新春吃鸡欢乐赛

Orange

2018 年 2 月 21 日

	地图	吃鸡	落地成盒
题目类型	传统型	传统型	传统型
源程序文件名	compress.cpp	pamffo.cpp	pdd.cpp
输入文件名	(stdin)	pamffo.in	pdd.in
输出文件名	(stdout)	pamffo.out	pdd.out
每个测试点时限	1 秒	1 秒	1 秒
内存限制	512 MiB	512 MiB	512 MiB
测试点数目	20	20	20
每个测试点分值	5	5	5
编译选项	-O2	(默认选项)	(默认选项)
比较答案方式	(默认方式)	(默认方式)	Special Judge

请仔细阅读本页内容

所有测试均**不**打开 -std=c++11 编译选项

比较答案的默认方式为全文比较，忽略文末回车，**但不忽略行末空格**。

评测配置：CPU Intel(R) Core(TM) i5-4210H，2.90GHz，内存 12 GB

题目难度与顺序无关
数据很弱，欢迎水过

1 地图 compress.cpp / stdin / stdout

1.1 题目描述

众而周知，8000m×8000m 的地图相当大，却还是满足不了萌萌哒选手 compress。

compress 说：“我要 xm×ym 的大地图!”既然你无法满足 compress 的地图梦，那你就帮他算算他要的地图到底有多大吧!

1.2 输入

输入包含两行。

第一行输入一个正整数 x，第二行输入一个正整数 y。

1.3 输出

输出包含一行一个正整数 s，$s = xy$。

1.4 样例

1.4.1 输入

```
8000
8000
```

1.4.2 输出

```
64000000
```

1.5 数据规模与约定

对于 5% 的数据，xy 在 int 范围内。

对于 10% 的数据，xy 在 long long 范围内。

对于 20% 的数据，x, y 在 long long 范围内。

对于 50% 的数据，$x, y \leq 10^{5000}$。

对于 100% 的数据，$x, y \leq 10^{80000}$。

1.6 提示

stdin / stdout 表示使用标准输入输出。

评测开启 -O2 编译选项，这意味着你可以认为评测机的运行速度比你使用的计算机快不少。

2 吃鸡 pamffo.cpp / pamffo.in / pamffo.out

2.1 题目描述

众而周知，每个人都想吃鸡。萌萌哒选手 pamffo 决定出来提需求：“我们想知道每个人吃了几次鸡!”
于是这个疯狂的需求就交给你完成：你要回答若干次询问。

2.2 输入

首先输入一行一个整数 n。
接下来有 n 行指令，有两种形式：

指令 1：1 [name] [time]
指令 2：2 [name] [qtime] [time]

指令 1 表示：在 time 时刻，一位昵称为 name 的人吃鸡了。
指令 2 表示：在 time 时刻，有人询问昵称为 name 的人在 qtime 时刻以及 qtime 之前的吃鸡次数。

其中，name 是一个长度不超过 100 的由小写字母组成的非空字符串，time 和 qtime 是在 int 范围内的正整数。
保证 time 递增，保证 qtime $\leq$ time。

2.3 输出

对于每个指令 2，输出一行一个整数，表示吃鸡次数。
可能会询问从未吃过鸡的人，此时请输出 0。

2.4 样例

2.4.1 输入

```
6
2 db 100 100
1 pamffo 101
1 db 102
2 db 100 103
2 db 102 104
2 db 101 105
```

2.4.2 输出

```
0
0
1
0
```

2.4.3 解释

没什么好解释的。

2.5 数据规模与约定

对于 30% 的数据，$n \leq 20$。
对于另外 30% 的数据，昵称为 short 范围内的正整数。
对于 100% 的数据，$n \leq 10^5$。

3 落地成盒 pdd.cpp / pdd.in / pdd.out

3.1 题目描述

众而周知，落地成盒是件很痛苦的事情，然而对于萌萌哒选手 pdd（不是 PDD）来说，只要有一种特殊的技巧，就能避免落地成盒。

这个技巧便是：不停地往前跑，永远不要回头，永远不要往低处走。尴尬的是，当不能再往高处前进时，pdd 还是不能避免成盒的命运。pdd 希望知道，自己最多能跑到多少地方。

整个地图可以抽象为长度为 n 的**正整数**序列 $\{a_i\}$，代表每个地方的高度。假设 pdd 在位置 i 处，那么 pdd 只能跑到 j 处（$i < j$，$a_i \leq a_j$）。由于游戏有随机性，因此 pdd 不仅想知道他最多可能跑到多少地方，还想知道从每个地方出发，他最多能跑到多少地方。

一句话题意：求出序列 $\{a_i\}$ 的最长**不下降**子序列的长度，以及以每个位置开头的**不下降**子序列的最长长度。

3.2 输入

第一行输入一个整数 n，表示序列长度。

接下来一行输入 n 个整数，表示序列中的每个数。

3.3 输出

第一行输入一个整数 n，表示序列的长度。

接下来一行输入 n 个整数，表示以每个位置开头的**不下降**子序列的最长长度。

3.4 Special Judge

本题使用 Special Judge。

如果你完成了第一个任务，即第一行输出正确，你将获得 40% 的分数。

如果你完成了第二个任务，即第二行输出正确，你将获得 60% 的分数。

如果你能完成第二个任务，但不能完成第一个任务，请在第一行任意输出一个整数，否则将会错判。

3.5 样例

3.5.1 输入

```
6
1 4 2 8 5 7
```

3.5.2 输出

```
4
4 3 3 1 2 1
```

3.5.3 解释

从 $a_1 = 1$ 出发，最长不下降子序列为 1 4 5 7 或者 1 2 5 7。
从 $a_2 = 4$ 出发，最长不下降子序列为 4 5 7。
从 $a_3 = 2$ 出发，最长不下降子序列为 2 5 7。
从 $a_4 = 8$ 出发，最长不下降子序列为 8。
从 $a_5 = 5$ 出发，最长不下降子序列为 5 7。
从 $a_6 = 7$ 出发，最长不下降子序列为 7。

3.6 数据规模与约定

对于 20% 的数据，$n \leq 20$。
对于 50% 的数据，$n \leq 5 \times 10^3$。
对于 100% 的数据，$n \leq 10^6$，$a_i \in \text{int}$。

3.7 提示

本题使用 Special Judge，将会忽略行末空格及文末回车。
请注意常数因子带来的程序效率上的影响。