# D. Censo (census)

Límite de tiempo: 1 segundos
Límite de memoria: 128 MiB

Un dato poco conocido sobre Cesenatico es que es el hogar de una sociedad secreta de $N$ informachicas (chicas informáticas). Esta sociedad es, de hecho, muy secreta; ninguna informachica conoce a las demás. Cada informachica tiene un ID único: un número entero no negativo $I$.

La única comunicación entre las informachicas es indirecta, mediante números escritos con tiza en diferentes ubicaciones por toda la ciudad. Cada 100 años, la sociedad realiza un censo para contar a sus integrantes. Después de completar el censo, cada informachica debe conocer el número total de integrantes de la sociedad.

El censo se lleva a cabo durante varios días. Cada día, cada integrante que aún participe en el proceso elegirá y realizará exactamente una acción: **leer**, **escribir** o **detenerse**.

- Si una integrante elige **leer**, elige una ubicación $P$. Durante el día, visita la ubicación $P$ y lee el número que está escrito ahí.
- Si una integrante elige **escribir**, elige una ubicación $P$ y un número $V$. Al atardecer, visita la ubicación $P$ y cambia el número que estaba escrito ahí por $V$. Como ya está oscuro, no puede leer el número anterior antes de escribir el nuevo.
- Si una integrante elige **detenerse**, ya no realiza ninguna acción en los días siguientes.

Si una integrante ve a otra escribir un número, podría reconocerla. Por lo tanto, está estrictamente prohibido que dos o más integrantes elijan escribir en la misma ubicación el mismo día. (No existe tal restricción para leer, ya que eso puede hacerse discretamente).

Si una o más integrantes leen de una ubicación donde otra integrante quiere escribir el mismo día, todas las lecturas ocurren antes de la escritura.

¿Cómo debería planear la sociedad su proceso de censo para minimizar el número de días hasta que todas conozcan el conteo correcto de informachicas?

## Implementación

⇨ Este es un problema interactivo, en el cual un número desconocido de instancias ($1 \le N \le 100$) de tu programa se ejecutarán simultáneamente. Cada instancia simula a una integrante de la sociedad.

Hay $10^{18}$ ubicaciones. El número $P$ de una ubicación debe satisfacer $0 \le P < 10^{18}$. Inicialmente, el valor escrito en todas las ubicaciones es $V = 0$.

El nuevo valor $V$ escrito en una ubicación siempre debe ser un entero tal que $0 \le V \le 10^9$. En la mayoría de las subtareas, $V$ solo puede ser 0 o 1. Consulta la sección "Puntuación" para más detalles.

Cuando una instancia de tu programa inicia, primero debe leer una línea con dos enteros, $I$ y $M$ ($0 \le I \le M - 1$): el ID único de la integrante de la sociedad representada por esta instancia y el número total de IDs posibles. Dentro de cada caso de prueba, todas las instancias obtendrán el mismo valor $M$ y valores distintos $I$. Nota que puede haber IDs que no estén asignados a ningún miembro.

Luego, por cada día en el proceso de censo, tu programa debe elegir la acción que quiere realizar e imprimir una línea de acuerdo a esto:

| Acción | Significado |
|---|---|
| `r` $P$ | **Leer** la ubicación $P$.<br>Después de imprimir esta línea, tu programa debe leer una línea con el valor actual escrito en $P$. |
| `w` $P$ $V$ | **Escribir** en la ubicación $P$ el nuevo valor $V$.<br>Si varias instancias de tu programa escriben en la misma ubicación $P$ el mismo día, obtendrás el veredicto *"Not correct"*.<br>Excepto por los ejemplos y la subtarea 3, debes escribir $0 \leq V \leq 1$; consulta la sección de Puntuación. |
| `!` $N$ | **Responder y detenerse**: reporta que hay $N$ informachicas y deja de participar en el censo.<br>Después de responder, **tu programa debe terminar normalmente**. (Toma en cuenta que otras instancias de tu programa pueden seguir ejecutándose por días adicionales antes de responder y terminar.) |

Si alguna instancia de tu programa responde con un valor incorrecto de $N$, viola el protocolo, usa más de 500 días, o excede el límite de tiempo/memoria (por proceso), tu envío será juzgado como *"Not correct"* para el caso de prueba dado.

