

Lift

Időlimit: 10 másodperc

Memórialimit: 64 MiB (egy függvényre)

Ez egy kommunikációs feladat.

Van egy épület, melynek emeletei 0-tól N -ig vannak számozva. A 0. emelet a földszint, az N . emelet felett pedig a tetőtér van. Így, a tetőtérrel is beleértve, az épületnek $N + 2$ szintje van.

Minden f emeleten ($0 \leq f \leq N$) pontosan egy ember lakik, és minden emelethez tartozik egy v_f , ($0 \leq v_f \leq 3$) egész szám, amelyet csak az ott lakó ismer. Valamint ott van még Karlsson, aki a tetőtérben lakik. Karlsson feladata, hogy minél több $v_0, v_1, \dots, v_N$ értéket meghatározzon, miközben minden lakó **együttműködik** vele annak érdekében, hogy a lehető legtöbbet sikerüljön kiderítenie.

A lakók a kommunikációhoz a ház liftjét használják a következő módon .

A lift a 0. emeletről indul, és csak felfelé halad. A lift belsejében 1-től N -ig számozott emeleteket jelző gombok vannak. Kezdetben egyik gomb sincs benyomva. Ha egy gombot egyszer benyomnak, az onnantól végig benyomva marad. A lift az f emeleten pontosan akkor áll meg és nyílik ki, ha vagy $f = 0$, vagy az f emelethez tartozó gombot egy korábbi megálláskor benyomták. Miután a lift elérte a legmagasabb emeletet, amelynek gombját benyomták (vagy a 0. emeletet, ha egyetlen gombot sem nyomtak meg), akkor a lift a többi emeleten való megállás nélkül közvetlenül a tetőtérbe megy.

Amikor a lift az f . emeleten megáll, a következő történik:

1. Az ott lakó ember látja az összes eddig benyomott gombot.
2. Kizárólag ezen információ és a v_f érték alapján kiválaszthatja az eddig nem benyomott gombok bármely részhalmazát, amelyek a saját emeleténél **szigorúan magasabban** lévő emeleteknek felelnek meg. A lakó benyomja ezeket a gombokat.
3. A lift felmegy a következő legalacsonyabban lévő emeletre, amelynek megfelelő gombot benyomták (vagy a tetőtérbe, ha az összes, a benyomott gombnak megfelelő emeleten már volt).

Ha a lift nem áll meg egy bizonyos emeleten, akkor az ott lakó ember nem nyomhat meg semmilyen gombot.

Amikor a lift eléri a tetőteret, Karlsson csak a benyomott gombok végső halmazát látja. Ezen információk alapján megpróbálja kitalálni a lehető legtöbbet $v_0, v_1, \dots, v_N$ értékekből.

A v értékei a program indítása előtt rögzítésre kerülnek, és a futtatás során nem változnak.

Megvalósítás

T teszt esetet kell megoldani.

Küldj be egy fájlt, ami két függvényt valósít meg.

Az első függvény, amelyet meg kell valósítanod:

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

- $subtask$: a részfeladat, amihez ezt a függvényt hívják meg, ahol $0 \leq subtask \leq 4$;
- N : az utolsó emelet száma;
- f : az aktuális emelet, ahol $0 \leq f \leq N$;
- v : az f . emelethez tartozó v_f érték;
- p : a jelenleg benyomott gombok növekvő sorrendben.

Ezt a függvényt akkor hívják meg, amikor a lift az f . emeleten megáll (vagyis, ha $f = 0$, vagy az f . emelethez tartozó gombot egy korábbi emeletnél már benyomták). Ez a függvény egy adott emelet esetében nem lesz meghívva, ha a hozzá tartozó gombot korábban nem nyomták meg.

A visszatérési tömb a most benyomott gombok listája. A számsorozatban szereplő értékek sorrendje nem fontos. Minden benyomott x gombnak teljesítenie kell a következő feltételeket:

- $f < x \leq N$;
- x nem szerepel már a p -ben, és x pontosan egyszer fordul elő a visszatérési tömbben.

