

Elevator

Временско ограничење: 10 секунди

Меморијско ограничење: 64 MB (по процесу)

Ово је проблем комуникације.

Постоји зграда са спратовима означеним од 0 до N . Ознаком 0 означено је приземље, а изнад спрата N је кров. Дакле, укључујући кров, зграда има $N + 2$ спратова.

На сваком спрату f ($0 \leq f \leq N$) је тачно један становник који има тајни цео број v_f ($0 \leq v_f \leq 3$), познат само њему. Такође, ту је и Карлсон који живи на крову ове зграде. Карлосов циљ је да одреди што више вредности $v_0, v_1, \dots, v_N$, док сваки становник **сарађује** како би максимизирао број вредности које може да одреди.

За комуникацију, станари користе лифт зграде на следећи начин.

Лифт креће на спрату 0 и креће се само навише. Унутар лифта налазе се дугмад означена бројевима спратова од 1 до N . На почетку, ниједно дугме није притиснуто. Када се дугме једном притисне, оно остаје трајно притиснуто. Лифт се зауставља и отвара на спрату f ако је $f = 0$ или је дугме које одговара спрату f притиснуто на претходном спрату. Након посете највишем спрату чије је дугме притиснуто (или спрату 0 ако се никада не притисне ниједно дугме), лифт наставља директно на кров без заустављања на преосталим спратовима.

Када се лифт заустави на спрату f , дешава се следеће:

1. Станар види комплетан скуп дугмади која су тренутно притиснута.
2. Само на основу ових информација и вредности v_f , може да изабере било који подскуп тренутно непритиснутих дугмади која одговарају спратовима строго изнад његовог спрата. Станар притиска ову дугмад.
3. Лифт иде до следећег најнижег (од преосталих изнад) спрата за који је одговарајуће дугме притиснуто (или до крова ако су посећени сви спратови који одговарају притиснутим дугмадима).

Ако се лифт не заустави на неком спрату, одговарајући станар не може да притисне ниједно дугме.

Када лифт стигне до крова, Карлсон види само последњи скуп притиснутих дугмади. Користећи ове информације, покушава да одреди што више вредности $v_0, v_1, \dots, v_N$.

Вредности v су фиксне пре покретања програма и не мењају се током процеса.

Детаљи имплементације

Биће T тест случајева.

Пошаљите једну фајл који имплементира ове две функције.

Прва функција коју треба да имплементирате је следећа:

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

- $subtask$: подзадатак за који се позива ова функција, где је $0 \leq subtask \leq 4$;
- N : број последњег спрата;
- f : тренутни спрат, где је $0 \leq f \leq N$;
- v : вредност v_f забележена на спрату f ;
- p : тренутно притиснута дугмад у растућем поредку.

Функција се позива када се врата лифта отварају на спрату f (ако је или $f = 0$ или је дугме које одговара спарту f било притиснуто раније). Ова функција неће бити позвана за одређени спрат ако одговарајуће дугме није било притиснуто раније.

Функција враћа низ нових дугмади које треба притиснути. Редослед вредности у враћеном низу није важан.

Свако притиснуто дугме x мора да задовољи следеће:

- $f < x \leq N$;
- x већ није у p и x се појављује тачно једном у враћеном низу.

Друга функција коју треба да имплементирате је следећа:

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

- $subtask$: подзадатак за који се позива ова функција, где је $0 \leq subtask \leq 4$;
- N : број последњег спрата;
- p : крајњи скуп притиснутих дугмади у растућем редоследу.

Ова функција се позива тачно једном по тестном случају када лифт стигне до крова.

Враћени низ мора имати дужину од $N + 1$, i -ти елемент у њему треба да буде једнак v_i , док вредности које Карлсон **не може да одреди** треба заменити са -1 .

Важно: Људи у згради не могу комуницирати ни на који други начин осим путем притиснутих дугмади. Да бисте то осигурали, ваш програм ће се извршавати као $N + 2$ одвојених процеса, по један за сваки спрат и један за кров. У сваком процесу за спратове, биће највише T позива функције `press_buttons`, а у процесу за кров – тачно T позива функције `answer`. Стога ваш програм не би требало да претпоставља никакво дељено стање (нпр. глобалне променљиве). Сваки процес ће имати на располагању 64 MB меморије која је одвојена од меморије других процеса, и сви ти позиви (до $(N + 1) \times T$ `press_buttons` и тачно T `answer`) морају се завршити за 10 секунди.

Ограничења

- $N = 60$
- $T \leq 10\,000$
- $0 \leq v_i \leq 3$ за свако $0 \leq i \leq N$

Пример

Пример теста има 1 тест пример са следећим вредностима спратова:

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

Ово је пример интеракције:

Програм учесника	Програм жирија
	<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>	

Ова интеракција враћа нулту и другу вредност исправно. Све друге враћене вредности су једнаке -1 како их Карлсон није открио.

Подзадаци

Подзадатак	Поени	Додатна ограничења	Број спратова за које треба одредити вредност да би се добили сви поени
0	0	Пример.	–
1	15	$0 \leq v_i \leq 1$ $v_0 = v_N = 1$ ако је $v_i = 0$ онда је $v_{i+1} = 1$ за свако $0 \leq i < N$	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

Бодовање

Ако ваш програм врати нетачну вредност или изврши неважећу операцију (нпр. врати `vector` погрешне дужине) у било ком од тест случајева за подзадатак, ваш предлог ће добити нула поена за тај подзадатак.

За тест случај, *откривени бројач* вашег послатог решења је број позиција чија враћена вредност није -1 . Нека K буде минимални откривени бројач између свих тест случајева подзадатка (сваки подзадатак има само 1 тест који садржи до 10 000 тест случајева). Затим се поени за сваки подзадатак израчунавају према табели испод.

Подзадатак	Интервал K	Поени
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

Пример грејдера

Грејдер врши све позиве функција за све тест примере током једног извршавања.

Формат улазних података је следећи:

- линија 1: три цела броја – број тест примера T , број последњег спарата N и број подзадатка S ;
- линија $1 + i$: $N + 1$ целих бројева $v_0, v_1, \dots, v_N$, где је v_f вредност забележена на спрату f за тест пример.

Формат илзаза је следећи:

- линија i : један цео број – број исправно откривених вредности v у тест примеру i ;
- линија $T + 1$: коначни поени.

За детаљније повратне информације, промените макро `DETAILED` из `false` у `true` у првој линији грејдера.

Можете да пустите грејдер да генерише тест случајеве (вредности v_f) за Вас тако што ћете променити макро `AUTO_GENERATE` из `false` у `true` у другој линији грејдера.