D. Попис (census)

Time limit: 1 seconds
Memory limit: 128 MiB

Помалку познат факт за Чезенатико е дека таму има тајно друштво од N информатичарки. Ова друштво е навистина многу тајно; ниедна членка не познава ниедна друга. Секоја членка има уникатно ID: ненегативен цел број I.

Единствената комуникација меѓу членките е индиректна, преку броеви напишани со креда на различни локации низ градот. На секои 100 години, друштвото прави попис за да ги изброи своите членки. Откако ќе заврши пописот, секоја членка треба да го знае вкупниот број на членки во друштвото.

Пописот се одвива повеќе денови. Секој ден, секоја членка што сè уште учествува во процесот ќе избере и направи точно едно нешто: да **чита**, да **пишува** или да **престане** со учество.

- Ако некоја членка избере да **чита**, таа бира локација P. Преку ден, таа оди на локацијата P и го чита бројот што е напишан таму.
- Ако избере да **пишува**, бира локација P и број V. Доцна вечерта, оди на локацијата P и го менува бројот што бил напишан таму во V. Бидејќи веќе е темно, не може да го прочита стариот број пред да го напише новиот.
- Ако избере да **престане**, веќе не презема никакви акции во следните денови.

Ако една членка види друга како пишува број, може да ја препознае. Затоа, строго е забрането две или повеќе членки да изберат да пишуваат на иста локација во истиот ден. (Ова не важи за читањето, бидејќи тоа може да се направи дискретно.)

Ако една или повеќе членки читаат од локација каде што друга членка сака да пишува истиот ден, сите читања се случуваат пред пишувањето.

Како треба друштвото да го испланира својот попис за да го минимизира бројот на денови дури сите не го дознаат точниот број на членки?

За да го разберете комплетно проблемот, внимателно прочитајте ги останатите делови од задачата.

Implementation

> ⇨ Ова е интерактивна задача, каде што непознат број инстанци ($1 \leq N \leq 100$) од твојата програма ќе се извршуваат истовремено. Секоја инстанца симулира една членка на друштвото.

Има 10^{18} локации. Бројот P на некоја локација мора да исполнува $0 \leq P < 10^{18}$. На почетокот, вредноста запишана на сите локации е $V = 0$.

Новата вредност V што се пишува на локација мора секогаш да биде цел број таков што $0 \leq V \leq 10^9$. Во повеќето подзадачи, V може да биде само 0 или 1. Види го делот Бодување за повеќе детали.

Кога ќе стартува инстанца од твојата програма, прво треба да прочита линија со два цели броеви, I и M ($0 \leq I \leq M - 1$): уникатното ID на членката на друштвото што ја претставува оваа инстанца

и вкупниот број на можни ID-а. Во секој тест пример, сите инстанци ќе добијат иста вредност M и различни вредности за I. Имај предвид дека може да има ID-а кои не се доделени на ниедна членка.

Потоа, за секој ден во процесот на попис, твојата програма треба да ја избере акцијата што сака да ја направи и соодветно да испечати линија:

Акција	**Значење**
`r` P	**Читај** од локација P. Откако ќе ја испечатиш оваа линија, твојата програма треба да прочита линија со моменталната вредност запишана на P.
`w` P V	**Запиши** на локација P нова вредност V. Ако повеќе инстанци запишуваат на иста локација P во истиот ден, ќе добиеш пресуда *Not correct*. Освен за примерите и подзадачата 3, мора да запишеш $0 \leq V \leq 1$; види го делот Бодување.
`!` N	**Одговори и престани**: пријави дека има N членки и престани со учество во пописот. По одговорот, **твојата програма треба нормално да заврши**. (Забележи дека други инстанци од твојата програма може да продолжат да работат уште неколку дена пред и тие да одговорат и завршат.)

Ако која било инстанца од твојата програма одговори со погрешна вредност за N, го прекрши протоколот, користи повеќе од 500 дена или го надмине лимитот за време/меморија (по процес), твоето решение ќе биде оценето како *Not correct* за дадениот тест пример.

