D. Перепись (census)

Ограничение по времени: 1 секунды
Ограничение по памяти: 128 MiB

Малоизвестный факт о Чезенатико заключается в том, что здесь находится тайное общество, состоящее из N программисток. Это общество очень тайное; его члены ничего не знают о друг друге. У каждой участницы есть уникальный идентификатор (ID): неотрицательное целое число I.

Единственный способ общения между участницами — косвенный: с помощью чисел, записанных мелом в разных местах города. Раз в 100 лет общество проводит перепись, чтобы подсчитать количество своих участниц. После завершения переписи каждая участница должна знать общее число членов общества.

Перепись проходит в течение нескольких дней. Каждый день каждая участница, которая всё ещё участвует в процессе, выбирает и выполняет ровно одно действие: **прочитать**, **записать** или **остановиться**.

- Если участница выбирает **прочитать**, она выбирает место P. Днём она посещает место P и читает число, которое там записано.
- Если участница выбирает **записать**, она выбирает место P и число V. Поздно вечером она посещает место P и меняет записанное там число на V. Поскольку уже темно, она не может прочитать старое число перед тем, как записать новое.
- Если участница выбирает **остановиться**, она больше не выполняет никаких действий в последующие дни.

Если одна участница увидит, как другая записывает число, она может её узнать. Поэтому строго запрещено двум или более участницам выполнять действие **записать** в одном и том же месте в один и тот же день. (Для операции **прочитать** такого ограничения нет, так как это можно сделать незаметно.)

Если одна или несколько участниц выбирают **прочитать** в том же месте, где другая участница выбрала **записать** в тот же день, то все операции чтения происходят перед записью.

Как обществу спланировать процесс переписи, чтобы минимизировать количество дней до того момента, когда каждая узнает правильное количество участниц?

Реализация

⇨ Это интерактивная задача, в которой одновременно запускается неизвестное количество экземпляров ($1 \leq N \leq 100$) вашей программы. Каждый экземпляр имитирует одну участницу общества.

Всего существует 10^{18} мест. Номер места P должен удовлетворять условию $0 \leq P < 10^{18}$. Изначально во всех местах записано значение $V = 0$.

Новое значение V, записываемое в выбранном месте, всегда должно быть целым числом, таким что $0 \leq V \leq 10^9$. В большинстве подзадач V может принимать только значения 0 или 1. Подробности см. в разделе «Система оценки».

При запуске экземпляра вашей программы он должен сначала считать строку с двумя целыми числами I и M ($0 \leq I \leq M-1$): уникальный ID участницы, которую представляет этот экземпляр, и общее число возможных ID. В рамках каждого теста все экземпляры получат одинаковое значение M и различные значения I. Обратите внимание, что могут быть ID, не присвоенные ни одной участнице.

Затем, для каждого дня процесса переписи ваша программа должна выбрать действие, которое она хочет выполнить, и вывести соответствующую строку:

Действие	**Значение**
`r` P	**Прочитать** место P. После вывода этой строки ваша программа должна считать строку с текущим значением, записанным в месте P.
`w` P V	**Записать** в место P новое значение V. Если несколько экземпляров записывают в одно и то же место P в один и тот же день, вы получите вердикт *Not correct*. За исключением примеров и подзадачи 3, вы должны записывать $0 \leq V \leq 1$; см. раздел «Система оценки».
`!` N	**Остановиться и ответить**: сообщить, что в обществе N членов, и перестать участвовать в переписи. После ответа **ваша программа должна нормально завершиться**. (Обратите внимание, что другие экземпляры вашей программы могут продолжать работать еще несколько дней, прежде чем они ответят и завершатся.)

Если какой-либо экземпляр вашей программы ответит неверным значением N, нарушит протокол, использует более 500 дней или превысит ограничение по времени/памяти (на процесс), ваша посылка получит вердикт *Not correct* для данного теста.

В противном случае ваша программа будет *(Partially) Correct* на данном тесте, а оценка будет зависеть от значения D: максимального количества дней, которое потребовалось любому из экземпляров для ответа. Для получения полного балла необходимо решить каждый тест с $D \leq 61$ и $V \leq 1$. Подробности см. в разделе «Система оценки».

Буферизация (Flushing). Если вы не используете предоставленные шаблоны, обязательно очищайте стандартный вывод после вывода каждой строки, иначе программа может получить вердикт *Not correct*. В Python это происходит автоматически, если вы используете `input()` для чтения строк. В C++ команда `cout << endl;` выполняет очистку буфера; если используете printf, используйте `fflush(stdout)`.

