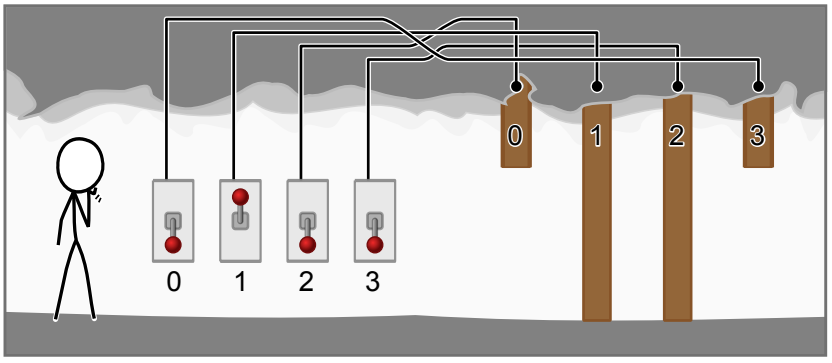
International Olympiad in Informatics 2013
6-13 July 2013
Brisbane, Australia
Day 2 tasks

cave

Czech — 1.0

Cestou na soutěž jste se ztratili a narazili na vchod do tajného podzemí pod univerzitou. Vstupní chodbu chrání `N` po sobě jdoucích bran, tj., abyste se dostali dovnitř, musíte postupně projít každou z nich. U vchodu do vstupní chodby je `N` přepínačů a každý z nich ovládá právě jednu z bran. Různé přepínače ovládají různé brány, každou bránu tedy ovládá právě jeden přepínač.



Brány jsou postupně dle pořadí v chodbě očíslovány od `0` do `N-1`, kde první je brána číslo 0. Přepínače jsou také očíslovány od `0` do `N-1`, ale nevíte, který přepínač ovládá kterou bránu.

Každý z přepínačů může být přepnut v poloze nahoru nebo dolů. Pro každý z přepínačů je právě jedna z těchto poloh správná. Je-li přepínač ve správné poloze, pak je jím ovládaná brána otevřená. Je-li přepínač v opačné poloze, pak je jím ovládaná brána zavřená. Správná poloha může být pro různé přepínače různá a dopředu ji neznáte.

Rádi byste pochopili, jak tento bezpečnostní systém funguje. Od přepínačů na brány nevidíte a brány jsou neprůhledné, takže můžete pouze opakovat následující postup: Nastavíte nějak přepínače a zajdete zjistit číslo té zavřené brány, která je nejblíže ke vchodu (tedy jejíž číslo je nejmenší).

Do začátku soutěže ale zbývá jen pár minut, takže máte čas tento postup opakovat nejvýše `70 000`-krát. Pro každý přepínač chcete zjistit jeho správnou polohu a kterou bránu ovládá.

Implementace

Odevzdejte soubor implementující proceduru `exploreCave()`. Tato procedura může až `70 000`-krát zavolat funkci `tryCombination()`, která je poskytnuta vyhodnocovačem. Poté musí zavolat proceduru `answer()` poskytnutou vyhodnocovačem. Význam a parametry těchto procedur a funkcí jsou vysvětleny níže.

Funkce vyhodnocovače: `tryCombination()`

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

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

Popis

Tato funkce je poskytnuta vyhodnocovačem a její pomocí můžete určit pro zadanou kombinaci přepínačů číslo první uzavřené brány. Jestliže jsou všechny brány otevřené, pak tato funkce vrátí `-1`. Časová složitost funkce `tryCombination` je `O(N)`, tedy nanejvýš přímo úměrná `N`.

Tuto funkci smíte zavolat nejvýše `70 000`-krát.

Parametry

- `S`: Jednoroznměrné pole délky `N`, popisující pozice jednotlivých přepínačů. Prvek `S[i]` je `0`, jestliže je přepínač číslo `i` v pozici nahoru. Prvek `S[i]` je `1`, jestliže je přepínač číslo `i` v pozici dolů.
- *Návratová hodnota*: Číslo první zavřené brány, nebo `-1` v případě, že všechny brány jsou otevřené.

Procedura vyhodnocovače: `answer()`

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

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

Popis

Jakmile určíte kombinaci přepínačů, která otevře všechny brány, a schéma propojení bran s přepínači, zavolejte tuto proceduru.

Parametry

- `S`: Jednorozměrné pole délky `N`, popisující správnou pozici všech přepínačů. Formát je stejný jako u výše popsané fuknce `tryCombimation()`.

- `D`: Jednorozměrné pole délky `N`, popisující propojení přepínačů a bran. Prvek `D[i]` musí obsahovat číslo brány ovládané přepínačem číslo `i`.
- *Návratová hodnota*: Tato procedura způsobí ukončení běhu programu, a tedy se nevrátí.

Vaše procedura: `exploreCave()`

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

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

Popis

Naimplementujte tuto proceduru.

Tato procedura by měla použít opakované volání funkce `tryCombination()` pro určení správných pozic přepínačů a schématu propojení přepínačů a bran. Poté s touto informací zavolá proceduru `answer()`.

Parametry

- `N`: Počet přepínačů a bran.

Ukázka průběhu programu

Nechť jsou přepínače a brány propojeny jako na obrázku nahoře.

Volání funkce	Návratová hodnota	Vysvětlení
`tryCombination([1, 0, 1, 1])`	`1`	Počáteční stav odpovídá obrázku. Přepínače 0, 2 a 3 jsou dole, přepínač 1 je nahoře. Funkce vrátí `1`, tedy číslo uzavřené brány, která je nejblíže ke vchodu.
`tryCombination([0, 1, 1, 0])`	`3`	Brány 0, 1 a 2 jsou otevřené, brána 3 je zavřená.
`tryCombination([1, 1, 1, 0])`	`-1`	Přepnutí přepínače 0 do polohy dole způsobí, že všechny brány jsou otevřené, jak ukazuje návratová hodnota `-1`.
`answer([1, 1, 1, 0], [3, 1, 0, 2])`	*(program skončí)*	Uhodneme, že správná kombinace poloh přepínačů je `[1, 1, 1, 0]` a že přepínače 0, 1, 2 a 3 po řadě ovládají brány 3, 1, 0 a 2.

Omezení

- Časový limit: 2 sekundy
- Paměťový limit: 32 MiB
- $1 \leq N \leq 5\,000$

Podúlohy

Podúloha	Body	Dodatečná omezení
1	12	Dveře číslo i ovládají bránu číslo i, pro každé i. Zbývá vám tedy pouze určit správnou polohu všech přepínačů.
2	13	Správná poloha všech přepínačů je 0. Zbývá vám tedy pouze určit, který přepínač ovládá kterou bránu.
3	21	$N \leq 100$
4	30	$N \leq 2\,000$
5	24	*(žádná)*

Experimentování

Vzorový testovač čte soubor `cave.in` v následujícím formátu:

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

Zde `N` je počet přepínačů a bran, `S[i]` je správná pozice přepínače číslo i a `D[i]` je číslo brány ovládané přepínačem číslo i.

Příklad na obrázku je tedy zadán následujícím souborem.

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

Poznámky k programovacím jazykům

C/C++	Na začátku souboru s řešením musí být řádek `#include "cave.h"`.
Pascal	Naimplementujte `unit Cave` a naimportujte funkce vyhodnocovače pomocí příkazu `uses GraderHelpLib`. Položky všech polí jsou číslovány od `0` (nikoliv od `1`).

Viz kostry řešení ve složce této úlohy.