最长上升子序列

最长上升子序列LIS(Longest Increasing Subsequence)

让我们举个例子：求 2 7 1 5 6 4 3 8 9 的最长上升子序列。我们定义d(i) (i∈[1,n])来表示以第i个数为结尾的最长上升子序列长度。则有如下：

索引	1	2	3	4	5	6	7	8	9
数据	2	7	1	5	6	4	3	8	9
LIS	1	2	1	2	3	2	2	4	5
子序列	2	2 7	1	2/1 5	2/1 5 6	2/1 4	2/1 3	2/1 5 6 8	2/1 5 6 8 9

最后，我们比较一下全部长度:d(i)=max{d(1),d(2),……,d(i)} 我们可以看出这9个数的LIS为d(9)=5

此外，在这里我们需要明确题目中的术语，它将会对结果产生十分重要的影响。例如：

最长上升子序列：不存在任意两数相等的情况

1、3、4、5、7（正确）

1、3、3、4、5（错误）

最长不下降子序列：可能存在两数相等的情况

1、2、3、3、4、5、7 （正确）

1、2、3、4、5、6、7 （正确）

子序列：原数据中非连续的数字组合，例

1、4、5、7

子串：原数据中必须连续的数字组合，例

2、7、1、5

.

1281：最长上升子序列

题目描述

一个数的序列bi，当b1<b2<...<bS的时候，我们称这个序列是上升的。对于给定的一个序列(a1,a2,...,aN)，我们可以得到一些上升的子序列(ai1,ai2,...,aiK)，这里1≤i1<i2<...<iK≤N。比如，对于序列(1,7,3,5,9,4,8)，有它的一些上升子序列，如(1,7),(3,4,8)等等。这些子序列中最长的长度是4，比如子序列(1,3,5,8)。

你的任务，就是对于给定的序列，求出最长上升子序列的长度。

输入

输入的第一行是序列的长度N(1≤N≤1000)。第二行给出序列中的N个整数，这些整数的取值范围都在0到10000。

输出

最长上升子序列的长度。

输入样例

```
7
1 7 3 5 9 4 8
```

输出样例

```
4
```

解析

直接套用上升子序列公式

```
#include <bits/stdc++.h>

using namespace std;
const int maxn = 1001; //序列长度最长为1000
const int INF = 0x7f7f7f7f; //负21亿
int a[maxn];  //原始数据
int f[maxn];  //动态规划表，表示以i结尾的LIS。
int n;  //原始序列长度
int ans = -INF;  //答案数据

int main() {
    //读入序列的长度
    scanf("%d", &n);
```

```
    //读入序列的数据
    for (int i = 1; i <= n; i++) {
        scanf("%d", &a[i]);
        //将每个序列最初的LIS定义为1
        f[i] = 1;
    }
    //遍历数组中的每一个数字
    for (int i = 1; i <= n; i++) {
        //开始向前寻找
        for (int j = 1; j < i; j++) {
            //找到比它小的数字
            //注意此处的符号，明确是求上升还是不下降
            if (a[j] < a[i]) {
                //比较前一个比他小的数字加1后的长度和自己当前的长度谁更大
                //需要比较是因为可能会出现波谷数，例如 1 3 4 1 4
                f[i] = max(f[i], f[j] + 1);
                //和全局最小值作比较
                ans = max(ans, f[i]);
            }
        }
    }
    printf("%d\n", ans);
    return 0;
}
```