D. 人口普查 (census)

Time limit: 1 seconds
Memory limit: 128 MiB

關於 Cesenatico 有個鮮為人知的事實，就是這裡有一個由 N 位女性資訊科學家組成的秘密社群。這個社群非常神秘；沒有任何成員知道其他人是誰。 每個成員有一個獨特的 ID: 一個整數 I，且 $0 \le I \le M-1$。

成員之間唯一的溝通方式是間接的，要透過在鎮上不同的地點用粉筆寫下數字來溝通。 每隔 100 年，這個社群會進行一次人口普查來清點成員人數。在普查完成後，每位成員應該會知道社群中成員的總數。

普查會持續好幾天。每一天，仍參與在普查過程中的每位成員都會選擇並執行恰好一個行動：**read**、**write**，或是 **stop** 參與普查。

- 如果成員選擇 **read** , 她會挑選一個地點 P。在白天, 她會前往地點 P 並讀取寫在那裡的數字。
- 如果成員選擇 **write**，她會挑選一個地點 P 和一個數字 V。在深夜，她會前往地點 P 把那裡的數字改成 V。因為天色已暗，在她寫下新數字前沒辦法讀舊有的數字。
- 如果成員選擇 **stop**，她在接下來的日子就不會再採取任何行動。

如果某個成員看到另一個成員正在 **write**，她可能會認出對方是誰。 因此，嚴格禁止兩位或更多的成員在同一天選擇在同一個地點寫下數字。 (read 則沒有這個限制，因為可以偷偷進行。)

如果一名或多名成員在同一天, 要在另一個成員想要寫下數字的地點進行 **read**, 所有的 **read** 都會在寫下之前發生。

這個社群應該要怎麼規劃它的普查過程，才能讓每個人得知正確成員總數所需要的天數降到最低呢？

Implementation

> ⇨ 這是一個互動題，在此任務，系統將會同時（simultaneously）執行 N 個你的程式實例 (instances)，其中 N 為未知 （$1 \le N \le 100$）。每個實例程式模擬社群中的一位成員。

總共有 10^{18} 個地點。 地點編號 P 必須滿足 $0 \le P < 10^{18}$。 一開始，所有地點寫的值都是 $V = 0$。

在某個地點寫下的新值 V 必須是一個整數，且滿足 $0 \le V \le 10^9$。 在大部分的子任務中，V 只能是 0 或 1。 更多細節請見計分 (Scoring) 區塊（section）。

當你的程式實例開始執行時，它應該要先讀入一行，有兩個整數 I 和 M ($0 \le I \le M-1$) ： 包含這個實例所代表的社群成員獨特 ID，以及可能的 ID 總數。在每一個測資中，所有的實例都會取得相同的 M 值，以及各自相異的 I 值。 請注意，可能會有沒有分配給任何成員的 ID。

在普查過程的每一天，你的程式應該選擇想要執行的行動，並印出對應的一行：

行動	代表意義

`r` P	在地點 P **Read**. 印出這行之後，你的程式應該讀入一行，內容為目前寫在 P 的值。
`w` P V	在地點 P **Write** 新數值 V。 如果多個實例在同一天寫下相同的 P，你會得到 *Not correct* 的判定。 除了範例和子任務 3 以外，你只能寫入 $0 \leq V \leq 1$；請見計分 (Scoring) 區塊。
`!` N	**Answer and stop**：回報有 N 個成員，並停止參與普查。 回答後，**你的程式應該要正常結束 (exit)**。(請注意，你的程式的其他實例在回答並結束(exit)之前，可能還會繼續執行好幾天。)

如果你的程式的任何一個實例回答了錯誤的 N 值、違反協定、使用了超過 500 天，或是超過了（每個實例的）時間或記憶體限制，你的提交在此測資就會被判定為 *Not correct*。

否則，你的程式在此測資將會是 *(Partially) Correct*，並且會根據 D 值（也就是所有實例回答問題所花費的最大天數）來計分。若要拿到滿分，你必須在每筆測資以 $D \leq 61$ 且 $V \leq 1$ 解決測試案例。詳情請見計分 (Scoring) 區塊。

清空輸出緩衝區 (Flushing)。如果你沒有使用我們提供的模板 (templates)，請確保在印出每一行後清空標準輸出，否則你的程式可能會被判定為 *Not correct*。在 Python 中，如果你使用 `input()` 讀取行，這將會自動發生。在 C++ 中，`cout << endl;` 除了印出換行之外也會清空緩衝區；如果使用 printf，請使用 `fflush(stdout)`。

Constraints

- $1 \leq N \leq 100$.
- $1 \leq M \leq 100\,000$.
- 你最多只能使用 500 天。

Scoring

你的程式會在多筆測資上進行測試，這些測資會被分組為多個子任務。要獲得某個子任務的分數，你必須正確解決該任務包含的所有測資。

- **Subtask 0 [0 points]**: 範例（你可以寫入任何符合 $0 \leq V \leq 1\,000\,000\,000$ 的整數）。
- **Subtask 1 [11 points]**: $M \leq 100$，且這 N 位成員的 ID 是 $0, 1, ..., N-1$。
- **Subtask 2 [12 points]**: $1 \leq N \leq 2$。
- **Subtask 3 [22 points]**: $M \leq 8000$，且你可以寫入任何符合 $0 \leq V \leq 1\,000\,000\,000$ 的整數。
- **Subtask 4 [55 points]**: 沒有額外限制。

