Assignment 1: Question 2

Algorithms

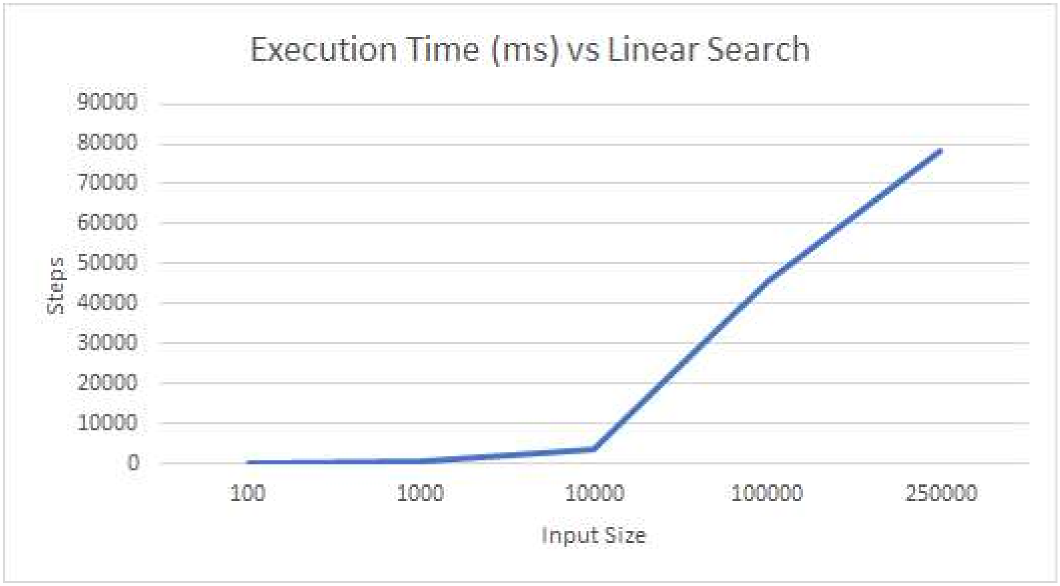
November 9th, 2021

By Tate Donnelly

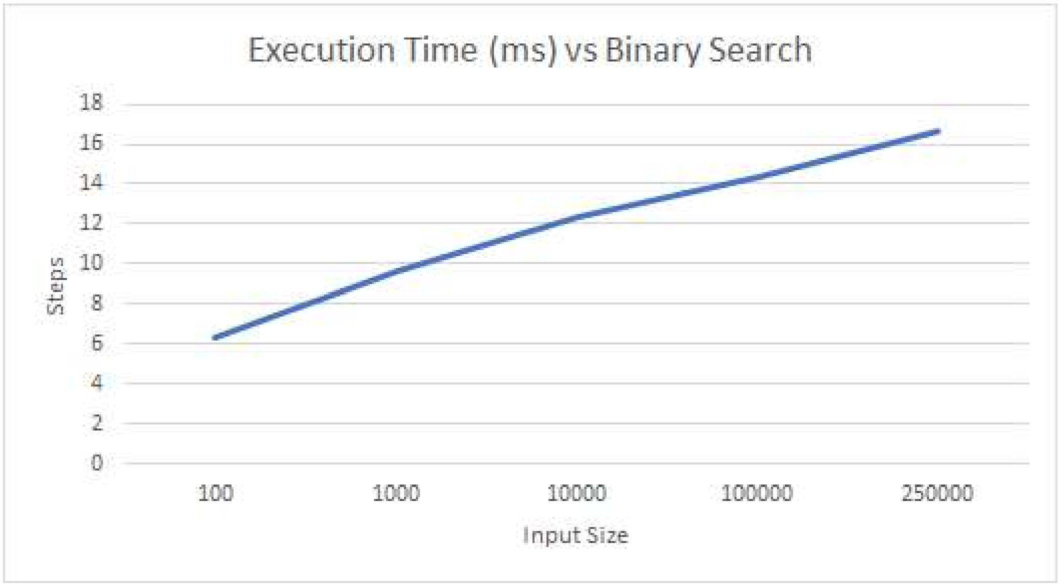
Comparing Linear Search, Binary Search, and Recursive Binary Search

In order to compare the efficiency of linear search, binary search, and recursive binary search algorithm, I ran a series of experiments where I tested how many steps it took each graph to find a single key in the same array. To measure the efficiency of each algorithm, I placed a step counter in logical places in the algorithms. Typically, the step counter was placed in areas where the algorithm had to look at an individual member of the array. You can find more information about the placement of the step counter can be found in the code for each algorithm. To get a good picture of how the number of steps varied based on the algorithm, I tested each algorithm with multiple input sizes(100, 1000, 10000, 100000, and 250000). I ran each algorithm with each input size 3 times and averaged those trials to produce my graphs to account for variance in the array's data. I also used the same array for every algorithm to ensure I was comparing them fairly.

