International Olympiad in Informatics 2013

6-13 July 2013

Brisbane, Australia

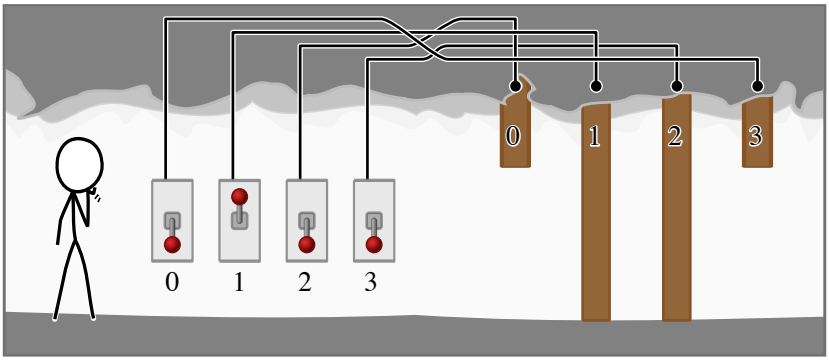
Day 2 tasks

cave

Lithuanian — 1.0

Kvinslando universiteto (UQ), kuriame vyksta IOI, teritorija didelė. Einant iš sporto centro į jūsų koledžą lengva paklysti. Kartą eidami paklydote ir atsidūrėte prie įėjimo į požeminius UQ koridorius, apie kurių egzistavimą nežinojote.

Įėjimą į požemius saugo sistema, sudaryta iš `N` viena po kitos esančių durų, sujungtų su `N` jungiklių: skirtingos durys sujungtos su skirtingais jungikliais.



Durys sunumeruotos iš eilės nuo `0` iki `(N - 1)`. Arčiausiai jūsų yra durys, kurių numeris 0. Jungikliai sunumeruoti nuo `0` iki `(N - 1)`. Deja, nežinote kuris jungiklis su kuriomis durimis yra sujungtas.

Jungikliai yra prie įėjimo į požemius. Kiekvienas jungiklis gali būti *pakeltas į viršų* arba *nuleistas žemyn*. Jei jungiklis yra tinkamoje pozicijoje, su juo sujungtos durys atidarytos. Jei jungiklio pozicija netinkama, atitinkamos durys užvertos.

Nėra žinoma, kurioje pozicijoje turi būti konkretus jungiklis (t.y. pakeltas ar nuleistas), kad su juo sujungtos durys būtų atvertos. Skirtingiems jungikliams šios pozicijos gali skirtis.

Norėdami įeiti turite suprasti kaip veikia apsauga. Galite parinkti bet kokią norimą jungiklių kombinaciją ir įeiti į požemius iki prieisite pirmas uždarytas duris. Deja, negalite matyti kas yra už uždarytų durų.

Turite laiko išbandyti ne daugiau kaip `70,000` jungiklių kombinacijų. Kiekvienam jungikliui nustatykite su kuriomis durimis jis yra sujungtas ir ar jungiklis turi būti pakeltas ar nuleistas, kad tos durys atsidarytų.

Realizacija

Pateikite failą, realizuojantį žemiau apibūdintą procedūrą `exploreCave()`. Ji gali iškviesti vertintojo funkciją `tryCombination()` ne daugiau kaip 70,000 kartų ir turi užbaigti darbą iškviesdama vertintojo procedūrą `answer()`. Šios funkcijos ir procedūros aprašytos žemiau.

Vertintojo funkcija: `tryCombination()`

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

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

Veikimas

Šią funkciją pateikia vertintojas. Ji leidžia išbandyti jungiklių kombinaciją ir įeiti į požemius iki prieisite pirmas užrakintas duris (t.y. leidžia nustatyti pirmų užrakintų durų numerį).

Jei visos durys atvertos, funkcija grąžins `-1`. Šios funkcijos sudėtingumas laiko atžvilgiu `O(N)`, t.y. pačiu blogiausiu atveju vykdymo laikas tiesiškai priklauso nuo `N`.

Šią funkciją galima iškviesti ne daugiau `70,000` kartų.

Parametrai

- `S`: `N` elementų ilgio masyvas, nusakantis kiekvieno jungiklio poziciją. Masyvo elementas `S[i]` atitinka `i`-ąjį jungiklį. Jei elemento reikšmė `0`, tai rodo, kad jungiklis yra pakeltas, o jei reikšmė `1`, tai rodo, kad jungiklis nuleistas.
- *return*: pirmųjų uždarytų durų numeris arba `-1`, jei visos durys atviros.

