Find Rotation Count

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.
```

Example 2:

```
Input: N = 5
Arr[] = {1, 2, 3, 4, 5}
Output: 0
Explanation: The given array is not rotated.
```

Your Task:
Complete the function **findKRotation()** which takes array **arr** and size **n**, as input parameters and returns an integer representing the answer. You don't to print answer or take inputs.

Expected Time Complexity: O(log(N))
Expected Auxiliary Space: O(1)

Constraints:

$1 <= N <= 10^5$

$1 <= Arr_i <= 10^7$

```python
#User function Template for python3
class Solution:
    def findKRotation(self,arr,  n):
        # code here
        lo = 0
        hi = len(arr)-1
        if arr[lo]<=arr[hi]:
            return 0
        while lo<=hi:
```

```
            if arr[lo]<=arr[hi]:
                return 0
            mid = (lo+hi)//2
            if arr[mid]<arr[((mid-1)+n)%n]:
                return mid
            elif arr[mid]>arr[(mid+1)%n]:
                return mid+1
            elif arr[lo]<=arr[mid]:
                lo = mid+1
            elif arr[mid]<=arr[hi]:
                hi = mid-1
        # return mid
```