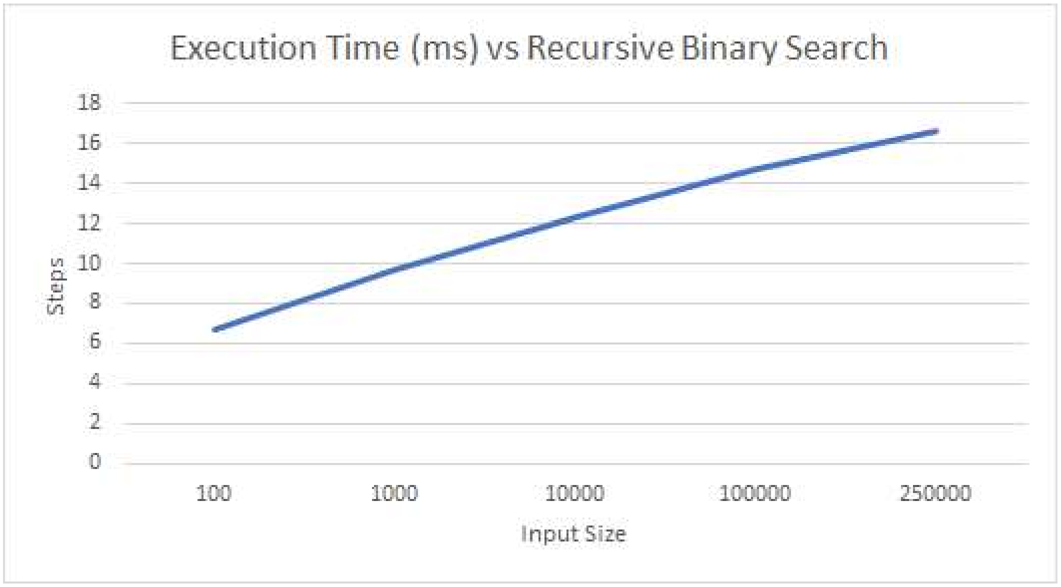
As you can see from “Execution Time vs Linear Search, Binary Search, and Recursive Binary Search (Log Scale),” the linear search algorithm took significantly more steps than both the binary search and recursive binary search algorithms. The linear search graph's line is also has a dramatically sharper path than the lines for the other two algorithm graphs. These factors indicate that the linear search algorithm is significantly less effective at finding data points in arrays than the binary search and recursive binary search algorithms. “Execution Time vs Linear Search, Binary Search, and Recursive Binary Search (Log Scale)” also shows that the binary search and recursive binary search algorithms are nearly identical in terms of efficiency. However, the recursive binary search graph line is just slightly smoother than the line for the binary search graph, and when you compare both algorithms' experiment results (like in "Execution Time vs Binary Search and Recursive Binary Search"), the graph shows that the binary search algorithm is just slightly more effective than the recursive binary search algorithm. Therefore, when it comes to searching arrays for one number, the binary search algorithm is the most efficient, though the recursive binary search comes very close to being as efficient, and the linear search algorithm is the least efficient.



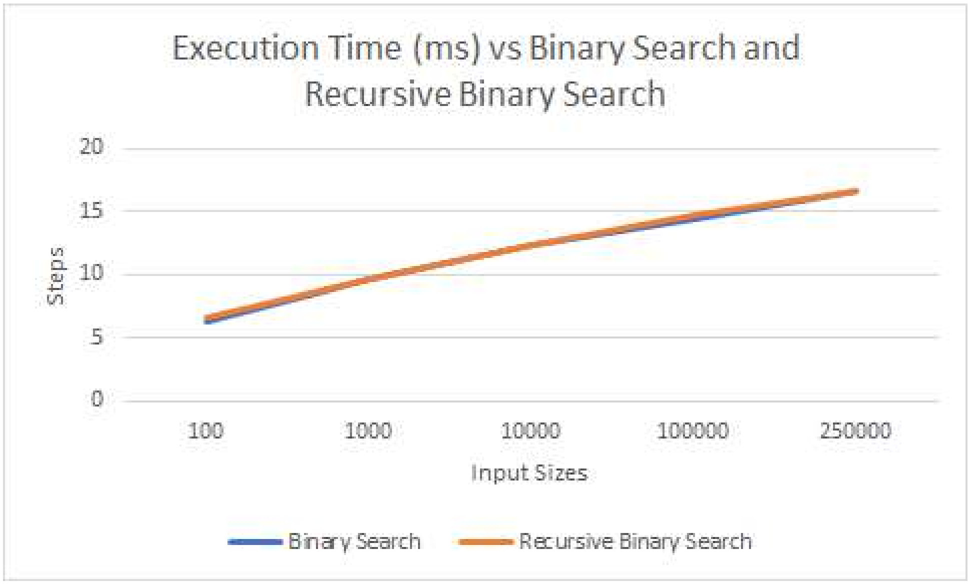
The linear search algorithm moves through each item until it finds a certain int (the key parameter), at which point it returns.

