Hang động

Bạn đang bị mắc kẹt trong một hang động và đang tìm cách thoát ra ngoài. Có N cánh cửa đang ở trước mặt bạn, chúng được đánh số từ 0 tới N - 1 theo thứ tự khoảng cách tới bạn tăng dần. Trước cánh cửa 0 có N công tắc, các công tắc được đánh số từ 0 tới N - 1, mỗi công tắc kết nối với đúng 1 cánh cửa, không có 2 cánh cửa nào được kết nối cùng 1 công tắc, tuy nhiên bạn không biết được công tắc nào kết nối với cửa nào.
Mỗi công tắc có 2 trạng thái: lên hoặc xuống. Khi công tắc ở đúng trạng thái được cài đặt sẵn, cánh cửa kết nối với nó sẽ được mở.
Nhiệm vụ của bạn là xác định được với mỗi công tắc cánh cửa mà nó kết nối tới và trạng thái đúng của công tắc là lên hay xuống. Bạn sẽ thử một số tổ hợp trạng thái của công tắc, với mỗi tổ hợp bạn sẽ biết được cánh cửa i bị đóng gần nhất (các cánh cửa từ 0 tới i - 1 đều mở, bạn không biết được trạng thái của các cửa sau i vì cửa i đóng).

Cài đặt:
Hãy viết một chương trình tương tác với chương trình chấm trong đó cài đặt hàm:
void exploreCave(int N)

- N (1 <= N <= 5000) là số lượng cánh cửa.
- Chương trình chấm sẽ gọi hàm này đúng một lần.

Hàm *exploreCave* có thể gọi 2 hàm sau:
int tryCombination(int S[])

- S là một mảng độ dài N, S[i] tương ứng với công tắc i và nhận giá trị 0 nếu bạn thử trạng thái lên cho cánh cửa i hoặc 1 nếu bạn thử trạng thái xuống.
- Hàm này sẽ trả về chỉ số của cánh cửa đầu tiên đóng. Nếu mọi cánh cửa đều mở, hàm trả về -1.
- Hàm này được phép gọi không quá 70000 lần.

void answer(int S[], int D[])

- S là một mảng độ dài N, S[i] tương ứng với công tắc i và nhận giá trị 0 nếu trạng thái đúng của cánh cửa thứ i là lên, ngược lại S[i] = 1.
- D là một mảng độ dài N, D[i] là chỉ số của cánh cửa mà công tắc i nối tới
- Hàm này nên được gọi đúng một lần.

Để gọi được 2 hàm trên, chương trình của bạn phải khai báo thư viện *cave.h*
#include "cave.h"

Một ví dụ khi N = 4 là:

Gọi hàm	**Giá trị trả về**
exploreCave(4)	
tryCombination([1, 0, 1, 1])	1
tryCombination([0, 1, 1, 0])	3

tryCombination([1, 1, 1, 0])	-1
answer([1, 1, 1, 0], [3, 1, 0, 2]	

Chấm điểm:
Với mỗi test, bạn sẽ nhận được điểm của test đó nếu các lời gọi hàm là hợp lệ và bạn xác định đúng được hệ thống công tắc - cửa.

Lưu ý: Chương trình của bạn không được viết hàm main()

Chương trình chấm mẫu:
Để tiện cho việc kiểm thử, bạn được cung cấp chương trình chấm mẫu *grader.c*, *graderlib.c,* file *cave.h* và một file bài làm mẫu *cave.cpp*. Để thuận tiện cho việc biên dịch, nên đặt 3 file này cùng một thư mục rồi biên dịch bằng lệnh: *g++ grader.c cave.cpp -std=c++11*.
Chương trình chấm mẫu *grader.c* đọc dữ liệu từ file *cave.in* theo định dạng:

- Dòng đầu: N
- Dòng hai: N số 0, 1 là trạng thái đúng của từng công tắc
- Dòng ba: N số, số thứ i là cánh cửa mà công tắc i nối tới.

Nếu chương trình của bạn đúng, chương trình chấm mẫu sẽ in ra CORRECT, ngược lại trình chấm mẫu sẽ in ra INCORRECT.