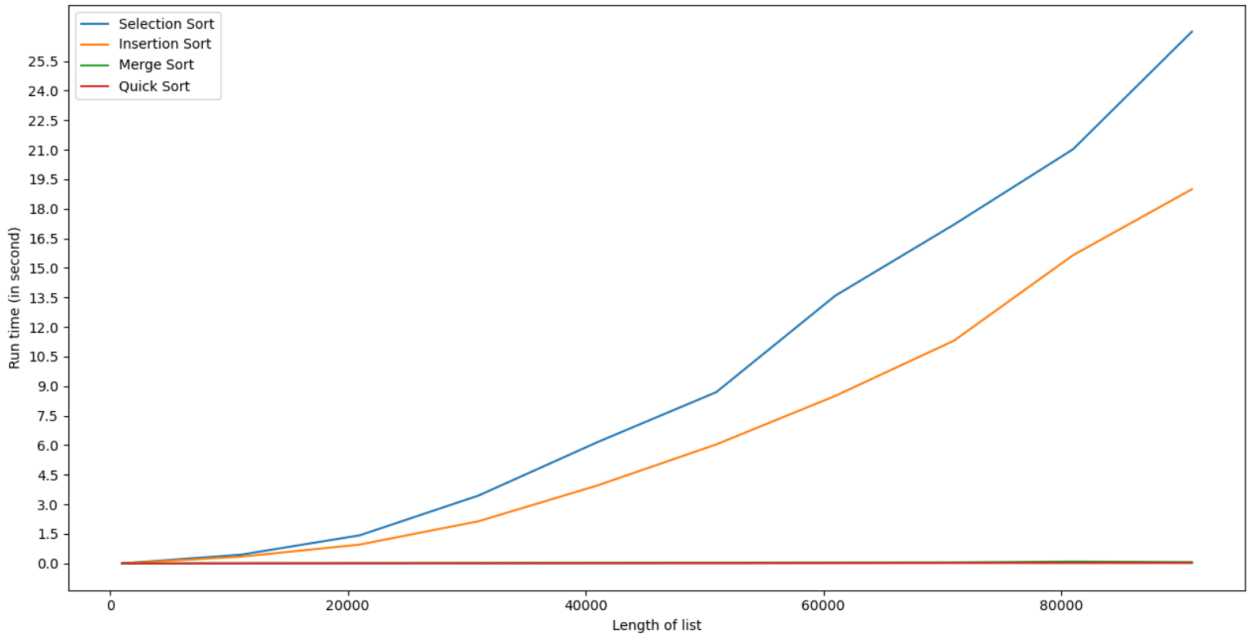
Note: You can scroll down to the bottom to see how to run the program and plot data.

