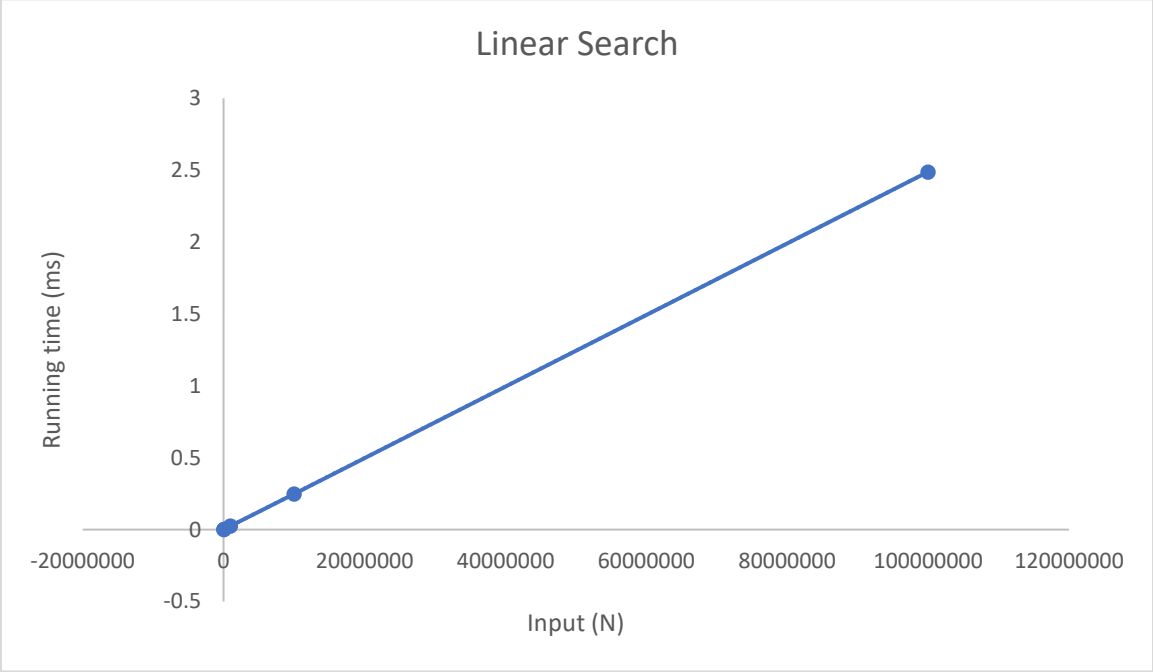
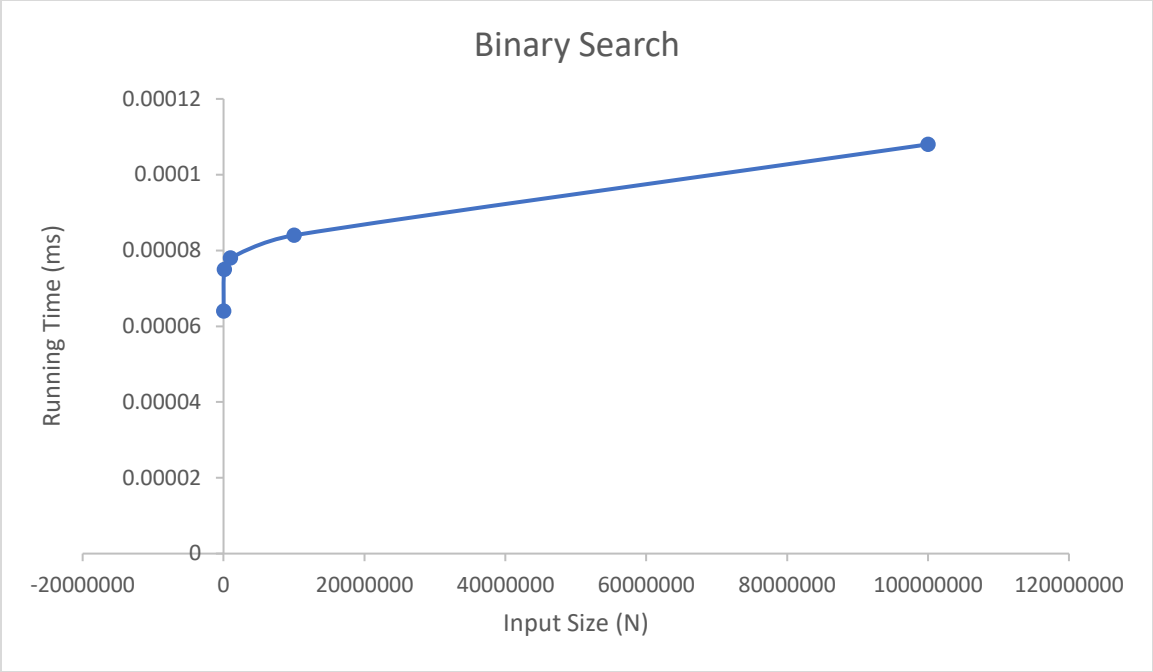
Key at 1% mark

