Lab 05 – Linear Search vs Binary Search

# Problem

Assigned the task of generating an array of size 1000, and then search through this array using two sort methods: linear search and binary search. We were then to conduct the same test 20 times, and calculate the amount of times the loops ran or the methods were recursively called in order to determine which method of searching was more efficient.

# Proposed Solution

Generate an array of size 1000, and then populate the array utilizing a Math.random() command. Ensure to force the Math.random() number generated to an integer type, as it normally produces a type of double. Sort the array utilizing a simple sort command, I chose bubble sort. Program a method to search through the array for a value x linearly, and ensure that a count value is increased each type we move forward in the array. Also program a method to utilize the binary search, where we separate the array in two halves and by using a divide and conquer algorithm, search the array.

# Tests and Results

After running the program 10 times, the rough average number of linear checks performed was 550, and the rough average for binary checks performed was 8. Binary search is a far superior method in terms of efficiency, but linear search is much easier to code. The trade off depending on the size of your array and the efficiency desired should be noted.

# Problems Encountered

I am still getting used to recursion, so I experienced some issues at first with binary search. I also generated a few erroneous binary search results due to an error in one of my recursive calls, as I was passing along the midIndex twice instead of the minIndex, which caused my binary searches to continue endlessly and the amount of checks performed was exponentially higher than it should have been.

# Conclusions and Discussions

Good practice again with recursion. This certainly proves that binary search is a superior search algorithm to linear search, but depending on the size of your array linear search could still be utilized. Recursion, to me at least, is still a bit harder to code than a strict search such as linear, so the additional time required for coding something like the binary search would only be offset by an array of sufficient size as to require a more efficient search algorithm.

# Additional Questions

**Lab Report Questions:**

1. What is the Big Oh complexity for linear search? Binary search?

Linear Search : **O**(n)

Binary Search : **O**(lg(n)

1. Plugging in the size of the data in this test into those complexities, did your tests validate those assumptions?  Why?

Linear search resulted in some cases where it had to parse through the entire array, of size 1000, and so 1000 searches were performed. This validates the big O complexity of linear search : **O**(n), where n in this case is 1000.

Binary search resulted in some cases where it had to parse through the entire array, of size 1000, and so 10 searches were performed. This validates the big O complexity of binary search: **O**(lg(n), where n in this case is 1000 and lg ( n ) = 9.96578428466.