Ограничения

- $1 \leq N \leq 100$.
- $1 \leq M \leq 100\,000$.
- Вы можете использовать не более 500 дней.

Система оценки

Ваша программа будет протестирована на нескольких наборах тестов, разбитых на подзадачи. Чтобы получить баллы за подзадачу, ваша программа должна пройти все тесты в ней.

- **Подзадача 0 [0 баллов]**: Примеры (вы можете записывать любое целое число $0 \leq V \leq 1\,000\,000\,000$).
- **Подзадача 1 [11 баллов]**: $M \leq 100$, и N членов имеют ID $0, 1, ..., N-1$.

- **Подзадача 2 [12 баллов]**: $1 \le N \le 2$.
- **Подзадача 3 [22 баллов]**: $M \le 8000$, и вы можете записывать любое целое число $0 \le V \le 1\,000\,000\,000$.
- **Подзадача 4 [55 баллов]**: Дополнительных ограничений нет.

В подзадачах 1, 2 и 4 вы можете записывать только $V = 0$ или $V = 1$ при каждом действии записи.

Пусть X_s — максимальный балл за подзадачу s (указан выше), а D_s — наибольшее количество дней, которое использовала любая из ваших программ на тесте в подзадаче s. Тогда:

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

Значение score_s округляется до ближайшего целого для каждой подзадачи, а ваш итоговый балл — это сумма баллов за все подзадачи. Чтобы получить полный балл за задачу, необходимо, чтобы $D \le 61$ и $V \le 1$ на каждом тесте.

