**Question 4**

Divyansh Mathur, 210357

***Simplifying the question:***

*The problem gives us an array such that it a shifted array version or a sorted array. We need to find the index of the last element in the given array in the sorted array.*

Given Array: A[0], A[1]……. A[i]…..A[n-2],A[n-1]

*We need to find i such that* ***A[i]<A[n-1] and A[i-1]>A[n-1].***

*Our answer would be* ***n-i-1***

**#Assumption made: “** he picked a random number k between 0 and n” {0 and n excluded}{

1. **To design an algorithm of complexity O(log(n)) for Joker to find the value of k**

The algorithm we would use be a modified version of binary search.

We check for middle element. There are three possibility:

1. A[mid]<A[n-1] && A[mid-1]>A[n-1]: in this case we return mid
2. A[mid]<A[n-1] && A[mid-1]<A[n-1]: in this case our ans will be on the right half of our segment so l=mid+1
3. A[mid]>A[n-1] && A[mid-1]>A[n-1]: in this case our ans will be on the left half of our segment so r=mid-1

\*no other case exists because the array is otherwise sorted

**NOTE:**

1. Shifted array version: Given a sorted and rotated array **A[]** of size **n** and a **key**, the task is to find the position of the key
2. All elements are distinct

**Pseudo Code:**



1. **To provide time complexity analysis for this strategy**

At each stage we are checking whether which condition should be followed in constant time

and after each iteration we are dividing the array in 2 equal segments of which we are considering only 1 segment.

Length of array =>

N -> N/2 -> N/4……1

Therefore time complexity is O( log(N))