Strivers-A2Z-DSA-Sheet-main\02.Binary Search\1D Arrays\13.Find_how_many_times_array_is_rotated.cpp

```
/*
QUESTION:
Given an ascending sorted rotated array Arr of distinct integers of size N. The array is right rotated K times. Find the value of K.

Example 1:

Input:
N = 5
Arr[] = {5, 1, 2, 3, 4}
Output: 1
Explanation: The given array is 5 1 2 3 4.
The original sorted array is 1 2 3 4 5.
We can see that the array was rotated
1 times to the right.

APPROACH:
To find the value of K, we can use binary search.
1. Initialize low = 0 and high = N-1, where N is the size of the array.
2. While low < high, calculate mid = low + (high - low) / 2.
3. Check if arr[mid] > arr[n-1].
    - If true, it means the rotation point lies on the right side of mid, so update low = mid + 1.
    - If false, it means the rotation point lies on the left side of mid or mid is the rotation point, so update high = mid.
4. After the loop ends, low will be pointing to the rotation point.
5. Return low as the value of K.

CODE:
*/

int findKRotation(int arr[], int n) {
    int low = 0, high = n - 1;
    while (low < high) {
        int mid = low + (high - low) / 2;
        if (arr[mid] > arr[n - 1])
            low = mid + 1;
        else
            high = mid;
    }
    return low;
}

// TIME COMPLEXITY: O(log n)
// SPACE COMPLEXITY: O(1)

```