Cs 313 project

Group: Luke Fang, Gomes Mikhaiel, Hila Ergys, Ibrat Nazara, Kaur Rupinder

Question # 2

The question asks to test how fast the different versions of binary search functions perform in comparison to each other. The binary searches are implemented on arrays of various large sizes, containing random numbers. To implement binary search on the array, the array content must be sorted. Starting with a value integer 1 for the first element in the array, each succeeding value is determined by skipping or adding a randomly generated amount from the previous value. After the array has been sorted, the execution time of the binary search function may be tested, using chrono library. As a result, the iterative binary seems to take less time to execute, in comparison to recursive, with linked-type binary search being the slowest. In this case, recursion is slower because the function calls within the function are added into the stack where copies of local memories are reallocated in the next function and finally a bool value is returned back after the base case to the previously called functions which take up time. Linked type binary search was shown to have a longer time by far, because it does not utilize the advantages of binary search. Linked list data structure has different properties as opposed to arrays, where memory can be accessed immediately by index location. The purpose of binary search is to create a more time effective search within the data structures, which has the main purpose of optimizing runtime complexity to O(log N). It does this by eliminating half of the list by the result of each comparison with less comparisons than traversing through the whole list. For the linked list to access a memory location it must traverse through the list to the location, as opposed to accessing the location immediately, as demonstrated when accessing middle elements in the list. This shows in the case of linked lists, linear search is more appropriate rather than binary search.