题目翻译

1. 变色龙的爱

内存：537MB 时限：2s 类型：交互题

/usr/bin/g++ -DEVAL -std=gnu++14 -O2 -pipe -static -s -o chameleon stub.cpp chameleon.cpp

题目描述：

在JOI动物园，有2N条变色龙，编号从1到2N。其中，N条变色龙的性别为X，剩下的N条性别为Y。

每条变色龙有它的**初始颜色**。以下是一些已知的初始颜色的有关信息：

•性别为X的N条变色龙初始颜色是不同的

•对于每条性别为X的变色龙，存在着唯一的和它颜色相同的性别为Y的变色龙

现在，JOI动物园正处于新恋爱的时期。每条变色龙都爱上了另外一条变色龙。以下是已知的它们的爱的有关信息：

•每条变色龙都爱着恰好一条和它性别不同的变色龙

•一条变色龙和它所爱的变色龙有不同的初始颜色

•不存在两条变色龙喜欢同一条变色龙

你可以聚集一些变色龙并且召开一次会面。对于参加这次会面的每条变色龙s，假设t是s所爱的变色龙，那么s的**皮肤颜色**由如下方式决定：

•如果t也参加了这次会面，那么s的皮肤颜色是t的初始颜色

•如果t没参加这次会面，那么s的皮肤颜色仍然是s的初始颜色

1. 变色龙的皮肤颜色可能因为不同的会面而改变。对于每一次你组织的会面，你可以计算出参加这次会面的变色龙的皮肤颜色的种类数

通过组织至多20000次会面，你想确定所有的初始颜色相同的变色龙编号对。

写出一个程序，给出变色龙的数量，通过组织至多20000次会面，确定所以初始颜色相同的变色龙的编号对。

实现细节：

你需要提交一个文件。

你提交的文件名应该为chameleon.cpp。它应该实现以下函数，程序应该包含chameleon.h

• void Solve(int N)

这个函数每个测试数据仅调用一次

参数N是题面中的N，变色龙中性别为X的数量。

你可以调用如下函数：

⋆ int Query(const std::vector<int> &p)

你可以通过调用这个函数来组织变色龙的会面

⋄ 参数p是参加这次会面的变色龙的列表

⋄ 返回值是参加这次会面的变色龙的皮肤颜色种类数

⋄每一个p中的元素应该是一个1至2N之间的整数。如果这个条件不满足，你的程序会提示错误信息Wrong Answer [1].

⋄ p中的元素应该不同，如果这个条件不满足，你的程序会提示错误信息Wrong Answer [2].

⋄ 你的程序不能调用这个函数超过20000次，如果这个条件不满足，你的程序会提示错误信息 Wrong Answer [3].

⋆ void Answer(int a, int b)

通过调用这个函数，你回答一对有相同初始颜色的变色龙编号

⋄ 参数a和b表示变色龙a和变色龙b的初始颜色相同。

⋄ 必须满足 1 ≤ a ≤ 2N and 1 ≤ b ≤ 2N. 如果这个条件不满足，你的程序会提示错误信息Wrong Answer [4].

⋄ 你的函数传入的参数a和b至多只能在一个程序里出现1次，如果这个条件不满足，你的程序会提示错误信息 Wrong Answer [5].

⋄ a和b的初始颜色必须相同如果这个条件不满足，你的程序会提示错误信息Wrong Answer [6].

⋄ 你的程序应该调用这个函数恰好N次，如果Solve函数结束时调用的次数不足N次，你的程序会提示错误信息Wrong Answer [7].

重要提示：

• 你的程序可以实现其他函数作为内部使用，或者使用全局变量

• 你的程序禁止使用标准输出和输出。你的程序禁止和其他文件以任何方式进行交互。然而，你的程序允许输出调试信息到标准错误流中。

编译运行：

你可以从竞赛网站上下载一个存档文件，里面包含了样例评测机来测试你的程序。存档文件同时包含了一个你应该提交的程序的样例源文件

样例评测机是文件grader.cpp. 为了测试你的程序，把grader.cpp, chameleon.cpp,

chameleon.h 放在同一路径下，并且运行如下指令来编译你的程序

g++ -std=gnu++14 -O2 -o grader grader.cpp chameleon.cpp

当编译完成后，会生成可执行文件 grader .