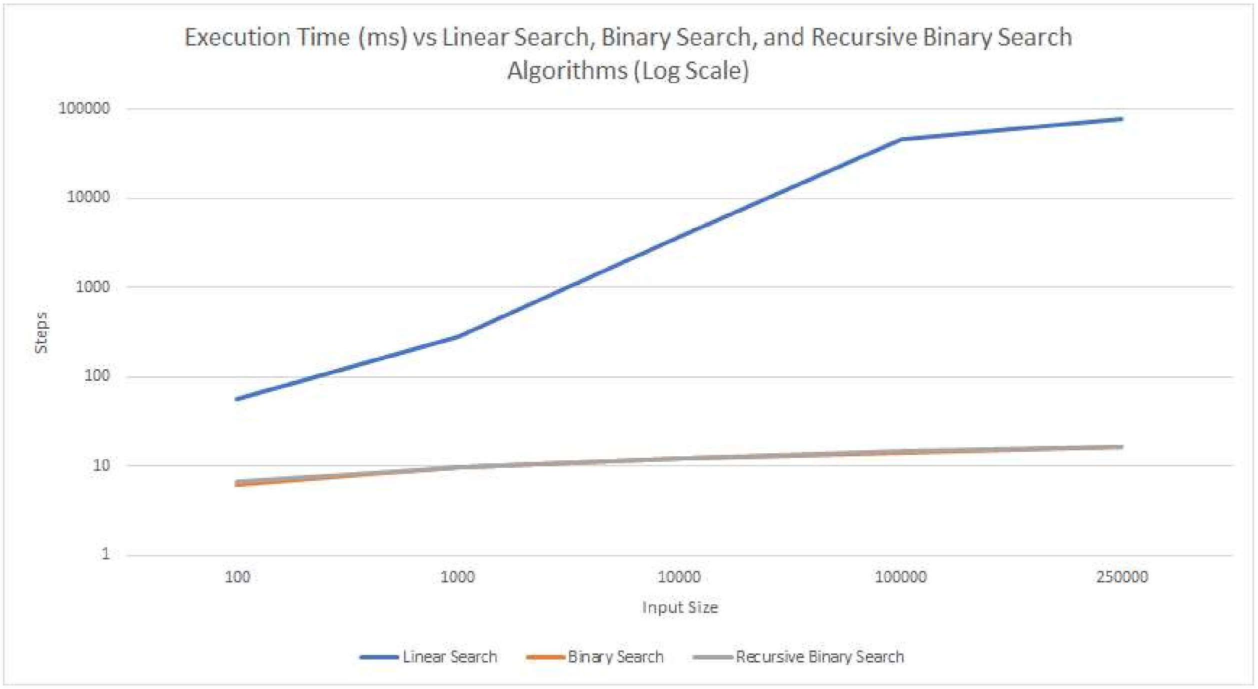


The binary search algorithm will search through a sorted array for a single int (the key parameter). It does this by examining if the current element is looking at is higher or less than the key with variables like lo (which is set to 0) and hi (which is set to the array's length-1). If the current element is less than the key, the function increments the lo variable and the array index. If the current element is higher than the key, the function decrements the hi variable and the array index. This goes on until it finds the key or until the loop ends.



The recursive binary search algorithm will search through a sorted array for a single int (the key parameter). It does this by examining if the current element is looking at is higher or less than the key with variables like lo (which is set to 0) and hi (which is set to the array's length-1). If the current element is less than the key, the function increments the lo variable and the array index. If the current element is higher than the key, the function decrements the hi variable and the array index. However, instead of running in the loop like the binary search algorithm, the recursive binary search algorithm simply implements recursion to test new values.




Execution Time (ms) vs Linear Search, Binary Search, and Recursive Binary Search Algorithms (Log Scale)
100000
10000
1000
100
10
1
Steps
100
1000
10000
100000
250000
Input Size
Linear Search
Binary Search
Recursive Binary Search