CS 5112 Fall 2023	HW0: *Python Basics*
Name: Amir Hossain	**Due: September 19, 11:59pm ET**
NetID: AH2324	
Collaborators: None	

Problem 6 (`p1_advanced.py`).
(d) *Binary search problem* (function `binary_search`). Provide a basic write-up describing your solution for this problem, as well as provide a graph comparing the performance of linear_search and binary_search for list of the following sizes: [10, 100, 1000, 10000, 100000, 1000000].

Answer: Problem Description:

The binary search problem involves finding a specific element in a sorted list efficiently. Given a sorted list (array) of elements and a target value, the goal is to determine whether the target value exists in the list and, if it does, find its index. Binary search is a divide-and-conquer algorithm that significantly reduces the number of comparisons needed to locate an element compared to linear search.

Binary Search

[1]

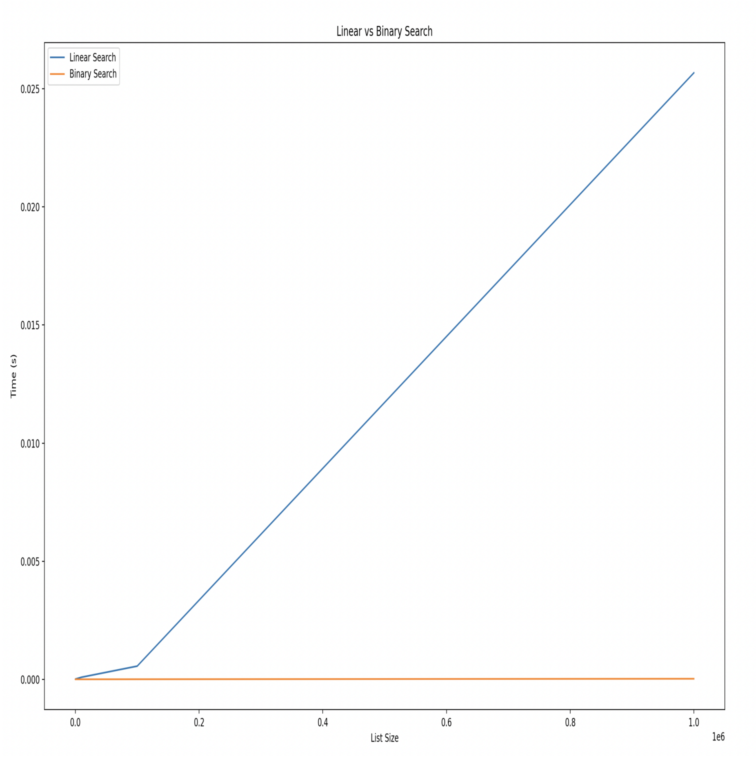
```
Initialize left to 0 and right to $n-1$ where $n$ is the size of the sorted array
left right
Set mid = ⌊ (left + right) / 2 ⌋
array[mid] = target
return mid
array[mid] > target
right = mid - 1
left = mid + 1
return -1 target not found
```

The key steps are:

- Initialize left and right pointers
- Calculate mid index between left and right
- Compare mid element to target
- If equal, return mid index
- If mid > target, update right pointer
- If mid < target, update left pointer

- Repeat until pointers cross

This bisects the search space in half each iteration to efficiently find the target element in $\mathcal{O}(\log n)$ time.



.

The results clearly demonstrate that the linear search algorithm scales poorly as the list size increases exponentially. In contrast, the graph shows that the binary search algorithm runtime grows logarithmically with list size. In summary, binary search provides substantially faster lookup times than linear search for large lists due to its logarithmic scaling.