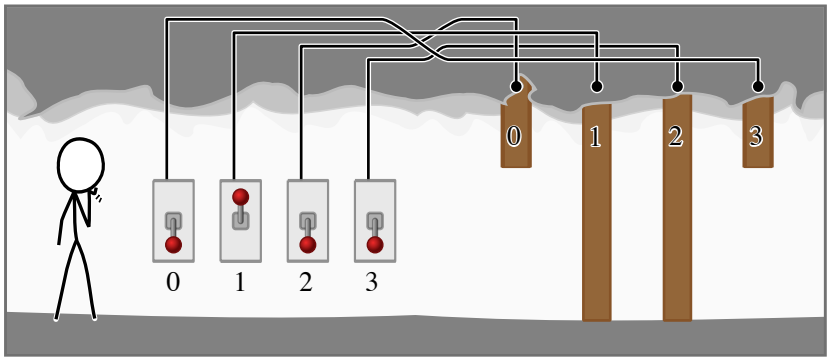
International Olympiad in Informatics 2013
6-13 July 2013
Brisbane, Australia
Day 2 tasks

cave

Romanian — 1.0

Pierdându-vă pe drumul lung de la colegiu la UQ Centre, ați dat peste intrarea într-un sistem secret de peșteri ce se întinde adânc sub universitate. Intrarea este blocată de un sistem de securitate consistând din `N` uși consecutive, aflate una în spatele alteia și `N` întrerupătoare, fiecare întrerupător conectat la o ușă diferită.



Ușile sunt numerotate `0, 1, …, (N - 1)` în ordine, ușa 0 fiind cea mai apropiată de voi. Întrerupătoarele sunt de asemenea numerotate `0, 1, …, (N - 1)`, dar nu cunoașteți care întrerupător este conectat la care ușă.

Întrerupătoarele se află la intrarea în peșteră. Fiecare întrerupător poate fi în poziția *up* sau în poziția *down*. Numai una din aceste poziții este corectă pentru fiecare întrerupător. Dacă un întrerupător este în poziția corectă atunci ușa la care este conectat se va deschide, iar dacă întrerupătorul se află în poziția greșita atunci ușa la care este conectat nu se va deschide. Poziția corectă poate fi diferită pentru diferite întrerupătoare, și voi nu știți care poziții sunt cele corecte.

Ați vrea să înțelegeți sistemul de securitate. Pentru a face asta, puteți seta întrerupătoarele în orice combinație, și apoi puteți merge în peșteră pentru a vedea care este prima ușă închisă. Ușile nu sunt transparente: o dată ce ați întâlnit prima ușă închisă, nu puteți vedea nicio ușă ce se află după aceasta.

Aveți timp să încercați `70,000` de combinații de întrerupătoare, dar nu mai mult de atât. Task-ul vostru este să determinați poziția corectă pentru fiecare întrerupător și cu care ușă este conectat fiecare întrerupător.

Implementare

Va trebui să submitați un fișier ce implementează procedura `exploreCave()`. Acesta poate apela funcția `tryCombination()` a grader-ului de maxim 70,000 de ori, și trebuie să termine apelând procedura `answer()` a grader-ului. Aceste funcții și proceduri sunt descrise mai jos.

Funcția grader-ului: `tryCombination()`

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

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

Descriere

Grader-ul va avea acestă funcție. Aceasta vă permite să încercați o combinație de întrerupătoare, și apoi să intrați în peșteră pentru a determina care este prima ușă închisă. Dacă toate ușile sunt deschise, atunci va returna `-1`. Acestă funcție se execută într-o complexitate temporală de `O(N)`; adică în cel mai rău caz se execută într-un timp propoțional cu `N`.

Acestă funcție poate fi apelată de maxim `70,000` de ori.

Parametrii

- `S`: Un array de dimensiune `N`, indicând poziția fiecarui întrerupător. Elementul `S[i]` corespunde cu întrerupătorul `i`. O valoare de `0` indică faptul că întrerupătorul este în poziția de up, iar o valoare de `1` indică faptul că este în poziția de down.
- *Returnează*: Numărul primei uși care este închisă, sau `-1` dacă toate ușile sunt deschise.

