D. 點人數 (census)

時間限制: 1 秒
空間限制: 128 MiB

關於切塞納蒂科（Cesenatico），鮮為人知的是：其中有個神秘組織，由 N 名女性信息學家組成。 該組織神秘到成員彼此互不認識。 每個成員有個獨有的 ID，是滿足 $0 \le I \le M-1$ 的整數 I.

成員之間祇在鎮上不同地點用粉筆寫下若干個數來間接交流。 每 100 年，組織就會點一次人數。 點人數完成時，每位成員都須知道組織的總人數。

點人數歷時多日。每日，每位仍在參與該過程的成員，都會選擇執行一種操作：**讀取 (read)**、**寫入 (write)** 或**停止 (stop)** 參與。

- 若成員選擇**讀取**，則她會選定一個地點 P。白天，她會前往地點 P，並讀取寫在該處的數。
- 若成員選擇**寫入**，則她會選定一個地點 P 和一個數 V。夜間，她會前往地點 P 並將寫在該處的數改為 V。由於天色已暗，她無法在寫入新數前讀取舊數。
- 若成員選擇**停止**，則她往後的日子都不再有任何操作。

若某成員目睹另一位成員寫入數字，她就可能認出對方。因此，嚴禁兩名或以上的成員在同一日選擇同一個地點寫入。 （讀取則沒有此限制，因為可以偷偷一瞥。）

若有（一名或多名）成員在另一成員打算寫入的同日同一地點讀取，各次讀取都會發生在該一次寫入之前。

組織該如何規劃點人數流程，纔可用最少的日子，使所有人都得知正確成員數?

實作須知

> ⇨ 本題為交互題，會同時運行你程式的 N $(1 \le N \le 100)$ 個實例。每個實例模擬組織的一名成員。

共有 10^{18} 個地點。 地點編號 P 須滿足 $0 \le P < 10^{18}$。 初始時，各地點寫下的值皆為 $V = 0$。

寫入某地點的新值 V 必須是整數，且滿足 $0 \le V \le 10^9$。 在大多數子任務中，V 只能為 0 或 1。 詳情請見「評分方式」一節。

當你的程式的實例啟動時，應先讀入一行，包含兩個整數 I 和 M $(0 \le I \le M-1)$，代表該實例所屬成員的唯一 ID，以及可能的 ID 總數。 每筆測資中，所有實例都會收到相同的 M 值和互異的 I 值。留意有些 ID 可能未分配給任何成員。

然後，對於點人數過程的每一天，你的程式應選定欲執行的操作，並相應列印一行：

動作	**含義**
`r` P	**讀取**地點 P。 列印此行後，你的程式應讀取一行，包含地點 P 當前寫有的值。

`w` P V	**寫入**新值 V 到地點 P。 若多個實例於同日同一地點 P 寫入，將被判定為「不正確」(*Not correct*)。 除了範例和子任務 3 外，你必須寫入 $0 \le V \le 1$；請見「評分方式」一節。
`!` N	**回答並停止**：回報有 N 名成員並停止參與點人數。 回答後，**你的程式應正常結束**。（留意其他實例可能繼續執行幾日，然後纔回答並停止。）

若你程式的任何實例回答了錯誤的 N 值、違反協議、使用超過 500 天，或是超過了（按每個進程計的）時間/空間限制，你的提交在該筆測資會被判為「不正確」（*Not correct*）。

否則，你的程式在該筆測資會判為「(部分）正確」(*(Partially) Correct*)，並根據 D 值（任何實例回答用到的最大天數）計分。要獲得滿分，你需要在 $D \le 61$ 且 $V \le 1$ 的條件下解決全部測資。 詳見「評分方式」一節。

沖刷緩衝區。若未使用提供的範本，請確保在列印每行後沖刷標準輸出，否則你的程式可能會被判為「不正確」。在 Python 中，若使用 `input()` 讀取行，這會自動完成。在 C++ 中，`cout << endl;` 除了列印換行符外還會沖刷緩衝區；若使用 `printf`，請使用 `fflush(stdout)`。

限制條件

- $1 \le N \le 100$.
- $1 \le M \le 100\,000$.
- 你最多可以使用 500 天。

評分方式

你的程式將以多筆測資進行評測，測資分成若干個子任務。要獲得某個子任務的分數，你必須正確解出其中全部測試。

- **子任務 0 [0 分]**: 範例（你可以寫入任何整數 $0 \le V \le 1\,000\,000\,000$）。
- **子任務 1 [11 分]**: $M \le 100$，且 N 名成員的 ID 分別為 $0, 1, ..., N-1$。
- **子任務 2 [12 分]**: $1 \le N \le 2$。
- **子任務 3 [22 分]**: $M \le 8000$，且你可以寫入任何整數 $0 \le V \le 1\,000\,000\,000$。
- **子任務 4 [55 分]**: 無額外限制。

