Search Element in a Sorted Rotated Array CodeStudio

The problem statement in the given code is to search for a target element in a sorted rotated array. The array can be sorted in ascending order and then rotated any number of times.

Input: nums = [8,9,4,5], target = 3

Output: -1

Input: nums = [4,5,6,7,0,1,2], target = 1

Output: 5

Approach 1: Using Linear Search to find target element.

performs a simple linear search through the array to find the target element. It iterates over each element and checks if it matches the target. If a match is found, the index is returned. If the loop completes without finding the target, -1 is returned.

This approach has a time complexity of O(N), where N is the size of the array, as it needs to traverse the entire array in the worst case.

This approach has a space complexity of O(1) since it does not use any additional data structures that scale with the input size.

Approach 2: Using Binary Search with the help of Pivot/Smallest element index.

The findPivotIndex function is used to find the index of the smallest element in the array, which serves as the pivot. This function uses binary search to locate the pivot element.

Then, in the findElementIndex function, the pivot index is used to determine whether the target element exists in the left or right-sorted halves of the array. Binary search is then applied to find the target element within the appropriate half.

This approach has a time complexity of O(log n) as it uses binary search to narrow down the search space.

This approach has a space complexity of O(1) since it does not use any additional data structures that scale with the input size.

Which Approach is Better:

The second approach, which uses binary search with the pivot index, is more efficient than the linear search approach. Binary search has a better time complexity, especially for large arrays, as it eliminates half of the search space with each iteration. In contrast, linear search needs to traverse the entire array in the worst case.