D. Mitgliederzählung (census)

Time Limit: 1 Sekunde
Memory Limit: 128 MiB

Ein wenig bekannter Fakt über Cesenatico ist, dass es die Heimat einer Geheimgesellschaft von N Informatikerinnen ist. Diese Gesellschaft ist in der Tat sehr geheim; kein Mitglied kennt ein anderes. Jedes Mitglied hat eine eindeutige ID: eine nicht-negative Ganzzahl I.

Die einzige Kommunikation zwischen den Mitgliedern ist indirekt, über Zahlen, die mit Kreide an verschiedenen Orten in der Stadt notiert werden. Alle 100 Jahre führt die Gesellschaft eine Mitgliederzählung durch, um ihre Mitglieder zu zählen. Nachdem die Mitgliederzählung abgeschlossen ist, sollte jedes Mitglied die Gesamtzahl der Mitglieder in der Gesellschaft kennen.

Die Mitgliederzählung findet über mehrere Tage statt. Jeden Tag wählt jedes Mitglied, das noch am Prozess teilnimmt, genau eine Aktion aus und führt diese aus: **Lesen**, **Schreiben** oder das Teilnehmen **beenden**.

- Wenn sich ein Mitglied für **Lesen** entscheidet, wählt es einen Ort P. Tagsüber besucht es den Ort P und liest die dort notierte Zahl.
- Wenn sich ein Mitglied für **Schreiben** entscheidet, wählt es einen Ort P und eine Zahl V. Am späten Abend besucht es den Ort P und ändert die dort notierte Zahl auf V. Da es bereits dunkel ist, kann es die alte Zahl nicht lesen, bevor es die neue schreibt.
- Wenn sich ein Mitglied für **Beenden** entscheidet, führt es an den folgenden Tagen keine Aktionen mehr aus.

Wenn ein Mitglied ein anderes eine Zahl schreiben sieht, könnte es diese erkennen. Daher ist es streng verboten, dass zwei oder mehr Mitglieder am selben Tag am selben Ort schreiben. (Für das Lesen gibt es keine solche Einschränkung, da dies diskret erfolgen kann.)

Wenn ein oder mehrere Mitglieder an einem Ort lesen, an dem ein anderes Mitglied am selben Tag schreiben möchte, finden alle Lesevorgänge vor dem Schreibvorgang statt.

Wie sollte die Gesellschaft ihren Mitgliederzählungsprozess planen, um die Anzahl der Tage zu minimieren, bis jeder die korrekte Mitgliederzahl kennt?

Implementierung

⇨ Dies ist ein interaktives Problem, bei dem eine unbekannte Anzahl von Instanzen ($1 \leq N \leq 100$) deines Programms gleichzeitig ausgeführt wird. Jede Instanz simuliert ein Mitglied der Gesellschaft.

Es gibt 10^{18} Orte. Die Zahl P eines Ortes muss die Bedingung $0 \leq P < 10^{18}$ erfüllen. Anfänglich ist der Wert, der an allen Orten notiert ist, $V = 0$.

Der neue Wert V, der an einem Ort geschrieben wird, muss immer eine Ganzzahl sein, sodass $0 \leq V \leq 10^9$. In den meisten Teilaufgaben kann V nur 0 oder 1 sein. Siehe den Abschnitt Bewertung für weitere Details.

Wenn eine Instanz deines Programms startet, sollte sie zuerst eine Zeile mit zwei Ganzzahlen, I und M ($0 \leq I \leq M-1$), lesen: die eindeutige ID des Mitglieds der Gesellschaft, das von dieser Instanz repräsentiert wird, und die Gesamtzahl der möglichen IDs. Innerhalb eines Testfalls erhalten alle Instanzen denselben Wert M und unterschiedliche Werte I. Beachte, dass es IDs geben kann, die keinem Mitglied zugewiesen sind.

Dann sollte dein Programm für jeden Tag im Mitgliederzählungsprozess die Aktion auswählen, die es ausführen möchte, und entsprechend eine Zeile ausgeben:

Aktion	**Bedeutung**
`r` P	**Lesen** von Ort P. Nach der Ausgabe dieser Zeile sollte dein Programm eine Zeile mit dem aktuellen Wert lesen, der an P notiert ist.
`w` P V	**Schreiben** des neuen Wertes V an Ort P. Wenn mehrere Instanzen am selben Tag am selben Ort P schreiben, wird dein Programm als *Not correct* bewertet. Ausser bei den Beispielen und Teilaufgabe 3 musst du $0 \leq V \leq 1$ schreiben; siehe den Abschnitt Bewertung.
`!` N	**Antworten und Beenden**: Melde, dass es N Mitglieder gibt und höre auf, an der Mitgliederzählung teilzunehmen. Nach der Antwort **sollte dein Programm normal beendet werden**. (Beachte, dass andere Instanzen deines Programms möglicherweise noch weitere Tage laufen, bevor sie antworten und sich beenden.)

Falls eine Instanz deines Programms einen falschen Wert für N antwortet, das Protokoll verletzt, mehr als 500 Tage benötigt oder das Time/Memory Limit (pro Prozess) überschreitet, wird deine Einreichung für den gegebenen Testfall als *Not correct* bewertet.

Andernfalls ist dein Programm für den Testfall *(Partially) correct* und wird basierend auf dem Wert D bewertet: die maximale Anzahl an Tagen, die eine Instanz bis zur Antwort benötigt hat. Für die volle Punktzahl musst du jeden Testfall mit $D \leq 61$ und $V \leq 1$ lösen. Siehe den Abschnitt Bewertung für Details.

