

Elevator

Vremensko ograničenje: 10 sekundi

Memorijsko ograničenje: 64 MiB (po procesu)

Ovo je komunikacijski zadatak.

Data je zgrada sa spratovima numerisanim od 0 do N . Sprat 0 je prizemlje, a iznad sprata N nalazi se krov. Dakle, uključujući krov, zgrada ima $N + 2$ nivoa.

Na svakom spratu f ($0 \leq f \leq N$) živi tačno jedan stanar i postoji tajni cijeli broj v_f ($0 \leq v_f \leq 3$), poznat samo tom stanaru. Dodatno, na krovu zgrade živi Karlson. Karlsonov cilj je odrediti što je moguće više vrijednosti $v_0, v_1, \dots, v_N$, pri čemu svaki stanar **sarađuje** sa ciljem da se maksimalno poveća broj vrijednosti koje on može rekonstruisati.

Za komunikaciju, stanari koriste lift u zgradi na sljedeći način.

Lift kreće sa sprata 0 i kreće se isključivo naviše. Unutar lifta nalaze se dugmad označena spratovima od 1 do N . Na početku, nijedno dugme nije pritisnuto. Kada se neko dugme pritisne, ono ostaje trajno pritisnuto. Lift će se zaustaviti i otvoriti na spratu f ako je $f = 0$ ili ako je dugme koje odgovara spratu f pritisnuto prilikom prethodnog zaustavljanja lifta. Nakon posjećivanja najvišeg sprata za koji je dugme pritisnuto (ili sprata 0 ako nijedno dugme nikada nije pritisnuto), lift se nastavlja kretati direktno do krova bez zaustavljanja na preostalim spratovima.

Kada se lift zaustavi na spratu f , dešava se naredno:

1. Stanar tog sprata vidi kompletan skup dugmadi koja su trenutno pritisnuta.
2. Na bazi isključivo te informacije i vrijednosti v_f , stanar može odabrati bilo koji podskup dugmadi koja trenutno nisu pritisnuta a koja odgovaraju spratovima koji su **striktno iznad** njegovog sprata. Stanar pritisne tu dugmad.
3. Lift ide naviše na naredni najniži sprat za koji je pritisnuto odgovarajuće dugme (ili na krov ako su svi spratovi za koje su odgovarajuća dugmad pritisnuta već posjećeni).

Ako se lift ne zaustavi na nekom spratu, stanar tog sprata ne može pritisnuti dugmad.

Kada lift stigne na krov, Karlson vidi samo konačni skup pritisnutih dugmadi. Koristeći tu informaciju, on će pokušati rekonstruisati što je moguće više vrijednosti $v_0, v_1, \dots, v_N$.

Vrijednosti v su određene prije pokretanja tvog programa i ne mijenjaju se tokom procesa.

Detalji implementacije

Biće ukupno T testnih slučajeva.

Trebaš poslati jedan fajl u kojem implementiraš dvije funkcije.

Prva funkcija koju trebaš implementirati je sljedeća:

```
std::vector<int> press_buttons(int subtask, int N,
                               int f, int v, std::vector<int> p)
```

- subtask : podzadatak za koji se funkcija poziva, pri čemu je $0 \leq \text{subtask} \leq 4$;
- N : broj posljednjeg sprata;
- f : trenutni sprat, pri čemu je $0 \leq f \leq N$;
- v : vrijednost v_f zapisana na spratu f ;
- p : trenutna pritisnuta dugmad, navedena u rastućem redoslijedu.

Ova funkcija se poziva kada se lift zaustavi i otvori na spratu f (ili je $f = 0$ ili je dugme koje odgovara spratu f pritisnuto na ranijem zaustavljanju). Funkcija se neće pozvati za specifičan sprat ako odgovarajuće dugme nije pritisnuto prilikom ranijih zaustavljanja lifta.

Funkcija treba vratiti niz novih dugmadi koje treba pritisnuti. Redoslijed vrijednosti u vraćenom nizu nije bitan. Svako pritisnuto dugme x mora zadovoljavati sljedeće uslove:

- $f < x \leq N$;
- x se ne nalazi u nizu p i x se pojavljuje samo jednom u vraćenom nizu.

Druga funkcija koju trebaš implementirati je sljedeća:

```
std::vector<int> answer(int subtask, int N, std::vector<int> p)
```

- subtask : podzadatak za koji se ova funkcija poziva, pri čemu je $0 \leq \text{subtask} \leq 4$;
- N : broj posljednjeg sprata;
- p : konačni skup pritisnutih dugmadi, navedenih u rastućem redoslijedu.

Ova funkcija se poziva tačno jednom po testnom slučaju, kada lift dođe na krov.

Vraćeni niz mora imati dužinu $N + 1$, pri čemu i -ti element tog niza treba imati vrijednost v_i , a ako se vrijednost **ne može rekonstruisati** onda vrijednost tog elementa treba biti -1 .

