Adrian Ornelas Ruvalcaba

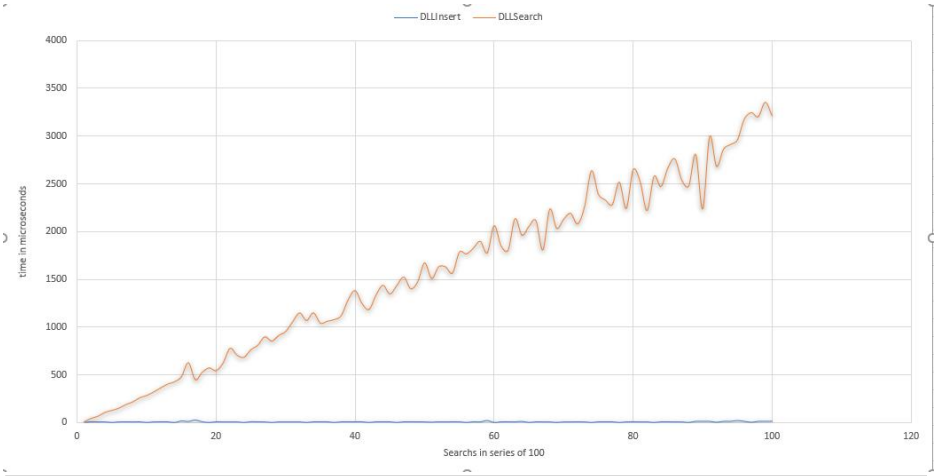
Final Project

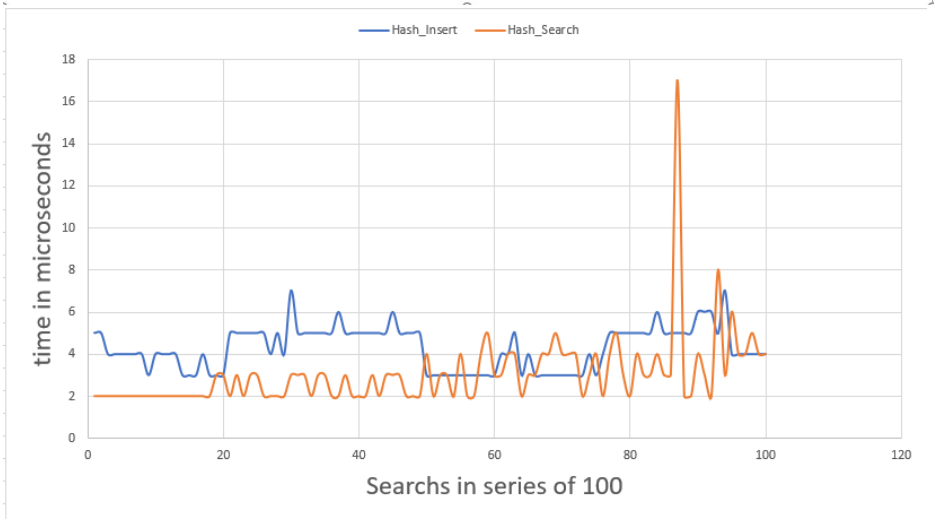
CSCI 2270

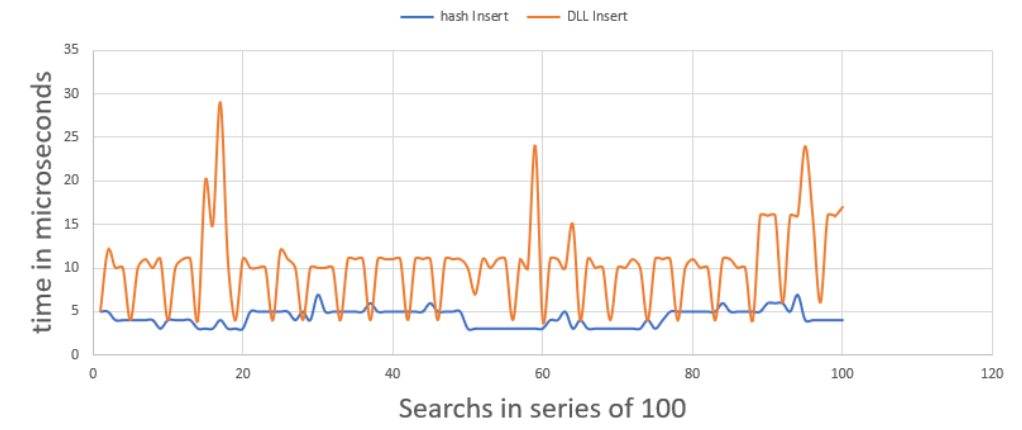
Part A:

