**Lab 8 – Sorting out the Sorts**

# Method:

The first step we did determine the time complexity of all of the given Greek letters. We did this through graphing the time each algorithm took across various values of n. At a very large n value of 100,000 Beta and Epsilon maintained a time of less than 20 while all of the other algorithms were well over times of 1500. This allowed us to draw the conclusion that Beta and Epsilon were O(nlogn) while the other algorithms were n^2.

After we began to analyze the movements of the O(n^2) algorithms. We noticed that in a sorted list only Alpha had a movement of 0. After looking at the code we noticed that Bubble Sort and Insertion sort had to have at least 1 movement when sorting and therefore Alpha had to be the O(n^2) algorithm, Selection sort.

Next, we were able to determine that Gamma was the bubble sort algorithm. This was also due to the number of movements. In the bubble sort code it had to make at least one movement when the list is sorted and no more unlike insertion which would create many movements even if the list was sorted. As we had assigned 2/3 of the O(n^2) algorithms (Alpha: Selection, Gamma: Bubble), Delta had to be insertion sort.

Finally, we looked at the O(nlogn) algorithms. We noticed that for Beta sort the amount of comparisons was equal whether the list was sorted in order versus reverse order. This allowed us to determine that it was Merge sort. The lists splits and compares in the same fashion whether the list is in order or reverse while a Sort like Quick sort will vary as the pivot point will change given the arrangement of the order of variables. Thus our final determination for the O(nlogn) sorts: Beta: Merge Sort, Epsilon: Quick Sort.

# Results:

Table alpha: Algorithm Alpha

Table

Description automatically generated

Chart, line chart

Description automatically generated

Table Beta: Algorithm Beta

Table

Description automatically generated

Chart, line chart

Description automatically generated

Table Gamma: Algorithm Gamma

Table

Description automatically generated

Chart, line chart

Description automatically generated

Table

Description automatically generated

Table Delta: Algorithm Delta

Chart, line chart

Description automatically generated

Table Epsilon: Algorithm Epsilon

Table

Description automatically generated

Chart, line chart

Description automatically generated

|  |  |
| --- | --- |
| **Greek Label** | **Time Complexity** |
| Alpha | O(n^2) |
| Beta | O(nlogn) |
| Gamma | O(n^2) |
| Delta | O(n^2) |
| Epsilon | O(nlogn) |

# Discussion:

We learned about the number of comparisons and movements algorithms had when sorted in order. Overall, if the list was sorted in order both Selection and Bubble sort made 1 or less comparisons. We also learned that Merge sort, if sorted in reverse or in order, made the same number of comparisons an equal amount of comparisons.

# Conclusion:

1. We found that in an inOrder list there were 0 movements because of this algorithm Alpha must be Selection Sort.
2. We found that in an inOrder list there was 1 movement because of this algorithm Gamma must be bubble sort. Which had movement++ in the do – while loop which only executed once because they were already in order.
3. We had already determined that Alpha, Delta, and Gamma were O(n^2), also knowing the time complexity of Selection, bubble, and insertion to be O(n^2), Algorithm Delta must be Insertion Sort.
4. We found that when an InOrder or Reversed Order list there were the same number of comparisons for Algorithm Beta which is true for Merge Sort Algorithm
5. By process of elimination Algorithm Epsilon must be Quick Sort

|  |  |
| --- | --- |
| Algorithm | Sort Name |
| Alpha | Selection |
| Beta | Merge |
| Gamma | Bubble |
| Delta | Insertion |
| Epsilon | Quick |