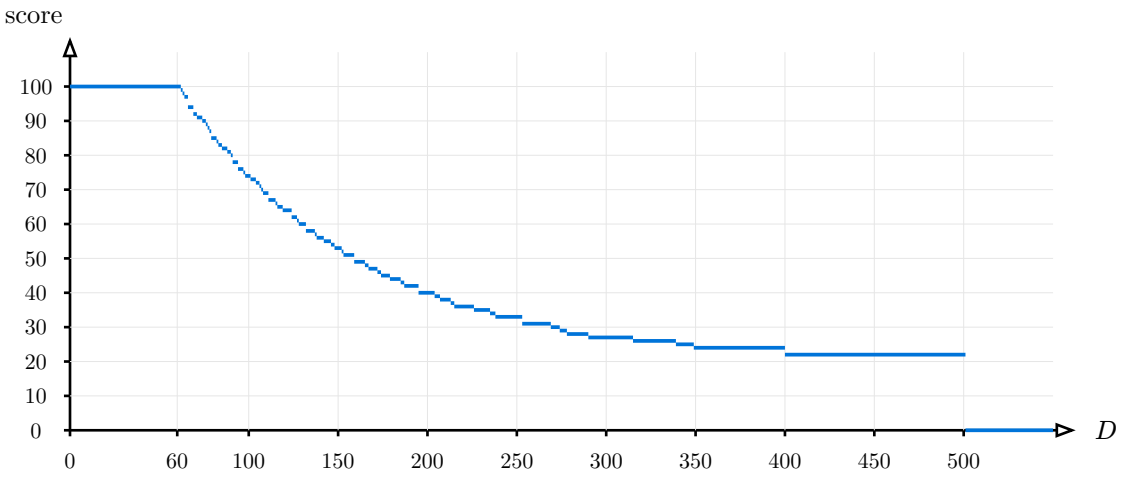


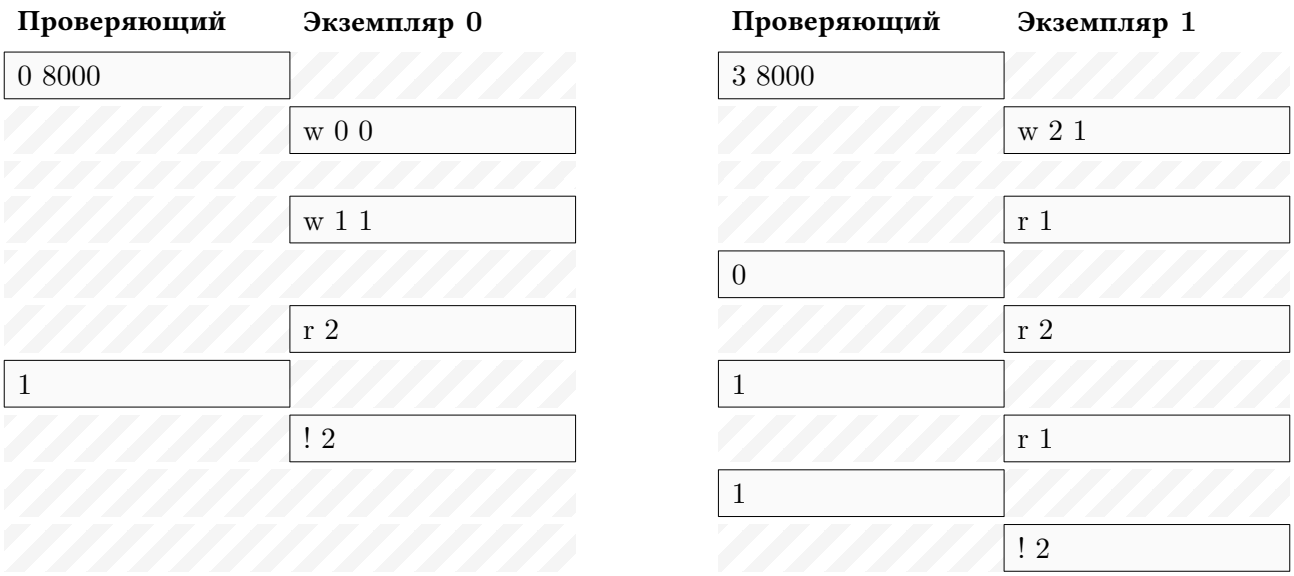
Рис. 1. Итоговый балл, если каждая подзадача решена с одинаковым максимальным D.

Примеры ввода/вывода

Первый пример. Каждая пара столбцов показывает взаимодействие между проверяющим и одним экземпляром.

Пров.	Экз. 0	Пров.	Экз. 1	Пров.	Экз. 2	Пров.	Экз. 3	Пров.	Экз. 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				

Второй пример.



Пояснение

Первый пример. У нас $N = 5$ членов с последовательными ID $0, 1, 2, 3, 4$ и $M = 100$ (подходит для подзадач 1, 3 и 4). Экземпляр i соответствует члену с ID i. Взаимодействие выше — это просто одна из возможных легальных последовательностей операций, она **не** является эффективной или разумной стратегией; она показана только для иллюстрации того, как работает протокол.

Второй пример. У нас $N = 2$ члена с ID 0 и 3, и $M = 8000$ (подходит для подзадач 2, 3 и 4). В первый день член с ID 0 записывает 0 в место 0 (без изменений), а член с ID 3 записывает 1 в место 2.

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

На второй день ID 0 записывает 1 в место 1, а ID 3 читает то же место. Заметим, что чтение происходит днём, до записи которая происходит вечером. Поэтому ID 3 всё ещё видит 0.

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

На третий день они оба читают место 2, где записана 1.

На четвёртый день ID 0 отвечает, что членов 2 (верно), в то время как ID 3 читает 1 из места 1. ID 0 немедленно завершает работу после этого и не участвует в последующие дни.

Наконец, на 5-й день оставшийся член также правильно отвечает $N = 2$.

Тестирование

Чтобы упростить тестирование вашего решения, мы предоставляем простой инструмент, который можно скачать из CMS. Использование инструмента опционально. Обратите внимание, что официальный проверяющий (grader) на CMS отличается от этого инструмента тестирования.

Для использования инструмента вам понадобится входной файл. Вы можете использовать примеры входных файлов `census.input0.txt` и `census.input1.txt` или создать свои. Входной файл должен начинаться с числа членов N и количества возможных ID M, за которыми следует строка с N числами, задающими ID членов общества.

Для программ на Python, скажем `census.py` (обычно запускаемых как `pypy3 census.py`), запустите инструмент тестирования следующим образом:

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

Для программ на C++ сначала скомпилируйте ваше решение:

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

а затем запустите инструмент тестирования:

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

Обратите внимание, что в этой задаче стандартный вывод используется для связи с проверяющим, поэтому его нельзя использовать для отладки. Вместо этого вы можете использовать стандартный вывод ошибок (stderr). В C++ можно использовать `cerr << msg << endl;`. В Python можно использовать `print(msg, file=sys.stderr)`.

Инструмент тестирования прочитает и выведет эти сообщения stderr вместе с запросами, выполненными всеми экземплярами вашей программы. Обратите внимание, что по техническим причинам они могут отображаться слегка рассинхронизированно друг относительно друга.