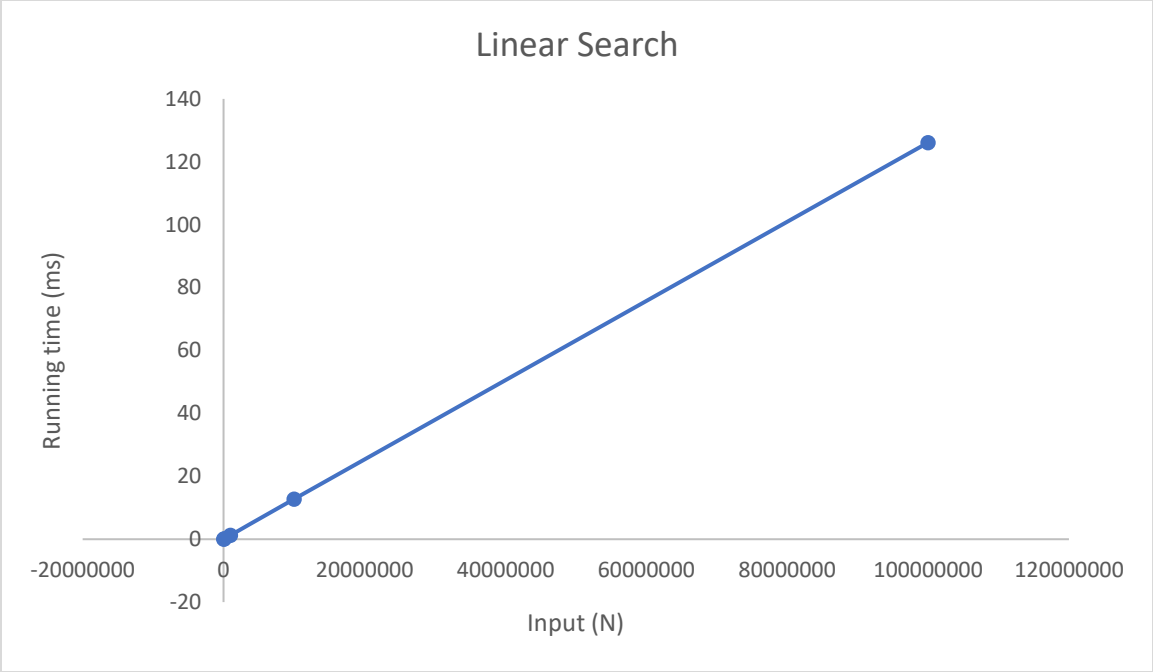
Input Size (N)	Linear, Time (ms)	Binary, Time (ms)
10000	0.00028	0.000064
100000	0.0026	0.000075
1000000	0.0258	0.000078
10000000	0.2476	0.000084
100000000	2.488	0.000108

