ASSIGNMENT D2

Binary Search and Building an Index for CS4DS

Submitted By: Ahmed Dider Rahat
Matriculation Number: 916146

Assignment Code: After implementing all the assignment I pushed my code on my Github link.

Answer to the question no. 1

After running the code and observng the documentation I have made the following decisions:-

Case a: list.sort() method only sort the exsiting list and return None. So, while assiginng and sorting a list in a same statement, the original list is sorted but the second list is assigning only the None from list.sort() method.

Case b: In this case, the list is first assigned to another list. As list assignment just assign the reference of one list to other, so list.sort() function sort both of the list.

Case c: In this case, sorted(list) method is used. This method sort the list and return the soterd iiterables instead of sorting the original list.

Answer to the question no. 2

I have implement the code using recursion. So, for the recusrssion process I consider the following assumtions.

1. **Base Conditions:** I assum base conditions are:
    a. if lower index is larger than upper then the element is not found.
    b. if the element is found in the middle of the given range.
2. **Recursive call:** If the middle point of the data is larger than the search element, we search the element into the left part otherwise into right part.

Answer to the question no. 3

Time Complexity of binary search: The binary search algorithm based on divide-and-conquere rule. So, each time its divide the data into two eual parts. So, the time complextiy follow $\log_2 n$ function.

Best case: O(1)

Worst case: O($\log_2 n$)

Average case: O($\log_2 n$)

Answer to the question no. 4

For searching 1000 elements the average and worstcase complexities are given bellow:

	Average case	Worstcase
Linear search	0(n/2) = 500	0(n) = 1000
Binary Search	O($\log_2 n$) = 10	O($\log_2 n$) = 10

Answer to the question no. 5

In python, there is a build-in data strucutre name dictionary which store the data as key-value pair. So, for storing the name as index and original index as value, I use the dictionary.

Answer to the question no. 6

In question-5, I used dictionary. And the space complexity of dictionary is O(n).

Answer to the question no. 7

I implement the code in Question_D2_7.py file.

Answer to the question no. 8

The screen shot of output is given bellow:

Answer to the question no. 9

Yes, python dictionary is an alternative for binary search. Python dictionary used hashmap implementation internally. So, access any key of a dictionary is approximately 0(1).