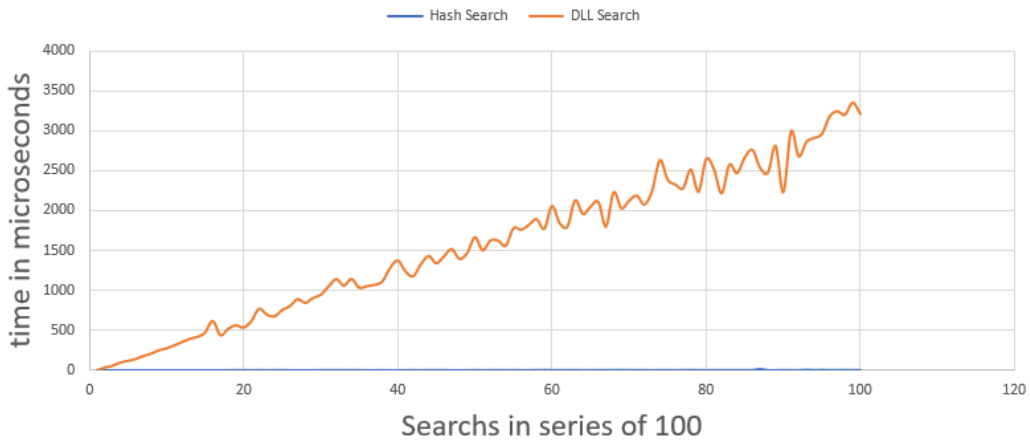
Explanation:

The Double linked list on graph form looks like it's the slower data structure and you would be right. For all the timed information presented were both on increments of 100 on insert and search. For a Double linked list we have an insert time of average is 4 microseconds which is still impressive but you must consider that its only adding at the end which means its hopping n amount of times which makes it O(n), for search in a double linked list the average time was 560 microseconds. That is a very high number but you must understand that a double linked list does not have a specific index to ensure where a specific key is at, it's always traversal but maintains an O(n). Hash table has an average of insert time of 1 micro seconds because of an index given, but keep in mind that the index is found by a hash function so when there is nothing inside that index than place it there and it's an O(1) but were there a duplicate (collision) you must apply quadratic probing which turns O(1) into O(n). Collisions ended with about 893 which is yet bad and didn't affect its time for the most part. Hash Table also has a search time average of 1 because of the same reason for insert. Now looking at the graphs and the Big O, the comparison makes it clear that Hash Functions are the faster data structure to insert and search through a large amount of dat


DLLInsert
DLLSearch
time in microseconds
Searchs in series of 100


Hash_Insert
Hash_Search
time in microseconds
Searchs in series of 100


hash Insert
DLL Insert
time in microseconds
Searchs in series of 100


Hash Search
DLL Search
time in microseconds
4000
3500
3000
2500
2000
1500
1000
500
0
0
20
40
60
80
100
120
Searchs in series of 100

Part B:

Explanation:

Bubble sort is a simple method in order to sort the information you want sorted, but it is not the most efficient. Bubble sort has an average time of 399 microseconds of the sorting increase. In Big O worlds this runs in a O(n^2) because of the 2 nested loops. Both loops running n times which leads to quadratic time. In other words if it was an array of 10 it must execute 100 times. Which shows the time increase drastically due to the amount of information they must go through. Heap sort is a different story, with an average time of 36 milliseconds which is about 10 times faster than bubble sort. Due to the fact it's a recursion it was expected to do O(2^n) times because of the fact that each addition of inputs increases the time it takes. But because it's a complete binary tree it causes a solution a lot more efficiently so it truly turns into O(nlogn). Heap sort is the faster data structure when organizing data when its compared to a nested loop.