Introduction

For this project, we were required to explore various sorting algorithms, six to be precise and analyze their time complexities. To do so, I have chosen 52768 sample I-TIN numbers, which will need to be sorted by the USCIS. I-TIN numbers are individual tan identification numbers that are given to U.S. foriegn immigrants without an SSN. After finding the data, I used excel to turn them into 4 data types: InOrder, RandomOrder, ReverseOrder, and AlmostOrder. AlmostOrder is a data set that is extremely similar to InOrder, except for the fact that its second and fourth I-TIN numbers are swapped. Every other data set represents its namesake.

Algorithm/Code:

To view the full code, please refer to the files attached in the zipped project folder.

Algorithm analysis (What I learned):

- **Insertion sort:** I learned that insertion sort is the fastest sort at O(N) linear time complexity for InOrder and AlmostOrder data types, however for the other two data types, insertion sort was some multiple of N^2 such as 0.25N^2 or 0.5 N^2. The graphs correspond to data in the excel file titled Project 1- Sorting Algorithms.
- **Selection sort**: Selection sort wasn't the fastest sort in any case. In fact, it was by far the slowest sort even when it made the least amount of swaps.
- Quick sort: Due to the immense size of the data, quick sort couldn't be implemented recursively. This is a point for further exploration of sorting algorithms where perhaps a non-recursive solution can be implemented for big data sets. However, the BigO ideally will be nlogn for RandomOrder and Almost Order. InOder and ReverseOrder will have a BigO of N^2 because they are the worst case scenarios with the furthest pivot points
- Merge sort: Merge sort was the fastest sort for RandomOrder and ReverseOrder with a BigO of nlogn.
- Heap Sort: Heap sort was very fast, and very reliable with a consistent BigO of NlogN
- Radix Sort: Radix sort was by far the easiest to comprehend and it makes sense that it has no comparisons, because it swaps from the 10th's place to however many digits there are in the number.

Graphs:

Graphs can be found in the project folder.