**Climbing the Leaderboard**

**locked**

https://d3keuzeb2crhkn.cloudfront.net/s3_pub/hr-avatars/2f8155f1-c82d-4109-943c-3a45ceafb29e/150x150.png**by [Shafaet](https://www.hackerrank.com/Shafaet)**

* [**Problem**](https://www.hackerrank.com/contests/world-codesprint-8/challenges/climbing-the-leaderboard)
* [**Submissions**](https://www.hackerrank.com/contests/world-codesprint-8/challenges/climbing-the-leaderboard/submissions)
* [**Leaderboard**](https://www.hackerrank.com/contests/world-codesprint-8/challenges/climbing-the-leaderboard/leaderboard)
* [**Discussions**](https://www.hackerrank.com/contests/world-codesprint-8/challenges/climbing-the-leaderboard/forum)
* [**Editorial**](https://www.hackerrank.com/contests/world-codesprint-8/challenges/climbing-the-leaderboard/editorial)

Alice is playing an arcade game and wants to climb to the top of the leaderboard. Can you help her track her ranking as she beats each level? The game uses [Dense Ranking](https://www.hackerrank.com/external_redirect?to=https://en.wikipedia.org/wiki/Ranking#Dense_ranking_.28.221223.22_ranking.29), so its leaderboard works like this:

* The player with the highest score is ranked number  on the leaderboard.
* Players who have equal scores receive the same ranking number, and the next player(s) receive the immediately following ranking number.

For example, four players have the scores , , , and . Those players will have ranks , , , and , respectively.

When Alice starts playing, there are already  people on the leaderboard. The score of each player  is denoted by . Alice plays for  levels, and we denote her total score after passing each level  as . After completing each level, Alice wants to know her current rank.

You are given an array, , of *decreasing* leaderboard scores, and another array, , of Alice's cumulative scores for each level of the game. You must print  integers. The  integer should indicate the current rank of alice after passing the  level.

**Input Format**

The first line contains an integer, , denoting the number of players already on the leaderboard.   
The next line contains  space-separated integers describing the respective values of .   
The next line contains an integer, , denoting the number of levels Alice beats.   
The last line contains  space-separated integers describing the respective values of .

**Constraints**

* for
* for
* The existing leaderboard, , is in *descending* order.
* Alice's scores are cumulative, so  is in *ascending* order.

**Subtask**

For  of the maximum score:

**Output Format**

Print  integers. The  integer should indicate the rank of alice after passing the  level.

**Sample Input 0**

7

100 100 50 40 40 20 10

4

5 25 50 120

**Sample Output 0**

6

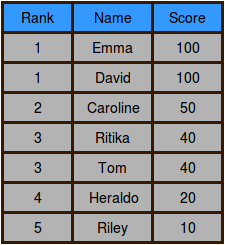
4

2

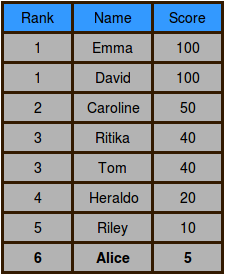
1

**Explanation 0**

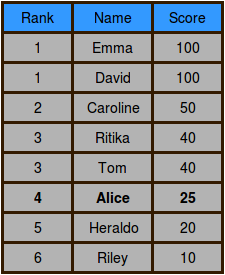
Alice starts playing with  players already on the leaderboard, which looks like this:



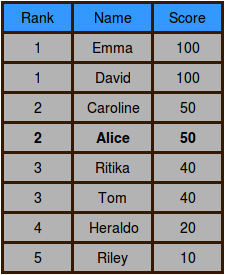
After Alice finishes level , her score is  and her ranking is :



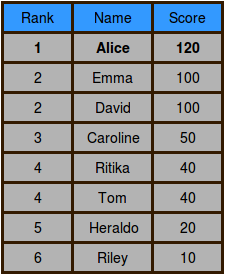
After Alice finishes level , her score is  and her ranking is :



After Alice finishes level , her score is  and her ranking is tied with Caroline at :



After Alice finishes level , her score is  and her ranking is :



<https://www.hackerrank.com/contests/world-codesprint-8/challenges/climbing-the-leaderboard>

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Climbing\_the\_Leaderboard

{

class Program

{

static void Main(string[] args)

{

Convert.ToInt32(Console.ReadLine());

int[] scores = Console.ReadLine().Split(' ').Select(x => Convert.ToInt32(x)).Distinct().ToArray();

int alice\_score\_count = Convert.ToInt32(Console.ReadLine());

int[] alice\_scores = Console.ReadLine().Split(' ').Select(y => Convert.ToInt32(y)).ToArray();

int last\_rank\_idx = scores.Length;

for (int i = 0; i < alice\_score\_count; i++)

{

last\_rank\_idx = GetScorePlacement(scores, last\_rank\_idx, alice\_scores[i]);

Console.WriteLine(last\_rank\_idx + 1);

}

}

private static int GetScorePlacement(int[] scores, int start, int val)

{

while (start != 0 && scores[start - 1] <= val)

{

start--;

}

return start;

}

}

}

-----------------------------

using System;

using System.Collections.Generic;

using System.IO;

using System.Linq;

class Solution

{

static void Main(String[] args)

