Name: Vankar Divyesh Kumar

Roll no.: 210101108

Binary Search Visualization Software - Report

Problem Description:

The objective of this software project is to implement a binary search algorithm on a sorted list of elements. The elements can be of any similar data type (integer, float, string), and the software will use Visual Basic and C/C++ to provide a visual representation of the binary search process.

Solution Overview:

User Interface:

- I will design a user-friendly interface for inputting the sorted list and the target element.
- I will implement a visual display of the sorted list to enhance user interaction.
- I will make visualise the binary search algorithm steps using graphics to make the search process transparent.

Binary Search Algorithm:

- I will implement the binary search algorithm to efficiently locate the target element.
- I track and display the step-by-step progress of the binary search, emphasizing elements being compared.

Visualization:

- I will utilized Visual Basic to create a visually appealing representation of the sorted list and binary search process.
- I incorporat graphical elements such as arrows and color changes to illustrate algorithmic steps.
- I will provid a dynamic view of the search process by updating the display in real-time.

Input Validation:

- I will Implement robust input validation to handle invalid user inputs and unsorted lists.

- I will Display appropriate error messages to guide users towards correct inputs.

- I have to handle input validation like if user input the string or integer but as input everything is in text so I will handle this to convert them in ASCII code or completely in Integer.(here for case of float I have to convert completely to float)

Testing:

- I will developed comprehensive test cases to validate the correctness and efficiency of the binary search implementation.

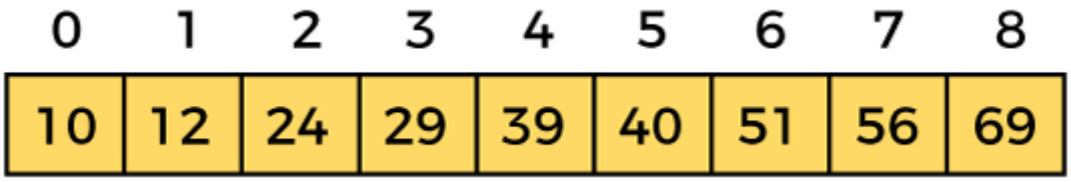
- Ensured the software handles various scenarios, including edge cases.

Visualization Example:

I will display the sorted list horizontally, visually highlighting elements during the binary search. This approach provides a clear representation of the algorithm's steps.

Binary Search Visualization:

Let the elements of array are -



Let the element to search is, **K = 56**

We have to use the below formula to calculate the **mid** of the array -

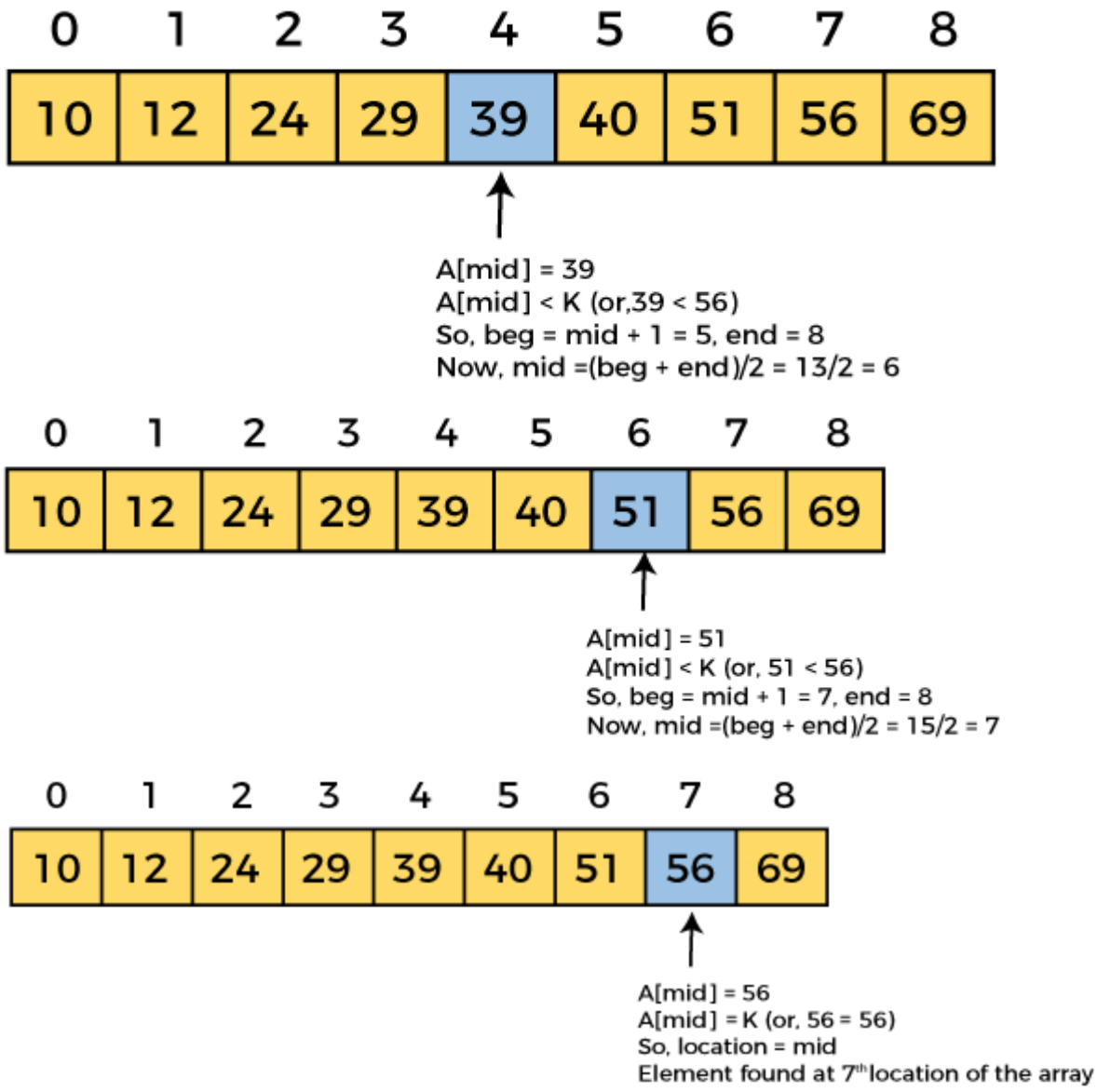
1. mid = (beg + end)/2

So, in the given array -

beg = 0

end = 8

mid = (0 + 8)/2 = 4. So, 4 is the mid of the array.



Now, the element to search is found. So algorithm will return the index of the element matched.

Pseudo Code:

```
function binary_search(A, n, T) is
  L := 0
  R := n − 1
  while L ≤ R do
    m := floor((L + R) / 2)
    if A[m] < T then
      L := m + 1
    else if A[m] > T then
      R := m − 1
    else:
      return m
  return unsuccessful
```

Time Complexity

Case	Time Complexity
Best Case	O(1)
Average Case	O(logn)
Worst Case	O(logn)

- **Best Case Complexity -** In Binary search, best case occurs when the element to search is found in first comparison, i.e., when the first middle element itself is the element to be searched. The best-case time complexity of Binary search is **O(1).**
- **Average Case Complexity -** The average case time complexity of Binary search is **O(logn).**
- **Worst Case Complexity -** In Binary search, the worst case occurs, when we have to keep reducing the search space till it has only one element. The worst-case time complexity of Binary search is **O(logn).**

Space Complexity

Space Complexity	O(1)

- The space complexity of binary search is O(1).

UI looks like this :