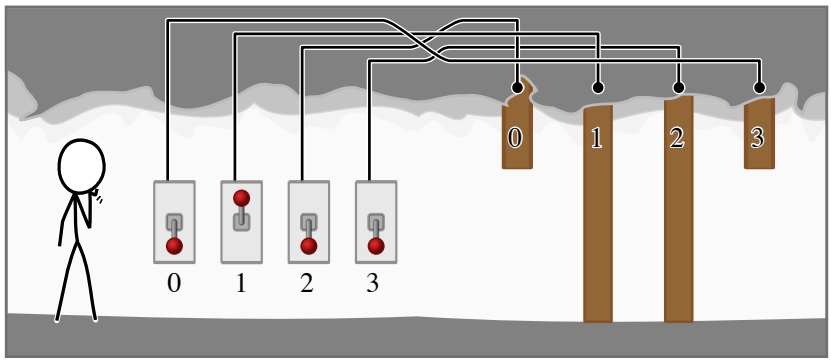
International Olympiad in Informatics 2013
6-13 July 2013
Brisbane, Australia
Day 2 tasks

cave

Thai — 1.0

คุณหลงทางระหว่างการเดินจากหอพักอันแสนไกลของคุณมายัง UQ Centre และคุณได้บังเอิญเจอทางเข้าไปยังถ้ำลับที่อยู่ใต้มหาวิทยาลัย ทางเข้านี้มีระบบรักษาความปลอดภัยซึ่งประกอบด้วยประตูจำนวน N ประตู แต่ละประตูเรียงอยู่ข้างหลังประตูก่อนหน้า และสวิตช์จำนวน N สวิตช์ โดยที่สวิตช์แต่ละอันนั้นเชื่อมต่อกับประตูที่แตกต่างกัน



ประตูมีหมายเลข $0, 1, \ldots,(N-1)$ ตามลำดับ โดยที่ประตู 0 คือประตูที่อยู่ใกล้คุณที่สุด สวิตช์มีหมายเลข $0, 1, \ldots, (N-1)$ เช่นกัน แต่คุณไม่รู้ว่าสวิตช์ใดเชื่อมกับประตูใด

สวิตช์ทั้งหมดอยู่ ณ ทางเข้าไปยังถ้ำ แต่ละสวิตช์จะอยู่ในตำแหน่ง *ขึ้น* หรือไม่ก็ *ลง* มีเพียงตำแหน่งเดียวเท่านั้นที่เป็นตำแหน่งที่ถูกต้องของสวิตช์แต่ละอัน ถ้าสวิตช์อยู่ในตำแหน่งที่ถูกต้องประตูที่เชื่อมต่อกับสวิตช์ดังกล่าวจะเปิดขึ้น และถ้าสวิตช์อยู่ในตำแหน่งที่ไม่ถูกต้องประตูที่เชื่อมต่อกับสวิตช์ดังกล่าวจะปิดลง ตำแหน่งที่ถูกต้องของสวิตช์แต่ละอันอาจจะไม่เหมือนกัน และคุณไม่รู้ตำแหน่งที่ถูกต้องของสวิตช์

คุณต้องการที่จะเข้าใจระบบรักษาความปลอดภัยนี้ คุณสามารถปรับสวิตช์เหล่านี้ ให้เป็นรูปแบบใดก็ได้ แล้วเดินเข้าไปในถ้ำเพื่อดูว่าประตูแรกที่ปิดอยู่นั้นคือประตูใด คุณไม่สามารถมองเห็นประตูอื่น ๆ ที่อยู่ด้านหลังประตูที่ปิดอยู่ได้

คุณมีเวลาที่จะทดลองรูปแบบต่าง ๆ ของสวิตช์ได้ $70{,}000$ รูปแบบ และไม่มากไปกว่านี้ งานของคุณคือคำนวณว่าตำแหน่งที่ถูกต้องของสวิตช์แต่ละอันคืออะไร และแต่ละสวิตช์เชื่อมต่อกับประตูใด

การเขียนโปรแกรม

คุณจะต้องส่งแฟ้มโปรแกรมที่เขียนโปรแกรมย่อย `exploreCave()` ซึ่งสามารถเรียกฟังก์ชัน `tryCombination()` ของเกรดเดอร์ ได้ไม่เกิน `70,000` ครั้ง และจะต้องจบการทำงานด้วยการเรียกโปรแกรมย่อย `answer()` โปรแกรมย่อยและฟังก์ชันเหล่านี้ถูกอธิบายดังต่อไปนี้

ฟังก์ชัน `tryCombination()` ของเกรดเดอร์

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

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

คำอธิบาย

เกรดเดอร์จะมีฟังก์ชันนี้ให้ ฟังก์ชันนี้ทำให้คุณสามารถทดลองรูปแบบต่าง ๆ ของสวิตช์ แล้วเดินเข้าไปในถ้ำเพื่อตรวจดูประตูแรกที่ปิดอยู่ได้ ถ้าประตูทั้งหมดเปิดอยู่ ฟังก์ชันนี้จะคืนค่า `-1` ฟังก์ชันนี้ใช้เวลาในการทำงานเป็น `O(N)` กล่าวคือ เวลาที่ใช้ในกรณีเลวร้ายที่สุดจะแปรผันตรงตามค่า `N`

ฟังก์ชันนี้ถูกเรียกใช้ได้ไม่เกิน `70,000` ครั้ง

พารามิเตอร์

- `S`: อาเรย์ความยาว `N` ซึ่งระบุถึงตำแหน่งของสวิตช์ต่าง ๆ ข้อมูล `S[i]` นั้นหมายถึงสวิตช์ตัวที่ `i` ค่า `0` หมายความว่าสวิตช์อยู่ในตำแหน่งขึ้น และ ค่า `1` หมายความว่าสวิตช์อยู่ในตำแหน่งลง
- *คืนค่า*: หมายเลขของประตูแรกที่ปิดอยู่ หรือ `-1` ถ้าทุกประตูเปิดหมด

