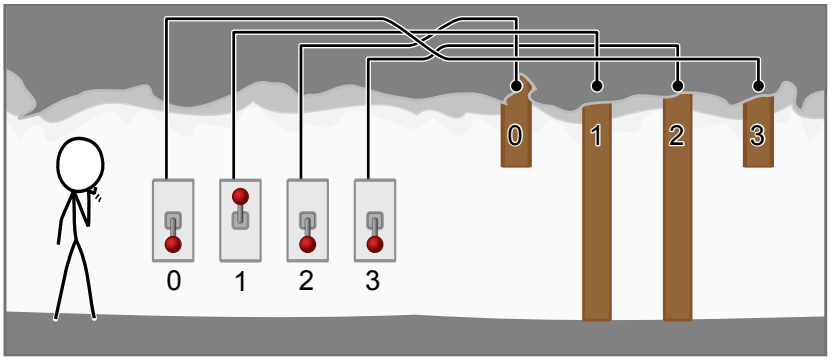
International Olympiad in Informatics 2013
6-13 July 2013
Brisbane, Australia
Day 2 tasks

cave

Armenian — 1.0

Մոլորվելով քոլեջից UQ կենտրոն տանող երկար ճանապարհին, դուք հայտնվել եք գաղտնի ժայռափոր համակարգում, որը տանում է համալսարանի տակ: Մուտքը փակված է անվտանգության համակարգով, որը բաղկացած է N իրար հաջորդող դռներից և N անջատիչներից, յուրաքանչյուր անջատիչ համապատասխանում է ճիշտ մեկ դռան:



Դռները համարակալված են $0, 1, \ldots, (N - 1)$ թվերով, 0 համարի դուռը ձեզ ամենամոտիկն է: Անջատիչները համարակալված են $0, 1, \ldots, (N - 1)$ թվերով, բայց դուք չգիտեք, թե որ անջատիչը որ դռանն է համապատասխանում:

Անջատիչները տեղադրված են քարանձավի մուտքի մոտ: Յուրաքանչյուր անջատիչ կարող է լինել “վերևի” կամ “ներքևի” դիրքում: Յուրաքանչյուր անջատիչի համար այդ դիրքերից միայն մեկն է ճիշտ: Եթե անջատիչը ճիշտ դիրքում է, ապա նրան միացված դուռը կբացվի, իսկ, եթե անջատիչը ճիշտ դիրքում չէ, ապա նրան միացված դուռը կմնա փակ: Մի անջատիչի համար ճիշտը կարող է լինել վերևը, մյուսի համար՝ ներքևը, և դուք չգիտեք, թե որոնք են ճիշտ դիրքերը:

Դուք ուզում եք պարզել անվտանգության համակարգը: Դրա համար կարող եք անջատիչները դնել ինչ-որ դիրքերում և մտնել քարանձավ ու քայլել մինչև առաջին փակ դուռը: Դռները թափանցիկ չեն. առաջին փակ դռանը հասնելով դուք չեք կարող տեսնել նրա հետևում գտնվող դռները:

Դուք կարող եք փորձել անջատիչների $70,000$ կոմբինացիա, ոչ ավել: Ձեր խնդիրն է պարզել յուրաքանչյուր անջատիչի ճիշտ դիրքը և, թե որ դուռը որ անջատիչի հետ է կապված:

Իրականացումը

Դուք պետք է ուղարկեք թեստավորման մի ֆայլ, որտեղ իրականացված է `exploreCave()` ենթածրագիրը: Այն կարող է մինչև 70 000 անգամ կանչել գրեյդերի `tryCombination()` ֆունկցիան, և պետք է ավարտվի գրեյդերի `answer()` ենթածրագրի կանչով: Այս ֆունկցիաները և ենթածրագրերը նկարագրված են ստորև:

Գրեյդերի ֆունկցա. **`tryCombination()`**

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

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

Նկարագրություն

Այս ֆունկցիան կտրամադրվի գրեյդերի կողմից: Այն թույլ է տալիս փորձել անջատիչների մի կոմբինացիա, ապա մտնել քարանձավ և գտնել առաջին փակ դուռը: Եթե բոլոր դռները բաց են, ֆունկցիան վերադարձնում է `-1`: Այս ֆունկցիան աշխատում է `O(N)` ժամանակում, այսինքն նրա կատարման ժամանակը համեմատական է `N`-ին:

Այս ֆունկցիան կարող է կանչվել առավելագույնը `70,000` անգամ:

Պարամետրերը

- `S`: `N` երկարության զանգված, ցույց է տալիս յուրաքանչյուր անջատիչի դիրքը: `S[i]` տարրը համապատասխանում է `i` անջատիչին: `0` արժեքը ցույց է տալիս, որ անջատիչը վերևի դիրքում է, իսկ `1` արժեքը ցույց է տալիս, որ անջատիչը ներքևի դիրքում է:
- Ֆունկցիան վերադարձնում է առաջին փակ դռան համարը, կամ `-1`, եթե բոլոր դռները բաց են:

