**Searching a number**

[array](http://www.practice.geeksforgeeks.org/tag-page.php?tag=array&isCmp=0)

Given an array of N elements and a integer K , return the index of first occurence of K in given array.  
Output -1 if the number is not found in an array.

**Input:**

The first line contains 'T' denoting the number of testcases. Then follows description of testcases.  
Each case begins with a two space separated integer N and K denoting the size of array and the value of K respectively. The next line contains the N space separated integers denoting the elements of array.

**Output:**

For each testcase, print the index of first occurence of given number K.  
Print -1 if the number is not found in array.

**Constraints:**

1<=T<=50  
1<=N<=1000  
1<=K<=100000  
1<=A[i]<=100000

**Example:**

Input :  
2   
5 16  
9 7 2 16 4  
7 98  
1 22 57 47 34 18 66

Output :   
4  
-1

\*\*For More Examples Use Expected Output\*\*

<http://www.practice.geeksforgeeks.org/problem-page.php?pid=237>

#include <iostream>

#include <stdio.h>

#include <set>

#include <map>

#include <vector>

#include <algorithm>

using namespace std;

int main() {

// TODO code application logic here

int t;

scanf("%d", &t);

while(t--) {

int n,x;

scanf("%d %d", &n, &x);

std::vector<int> v;

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

int elem;

scanf("%d", &elem);

v.push\_back(elem);

}

std::vector<int>::iterator i = std::find(v.begin(), v.end(), x) ;

int pos;

if(i != v.end()) {

pos = distance(v.begin(), i);

//cout << pos << endl;

}

if(i == v.end()) {

printf("-1\n");

}else {

printf("%d\n", pos + 1);

}

}

}