# Given a sorted array of size **N** and an integer **K**, find the position(0-based indexing) at which **K** is present in the array using binary search.

import java.io.\*;

import java.util.\*

public class Demo {

public static void main(String[] args) throws IOException {

Scanner sc = new Scanner(System.in);

int T = sc.nextInt();

while (T > 0) {

int n = sc.nextInt();

int arr[] = new int[n];

for (int i = 0; i < n; i++) {

arr[i] = sc.nextInt();

}

int key = sc.nextInt();

Solution g = new Solution();

System.out.println(g.binarysearch(arr, n, key));

T--;

}

}

}

}

class Solution {

int binarysearch(int arr[], int n, int k) {

int low=0;

int high=n-1;

while (low <= high) {

int mid = (high +low) / 2;

if (arr[mid] == k)

return mid;

if (arr[mid] < k)

low = mid + 1;

else

high = mid - 1;

}

return -1;

}

}