Գրեյդերի ենթածրագիր. **answer()**

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

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

Նկարագրություն

Կանչեք այս ենթածրագիրը, եթե դուք գտել եք բոլոր դռները բացող անջատիչների կոմբինացիա, և պարզել եք, թե որ անջատիչը որ դռանն է համապատասխանում:

Պարամետրերը

- `S`: `N` երկարության զանգված, ցույց է տալիս յուրաքանչյուր անջատիչի ճիշտ դիրքը: Ձևաչափը համապատասխանում է `tryCombination()` ֆունկցիայի՝ վերևում նկարագրված ձևաչափին:
- `D`: `N` երկարության զանգված, ցույց է տալիս, թե որ անջատիչը որ դռանն է համապատասանում: Մասնավորապես, `D[i]` տարրը պետք է պարունակի այն դռան համարը, որը համապատասխանում է `i` անջատիչին:
- *Returns*: Այս ենթածրագիրը ոչինչ չի վերադարձնում: Բայց այն կարող է ծրագրի ավարտի պատճառ հանդիսանալ:

Ձեր ենթածրագիրը. **exploreCave()**

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

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

Նկարագրություն

Ձեր ուղարկած ֆայլում պետք է լինի հետևյալ ենթածրագրի իրականացումը.

Այս ֆունկցիան պետք է օգտագործի գրեյդերի `tryCombination()` ենթածրագիրը պարզելու համար, անջատիչների ճիշտ դիրքերը, և, թե որ անջատիչը որ դռանն է համապատասխանում, և այդ ինֆորմացիան պազրելուց հետո պետք է կանչի `answer()` ֆունկցիան:

Պարամետրերը

- `N`: Քարանձավում անջատիչների և դռների քանակը:

Ծրագրի աշխատանքի օրինակ

Դիցուք, դռները և անջատիչները դասավորված են այնպես, ինչպես պատկերված է վերևի նկարում.

Ֆունկցիայի կանչ	Վերադարձնում է	**Explanation**
`tryCombination([1, 0, 1, 1])`	`1`	Սա համապատասխանում է նկարին: 0, 2 և 3 անջատիչները ներքևի դիրքում են, 1-ը՝ վերևի: Ֆունկցիան վերադարձնում է `1`, ինչը նշանակում է, որ ձախից առաջին փակ դուռը 1 դուռն է:
`tryCombination([0, 1, 1, 0])`	`3`	0, 1 և 2 դռները բաց են, իսկ 3 դուռը փակ է:
`tryCombination([1, 1, 1, 0])`	`-1`	0 անջատիչը ներքևի դիրքի փոխելուց հետո բոլոր դռները բաց են, քանի որ վերադարձի արժեքը `-1` է:
`answer([1, 1, 1, 0], [3, 1, 0, 2])`	*(Ծրագիրն ավարտվում է)*	Մենք գտել ենք, որ ճիշտ կոմբինացիան հետևյալն է. `[1, 1, 1, 0]`, և, որ 0, 1, 2 և 3 անջատիչները միացված են, համապատասխանաբար, 3, 1, 0 և 2 դռներին:

Սահմանափակումները

- Time limit: 2 seconds
- Memory limit: 32 MiB
- `1 ≤ N ≤ 5,000`

Ենթախնդիրներ

Ենթախնդիր	Միավոր	Լրացուցիչ սահմանափակումներ
1	12	Յուրաքանչյուր `i`-ի համար, `i` անջատիչը միացված է `i` դռանը: Ձեր խնդիրը, պարզապես, ճիշտ կոմիբնացիան գտնելն է:
3	21	`N ≤ 100`
4	30	`N ≤ 2,000`
5	24	*(None)*

Փորձարկում

Ձեր համակարգչում գտնվող գրեյդերը տվյալները կարդալու է `cave.in` ֆայլից, որը պետք է ունենա հետևյալ ձևաչափը.

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

Այստեղ `N`-ը դռների և անջատիչների քանակն է, `S[i]`-ն `i` անջատիչի ճիշտ դիրքն է, իսկ `D[i]`-ն այն դուռն է, որը միացված է `i` անջատիչին։

Վերևի օրինակի դեպքում այդ ֆայլն այսպիսի տեսք կունենա.

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

Դիտողություններ լեզուների վերաբերյալ

C/C++	You must `#include "cave.h"`.
Pascal	You must define the `unit Cave`, and you must also import the grader routines via `uses GraderHelpLib`. All arrays are numbered beginning at `0` (not `1`).

Օրինակների համար նայեք ձեր մեքենայում գտնվող ձևական լուծումները։