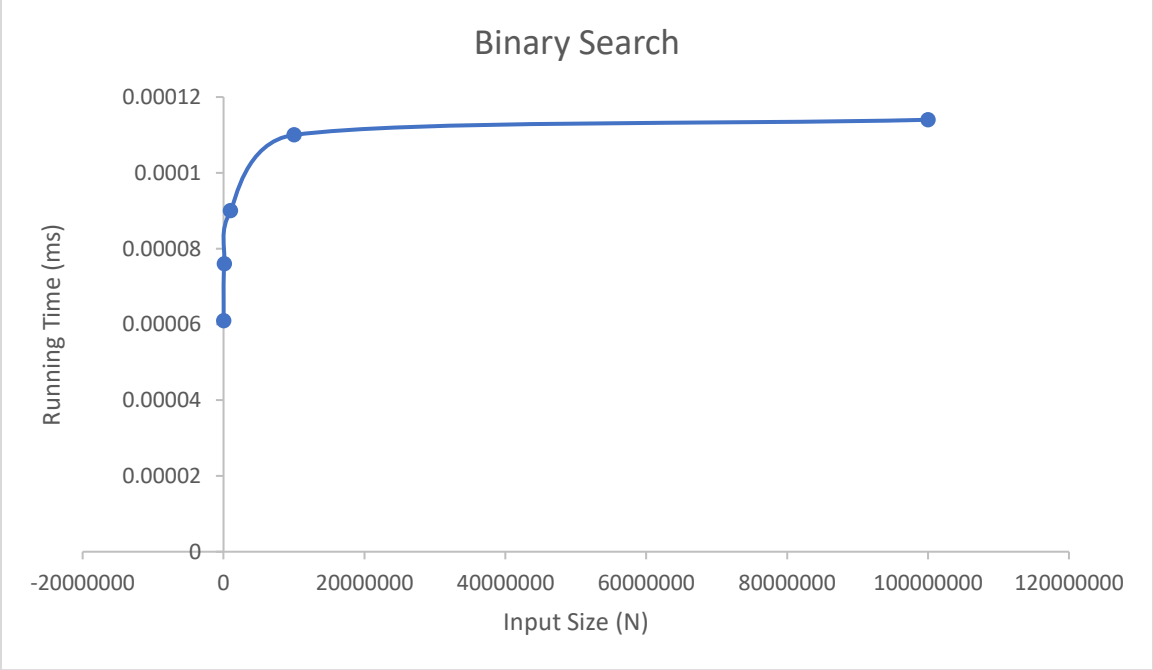




Key is at 50% mark

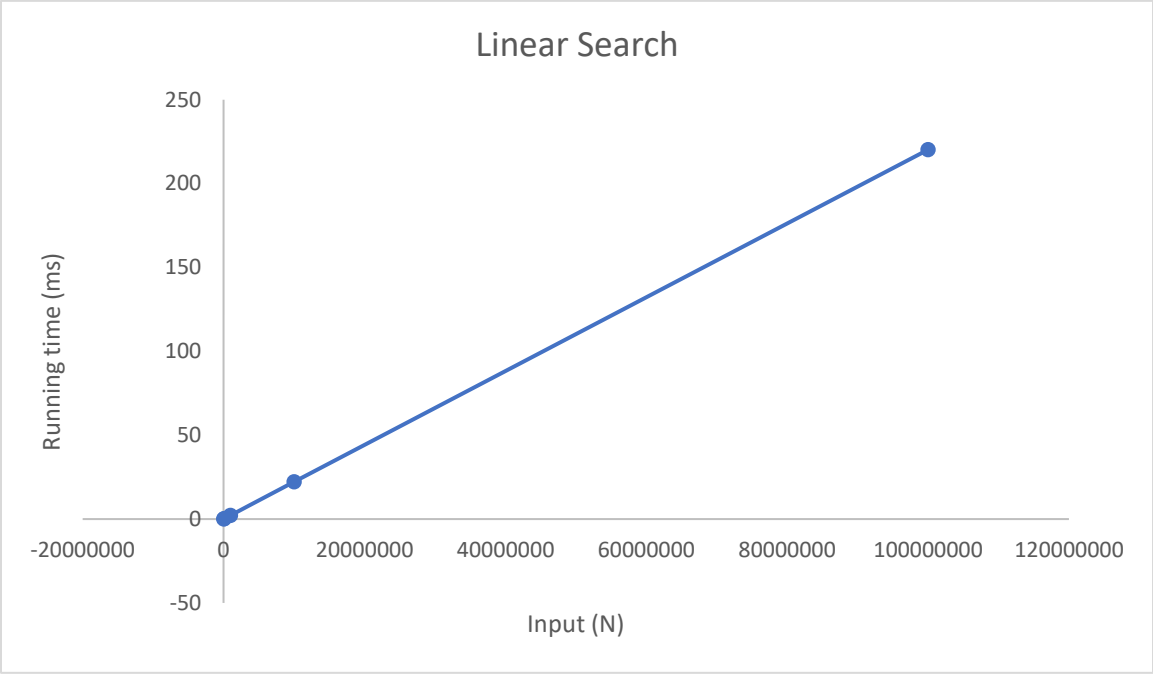
Input Size (N)	Linear, Time (ms)	Binary, Time (ms)
10000	0.012	0.000061
100000	0.131	0.000076
1000000	1.263	0.000090
10000000	12.75	0.000110
100000000	126.1	0.000114