在子任務 1、2 和 4 中，你在每次的 Write 行動只能寫入 $V = 0$ 或 $V = 1$。

令 X_s 表示子任務 s（如上所示）的最高分數，且 D_s 表示任一個你的程式在子任務 s 的一筆測資中所使用的最大天數。然後：

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

score_s 的值會在每個子任務四捨五入到最近的整數，你的總分為這些分數的總和。若要拿到滿分，你必須在每筆測資都達成 $D \leq 61$ 且 $V \leq 1$。

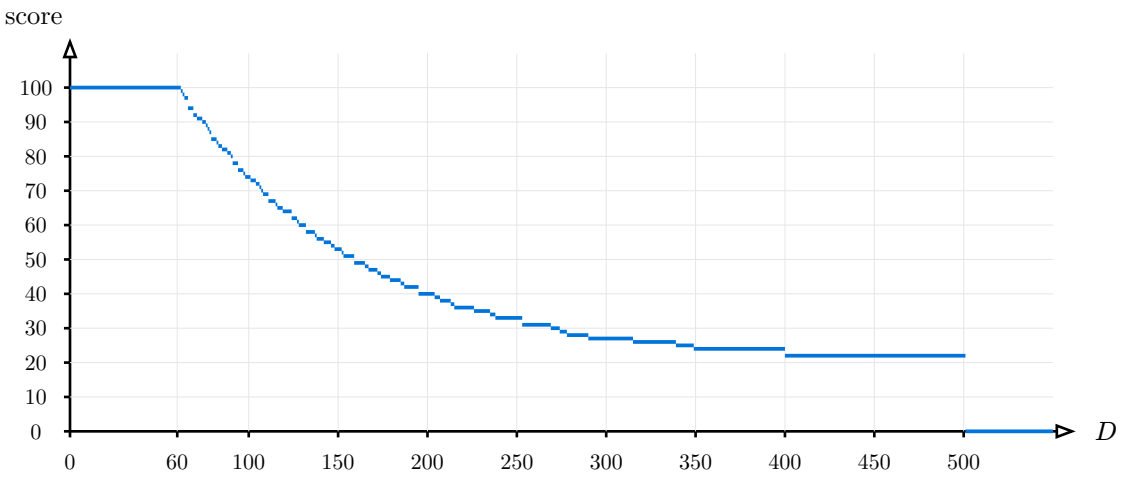

Figure 1: 總分（假設每個子任務都以相同的最大天數 D 解出）。

Examples

第一個範例。每對欄位顯示了評測程式與一個實例之間的溝通。

Gra.	Inst. 0	Gra.	Inst. 1	Gra.	Inst. 2	Gra.	Inst. 3	Gra.	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				

第二個範例。

Grader	Instance 0
0 8000	
	w 0 0
	w 1 1
	r 2
1	
	! 2

Grader	Instance 1
3 8000	
	w 2 1
	r 1
0	
	r 2
1	
	r 1
1	
	! 2

Explanation

第一個範例。 我們有 $N = 5$ 位成員，其連續 ID 為 $0, 1, 2, 3, 4$，且 $M = 100$（適用於子任務 1、3 和 4）。解決方案 i 對應 ID 為 i 的成員。上面的互動過程只是其中一種合法的操作序列，**並不是** 一個有效率或合理的策略；這只是用來示協定是如何運作的。

第二個範例。 我們有 $N = 2$ 位成員，ID 為 0 和 3，且 $M = 8000$（適用於子任務 2、3 和 4）。 在第一天，ID 0 的成員在地點 0 **write** 0（沒有變化），而 ID 3 的成員在地點 2 **write** 1。

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

在第二天，ID 0 在地點 1 **write** 1，而 ID 3 **read** 同一個地點。請注意，**read** 發生在白天，也就是在晚上的 **write** 之前。因此，ID 3 看到的還是 0。

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

在第三天，他們都 **read** 地點 2，那邊寫著 1。

在第四天，ID 0 回答有 2 名成員（正確），而 ID 3 **read** 地點 1 的 1。ID 0 在這之後立刻結束執行 (exits)，且不會參與接下來幾天的過程。

最後，在第 $D = 5$ 天，剩下的一位成員也正確回答 $N = 2$。

測試

為了方便你測試自己的解答，我們提供了一個簡單的工具，你可以從 CMS 下載。 這個工具可選擇是否使用。請注意，CMS 上的官方評測程式與這個測試工具不同。

若要使用這個工具，你需要一個輸入檔。你可以使用我們提供的範例輸入 `census.input0.txt` 和 `census.input1.txt`，或是自己寫一個。輸入檔的開頭應該要是成員數量 N 以及可能 ID 的總數 M，接著下一行包含 N 個數字，用來指定社團成員的 ID。

對於 Python 程式，假設檔名是 `census.py`（通常會以 `pypy3 census.py` 來執行），請按以下方式執行測試工具：

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

對於 C++ 程式，請先編譯你的解答(solution)：

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

然後執行測試工具：

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

請注意，在這道題目中，標準輸出 (standard output) 是用來與評測程式進行溝通的，所以它不應該被用來印出除錯訊息。取而代之的是，你可以使用標準錯誤輸出 (stderr)。在 C++ 中，你可以使用 `cerr << msg << endl;`。在 Python 中，你可以使用 `print(msg, file=sys.stderr)`。

測試工具會讀取並呈現這些 stderr 的訊息，以及你所有程式實例所執行的查詢 (queries)。請注意，由於技術上的原因，它們印出來的順序可能會稍微有一點點不同步 (out of sync)。