{

int n = Convert.ToInt32(Console.ReadLine());

string[] scores\_temp = Console.ReadLine().Split(' ');

int[] scores = Array.ConvertAll(scores\_temp, e => int.Parse(e));

int m = Convert.ToInt32(Console.ReadLine());

string[] alice\_temp = Console.ReadLine().Split(' ');

int[] alice = Array.ConvertAll(alice\_temp, e => int.Parse(e));

scores = scores.Distinct().ToArray();

foreach (var v in alice)

{

int left = 0;

int right = scores.Length - 1;

while (left <= right)

{

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

if (v < scores[mid])

left = mid + 1;

else

right = mid - 1;

}

Console.WriteLine(left + 1);

}

}

}

using System;

using System.Linq;

class Solution

{

static void Main(String[] args)

{

int n = int.Parse(Console.ReadLine());

var scores = Console.ReadLine().Split(' ').Distinct().Select(x => int.Parse(x)).ToList();

int m = int.Parse(Console.ReadLine());

var alice = Console.ReadLine().Split(' ').Select(x => int.Parse(x)).ToList();

int rank = scores.Count;

for (int i = 0; i < alice.Count; i++)

{

for (int j = rank - 1; j >= 0; j--)

{

if (alice[i] >= scores[j])

{

rank = j;

continue;

}

break;

}

Console.WriteLine(rank + 1);

}

}

}

using System;

using System.Collections.Generic;

using System.IO;

using System.Linq;

class Solution

{

static void Main(String[] args)

{

int n = Convert.ToInt32(Console.ReadLine());

string[] scores\_temp = Console.ReadLine().Split(' ');

int[] scores = Array.ConvertAll(scores\_temp, Int32.Parse);

int m = Convert.ToInt32(Console.ReadLine());

string[] alice\_temp = Console.ReadLine().Split(' ');

int[] alice = Array.ConvertAll(alice\_temp, Int32.Parse);

List<int> score = new List<int>();

score.Add(scores[0]);

for (int i = 1; i < n; i++)

{

if (scores[i] < scores[i - 1])

{

score.Add(scores[i]);

}

}

score.Reverse();

//for(int i=0;i<score.Count;i++)

//Console.WriteLine(score[i] + " " + count[i]);

int bs;

for (int i = 0; i < m; i++)

{

bs = score.BinarySearch(alice[i]);

if (bs >= 0)

Console.WriteLine(score.Count - bs);

else

Console.WriteLine(score.Count + 2 + bs);

}

}

}

#define \_CRT\_SECURE\_NO\_WARNINGS

#pragma comment(linker, "/STACK:67108864")

#include <iostream>

#include <sstream>

#include <cstdio>

#include <cstdlib>

#include <cmath>

#include <string>

#include <vector>

#include <queue>

#include <stack>

#include <map>

#include <set>

#include <algorithm>

#include <functional>

#include <numeric>

#include <memory.h>

#include <time.h>

using namespace std;

typedef long long LL;

int n, m;

int main()

{

scanf("%d", &n);

set<int> s;

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

{

int x;

scanf("%d", &x);

s.insert(x);

}

scanf("%d", &m);

vector<int> v(s.begin(), s.end());

for (int i = 0; i < m; ++i)

{

int x;

scanf("%d", &x);

int pos = upper\_bound(v.begin(), v.end(), x) - v.begin();

printf("%d\n", (int)v.size() - pos + 1);

}

return 0;

}

#define pb push\_back

#define sqr(x) (x)\*(x)

#define sz(a) int(a.size())

#define reset(a,b) memset(a,b,sizeof(a))

#define oo 1000000007

using namespace std;

typedef pair<int,int> pii;

typedef long long ll;

const int maxn=200007;

int n,m,a[maxn],r[maxn];

int main(){

// freopen("input.txt","r",stdin);

cin>>n;

for(int i=1; i<=n; ++i) cin>>a[i];

cin>>m;

r[1]=1;

for(int i=2; i<=n; ++i){

if(a[i]==a[i-1]) r[i]=r[i-1];

else r[i]=r[i-1]+1;

}

int x=n;

r[0] = 0;

for(int i=1; i<=m; ++i){

int v;

cin>>v;

while(x>=1 && a[x] < v) --x;

if(x>0 && a[x]==v) cout<<r[x]<<endl;

else cout<<r[x]+1<<endl;

}

}

#include <algorithm>

#include <bitset>

#include <cassert>

#include <climits>

#include <cmath>

#include <complex>

#include <cstdio>

#include <cstdlib>

#include <cstring>

#include <ctime>

#include <deque>

#include <fstream>

#include <functional>

#include <iostream>

#include <limits>

#include <list>

#include <map>

#include <numeric>

#include <queue>

#include <random>

#include <set>

#include <sstream>

#include <stack>

#include <string>

#include <tuple>

#include <vector>

#include <unordered\_map>

#include <unordered\_set>

#include <utility>

using namespace std;

int main()

{

unsigned T, x;

cin >> T;

set<unsigned> s;

for (unsigned it = 0; it < T; ++it)

{

cin >> x;

s.insert(x);

}

vector<unsigned> v;

for (auto xx : s)

{

v.push\_back(xx);

}

cin >> T;

unsigned cs = 0;

for (unsigned it = 0; it < T; ++it)

{

cin >> x;

cs += x;

auto it1 = upper\_bound(v.begin(), v.end(), x);

cout << (v.end() - it1) + 1 << endl;

}

return 0;

}