De lo contrario, tu programa será juzgado *"(Partially) Correct"* en el caso de prueba y calificado con base en el valor $D$: el número máximo de días que cualquier instancia tardó en responder. Para obtener el puntaje completo, necesitas resolver cada caso de prueba con $D \leq 61$ y $V \leq 1$. Consulta la sección "Puntuación" para más detalles.

**Limpieza de búfer (Flushing).** Si no estás usando las plantillas proporcionadas, asegúrate de limpiar la salida estándar (flush) después de imprimir cada línea, o de lo contrario tu programa podría trabarse y ser juzgado como *"Not correct"*. En Python, esto sucede automáticamente si usas `input()` para leer líneas. En C++, `cout << endl;` limpia el búfer además de imprimir un salto de línea; si usas printf, usa `fflush(stdout)`.

## Restricciones

- $1 \leq N \leq 100$.
- $1 \leq M \leq 100\,000$.
- Puedes usar como máximo 500 días.

## Puntuación

Tu programa será probado en varios casos de prueba agrupados en subtareas. Para obtener el puntaje de una subtarea, debes resolver correctamente todas las pruebas que contiene.

- **Subtarea 0 [ 0 puntos]**: Ejemplos (puedes escribir cualquier entero $0 \leq V \leq 1\,000\,000\,000$).
- **Subtarea 1 [11 puntos]**: $M \leq 100$, y los $N$ miembros tienen IDs $0, 1, ..., N-1$.
- **Subtarea 2 [12 puntos]**: $1 \leq N \leq 2$.
- **Subtarea 3 [22 puntos]**: $M \leq 8000$, y puedes escribir cualquier entero $0 \leq V \leq 1\,000\,000\,000$.
- **Subtarea 4 [55 puntos]**: Sin restricciones adicionales.

**En las subtareas 1, 2 y 4, solo puedes escribir $V = 0$ o $V = 1$ en cada acción Escribir.**

Sea $X_s$ el máximo de puntos para la subtarea $s$ (mostrado arriba), y $D_s$ el mayor número de días que cualquiera de tus programas usa en cualquier caso de prueba de la subtarea $s$. Entonces:

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

El valor de $\text{puntaje}_s$ se redondea al entero más cercano por subtarea, y tu puntaje total es la suma de estos. Para obtener el puntaje completo, necesitas $D \leq 61$ y $V \leq 1$ en cada caso de prueba.

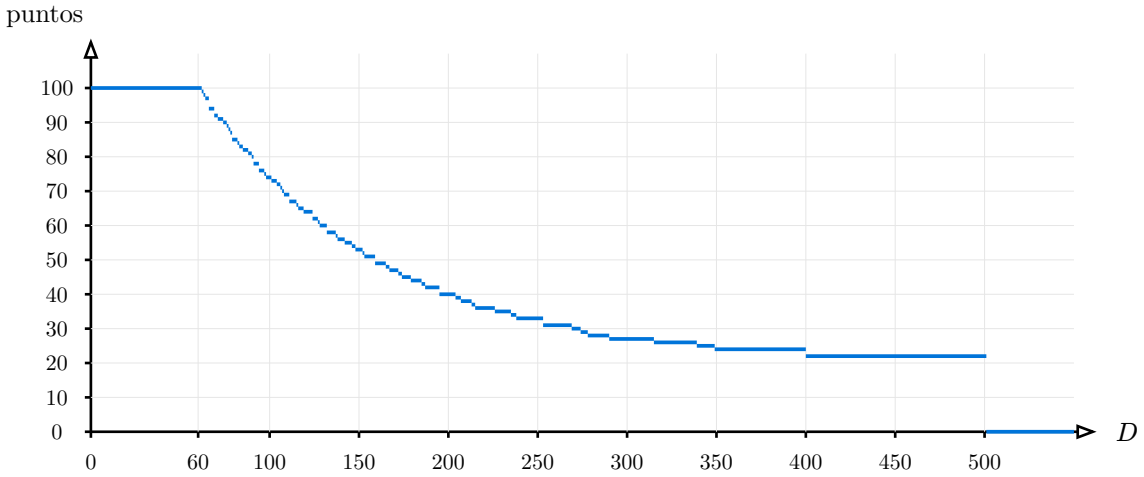


Figura 1: Puntaje total, asumiendo que cada subtarea se resuelve con el mismo máximo $D$.