注意实际的评分程序和样例评分程序不同。样例评分程序会以单进程执行，从标准输入读入数据并把结果输出到标准输出

样例评测机的输入：

N

Y1 Y2 ... Y2N

C1 C2 ... C2N

L1 L2 ... L2N

Yi (1 ≤ i ≤ 2N) 是变色龙i的性别，0表示X，1表示Y

Ci (1 ≤ i ≤ 2N) 是变色龙i的初始颜色，整数且属于1至N之间。

Li (1 ≤ i ≤ 2N) 是变色龙i喜欢的变色龙的编号。

样例评测机的输出：

当程序成功结束时，样例评测机将以下信息输出到标准输出中（引号中的内容）

• 如果你的程序判断是正确的，输出“Accepted: 100”.

• 如果你的程序错误，输出“Wrong Answer [1]”.

如果你的程序有多种错误，仅输出其中的一种错误信息

数据限制：

• 2 ≤ N ≤ 500.

• 0 ≤ Yi ≤ 1 (1 ≤ i ≤ 2N).

• 1 ≤ Ci ≤ N (1 ≤ i ≤ 2N).

• For each j (1 ≤ j ≤ N), there exists a unique i (1 ≤ i ≤ 2N) satisfying Yi = 0 and Ci = j.

• For each j (1 ≤ j ≤ N), there exists a unique i (1 ≤ i ≤ 2N) satisfying Yi = 1 and Ci = j.

• 1 ≤ Li ≤ 2N (1 ≤ i ≤ 2N).

• Yi , YLi (1 ≤ i ≤ 2N).

• Ci , CLi (1 ≤ i ≤ 2N).

• Lk , Ll (1 ≤ k < l ≤ 2N).

子任务：

1. (4 points) LLi = i (1 ≤ i ≤ 2N).

2. (20 points) N ≤ 7.

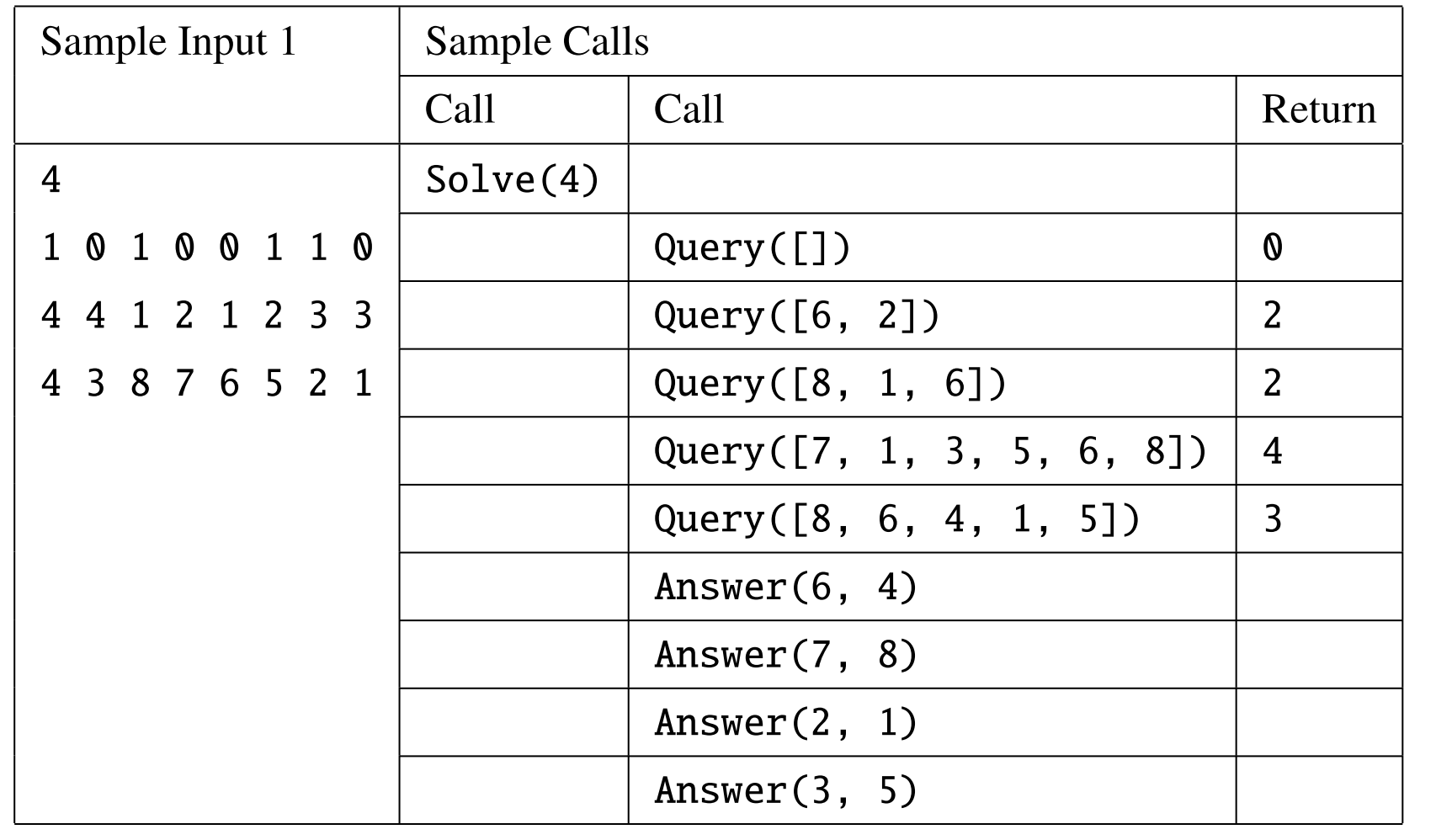
3. (20 points) N ≤ 50.