Bitno: Stanari zgrade ne mogu komunicirati na bilo koji drugi način osim putem pritisnutih dugmadi. Da bi se to osiguralo, tvoj program će se pokrenuti kao $N + 2$ odvojenih procesa, po jedan za svaki sprat i jedan za krov. U svakom procesu koji odgovara jednom spratu, biće najviše T poziva funkcije `press_buttons`, a u procesu za krov – tačno T poziva `answer` funkcije. Zbog toga tvoj program ne smije čuvati dijeljeno stanje (npr. globalne varijable). Svakom procesu biće

dostupno 64 MiB memorije, odvojene od memorije za ostale procese, a svi pozivi (najviše do $(N + 1) \times T$ poziva funkcije `press_buttons` i tačno T poziva funkcije `answer`) trebaju završiti u roku od 10 sekundi.

Ograničenja

- $N = 60$
- $T \leq 10\,000$
- $0 \leq v_i \leq 3$ za svako $0 \leq i \leq N$

Primjer

Testni primjer ima 1 testni slučaj sa sljedećim vrijednostima po spratovima:

```
1 1 0 1 1 1 1 1 1 0 1 1 1 0 1 0 1 1 0 1 1 0 1 0 1 0 1 1 1
1 0 1 1 0 1 1 1 0 1 0 1 0 1 1 0 1 1 1 1 0 1 1 1 1 0 1 1
```

Slijedi primjer interakcije:

Takmičarski program	Sudijski program
	<code>press_buttons(0, 60, 0, 1, {})</code>
<code>return {2}</code>	
	<code>press_buttons(0, 60, 2, 0, {2})</code>
<code>return {13, 42}</code>	
	<code>press_buttons(0, 60, 13, 0, {2, 13, 42})</code>
<code>return {}</code>	
	<code>press_buttons(0, 60, 42, 1, {2, 13, 42})</code>
<code>return {}</code>	
	<code>answer(0, 60, {2, 13, 42})</code>
<code>return {1, -1, 0, -1, -1, ..., -1}</code>	

Ova interakcija rekonstruiše 0-tu i 2-u vrijednost ispravno. Sve ostale vraćene vrijednosti su jednake -1 jer ih Karlson nije uspio rekonstruisati.

Podzadaci

Podzadatak	Bodovi	Dodatna ograničenja	Broj spratova čije vrijednosti treba rekonstruisati za puni broj bodova
0	0	Primjer iz teksta zadatka.	–
1	15	Ako je $v_i = 0$, tada je $v_{i+1} = 1$ za svako $0 \leq i < N$	60
2	35	$0 \leq v_i \leq 1$	40
3	15	$0 \leq v_i \leq 2$	30
4	35	$0 \leq v_i \leq 3$	25

Bodovanje

Ako tvoj program vrati netačnu rekonstruisanu vrijednost ili izvede neispravnu operaciju (npr. vrati niz pogrešne dužine) u bilo kojem testnom slučaju nekog podzadatka, tvoj pokušaj će dobiti 0 bodova za taj podzadatak.

Za jedan testni slučaj, *broj rekonstruisanih vrijednosti* tvog pokušaja jednak je broju pozicija čija vraćena vrijednost nije –1. Neka je K najmanji broj rekonstruisanih vrijednosti među svim testnim slučajevima jednog podzadatka (svaki podzadatak sadrži samo 1 test koji obuhvata do 10 000 testnih slučajeva). Bodovi za svaki podzadatak izračunavaju se prema tabeli u nastavku.

Podzadatak	Opseg K	Bodovi
0	–	0
1	$K < 60$	$0.25K$
	$K \geq 60$	15
2	$K < 30$	$0.5K$
	$30 \leq K < 40$	$2K - 45$
	$K \geq 40$	35
3	$K < 30$	$0.5K$
	$K \geq 30$	15
4	$K < 20$	$0.7K$
	$20 \leq K < 25$	$4K - 66$
	$K \geq 25$	35

Sample grader

Sample grader pravi sve pozive funkcija za sve testne slučajeve pri jednom pokretanju.

Format ulaza je sljedeći:

- linija 1: tri cijela broja – broj testnih slučajeva T , broj posljednjeg sprata N i broj podzadatka S ;
- linija $1 + i$: $N + 1$ cijeli brojevi $v_0, v_1, \dots, v_N$, pri čemu je v_f vrijednost zapisana na spratu f za taj testni slučaj.

Format izlaza je sljedeći:

- linija i : jedan cijeli broj – broj tačno rekonstruisanih vrijednosti v u testnom slučaju i ;
- linija $T + 1$: konačan broj bodova.

Za detaljnije povratne informacije, promijenite makro `DETAILED` iz `false` u `true` u prvoj liniji grader-a.

Možete pustiti grader-a da generiše testne slučajeve (vrijednosti v_f) izmjenom makro-a `AUTO_GENERATE` sa `false` na `true` u drugoj liniji grader-a.