A második függvény, amelyet meg kell valósítanod:

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

- $subtask$: a részfeladat, amihez ezt a függvényt hívják meg, ahol $0 \leq subtask \leq 4$;
- N : az utolsó emelet száma;
- p : a benyomott gombok végső halmaza növekvő sorrendben.

Ez a függvény minden teszt esetben pontosan egyszer lesz meghívva, amikor a lift eléri a tetőteret.

A visszatérési tömb hossza $N + 1$ kell, hogy legyen, az i -edik eleme v_i , azokat az értékeket, amiket Karlsson **nem tud kitalálni** -1 -re kell állítani.

Fontos: Az épületben tartózkodó emberek kizárólag a benyomott gombok segítségével tudnak kommunikálni. Ennek biztosítása érdekében a programod $N + 2$ önálló folyamatként fog futni: egy-egy minden emeletre, valamint egy a tetőtérre. Az egyes emeleteket kezelő folyamatokban legfeljebb $T_{\text{press_buttons}}$ függvényhívás lesz, a tetőtérre vonatkozó folyamatban pedig pontosan T_{answer} függvényhívás. Ezért a programod nem dolgozhat semmilyen megosztott állapottal (például globális változókkal). Minden folyamathoz 64 MiB memória áll rendelkezésére, amely elkülönül a többi folyamat memóriájától, és az összes hívásnak (legfeljebb $(N + 1) \times T_{\text{press_buttons}}$ és pontosan T_{answer}) 10 másodpercen belül kell lefutnia.

Korlátok

- $N = 60$
- $T \leq 10000$
- $0 \leq v_i \leq 3$ minden $0 \leq i \leq N$ esetén

Példa

Az első példában 1 teszteset van, a következő értékekkel:

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

Itt egy példa interakció:

Saját programod	Értékelőprogram
	<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>	

Ez az interakció a 0. és a 2. értéket helyesen kitalálja. Az összes többi visszatérési érték -1 , mivel Karlsson azokat nem tudta kitalálni.

Részfeladatok

Részfeladat	Pontszám	Egyéb korlátok	A teljes pontszámhoz kitalálendő emeletek száma
0	0	A példa.	–
1	15	$0 \leq v_i \leq 1$ $v_0 = v_N = 1$ ha $v_i = 0$, akkor $v_{i+1} = 1$ minden $0 \leq i < N$ esetén	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

Pontozás

Ha a programod egy részfeladat bármelyik tesztelésében helytelenül talál ki egy értéket, vagy érvénytelen műveletet hajt végre (például rossz hosszúságú `vector`-t ad vissza), arra a részfeladatra 0 pontot kapsz.

Egy tesztelésre a megoldásod *visszaállítási száma* azon pozíciók száma, ahol visszatérési érték nem -1 . Legyen K a részfeladat összes tesztelése közötti legkisebb visszaállítási szám (minden részfeladatnak csak 1 tesztje van, amely legfeljebb 10 000 tesztelést tartalmaz). Ekkor a pontszámod az adott részfeladatra az alábbi táblázat alapján lesz kiszámolva.

Részfeladat	K intervalluma	Pontszám
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

Mintaértékelő

A mintaértékelő egyetlen futtatás során végrehajtja az összes tesztelés összes függvényhívását.

A bemenet formátuma a következő:

- 1. sor: három egész szám – a tesztelések száma (T), az utolsó emelet száma (N), és a részfeladat száma (S);
- $1 + i$. sor: $N + 1$ egész szám $v_0, v_1, \dots, v_N$, ahol v_f az f . emelethez tartozó érték, az adott tesztelésben.

A kimenet formátuma a következő:

- i . sor: egy egész szám – az i . tesztelésben helyesen visszaállított v értékek száma;
- $T + 1$. sor: a végső pontszám.

Részletesebb visszajelzésért állítsd a `DETAILED` makrót `false`-ról `true`-ra az értékelőprogram első sorában.

Az értékelőprogramban teszteseteket (v_f értékeket) generálhatsz úgy, hogy az `AUTO_GENERATE` makrót `false`-ról `true`-ra állítod az értékelőprogramban második sorában.