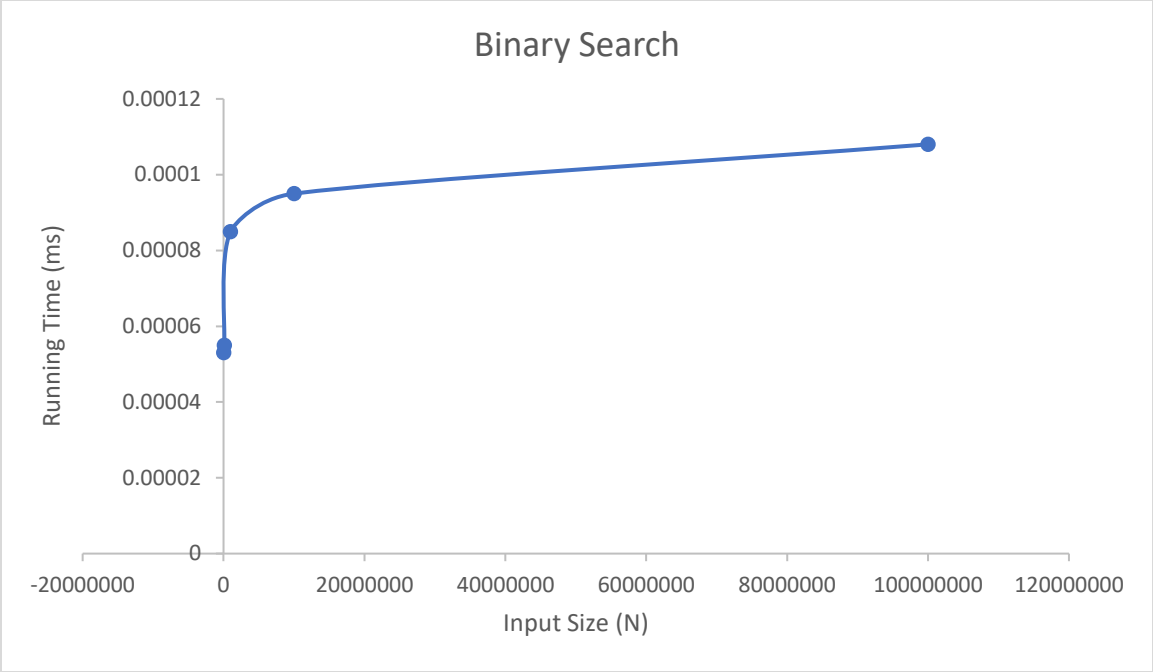




Key is at 90% mark

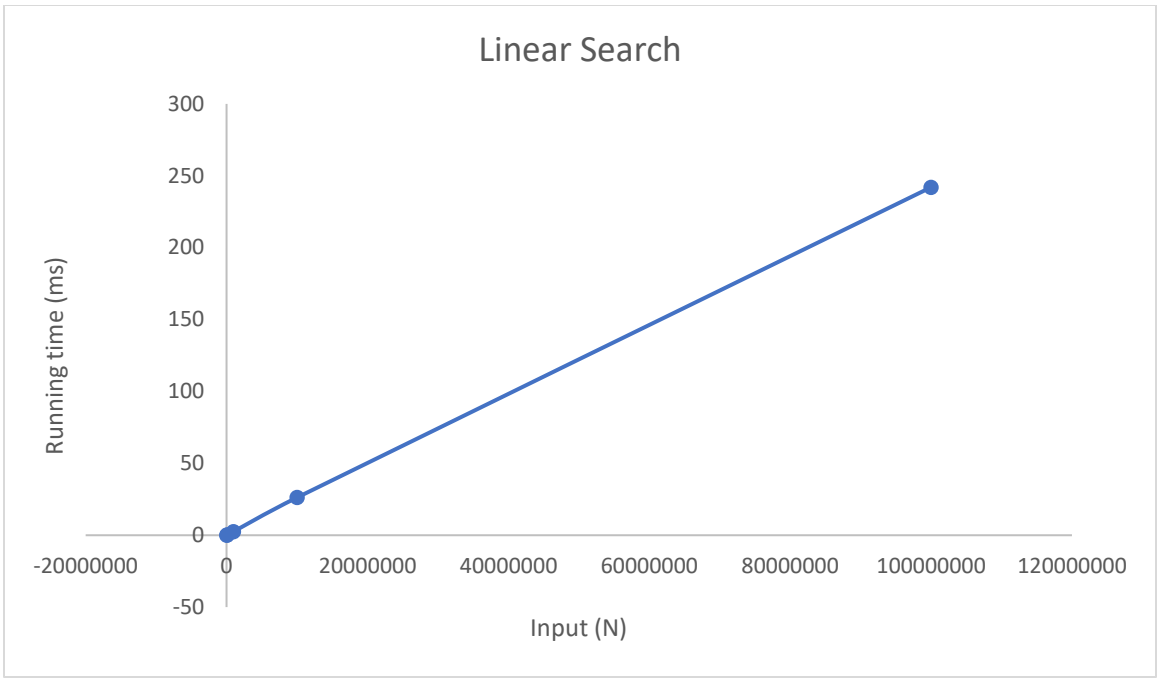
Input Size (N)	Linear, Time (ms)	Binary, Time (ms)
10000	0.022	0.000053
100000	0.224	0.000055
1000000	2.195	0.000085
10000000	22.18	0.000095
100000000	220.2	0.000108

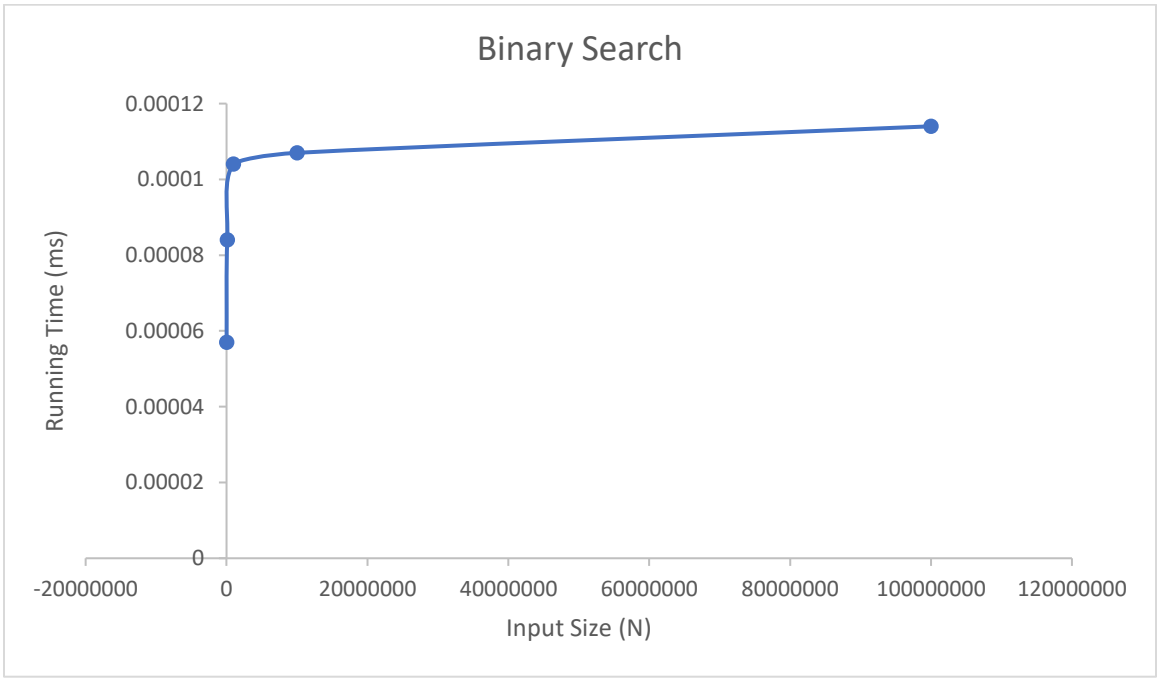




Key is not in the array

Input Size (N)	Linear, Time (ms)	Binary, Time (ms)
10000	0.0253	0.000057
100000	0.255	0.000084
1000000	2.51	0.000104
10000000	26.2	0.000107
100000000	242	0.000114


Linear Search
300
250
200
150
100
50
0
-50
Running time (ms)
-20000000
0
20000000
40000000
60000000
80000000
100000000
120000000
Input (N)


Binary Search
0.00012
0.0001
0.00008
0.00006
0.00004
0.00002
0
Running Time (ms)
-20000000
0
20000000
40000000
60000000
80000000
100000000
120000000
Input Size (N)

