

Name: Bishal Sarangkoti

CE III

Roll: 45

Lab Works 1 Link:

- <https://github.com/sarangbishal/COMP-314-Algorithms-and-Complexity-Labworks/tree/master/Lab-1%20Searching>

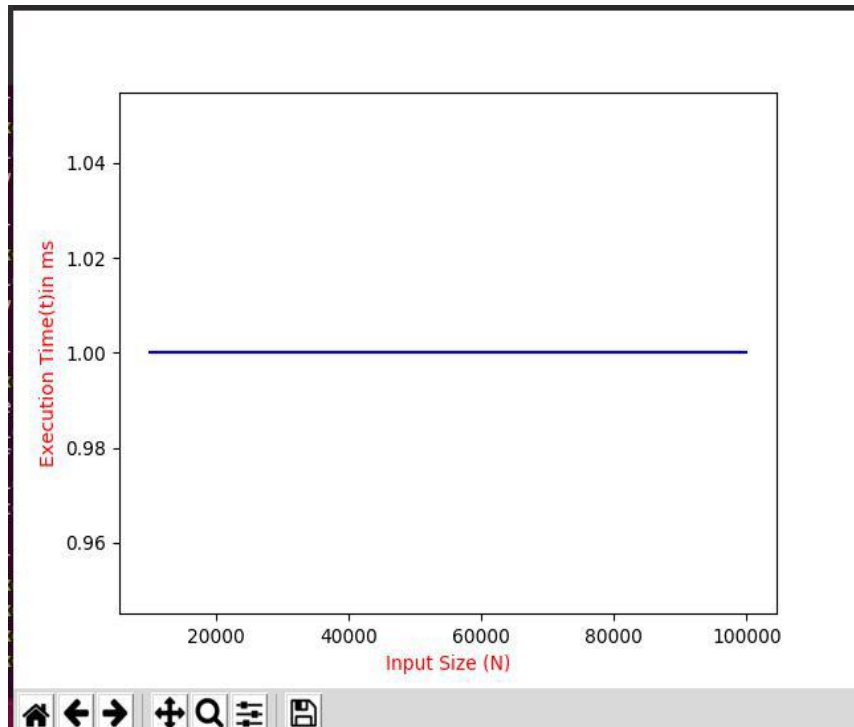
Tasks:

1. In file *binarysearch.py* and *linear_search.py*
2. In file *tester.py* and *stress_test.py*
3. Inside folder *Complexities*
4. By plotting the graph of input size vs total number of comparisons and input size and execution time, we have the following conclusion:

- Linear Search

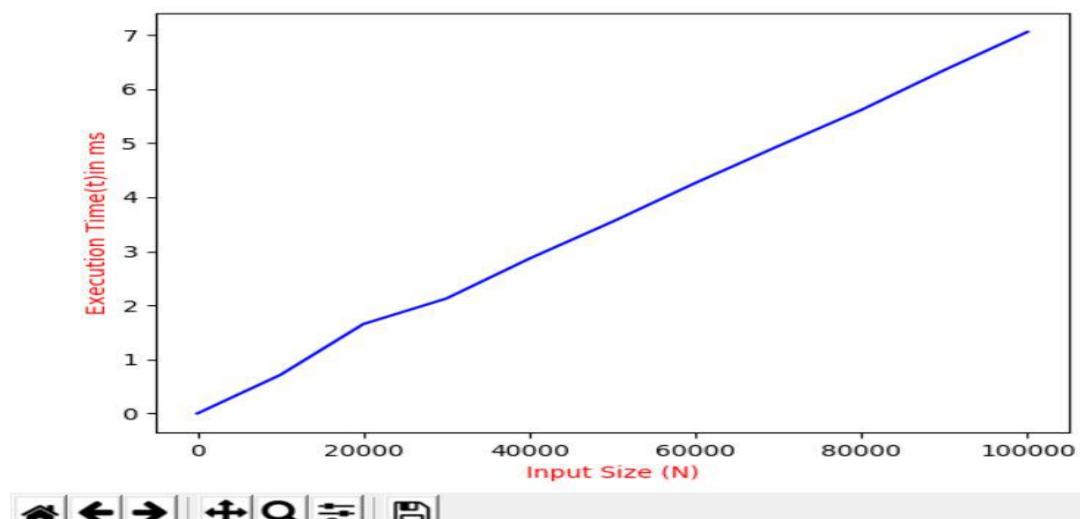
1. **Bestcase: $O(1)$**

For linear search the best case is when the target element lies in the beginning of the array. In this case we can find the element with constant time complexity regardless of input size N .



2. **Worstcase: $O(N)$**

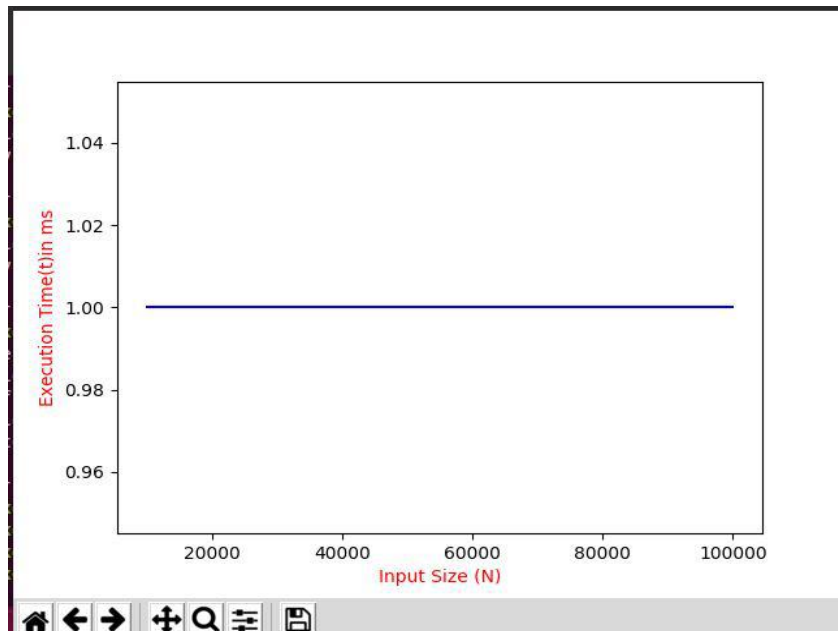
For linear search the worstcase is when the element lies at the end of the list. In this case for the array with size N we have to traverse the whole array once. So the total time complexity of linear search in worst case is $O(N)$



- Binary Search

1. **Bestcase: $O(1)$**

For binary search the bestcase is when the target element lies exactly in the middle of the array. In this case regardless of input size of an array N , we can find the element in 1 comparison. So the total time complexity of binary search in best case is also **$O(1)$** .



2. **Worstcase: $O(\log(N))$**

For binary search the worst case is when the target element lies at the beginning or the end of the array. In this case, we need $\log(N)$ total comparisons for an array of size N .