4. (20 points) Yi = 0 (1 ≤ i ≤ N).

5. (36 points) No additional constraints.

样例交互：

以下是一个样例评测机的样例输入以及各函数的调用情况



在可以从竞赛网站下载的文件中，sample-02.txt满足子任务1的限制,sample-03.txt 满足子任务 4的限制.

2、在Joitter上交朋友很有趣

内存：1.07GB 时限：3s 类型：传统题

/usr/bin/g++ -DEVAL -std=gnu++14 -O2 -pipe -static -s -o joitter2 joitter2.cpp

题目描述：

Joitter是一种流行的社交媒体，在上面你可以和朋友分享你的回忆。

在Joitter上，你可以**关注**其他的用户。比如，当一个用户a关于另一个用户b时，用户a可以阅读b的时间线上的帖子。在这种情况下，b用户可以关注a也可以不关注a。然而，用户a不可能关注他自己或者关注一个用户b超过1次。

N个用户，编号1至N，开始使用Joitter。刚开始的时候，没有任何人关注别人。

从此时开始，持续M天，每天发生一个事件：第i天用户Ai会关注Bi。 (1 ≤ i ≤ M).

Joitter的官方准备在他们的服务器上于这M天内举行一次社会交流活动。一次社会交流活动流程如下：

1. 选择一个用户x。
2. 同时选择一个用户x关注的用户y
3. 选择一个用户z，满足z和x不同，x没有关注z，y关注了z，z也关注了y。
4. 让x关注z
5. 重复这个过程直到无法找到一个合法的三元组(x,y,z)

Joitter的官方还没决定什么时候举行这个社会交流活动。所以，对于M天中的每一天i，当第i天的事件发生后，他们想知道在当天马上接着举行一次社会交流活动结束后所有用户关注的人数的和的最大值。假设社会交流活动在第i天举行时，在i+1天的事件发生前一定可以结束。

写一个程序，给出用户的数量和M天发生的事件，对于每个i计算第i天事件发生后举行社会交流活动后所有用户关注人数的和的最大值。

题目输入：

所有输入都是整数。

N M

A1 B1

...

AM BM

题目输出：

输出共M行，每行按照题目要求输出最大值。

数据限制：

• 2 ≤ N ≤ 100 000.

• 1 ≤ M ≤ 300 000.

• 1 ≤ Ai ≤ N (1 ≤ i ≤ M).

• 1 ≤ Bi ≤ N (1 ≤ i ≤ M).

• Ai , Bi (1 ≤ i ≤ M).

• (Ai, Bi) , (Aj, Bj) (1 ≤ i < j ≤ M).

子任务：

1. (1 point) N ≤ 50.

2. (16 points) N ≤ 2000.

3. (83 points) No additional constraints.