Experimental Analysis

	Linear Search	Binary Search
Best Case	Key at 1% mark	All 3 are same
Average Case	Key is at 50% mark	All 3 are same
Worst Case	Key is not in array	All 3 are same

Computer Specifications:

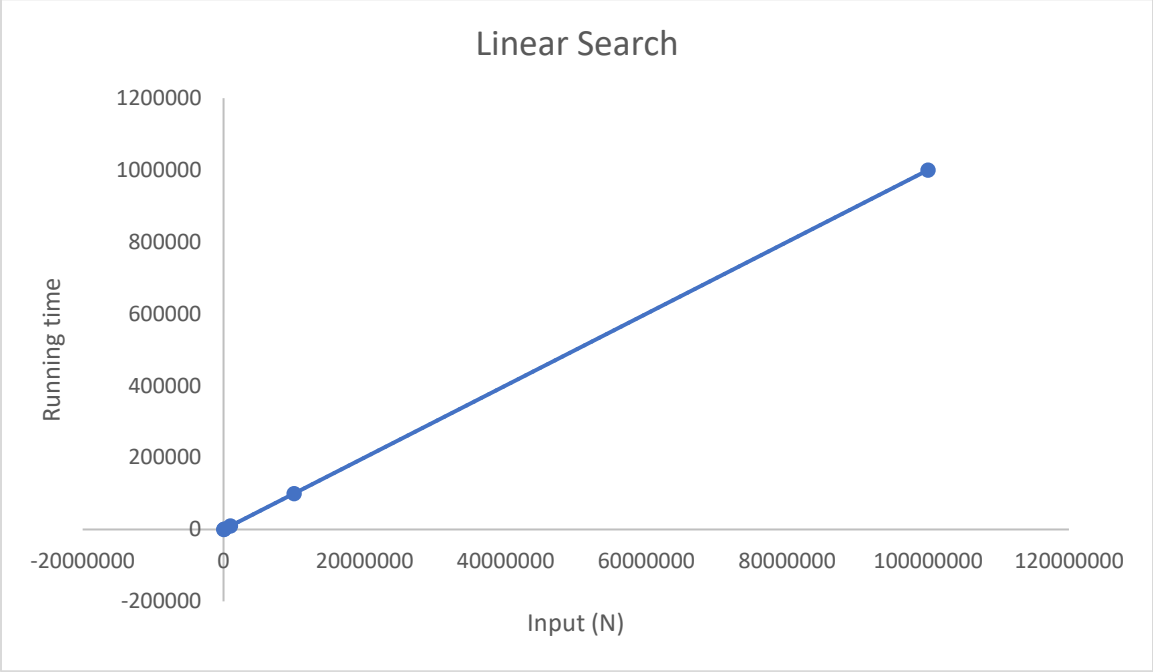
Processor: Intel(R) Core(TM) i7-7700HQ CPU @ 2.80GHz, 2808 MHz, 4 Core(s), 8 Logical Processor(s)

