Badge Progress (Details)

Points: 2252.04 Rank: 4299

The Longest Increasing Subsequence

by vkristijan

Problem Submissions Leaderboard Discussions

An Introduction to the Longest Increasing Subsequence Problems

The task is to find the length of the longest subsequence in a given array of integers such that all elements of the subsequence are sorted in ascending order. For example, the length of the LIS for { 15, 27, 14, 38, 26, 55, 46, 65, 85 } is 6 and the longest increasing subsequence is {15, 27, 38, 55, 65, 85}.

Here's a great Youtube video of a lecture from MIT's Open-Courseware, covering the topic.

Here is one approach which solves this in quadratic time using dynamic programming. A more efficient algorithm which solves the problem in N Log N time is available here.

In this challenge you simply have to find the length of the longest strictly increasing sub-sequence of the given sequence.

Input Format

In the first line of input, there is a single number *N*.
In the next N lines input the value of *a[i]*.

Constraints
$1 \leq N \leq 10^6$
$1 \leq a[i] \leq 10^5$

Output Format

In a single line, output the length of the longest increasing sub-sequence.

Sample Input

```
5
2
7

4
3
8
```

Sample Output

```
3
```

Explanation

{2,7,8} is the longest increasing sub-sequence, hence the answer is 3 (the length of this sub-sequence).

Submissions: 10247
Max Score: 60
Difficulty: Advanced

Rate This Challenge:
☆☆☆☆☆

More

Current Buffer (saved locally, editable)

C#

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

class Solution {


    // Binary search (note boundaries in the caller)
        static int CeilIndex(List<int> v, int l, int r, int key)
        {
            while (r - l > 1)
            {
                int m = l + (r - l) / 2;
                if (v[m] >= key)
                    r = m;
                else
                    l = m;
            }

            return r;
        }

        static int LongestIncreasingSubsequenceLength(List<int> v)
        {
            if (v.Count == 0)
                return 0;

            List<int> tail = new List<int>();
            for (int i = 0; i < v.Count; i++)
            {
                tail.Add(0);
            }
            int length = 1; // always points empty slot in tail

            tail[0] = v[0];
            for (int i = 1; i < v.Count; i++)
            {
                if (v[i] < tail[0])
                    // new smallest value
                    tail[0] = v[i];
                else if (v[i] > tail[length - 1])
                    // v[i] extends largest subsequence
                    tail[length++] = v[i];
                else
                    // v[i] will become end candidate of an existing subsequence or
                    // Throw away larger elements in all LIS, to make room for upcoming grater elements than v[i]
                    // (and also, v[i] would have already appeared in one of LIS, identify the location and replace it)
                    tail[CeilIndex(tail, -1, length - 1, v[i])] = v[i];
            }

            return length;
        }


        static void Main(string[] args)
        {

               //List<int> v = new List<int>(new int[]{ 2, 5, 3, 7, 11, 8, 10, 13, 6 });
               //Console.WriteLine(LongestIncreasingSubsequenceLength(v));
            int n = int.Parse(Console.ReadLine());

            int[] a = new int[n];
            for (int i = 0; i < n; i++)
            {
                a[i] = int.Parse(Console.ReadLine());
            }

            Console.WriteLine(LongestIncreasingSubsequenceLength(a.ToList()));

            Console.ReadLine();
        }


}
```

Line: 5 Col: 1

Upload Code as File □ Test against custom input Run Code Submit Code

Congrats, you solved this challenge!

✓ Test Case #0 ✓ Test Case #1 ✓ Test Case #2

✓ Test Case #3 ✓ Test Case #4 ✓ Test Case #5

✓ Test Case #6 ✓ Test Case #7 ✓ Test Case #8

✓ Test Case #9

Next Challenge

Join us on IRC at #hackerrank on freenode for hugs or bugs.

Contest Calendar | Interview Prep | Blog | Scoring | Environment | FAQ | About Us | Support | Careers | Terms Of Service | Privacy Policy | Request a Feature