Procedura grader-ului: `answer()`

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

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

Descriere

Apelați acestă procedură atunci când ați identificat o combinație de întrerupătoare care să deschidă toate ușile, și ușa la care este conectat fiecare întrerupător.

Parametrii

- `S`: Un array de lungime `N`, indicând poziția corectă a fiecărui întrerupător. Formatul acestuia este la fel cu formatul din funcția `tryCombination()` de mai sus.
- `D`: Un array de lungime `N`, indicând ușa la care este conectat fiecare întrerupător. Mai precis, elementul `D[i]` ar trebui să conțină numărul ușii la care întrerupatorul `i` este conectat.
- *Returns*: Această procedură nu returneaza, ci va cauza programul sa facă exit.

Procedura voastră: `exploreCave()`

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

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

Descriere

Submisia voatră trebuie să implementeze această procedură.

Această procedură ar trebui să folosească funcția `tryCombination()` a grader-ului pentru a determina poziția corectă a fiecărui întrerupător, și ușa la care este conectat fiecare întrerupător, iar apoi, când are aceste informații, să apeleze `answer()`.

Parametrii

- `N`: Numărul de întrerupătoare și uși din peșteră.

Exemplu

Presupuneți că ușile si întrerupătoarele sunt aranjate ca în imaginea de mai sus:

Apel de funcție	Returnează	Explicație
`tryCombination([1, 0, 1, 1])`	`1`	Aceasta corespunde imaginii. Întrerupătoarele 0, 2 și 3 sunt down, iar întrerupătorul 1 este up. Funcția returnează `1`, indicând că ușa 1 este prima ușă de la stânga care este închisă.
`tryCombination([0, 1, 1, 0])`	`3`	Ușile 0, 1 și 2 sunt deschise, iar ușa 3 este închisă.
`tryCombination([1, 1, 1, 0])`	`-1`	Mutând înterupătorul 0 în poziția down cauzează toate ușile să se deschidă, lucru indicat de valoarea de retur `-1`
`answer([1, 1, 1, 0], [3, 1, 0, 2])`	*(Programul își încheie execuția)*	Ne putem da seama că răspunsul este `[1, 1, 1, 0]`, iar întrerupătoarele 0, 1, 2 și 3 sunt conectate la ușile 3, 1, 0 și 2.

Constrângeri

- Limită de timp: 2 secunde
- Limită de memorie: 32 MiB
- $1 \le N \le 5{,}000$

Subtask-uri

Subtask	Punctaj	Constrângeri adiționale
1	12	Pentru fiecare `i`, întrerupătorul `i` este conectat la ușa `i`. Task-ul vostru este să determinați poziția corectă a întrerupătoarelor.
2	13	Poziția corectă va fi tot timpul `[0, 0, 0, …, 0]`. Task-ul vostru este să determinați cu ce ușă este conectat fiecare întrerupător.
3	21	$N \leq 100$
4	30	$N \leq 2{,}000$
5	24	*(Nimic)*

Testare

Grader-ul de pe computerul vostru va citi datele de intrare din fișierul `cave.in`, care trebuie să fie în următorul format:

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

Aici `N` este numărul de uși și întrerupătoare, `S[i]` este poziția corectă pentru întrerupătorul `i`, și `D[i]` este ușa la care este conectat întrerupătorul `i`.

Astfel, exemplul de mai sus ar fi dat în următorul format:

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

Note de limbaj

C/C++	Trebuie să faceți `#include "cave.h"`.
Pascal	Trebuie să definiți `unit Cave`, și trebuie să importați rutinele grader-ului facând `uses GraderHelpLib`. Toate array-urile sunt indexate de la `0` (nu de la `1`).

Vedeți template-urile de soluții de pe computer-ul vostru pentru exemple.