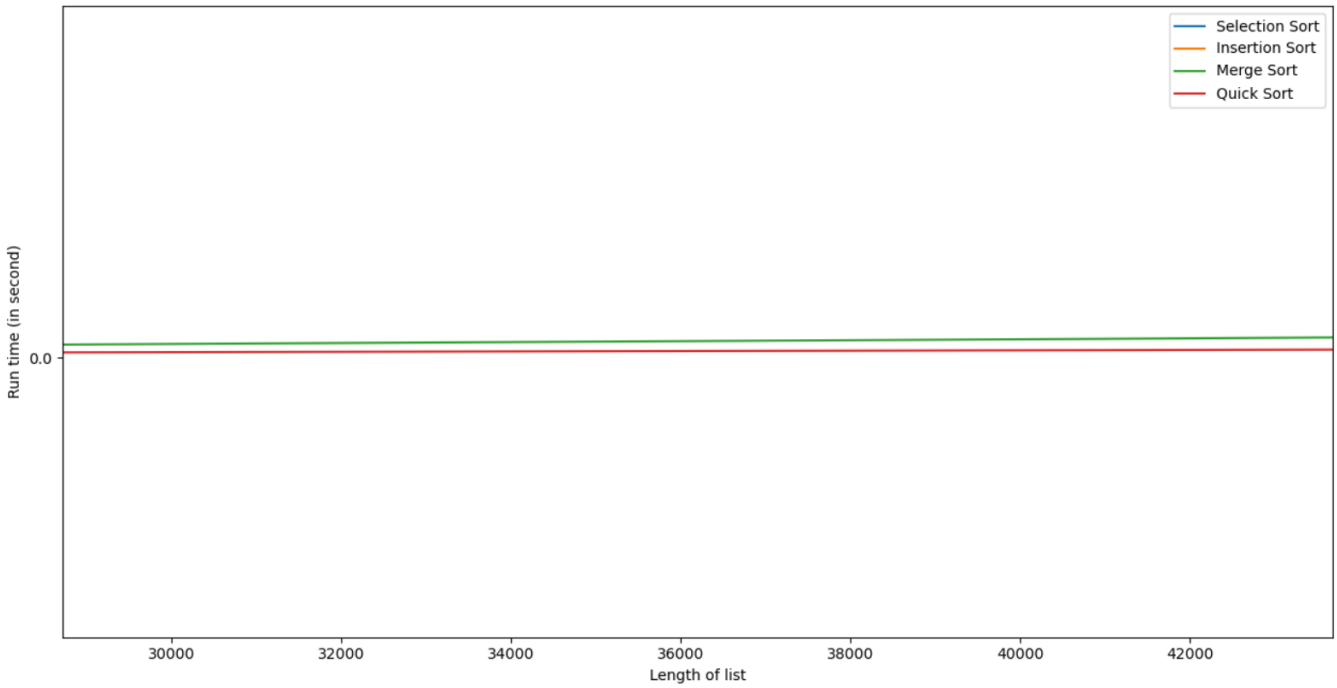
Here is the plot of run times of different sorting algorihtms:



- At the beginning where the list's length is small, all four sorting algorithms have roughly the same run time.

- However, as the list's length grow larger, the run times of Selection Sort and the Insertion Sort become drastically bigger than the run times of Merge Sort and Quick Sort, while the run times of Merge Sort and Quick Sort are roughly the same as before.

- As we can see, the run times of Merge Sort and Quick Sort are so close that we hardly notice the difference between them. Here is the image when I zoom in to see the difference between Merge Sort and Quick Sort more clearly:



- This result is quite reasonable, since the time complexity of the Selection Sort is always O(n^2), even when the list is already sorted.

- The worst-case time complexity of the Insertion Sort is also O(n^2), but if the list is nearly sorted or partially sorted, the run time of Insertion Sort is faster than the worst case. In the case that the list is nearly sorted, the Insertion Sort takes O(n). Therefore, although the worst-case time complexity is O(n^2), the Insertion Sort is faster than Selection Sort in some cases, and we can see this difference in the graph.

- The time complexity of the Merge Sort is always O($n\ log(n)$), which is significantly smaller than O(n^2) and remains low even when the input is large. For example, when n = 1000, $n\ log(n)\ =$ 9965 and $n^2\ =\ 1000000$. When n = 91000, $n\ log(n)\ =$ 1499095 and $n^2\ =\ 8281000000$. That's the reason why the run time of Merge Sort is so small compare to Selection Sort and Insertion Sort.

- It is a bit special when it comes to Quick Sort. The worst-case time complexity of Quick Sort is O(n^2). But if our algorithm is implemented correctly, this worst case rarely happens. However, the **average or typical** run time of Quick Sort is O($n\ log(n)$). That's the reason why in the graph we see that the run time of Merge Sort and Quick Sort is roughly equal. In fact, Quick Sort is a little faster than Merge Sort, maybe because Quick Sort doesn't need much extra space, while Merge Sort requires O(n) extra space.

Instructions to run the program

1. `cd` into the program directory.
2. Run the command `make`.
3. The command `./main` will print the output including run time of different sorting algorithms regarding different input sizes.
4. The command `./main > data.csv` will export the data, including run time of different sorting algorithms regarding different input sizes, into **data.csv** file.
5. To plot the result, run the file **plot.py**
6. To clean objects file or executable file from the directory, run the command `make clean`.