**CPSC 331**

**Calder Sloman, 30107102**

**Carlene Kalin, 10162051**

**June 2nd, 2023**

**CPSC 331 - Assignment #4**

**Hash Tables and Searching**

Comparing Linear Search and Binary Search

Linear search checks each element in the array sequentially until a match is found, or the entire array has been searched without finding a match. The time complexity is O(n) where n is the size of the array, so it works best with smaller array sizes. Sorting the array doesn’t change the complexity, as each element will still need to be checked one by one in order.

Binary search is much more efficient than linear search when searching a large, sorted array. Its complexity is O(long(n)), where n is the size of the array, because it uses a divide-and-conquer method of searching and eliminates half the search space each iteration.

The turning point is when the array size is in the 1000’s or more, at which point binary search has the advantage. If the array is sorted, binary search is the better choice because of its faster time complexity. However, if the array is unsorted or is small, linear search is a simpler option. If we have a large array and perform multiple searches, it could be more efficient to sort the array first and then use binary search.

Comparing Binary Search and Hash Search

Binary search is used on sorted arrays and follows a divide-and-conquer approach where it compares the middle value of the array and eliminates half of the search space on each iteration. Because the arrays need to be sorted beforehand, it may take additional time.

Hash search uses a hash table, which maps keys to ‘buckets’ of values. This allows access to elements without needing to search sequentially. It’s time complexity is O(1) on average, so the size of the array won’t affect the search time. However, all the elements need to be hashed, which means that populating the hash table is O(n). What this means, is that if you were only planning on conducting a single search on a set of values, there would be no real benefit. The efficiency of a hash table is best realized when a high volume of searches are to be performed on a data set.

The turning point at which the hash search becomes more effective depends on the size of the array and the expected number of operations to be performed. The hash search is good for large array sizes and when the number or search operations is expected to be high, especially compared to the cost of building the hash table first. If the cost of building the hash table is too high, it could be more beneficial to use binary search. In the end, it depends on whether it’s more costly to build the hash table or to perform multiple search operations.