子任務 1、2、4 中，每次寫入動作只能寫入 $V = 0$ 或 $V = 1$。

令 X_s 為子任務 s 的滿分（如上所示），D_s 為你的程式在子任務 s 各測試中使用的最大天數。則：

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

分_s 的值會四捨五入到最接近的整數，你的總分是這些分數的總和。欲取得全題的滿分，你需要在全部測資做到 $D \le 61$ 且 $V \le 1$。

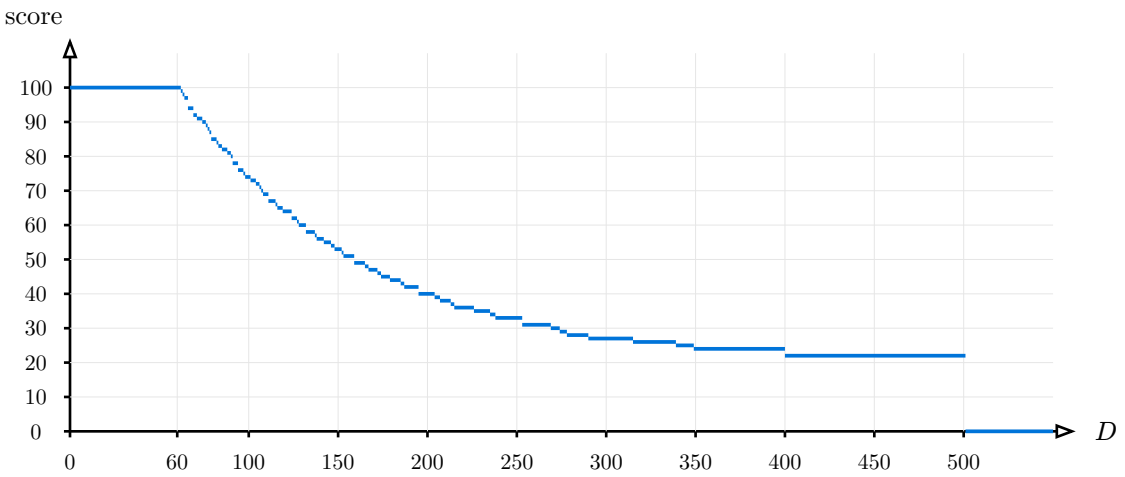

圖 1　總分，假設每個子任務皆以相同的最大值 D 解出。

範例

第一個範例。每兩欄展示評測機與一個實例如何交互。

機	例 0	機	例 1	機	例 2	機	例 3	機	例 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				

第二個範例。

評測機	實例 0
0 8000	
	w 0 0
	w 1 1
	r 2
1	
	! 2

評測機	實例 1
3 8000	
	w 2 1
	r 1
0	
	r 2
1	
	r 1
1	
	! 2

說明

第一個範例。 有 $N = 5$ 名成員，ID 分別為 $0, 1, 2, 3, 4$，且 $M = 100$（適用於子任務 1、3、4）。實例 i 對應 ID 為 i 的成員。上面的交互衹是一種可能的合法操作序列，**未必**是高效或合理的策略；僅用於說明交互協議如何運作。

第二個範例。 有 $N = 2$ 名成員，ID 分別為 0 和 3，且 $M = 8000$（適用於子任務 2、3、4）。 第一天，ID 為 0 的成員在地點 0 寫入 0（無變化），而 ID 為 3 的成員在地點 2 寫入 1。

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

第二天，ID 0 在地點 1 寫入 1，而 ID 3 讀取同一個地點。請注意，讀取發生在白天，在晚間寫入之前。因此 ID 3 看到的仍是 0。

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

第三天，她們都讀取地點 2，該處寫入的值為 1。

第四天，ID 0 回答有 2 名成員（正確），同時 ID 3 讀取地點 1 處的 1。ID 0 此後立即結束，不再參與隨後的日子。

最後，第 $D = 5$ 天，剩餘的成員也正確回答 $N = 2$。

測試

為方便你測試自己的解答, 我們提供一個簡單工具, 可從 CMS 下載。你可自由選用該工具。留意 CMS 的正式評測機與此測試工具有差異。

要使用該工具，需要有輸入檔案。你可以使用提供的範例輸入 `census.input0.txt` 和 `census.input1.txt`，或自行預備。 輸入檔案應以成員數目 N 和可能 ID 數目 M 起首，其後是一行，包含 N 個數，指定組織各成員的 ID。

對於 Python 程式，假設為 `census.py`（通常運行方式為 `pypy3 census.py`），運行測試工具的方式如下:

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

對於 C++ 程式，先編譯你的解答:

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

然後運行測試工具:

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

留意本題中，標準輸出是用於與評測機交互，因此不可用於調試。取而代之，你可以用標準錯誤輸出（stderr）。C++ 中，可以用 `cerr << msg << endl;`。Python 中，可以用 `print(msg, file=sys.stderr)`。

測試工具會讀取該些 stderr 訊息，連同你程式的實例的各操作一同列出。惟因技術理由，其列出的時間可能稍為錯開。