Vertintojo procedūra: `answer()`

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

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

Veikimas

Šią procedūrą iškvieskite tuomet, kai žinosite, su kuriomis durimis sujungtas *kiekvienas* jungiklis ir kokioje pozicijoje jis turi būti, kad tos durys atsidarytų.

Parametrai

- `S`: `N` elementų ilgio masyvas, rodantis teisingą kiekvieno jungiklio poziciją; masyvas nusakomas taip pat kaip jau aprašytoje funkcijoje `tryCombination()`.
- `D`: `N` elementų ilgio masyvas, nurodantis su kuriomis durimis sujungtas atitinkamas jungiklis; masyvo elemente `D[i]` turi būti įrašytas durų, su kuriomis sujungtas jungiklis `i`, numeris.
- *return*: procedūra nieko negrąžina; ją iškvietus programa sustabdoma.

Jūsų procedūra: `exploreCave()`

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

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

Veikimas

Realizuokite šią procedūrą.

Ši funkcija turi naudoti vertintojo funkciją `tryCombination()`, kuri padės nustatyti, kurias duris valdo konkretus jungiklis ir kokioje pozicijoje jis turi būti, kad tos durys atsidarytų. Kai jūsų procedūra turės šią informaciją, ji turi iškviesti procedūrą `answer()`.

Parametrai

- `N`: jungiklių ir durų skaičius.

Pavyzdinis veikimas

Tarkime, jungikliai sujungti su durimis kaip parodyta paveiksle sąlygos pradžioje:

Kreipinys į funkciją	Grąžinama	Paaiškinimas
`tryCombination([1, 0, 1, 1])`	`1`	Atitinka situaciją paveiksle. Jungikliai su numeriais 0, 2 ir 3 nuleisti žemyn, o jungiklis nr. 1 yra pakeltas. Funkcija grąžina `1`. Tai reiškia, kad durys nr. 1 yra pirmosios uždarytos durys.
`tryCombination([0, 1, 1, 0])`	`3`	Durys su numeriais 0, 1 ir 2 yra atidarytos, durys nr. 3 - uždarytos.
`tryCombination([1, 1, 1, 0])`	`-1`	Jungiklį nr. 0 nuleidus žemyn visos durys atsivers, todėl bus grąžinta reikšmė `-1`.
`answer([1, 1, 1, 0], [3, 1, 0, 2])`	*(Programa užbaigia darbą)*	Spėjama, kad teisinga jungiklių pozicijų kombinacija yra `[1, 1, 1, 0]`, ir kad jungikliai nr. 0, 1, 2 bei 3 atitinkamai sujungti su durimis kurių nr. 3, 1, 0 ir 2.

Ribojimai

- Laiko ribojimas: 2 sekundės
- Atminties ribojimas: 32 MiB
- $1 \le N \le 5,000$

Dalinės užduotys

Dalinė užduotis	Taškai	Papildomi duomenų ribojimai
1	12	Kiekvienam `i`, jungiklis nr. `i` sujungtas su durimis nr. `i`. Jūsų užduotis: nustatyti korektiškas jungiklių pozicijas.
2	13	Visos durys bus atvertos, jei visi jungikliai bus pakelti: `[0, 0, 0, …, 0]`. Jums reikia nustatyti, kuris jungiklis kurias duris valdo.
3	21	$N \le 100$
4	30	$N \le 2,000$
5	24	*(Nėra)*

Eksperimentavimas

Jums pateiktas pavyzdinis vertintojas skaitys failą `cave.in`, kurio formatas toks:

- 1-a eilutė: `N`
- 2-a eilutė: `S[0] S[1] … S[N - 1]`
- 3-ia eilutė: `D[0] D[1] … D[N - 1]`

Čia `N` yra jungiklių ir durų skaičius, `S[i]` yra teisinga `i`-ojo jungiklio pozicija, o `D[i]` yra durų, su kuriomis sujungtas jungiklis `i`, numeris.

Pavyzdžiui, aprašytas pavyzdys būtų pateiktas tokiu formatu:

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

Pastabos apie programavimo kalbas

C/C++	Įtraukite `#include "cave.h"`.
Pascal	Apibrėžkite `unit Cave` ir įkelkite vertintojo procedūras `uses GraderHelpLib`. Visi masyvai numeruojami nuo `0` (ne nuo `1`).

Naudokite pateiktą sprendimo šabloną kaip pavyzdį.