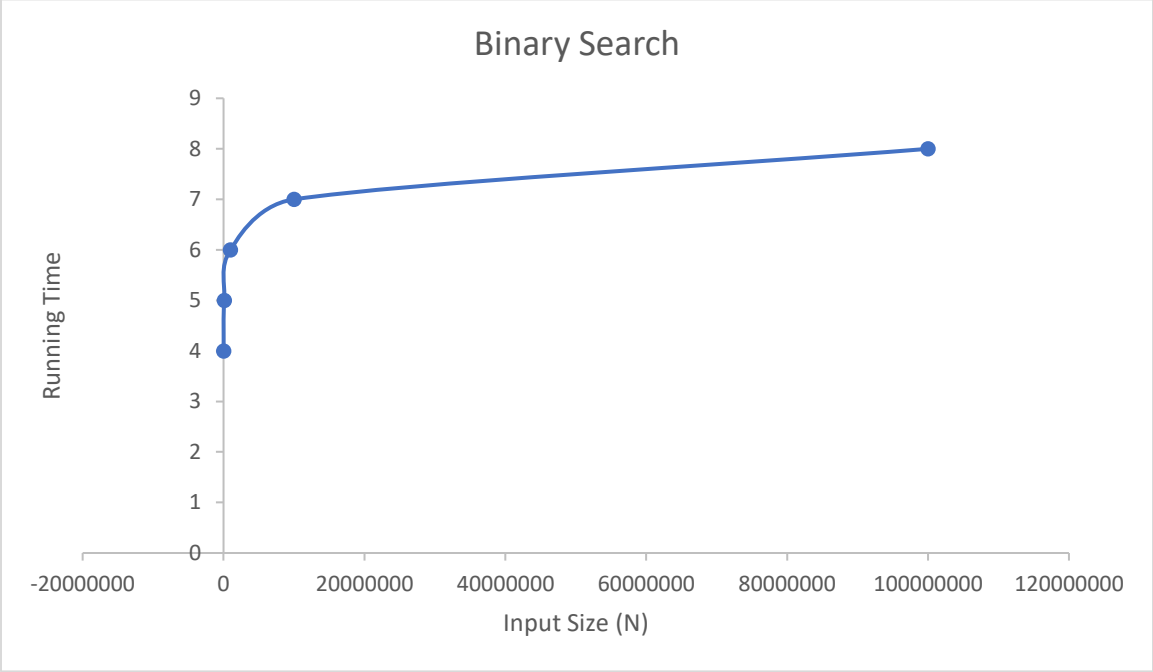
RAM: 8GB

OS Name: Microsoft Windows 10 Home Single Language

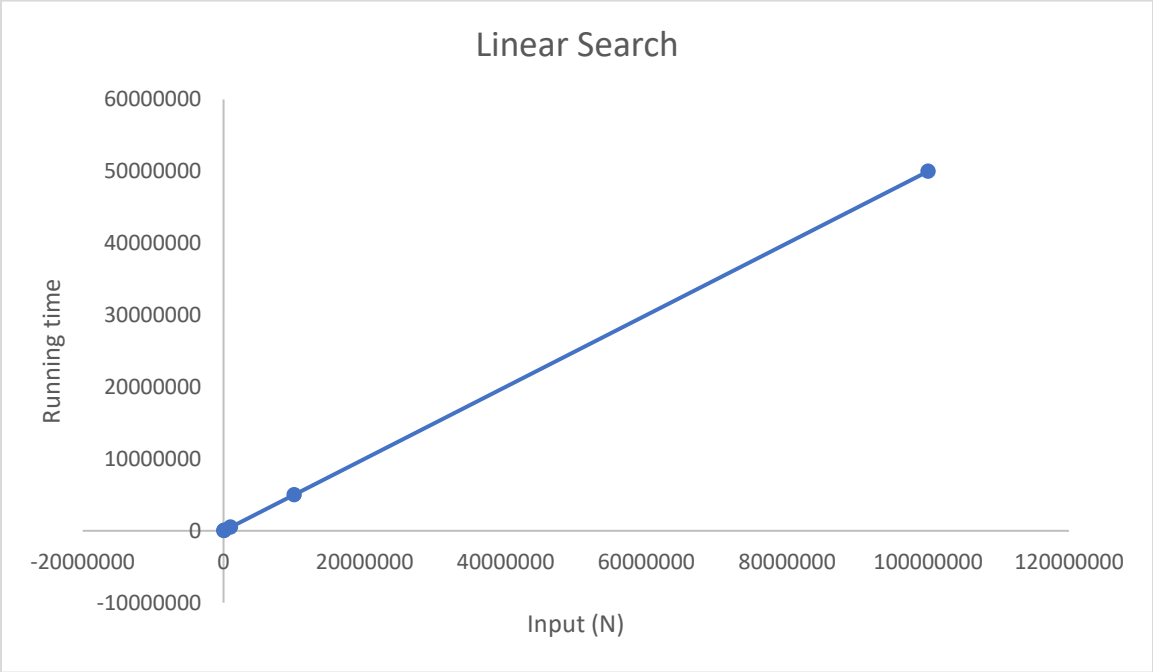
Theoretical Analysis

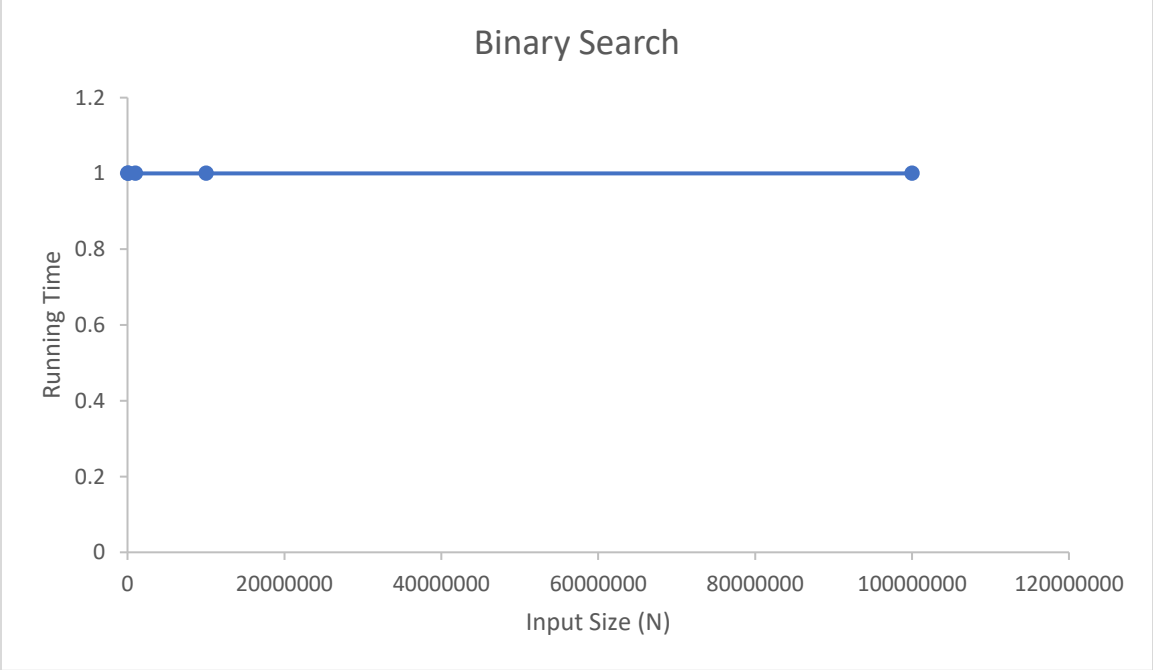
Key at 1% mark



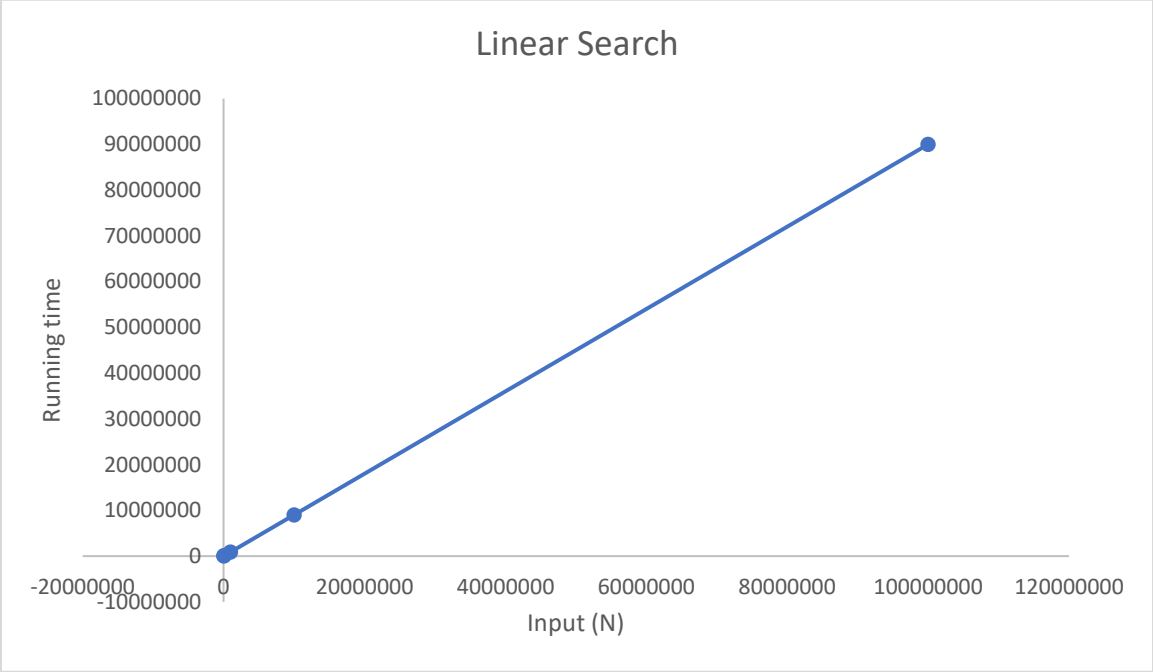


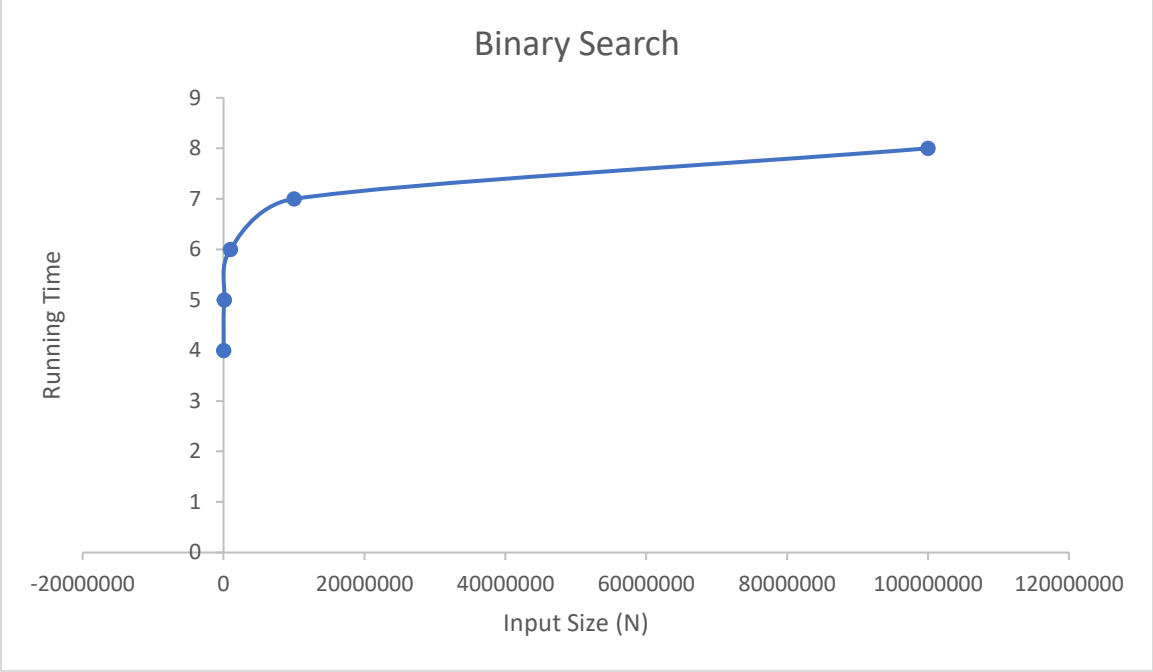
Key is at 50% mark



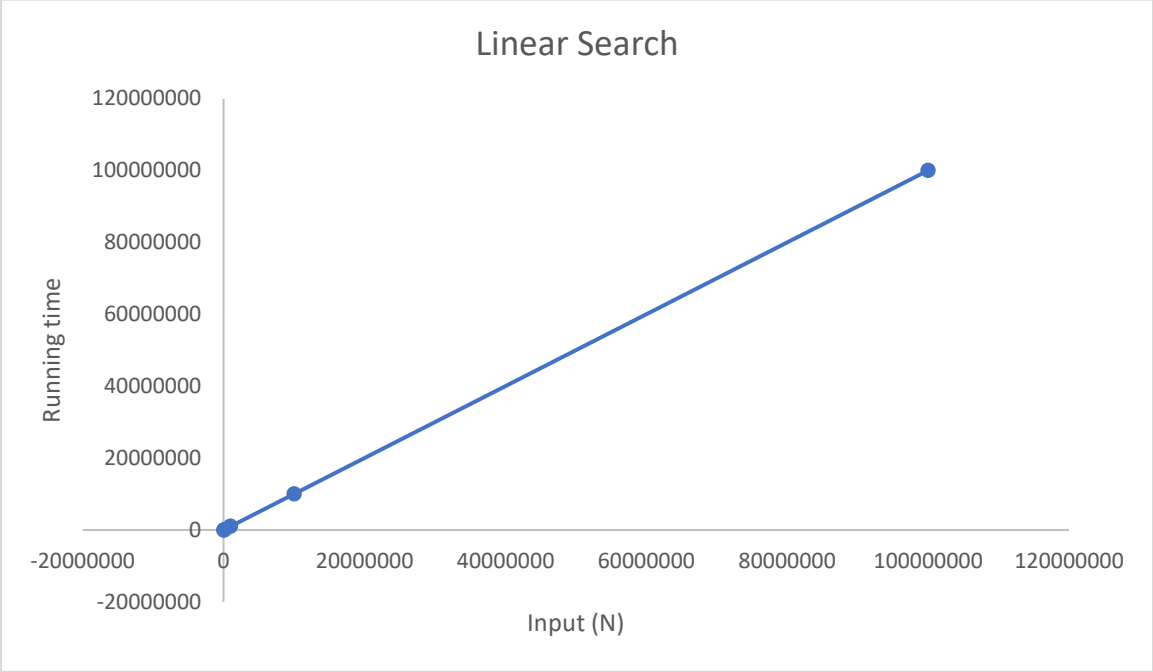


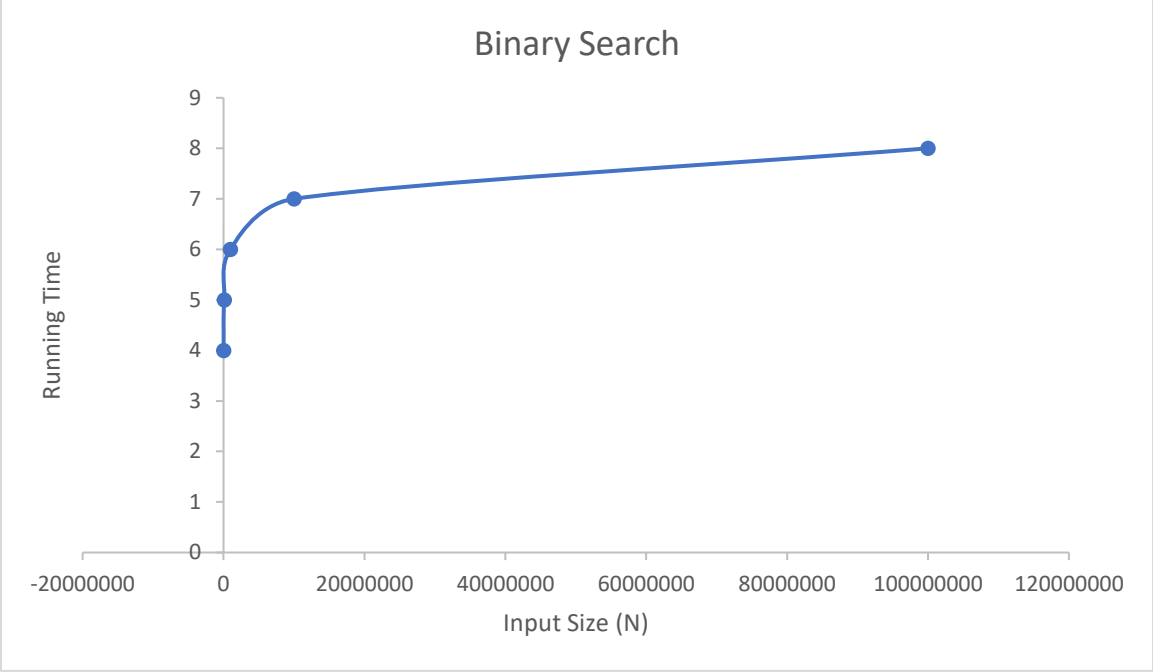
Key is at 90% mark





Key is not in the array





Theoretical Analysis

	Linear Search	Binary Search
Best Case	Key at 1% mark	Key is at 50% mark
Average Case	Key is at 50% mark	Key is 1% or 90% mark
Worst Case	Key is not in array	Key is not in the array

Conclusion:

The time taken to do a linear search is O(N), if we take a definite value e.g. first value is the value to find, then no matter the size of the input array, the time taken to do the search would remain constant. For all other cases, we can expect the graph to grow linearly due to the fact that the linear search goes through all values one by one. My results differ due to the fact that I took a percentage value for all graphs (Key at 1% mark etc.) and that it why my graph still grows linearly for the case where the key is close to the start of the array.

For a binary search, the best case scenario is when the key is in the middle, then O(1) is the time complexity. But for all other cases, the time taken is O(logN), this is shown by both the graphs of the experimental and theoretical analysis. One error though is that the experimental graph does not show the key in the middle as the best case. This is due to the way our binary search algorithm is implemented. For example, for input size 10000, using the key 5000, the middle value is 5000, but the graph doesn't find it in O(1) time. This is because the binary search takes low as 0 and high as N – 1, which is 9999 in this case. Therefore, mid is calculated to be 4999.5 and rounded down to 4999 and hence, the value of the mid isn't found at time O(1).