Инаку, твојата програма ќе биде *(Делумно) Точна* на тест примерот и ќе биде бодувана врз основа на вредноста D: максималниот број на денови што ѝ требале на која било инстанца за да одговори. За целосни поени, треба да го решиш секој тест пример со $D \leq 61$ и $V \leq 1$. Види го делот Бодување за детали.

Чистење на баферот (Flushing). Ако не ги користиш дадените темплејти, погрижи се да го исчистиш стандардниот излез (flush) по печатењето на секоја линија, инаку програмата може да добие *Not correct*. Во Python, ова се случува автоматски ако користиш `input()` за читање линии. Во C++, `cout << endl;` прави flush покрај тоа што печати нов ред; ако користиш printf, користи `fflush(stdout)`.

Constraints

- $1 \leq N \leq 100$.
- $1 \leq M \leq 100\,000$.
- Можеш да користиш најмногу 500 дена.

Scoring

Твојата програма ќе биде тестирана на повеќе тест примери групирани во подзадачи. За да добиеш поени за дадена подзадача, мора точно да ги решиш сите тестови во неа.

- **Subtask 0** [**0 points**]: Примери (можеш да го запишеш кој било цел број $0 \leq V \leq 1\,000\,000\,000$).
- **Subtask 1** [**11 points**]: $M \leq 100$, и N-те членки имаат ID-а $0, 1, ..., N-1$.
- **Subtask 2** [**12 points**]: $1 \leq N \leq 2$.
- **Subtask 3** [**22 points**]: $M \leq 8000$, и да го запишеш кој било цел број $0 \leq V \leq 1\,000\,000\,000$.
- **Subtask 4** [**55 points**]: Нема дополнителни ограничувања.

Во подзадачите 1, 2 и 4, можеш да запишеш само $V = 0$ или $V = 1$ при секое запишување.

Нека X_s бидат максималните поени за подзадачата s (прикажани погоре), а D_s најголемиот број на денови што која било од твоите програми го користи на тест во подзадачата s. Тогаш:

$$\text{score}_s = \begin{cases} X_s & \text{ако } D_s \leq 61 \\ X_s \cdot \left(0.2 + 0.8 \cdot 1.01^{(60-D_s)}\right) & \text{ако } 61 < D_s \leq 500 \\ 0 & \text{ако } 500 < D_s. \end{cases}$$

Вредноста на score_s се заокружува на најблискиот цел број по подзадача, а твоите вкупни поени се збир од овие вредности. За да ги добиеш сите поени за задачата, ти треба $D \leq 61$ и $V \leq 1$ на секој тест пример.

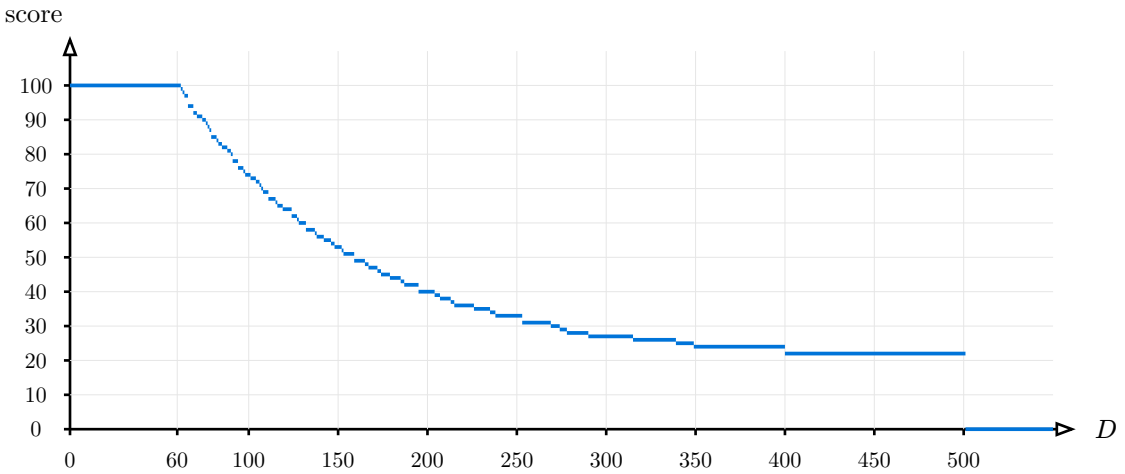

Figure 1: Вкупно поени, под претпоставка дека секоја подзадача е решена со исто максимално D.

