**Speed**

Attempted by: **3264**

/

Accuracy: **90%**

/

Maximum Score: **20**

/

27 Votes

Tag(s):

Algorithms, Easy, Implementation

**PROBLEM**

**EDITORIAL**

**MY SUBMISSIONS**

**ANALYTICS**

Rash is known about his love for racing sports. He is an avid Formula 11 fan. He went to watch this year's Indian Grand Prix at New Delhi. He noticed that one segment of the circuit was a long straight road. It was impossible for a car to overtake other cars on this segment. Therefore, a car had to lower down its speed if there was a slower car in front of it. While watching the race, Rash started to wonder how many cars were moving at their maximum speed. Formally, you're given the maximum speed of NN cars in the order they entered the long straight segment of the circuit. Each car will prefers to move at its maximum speed. If that's not possible because of the front car being slow, it might have to lower its speed. It still moves at the fastest possible speed while avoiding any collisions. For the purpose of this problem, you can assume that the straight segment is infinitely long. Count the number of cars which were moving at their maximum speed on the straight segment.

Input

The first line of the input contains a single integer TT denoting the number of test cases to follow. Description of each test case contains 22 lines. The first of these lines contain a single integer NN, the number of cars. The second line contains NN space separated integers, denoting the maximum speed of the cars in the order they entered the long straight segment.

Output

For each test case, output a single line containing the number of cars which were moving at their maximum speed on the segment.

Constraints

1≤T≤1001≤T≤100  
1≤N≤1051≤N≤105   
1≤speed≤1091≤speed≤109

**SAMPLE INPUT**

3

1

10

3

8 3 6

5

4 5 1 2 3

**SAMPLE OUTPUT**

1

2

2

**Time Limit:**3.0 sec(s) for each input file.

**Memory Limit:**256 MB

**Source Limit:**1024 KB

**Marking Scheme:**Marks are awarded when all the testcases pass.

**Allowed Languages:**C, C++, Clojure, C#, D, Erlang, F#, Go, Groovy, Haskell, Java, Java 8, JavaScript(Rhino), JavaScript(Node.js), Lisp, Lisp (SBCL), Lua, Objective-C, OCaml, Octave, Pascal, Perl, PHP, Python, Python 3, R(RScript), Racket, Ruby, Rust, Scala, Scala 2.11.8, Swift, Visual Basic

<https://www.hackerearth.com/practice/data-structures/arrays/1-d/practice-problems/algorithm/speed-7/#c87089>

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace ConsoleApplication2

{

class Program

{

static void Main(string[] args)

{

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

while (t-- > 0)

{

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

long[] a = Array.ConvertAll(Console.ReadLine().Trim() .Split(' '), e => long.Parse(e));

// int[] a = { 4, 5, 1, 2, 3 };

//int[] a = { 8, 3, 6 };

long[] real\_speed = new long[a.Length];

real\_speed[0] = a[0];

for (int i = 1; i < a.Length; i++)

{

if (a[i] >= real\_speed[i - 1])

{

real\_speed[i] = real\_speed[i - 1];

}

else if (a[i] < real\_speed[i - 1])

{

real\_speed[i] = a[i];

}

}

int cont\_max = 0;

for (int i = 0; i < real\_speed.Length; i++)

{

//Console.Write(real\_speed[i] + " ");

if (a[i] == real\_speed[i])

{

cont\_max++;

}

}

Console.WriteLine(cont\_max);

}

Console.ReadLine();

}

}

}