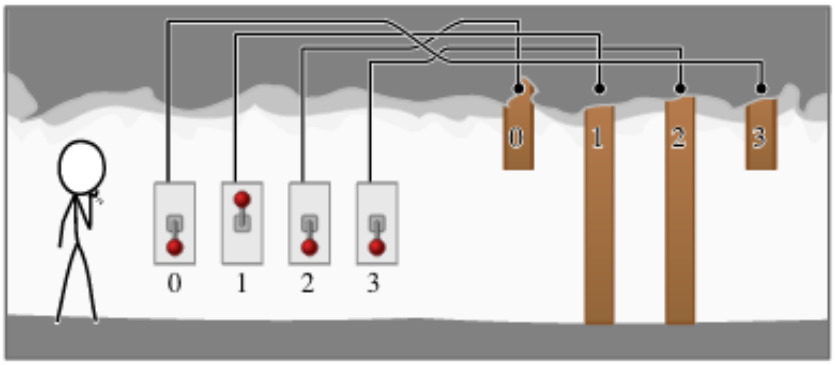
International Olympiad in Informatics 2013
6-13 July 2013
Brisbane, Australia
Day 2 tasks

# 洞穴

中文 — 1.0

在从学院去往昆士兰大学中心礼堂的路上，你迷路了。你跌跌撞撞地来到了一个秘密地下洞穴系统的入口处。这个入口安装了一套复杂的安全系统。该系统由 N 道连续的门和 N 个开关组成。N 道门按前后顺序依次排列,每个开关控制一道不同的门（即开关和门一一对应）。




门按 0, 1, ..., (N - 1) 的顺序编号。0号门离你最近。开关也按 0, 1, ..., (N - 1) 的顺序编号。所有开关都位于入口处。但是你并不知道哪个开关控制哪道门。

每个开关都有“上”和“下”两种状态，其中只有一种状态是正确的。如果一个开关处于正确的状态，它所对应的门就会打开；如果它处于错误的状态，与之对应的门就会关闭。不同的开关有不同的正确状态，但你并不知道哪个开关在哪种状态下是正确的。

现在你试图了解这套安全系统。你可以这样做，将所有开关设置为某种状态组合，然后走进地下洞穴系统去查看哪道门是第一道被关闭的门。因为门是不透明的，所以你不会知道这道关闭的门之后的门是打开还是关闭的。

你有时间尝试 70,000 次开关状态的不同组合，但是不能再多了。你的任务是确定每个开关的正确状态是什么，以及门和开关之间的对应关系。

## 实现

你需要提交一个文件来实现 `exploreCave()`。在这个函数中最多可以调用 `tryCombination()` 函数70,000次，并在最后调用一次 `answer()` 函数。这些函数的描述如下：

### 评分函数: `tryCombination()`

C/C++
```
int tryCombination(int S[]);
```

Pascal
```
function tryCombination(var S: array of LongInt) : LongInt;
```

#### 描述

评分系统会提供这个函数。它允许你尝试一种开关状态组合，并让你了解该组合下第一道关闭的门是哪一道。如果所有门都被打开了，该函数返回 `-1`。这个函数会在 O(N) 时间内返回，即最坏情况下，该函数会在 N 的线性时间内返回。

这个函数最多可以被调用 70,000 次。

#### 参数

- `S`：长度为 N 的数组，表示每个开关的状态。元素 S[i] 对应开关 i。`0` 表示开关状态为“上”，`1` 表示开关状态为“下”。
- *Returns*：第一道关闭的门的编号，或者 `-1`，表示所有门都是打开的。

### 评分函数: `answer()`

C/C++
```
void answer(int S[], int D[]);
```

Pascal
```
procedure answer(var S, D: array of LongInt);
```

#### 描述

你确定了所有开关的正确状态，以及开关和门的对应关系后，可以调用该函数。

参数

- `S`：长度为`N`的数组，表示每个开关的正确状态。格式与函数`tryCombination()`描述的一样。
- `D`：长度为`N`的数组，表示每个开关对应的门的编号。具体来说，元素`D[i]`应该包含开关`i`所对应的门的编号。
- *Returns*：本函数没有返回值，它被调用后整个程序将退出。

## 你的函数：`exploreCave()`

C/C++
```
void exploreCave(int N);
```

Pascal
```
procedure exploreCave(N: longint);
```

描述

你的提交必须实现该函数。

该函数必须调用`tryCombination()`函数来确定开关的正确状态和开关与门之间的对应关系，并且在最后调用一次`answer()`函数。

参数

- `N`：开关和门的数目。

## 样例

假设门和开关的排布如首页中的图例所示:

| 函数调用 | 返回值 | 解释 |
|---|---|---|
| `tryCombination([1, 0, 1, 1])` | `1` | 与图示相对应。开关 0, 2 和 3 状态为“下”，开关 1状态为“上”。 函数返回 `1`, 表示 1 号门是关闭的。 |
| `tryCombination([0, 1, 1, 0])` | `3` | 门 0, 1 和 2 是打开的, 3 号门是关闭的。 |
| `tryCombination([1, 1, 1, 0])` | `-1` | 返回值为 `-1` 表示将开关 0的状态改为“下”，使得所有的门都打开了。 |
| `answer([1, 1, 1, 0], [3, 1, 0, 2])` | *(程序退出)* | 我们猜想的开关正确状态的组合是 `[1, 1, 1, 0]`, 并且开关 0, 1, 2 和 3 分别对应门 3, 1, 0 和 2 。 |

## 限制

- 时间限制: 2 秒
- 内存限制: 32 MB
- $1 \leqslant N \leqslant 5{,}000$

## 子任务

| 子任务 | 分数 | 输入限制 |
|---|---|---|
| 1 | 12 | 对于每个 i, 开关 i 与 i 号门对应。 你的任务仅是确定开关正确状态的组合。 |
| 2 | 13 | 开关正确状态组合永远是 `[0, 0, 0, …, 0]`。你的任务仅是确定开关与门的对应关系。 |
| 3 | 21 | $N \leqslant 100$ |
| 4 | 30 | $N \leqslant 2{,}000$ |
| 5 | 24 | *(无)* |

## 评测

你电脑上的样例评分程序从文件`cave.in`中读入，该文件格式如下：

- 第1行: `N`
- 第2行: `S[0] S[1] … S[N - 1]`
- 第3行: `D[0] D[1] … D[N - 1]`

这里`N`是门和开关的数目，`S[i]`是开关`i`的正确状态，`D[i]`是开关`i`对应的门的编号。

例如，上面的例子的输入文件格式如下：

```
4
1 1 1 0
3 1 0 2
```

## 编程语言说明

| | |
|---|---|
| C/C++ | 你必须`#include "cave.h"`. |
| Pascal | 你必须定义`unit Cave`，并且通过`uses GraderHelpLib`导入评分程序。所有数组从`0`（而非`1`）开始。 |

参考你电脑上的解题模板。