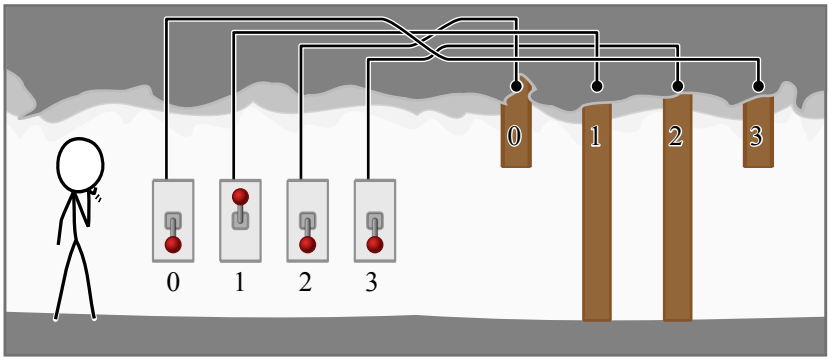
International Olympiad in Informatics 2013
6-13 July 2013
Brisbane, Australia
Day 2 tasks

cave

Srpski — 1.0

Vi ste se izgubili na dugom putu od koledža do univerzitetskog centra i nabasali na ulaz u tajnu pećinu koja se proteže duboko ispod Univerziteta. Ulaz u pećinu je blokiran bezbedonosnim sistemom koji se sastoji od `N` uzastopnih vrata poređanih jedna za drugim i `N` prekidača od kojih je svaki povezan sa jednim od vrata (različiti prekidači sa različitim vratima).



Vrata su numerisana (označena) brojevima `0, 1, …, (N - 1)` u redosledu u kome su postavljena, pri čemu su vrata numerisana brojem 0 najbliža Vama. Prekidači su takođe numerisani brojevima `0, 1, …, (N - 1)`, i Vi ne znate koji prekidač je povezan sa kojim vratima.

Prekidači se nalaze na ulazu u pećinu. Svaki prekidač može biti u jednoj od pozicija *gore* ili *dole*. Samo jedna od ovih pozicija je ispravna za svaki od prekidača. Ako je prekidač u ispravnoj poziciji, vrata na koji jeon povezan su otvorena. Ako je prekidač u neispravnoj poziciji, vrata na koji je povezan će biti zatvorena. Različiti prekidači mogu imati različite ispravne pozicije i Vi ne znate njihove ispravne pozicije.

Vi želite da proučite taj bezbedonosni sistem. Da bi to uradili, možete da postavite sve prekidače u proizvoljnu poziciju i tako napravite proizvoljnu kombinaciju, a nakon toga krenete u pećinu kako bi stigli do prvih vrata koja su zatvorena. Vrata nisu transparentna: kada pronađete prva zatvorena vrata, Vi ne možete videti nijedna vrata posle njih.

Možete da probate `70,000` kombinacija prekidača (ne više od toga). Vaš zadatak je da odredite ispravnu poziciju za svaki od prekidača, kao i vrata sa kojima su povezani prekidači.

Implementacija

Vi treba da priložite datoteku koja sadrži implementaciju procedure `exploreCave()`. Ona može pozivati funkciju `tryCombination()` ne više od 70,000 puta, i mora završiti pozivom procedure `answer()`. Ove funkcije i procedure su opisane u nastavku.

Funkcija (grejdera): `tryCombination()`

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

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

Opis

Grejder obezbeđuje implementaciju ove funkcije. Ona obezbeđuje da možete napraviti bilo koju kombinaciju prekidača, a zatim ući u pećinu i pronaći prva zatvorena vrata. Ako su sva vrata otvorena, funkcija će vratiti `-1`. Ova funkcija se izvršava u `O(N)` vremenu; što znači da je vreme izvršavanja u najlošijem (najgorem) slučaju proporcionalno broju `N`.

Ova funkcija može biti pozvana najviše `70,000` puta.

Argumenti (parametri)

- `S`: Niz dužine `N`, koji govori o pozicijama svih prekidača. Element `S[i]` odgovara prekidaču `i`. Vrednost `0` označava da je taj prekidač u poziciji gore, a vrednsot `1` označava da je taj prekidač u poziciji dole.
- *Vraća*: Redni broj (oznaku) prvih vrata koja su zatvorena, ili `-1` ako su sva vrata otvorena.