## Ejemplos

Primer ejemplo. Cada par de columnas muestra la comunicación entre el evaluador y una instancia.

| Eval. | Inst. 0 | Eval. | Inst. 1 | Eval. | Inst. 2 | Eval. | Inst. 3 | Eval. | Inst. 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 | | | | |

Segundo ejemplo.

| Evaluador | Instancia 0 |
|---|---|
| 0 8000 | |
| | w 0 0 |
| | w 1 1 |
| | r 2 |
| 1 | |
| | ! 2 |

| Evaluador | Instancia 1 |
|---|---|
| 3 8000 | |
| | w 2 1 |
| | r 1 |
| 0 | |
| | r 2 |
| 1 | |
| | r 1 |
| 1 | |
| | ! 2 |

## Explicación

**Primer ejemplo.** Tenemos $N = 5$ integrantes con IDs consecutivos $0, 1, 2, 3, 4$ y $M = 100$ (válido para las subtareas 1, 3 y 4). La instancia $i$ corresponde al miembro con ID $i$. La interacción en la sección anterior es solo una posible secuencia legal de operaciones y **no** pretende ser una estrategia eficiente o bien pensada; se muestra solo para ilustrar cómo funciona el protocolo.

**Segundo ejemplo.** Tenemos $N = 2$ miembros, con IDs 0 y 3, y $M = 8000$ (válido para las subtareas 2, 3 y 4). En el primer día, el miembro con ID 0 escribe un 0 en la ubicación 0 (sin cambios), y el miembro con ID 3 escribe un 1 en la ubicación 2.

| **ubicación** | 0 | 1 | 2 | 3 | 4 | ... |
|---|---|---|---|---|---|---|
| **valor** | 0 | 0 | 1 | 0 | 0 | ... |

En el segundo día, el ID 0 escribe un 1 en la ubicación 1, y el ID 3 lee esa misma ubicación. Observa que la lectura ocurre durante el día, antes de la escritura en la noche. Por lo tanto, el ID 3 todavía ve un 0.

| **ubicación** | 0 | 1 | 2 | 3 | 4 | ... |
|---|---|---|---|---|---|---|
| **valor** | 0 | 1 | 1 | 0 | 0 | ... |

En el tercer día, ambos leen la ubicación 2, donde hay un 1 escrito.

En el cuarto día, el ID 0 responde que hay 2 miembros (correcto), mientras que el ID 3 lee el 1 en la ubicación 1. El ID 0 termina inmediatamente después de esto y no participa en los días siguientes.

Finalmente, en el quinto día, el miembro restante también responde correctamente $N = 2$.

## Herramienta de pruebas

Para facilitar la prueba de tu solución, proporcionamos una herramienta sencilla que puedes descargar desde el sistema de evaluación CMS. La herramienta es opcional. Observa que el evaluador oficial en CMS es diferente a la herramienta de pruebas.

Para usar la herramienta necesitas un archivo de entrada. Puedes usar los ejemplos proporcionados `census.input0.txt` y `census.input1.txt`, o crear el tuyo. El archivo de entrada debe comenzar con el número de miembros $N$ y los IDs posibles $M$, seguidos de una línea con $N$ números especificando los IDs de los miembros de la sociedad.

Para programas en Python, digamos `census.py` (normalmente ejecutado como `pypy3 census.py`) ejecuta la herramienta de pruebas de la siguiente manera:

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

Para programas en C++, primero compila tu solución:

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

y luego ejecuta la herramienta de pruebas:

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

Toma en cuenta que en este problema se usa la salida estándar para la comunicación con el evaluador, por lo que no debe usarse para depurar. En su lugar, puedes usar la salida de error estándar (stderr). En C++ puedes usar `cerr << msg << endl;`. En Python puedes usar `print(msg, file=sys.stderr)`.

La herramienta de pruebas leerá y mostrará estos mensajes de stderr junto con las consultas realizadas por todas las instancias de tu programa. Es importante mencionar que por razones técnicas, los mensajes de depuración pueden aparecer ligeramente desincronizados entre sí.