**Approach : Modified Binary Search ( Hint)**

1. Given array will be unsorted
2. To find index such that number before index must be lesser and number after is also lesser

a b c d e f g

Mid-1 Mid mid+1

Mid = left + (right-left)//2

If arr[mid] > left and > right:

Mid is required index

Otherwise if right > mid:

Move right

Otherwise

Move left

***Edge conditions***

*For terminal elements (0 and n-1) check only right and left neighbours respectively*

*Eg : for nums[0] no left neighbour*

*For nums[n-1] no right neighbour*

**CODE:**

left = 0

right = len(nums)-1

n = len(nums)

# 1 element in array

if n==1:

return 0

while left <= right:

mid = left + (right-left)//2

if mid > 0 and mid < n-1:

if nums[mid-1] < nums[mid] and nums[mid+1] < nums[mid]:

return mid

elif nums[mid+1] > nums[mid]:

left = mid + 1

else:

right = mid - 1

elif mid==0:

if nums[0] > nums[1]:

return 0

else:

return 1

elif mid==n-1:

if nums[n-1] > nums[n-2]:

return n-1

else:

return n-2