样例一：

输入：

4 6

1 2

2 3

3 2

1 3

3 4

4 3

输出：

1

2

4

4

5

9

解释：

• 第一天，用户1关注了用户2。社会交流活动不会新增关注，所以总数为1

• 第二天，用户2关注了用户3。社会交流活动不会新增关注，所以总数为2

• 第三天，用户3关注了用户2。在社会交流活动中如果用户1关注用户3，此时总数为4，并且这是最大的可能的总数。

• 第四天，用户1关注了用户3，社会交流活动不会新增关注，所以总数为4

• 第五天，用户3关注了用户4，社会交流活动不会新增关注，所以总数为5

• 第六天，用户4关注了用户3，在社会交流活动中如果用户1关注用户4，用户2关注用户4，用户4关注用户2，此时总数为9，并且这是最大的可能的总数。

样例二：

输入：

6 10

1 2

2 3

3 4

4 5

5 6

6 5

5 4

4 3

3 2

2 1

输出：

1

2

3

4

5

7

11

17

25

30

1. 遗迹3

内存：537MB 时限：4s 类型：传统题

/usr/bin/g++ -DEVAL -std=gnu++14 -O2 -pipe -static -s -o ruins3 ruins3.cpp

题目描述：

JOI教授是IOI王国历史学前沿的专家。当他调查IOI王国一个旧寺庙时，他发现了一个石柱建造的遗迹。同时，他还发现了一份IOI王国古人写的旧文档。这份旧文档中包含了对于遗迹石柱的描述。更准确的说，文档中写着如下内容：

• 刚完工时，这里有2N个石柱，编号1至2N

• 刚完工时，对于高度k(1 ≤ k ≤ N)，恰好有两根石柱高度为k。

• 地震发生了N次。每次地震发生后，一些石柱会崩塌，使得其高度减少1。然而，其余的石柱都被古人保护了起来。这些石柱没有崩塌，高度保持不变。

• 当地震发生时，对于每一个高度k(1 ≤ k ≤ N)，只有一个高度为k的石柱会被古人保护。如果地震发生时有多个石柱高度为k，那么编号最大的石柱会被保护。换句话说，如果石柱i(1 ≤ i ≤ 2N)的高度在地震发生前为hi，那么石柱i被保护当且仅当hi>=1且对于所有的j>i满足hj != hi。

• 在N次地震发生后，恰好有N根石柱被保留下来。（恰好有N根石柱高度至少为1）

JOI教授觉得如果他能还原出这2N个石柱刚完工时的高度将会是一个伟大的发现。他更加详细的调查了遗迹，发现地震发生N次后，剩下的N根石柱的编号分别为A1,A2....AN。

JOI教授想知道2N个石柱初始高度的可能情况的数量。作为他的学生，你被要求写一个程序来计算可能的情况数。

写一个程序，告诉你N次地震发生后剩余石柱的编号，计算2N根石柱初始高度的可能情况数，对1 000 000 007取模。

题目输入：

所有输入为整数。

N

A1 A2 .... AN

题目输出：

输出题目要求的情况数对1 000 000 007取模的余数

数据限制：

• 1 ≤ N ≤ 600.

• 1 ≤ Ai ≤ 2N (1 ≤ i ≤ N).

• Ai < Ai+1 (1 ≤ i ≤ N-1).

子任务：

1. (6 points) N ≤ 13.

2. (52 points) N ≤ 60.

3. (42 points) No additional constraints.

样例一：

输入：

3

3 4 6

输出：

5

解释：

比如，假设从石柱1开始各石柱初始高度为 (2, 2, 3, 3, 1, 1). 因为每个高度k(1 ≤ k ≤ 3)都恰好有两根石柱，所以符合旧文档的要求。

• 当第一次地震发生时，2、4、6号石柱被保护，地震发生后高度变为 (1, 2, 2, 3, 0, 1).

• 当第二次地震发生时，3、4、6被保护，地震发生后高度变为(0, 1, 2, 3, 0, 1).

• 当第三次地震发生时，3、4、6被保护，地震发生后高度变为 (0, 0, 2, 3, 0, 1).

三次地震过后，剩下了3、4、6号石柱，与我们的输入信息相符。

除了上述情况以外，还有四种其他的可能初始高度(2, 3, 2, 3, 1, 1), (2, 3, 3, 2, 1, 1), (3, 2, 2, 3, 1, 1), (3, 2, 3, 2, 1, 1).

因此，共有5种可能性满足题目的条件。

样例二：

输入：

1

1

输出：

0

解释：

这个输入中，只有(1,1)是可能的初始高度，但是第一次地震之后2号石柱被保护，高度变为(0,1)，所以2号石柱被剩下，与输入信息不符合，因此没有符合输入要求的初始高度组合。

样例三：

输入：

10

5 8 9 13 15 16 17 18 19 20

输出：

147003663

解释：

共有111 147 004 440可能性，对1 000 000 007取模后得到147003663