Examples

Прв пример. Секој пар колони ја покажува комуникацијата помеѓу оценувачот и една инстанца.

Оцен.	Инст. 0	Оцен.	Инст. 1	Оцен.	Инст. 2	Оцен.	Инст. 3	Оцен.	Инст. 4
0 100		1 100		2 100		3 100		4 100	
	w 12 1		w 50 1		w 99 0		w 7 1		r 5
								0	
	r 50		r 7		r 12		w 1 1		! 5
1		1		1					
	! 5		r 1		w 0 0		! 5		
		1							
			! 5		! 5				

Втор пример.

Оценувач	Инстанца 0
0 8000	
	w 0 0
	w 1 1
	r 2
1	
	! 2

Оценувач	Инстанца 1
3 8000	
	w 2 1
	r 1
0	
	r 2
1	
	r 1
1	
	! 2

Explanation

Прв пример. Имаме $N = 5$ членки со последователни ID-а $0, 1, 2, 3, 4$ и $M = 100$ (валидно за подзадачите 1, 3 и 4). Инстанцата i одговара на членката со ID i. Интеракцијата погоре е само една можна валидна низа од операции и **не** е замислена да биде ефикасна или логична стратегија; прикажана е само за да илустрира како работи протоколот.

Втор пример. Имаме $N = 2$ членки, со ID-а 0 и 3, и $M = 8000$ (валидно за подзадачите 2, 3 и 4). Првиот ден, членката со ID 0 запишува 0 на локација 0 (нема промена), а членката со ID 3 запишува 1 на локација 2.

location	0	1	2	3	4	...
value	0	0	1	0	0	...

Вториот ден, ID 0 запишува 1 на локација 1, а ID 3 чита од истата локација. Забележи дека читањето се случува преку ден, пред запишувањето вечерта. Затоа, ID 3 сè уште гледа 0.

location	0	1	2	3	4	...
value	0	1	1	0	0	...

Третиот ден, двете читаат од локација 2, каде што е запишана вредноста 1.

Четвртиот ден, ID 0 одговара дека има 2 членки (точно), додека ID 3 ја чита 1-ката на локација 1. ID 0 веднаш завршува по ова и не учествува во следните денови.

На крај, на денот $D = 5$, преостанатата членка исто така точно одговара дека $N = 2$.

Тестирање

За да го олесниме тестирањето на твоето решение, обезбедуваме едноставна алатка која можеш да ја преземеш од CMS. Користењето на алатката е по избор. Имај предвид дека официјалниот оценувач на CMS е различен од оваа алатка за тестирање.

За да ја користиш алатката ти треба влезна датотека. Можеш да ги користиш дадените влезни примери `census.input0.txt` и `census.input1.txt`, или да направиш свои. Влезната датотека треба да почнува со бројот на членки N и можни ID-а M, по што следи линија со N броеви кои ги претставуваат ID-ата на членките на друштвото.

За програми во Python, на пример `census.py` (кои нормално се стартуваат со `pypy3 census.py`) пушти ја алатката за тестирање вака:

```
python3 testing_tool.py pypy3 census.py < census.input0.txt
```

За програми во C++, прво компајлирај го твоето решение:

```
g++ -DEVAL -std=gnu++20 -O2 -pipe -static -s -o census census.cpp
```

и потоа пушти ја алатката за тестирање:

```
python3 testing_tool.py ./census < census.input0.txt
```

Имај предвид дека во оваа задача стандардниот излез се користи за комуникација со оценувачот, па не треба да го користиш за дебагирање. Наместо тоа, можеш да го користиш стандардниот излез за грешки (stderr). Во C++ можеш да користиш `cerr << msg << endl;`. Во Python можеш да користиш `print(msg, file=sys.stderr)`.

Алатката за тестирање ќе ги прочита и прикаже овие stderr пораки заедно со прашањата направени од сите инстанци на твојата програма. Имај предвид дека поради технички причини, тие може да се појават малку несинхронизирано едни со други.