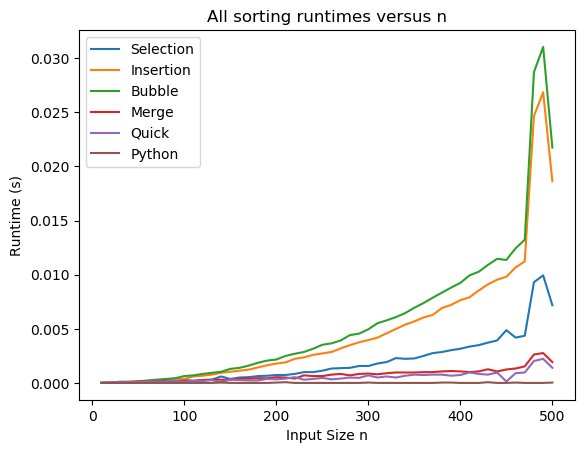


Figure-1 All sorting runtimes versus input size n

According to the theory of sorting algorithms, the total work required in Selection Sort, Insertion Sort and Bubble Sort are all *O*(*n*2) for average case, while the runtime for Merge Sort and Quick Sort are both *O*(*n*log*n*). We can also see this feature from *Figure-1*, because Merge Sort and Quick takes much less runtime than the other three algorithms.

In addition, the log-log plot of runtime versus input size can help analyze the Big-O notation of different algorithms in a better way. The slope of the line in log-log plot represents the value of *k* in *O*(*nk*). According to *Figure-2* and *Figure-3*, when *n* is large, the log-log slopes of Selection Sort, Insertion Sort and Bubble Sort are all between 2.4 and 2.5, which approximately accords with *O*(*n*2). #how to explain why larger than 2?# On the other hand, the log-log slopes of Merge Sort and Quick Sort are around 1.35, which is smaller than the theoretical slope of *O*(*n*2) and larger than that of *O*(*n*), and also roughly accords with *O*(*n*log*n*).

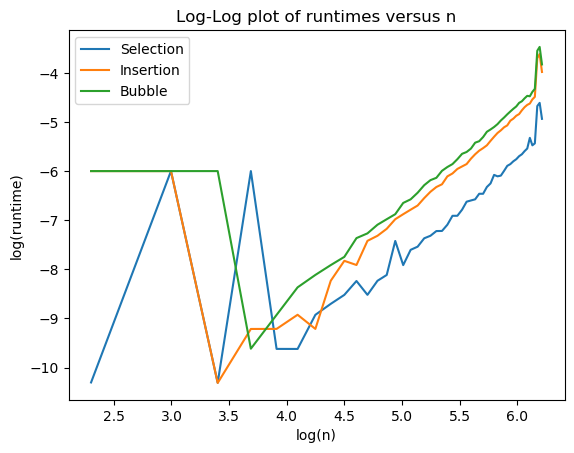


Figure-2 Log-log plot of runtimes versus n

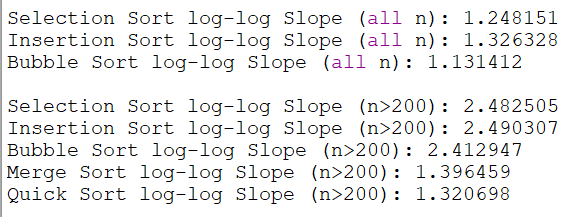


Figure-3 Log-log slope for different algorithms

In *Figure-3*, another point we should pay attention to is that log-log slope is much smaller than the theoretical value for all *n*. In *Figure-2*, we also notice that the value of log(runtime) does not accord with the general trend and fluctuates sharply when log(*n*) is small. Intuitively, when the input size is small, the extreme cases would have larger influence on the runtime, making it deviate from the theoretical value. Meanwhile, when the input size is large, extreme cases would have smaller influence on the runtime, so the slope becomes more stable and closer to the theoretical value. Therefore, asymptotically large values of *n* make better sense and should be taken into consideration.

Moreover, it is meaningful and significant to average the runtime across multiple trials, rather than use only one trial. The reason is that some algorithms can have best cases and worst cases other than averages cases. If we only use one trial and it happens to be a best or worst case, the experimental runtime would be more likely to deviate from the theoretical value. We can largely avoid this drawback by taking a number of measurements and averaging them.