Procedura (grejdera): `answer()`

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

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

Opis

Pozivate ovu proceduru kada odredite kombinaciju prekidača koja otvara sva vrata, kao i na koja vrata su vezani pojedini prekidači.

Argumenti (parametri)

- `S`: Niz dužine `N`, koji govori o ispravnim pozicijama svih prekidača. Format se poklapa sa formatom u pozivu funkcije `tryCombination()`, što je već opisano.

- `D`: Niz dužine `N`, koji govori o vratima na koja su povezani pojedini prekidači. Preciznije, element `D[i]` sadrži redni broj (oznaku) vrata koja su povezana sa prekidačem čiji je redni broj (oznaka) `i`.
- *Vraća*: Ova procedura ne vraća rezultat, ali prouzrokuje da se prekida izvršavanje programa.

Vaša procedura: `exploreCave()`

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

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

Opis

Vi morate priložiti implementaciju ove procedure.

Ova funkcija treba da koristi funkciju `tryCombination()` da bi odredili ispravne pozicije prekidača, kao i oznake vrata na koje su povezani pojedini prekidači i mora pozvati funkciju `answer()` samo jednom, kada su određeni svi rezultati.

Parametri

- `N`: Broj prekidača i vrata u pećini.

Primer izvršavanja

Pretpostavimo da su vrata i prekidači raspoređeni kao što pokazuje gornja slika:

Poziv funkcije	Vraća	Objašnjenje
`tryCombination([1, 0, 1, 1])`	`1`	Ovo odgovara slici. Prekidači 0, 2 i 3 su u poziciji dole, a prekidač 1 u poziciji gore. Funkcija vraća `1`, označavajući da su vrata broj 1 prva sleva koja su zatvorena.
`tryCombination([0, 1, 1, 0])`	`3`	Vrata 0, 1 i 2 su otvorena, dok su vrata 3 zatvorena.
`tryCombination([1, 1, 1, 0])`	`-1`	Prebacujući prekidač 0 u poziciju dole, sva vrata postaju otvorena što se manifestuje tako što funkcija vraća vrednost `-1`.
`answer([1, 1, 1, 0], [3, 1, 0, 2])`	*(Prekida se izvršavanje programa)*	Pogodili ste da su ispravne pozicije `[1, 1, 1, 0]`, i da su prekidači 0, 1, 2 i 3 povezani sa vratima 3, 1, 0 i 2 redom.

Ograničenja

- Vremensko ograničenje: 2 sekunde
- Memorijsko ograničenje: 32 MiB
- $1 \le N \le 5,000$

Podzadaci

Podyadatak	Poeni	Dodatna ograničenja u ulaznim podacima
1	12	Za svako `i`, prekidač broj `i` je povezan na vrata `i`. Vaš zadatak je samo da odredite ispravnu kombinaciju pozicija prekidača.
2	13	Ispravna pozicija prekidača je obavezno `[0, 0, 0, …, 0]`. Vaš zadatak je da odredite koji prekidač je vezan na koja vrata.
3	21	$N \le 100$
4	30	$N \le 2,000$
5	24	*(Nema)*

Lokalno testiranje

Jednostavan primer grejdera na Vašem računaru čita ulazne podatke iz datoteke `cave.in`, koja treba da ima sledeći format:

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

Ovde je `N` broj vrata i prekidača, `S[i]` je ispravna pozicija prekidača `i`, i `D[i]` je redni broj (oznaka) vrata sa kojim je povezan prekidač `i`.

Npr. za primer sa prethodnih strana, datoteka treba da ima sledeći format:

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

Napomene vezane za programski jezik

C/C++	Vi treba da dodate red `#include "cave.h"`.
Pascal	Vi treba da definišete `unit Cave`, i treba da importujete grejder rutine dodajući `uses GraderHelpLib`. Svi nizovi su numerisani počev od `0` (ne od `1`).

Pogledajte uzorak rešenja na Vašim računarima.