โปรแกรมย่อย `answer()` ของเกรดเดอร์

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

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

คำอธิบาย

ให้เรียกโปรแกรมย่อยนี้เมื่อคุณรู้รูปแบบของสวิตช์ที่เปิดประตูทั้งหมด และรู้ว่าประตูใดเชื่อมต่อกับสวิตช์ใดทั้งหมด

พารามิเตอร์

- `S`: อาเรย์ความยาว `N` ซึ่งระบุถึงตำแหน่งที่ถูกต้องของสวิตช์ต่าง ๆ รูปแบบของข้อมูลนั้นเหมือนกับที่อธิบายไว้แล้วใน `tryCombination()`
- `D`: อาเรย์ความยาว `N` ซึ่งระบุถึงประตูที่แต่ละสวิตช์เชื่อมต่ออยู่ กล่าวคือ ข้อมูล `D[i]` จะต้องระบุหมายเลขของประตูที่สวิตช์หมายเลข `i` เชื่อมต่ออยู่
- *คืนค่า*: โปรแกรมย่อยนี้ไม่คืนค่าใด ๆ แต่จะทำให้โปรแกรมหยุดทำงาน

โปรแกรมย่อย `exploreCave()` ของคุณ

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

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

คำอธิบาย

โปรแกรมของคุณจะต้องมีโปรแกรมย่อยนี้

ฟังก์ชันนี้ควรจะเรียกใช้โปรแกรมย่อย `tryCombination()` ของเกรดเดอร์ เพื่อหาตำแหน่งที่ถูกต้องของสวิตช์แต่ละอันและประตูที่แต่ละสวิตช์เชื่อมต่ออยู่ และจะต้องเรียก `answer()` หลังจากที่คำนวณข้อมูลดังกล่าวเสร็จแล้ว

พารามิเตอร์

- `N`: จำนวนของสวิตช์และประตูในถ้ำ

ตัวอย่างการติดต่อ

สมมติว่าประตูและสวิตช์เรียงตัวตามรูปด้านบน

การเรียกฟังก์ชัน	คืนค่า	คำอธิบาย
`tryCombination([1, 0, 1, 1])`	`1`	สถานะเริ่มต้นเป็นดังรูปด้านบน สวิตช์ 0, 2, 3 อยู่ในตำแหน่งลง ส่วนสวิตช์ 1 อยู่ในตำแหน่งขึ้น ฟังก์ชันคืนค่า `1` ซึ่งหมายความว่าประตู 1 เป็นประตูแรกที่ปิดอยู่
`tryCombination([0, 1, 1, 0])`	`3`	ประตู 0, 1 และ 2 เปิดอยู่ทั้งหมด ในขณะที่ประตู 3 ปิดอยู่
`tryCombination([1, 1, 1, 0])`	`-1`	เลื่อนสวิตช์ 0 ลง ทำให้ประตูทั้งหมดเปิดขึ้น ซึ่งเห็นได้จากค่าที่คืนมาเป็น `-1`
`answer([1, 1, 1, 0], [3, 1, 0, 2])`	*(โปรแกรมจบการทำงาน)*	เราตอบว่ารูปแบบที่ถูกต้องคือ `[1, 1, 1, 0]` และสวิตช์ 0, 1, 2 และ 3 ต่ออยู่กับประตู 3, 1, 0, และ 2 ตามลำดับ

เงื่อนไขบังคับ

- จำกัดเวลา 2 วินาที
- จำกัดหน่วยความจำ 32 MiB
- $1 \le N \le 5,000$

ปัญหาย่อย

ปัญหาย่อย	คะแนน	เงื่อนไขบังคับเพิ่มเติมของข้อมูลนำเข้า
1	12	สำหรับแต่ละค่า `i` สวิตช์หมายเลข `i` เชื่อมต่ออยู่กับประตูหมายเลข `i` งานของคุณคือเพียงแค่คำนวณหารูปแบบของสวิตช์ที่ถูกต้องเท่านั้น
2	13	รูปแบบที่ถูกต้องจะเป็น `[0, 0, 0,…,0]` เสมอ งานของคุณคือการหาว่าสวิตช์ใดเชื่อมกับประตูใดเท่านั้น
3	21	$N \le 100$
4	30	$N \le 2,000$
5	24	*(ไม่มี)*

การทดลอง

เกรดเดอร์ตัวอย่างในคอมพิวเตอร์ของคุณจะอ่านข้อมูลนำเข้าจากแฟ้มชื่อ `cave.in` ซึ่งต้องมีรูปแบบดังต่อไปนี้

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

ในที่นี้ `N` คือจำนวนของประตูและสวิตช์ `S[i]` คือตำแหน่งที่ถูกต้องของสวิตช์ `i` และ `D[i]` คือประตูที่สวิตช์ `i` เชื่อมต่ออยู่

ยกตัวอย่างเช่น ตัวอย่างด้านบนจะระบุในรูปแบบต่อไปนี้

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

หมายเหตุด้านภาษา

C/C++	คุณจะต้องระบุ `#include "cave.h"` ที่ส่วนหัวของโปรแกรม
Pascal	คุณจะต้องนิยาม `unit Cave` และคุณจะต้องนำเข้าโปรแกรมย่อยของเกรดเดอร์โดยระบุคำสั่ง `uses GraderHelpLib` อาเรย์ทั้งหมดจะเริ่มนับที่ `0` (ไม่ใช่ `1`).

คุณสามารถดูตัวอย่างได้จากเทมเพลตในเครื่องคุณ