Flushing. Wenn du nicht die bereitgestellten Vorlagen verwendest, stelle sicher, dass du die Standardausgabe nach dem Drucken jeder Zeile flushst, sonst könnte dein Programm als *Not correct* bewertet werden. In Python geschieht dies automatisch, wenn du `input()` zum Lesen von Zeilen verwendest. In C++ flushed `cout << endl;` zusätzlich zum Einfügen eines Zeilenumbruchs; wenn du `printf` verwendest, nutze `fflush(stdout)`.

Einschränkungen

- $1 \leq N \leq 100$.
- $1 \leq M \leq 100\,000$.
- Du darfst maximal 500 Tage verwenden.

Bewertung

Dein Programm wird auf mehreren Testfällen getestet, die in Teilaufgaben gruppiert sind. Um die Punktzahl für eine Teilaufgabe zu erhalten, musst du alle darin enthaltenen Tests korrekt lösen.

- **Teilaufgabe 0 [0 Punkte]**: Beispiele (du kannst jede Ganzzahl $0 \leq V \leq 1\,000\,000\,000$ schreiben).
- **Teilaufgabe 1 [11 Punkte]**: $M \leq 100$ und die N Mitglieder haben die IDs $0, 1, ..., N-1$.
- **Teilaufgabe 2 [12 Punkte]**: $1 \leq N \leq 2$.
- **Teilaufgabe 3 [22 Punkte]**: $M \leq 8000$ und du kannst jede Ganzzahl $0 \leq V \leq 1\,000\,000\,000$ schreiben.
- **Teilaufgabe 4 [55 Punkte]**: Keine weiteren Einschränkungen.

In den Teilaufgaben 1, 2 und 4 kannst du bei jeder Schreib-Aktion nur $V = 0$ oder $V = 1$ schreiben.

Sei X_s die maximale Punktzahl für Teilaufgabe s (oben angezeigt) und D_s die grösste Anzahl an Tagen, die eines deiner Programme bei einem Test in Teilaufgabe s verwendet. Dann gilt:

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

Der Wert von score_s wird pro Teilaufgabe auf die nächste Ganzzahl gerundet, und deine Gesamtpunktzahl ist die Summe dieser Werte. Um die volle Punktzahl für die Aufgabe zu erhalten, musst du $D \leq 61$ und $V \leq 1$ bei jedem Testfall erreichen.

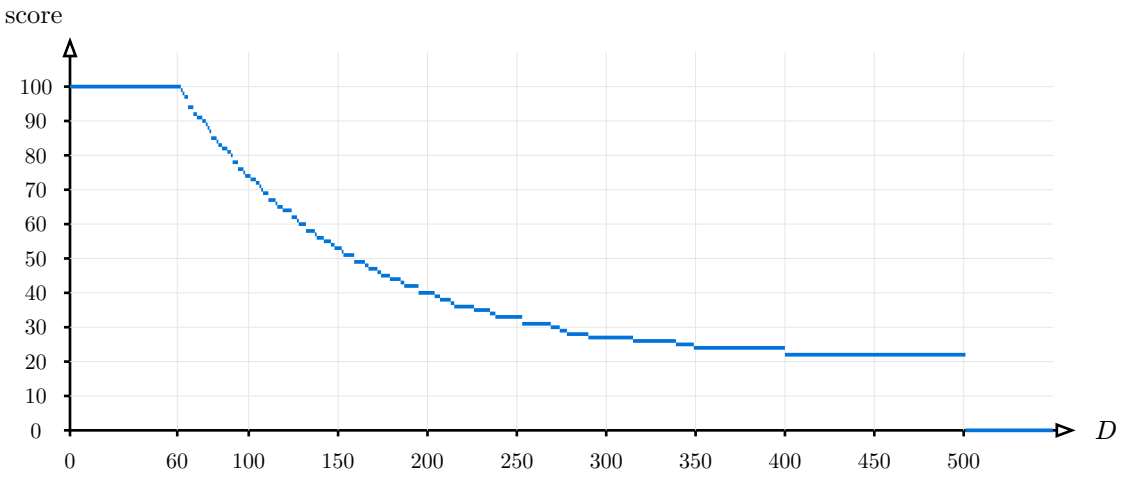

Abbildung 1: Gesamtpunktzahl unter der Annahme, dass jede Teilaufgabe mit dem gleichen maximalen D gelöst wird.

Beispiele

Erstes Beispiel. Jedes Spaltenpaar zeigt die Kommunikation zwischen dem Grader und einer Instanz.

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				

Zweites Beispiel.

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

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

Erklärung

Erstes Beispiel. Wir haben $N = 5$ Mitglieder mit fortlaufenden IDs $0, 1, 2, 3, 4$ und $M = 100$ (gültig für die Teilaufgaben 1, 3 und 4). Instanz i entspricht dem Mitglied mit der ID i. Die Interaktion oben ist nur eine mögliche legale Operationssequenz und ist **nicht** als effiziente oder sinnvolle Strategie gedacht; sie wird nur gezeigt, um zu illustrieren, wie das Protokoll funktioniert.

Zweites Beispiel. Wir haben $N = 2$ Mitglieder mit den IDs 0 und 3 und $M = 8000$ (gültig für die Teilaufgaben 2, 3 und 4). Am ersten Tag schreibt das Mitglied mit der ID 0 eine 0 an Ort 0 (keine Änderung), und das Mitglied mit der ID 3 schreibt eine 1 an Ort 2.

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

Am zweiten Tag schreibt ID 0 eine 1 an Ort 1, und ID 3 liest denselben Ort. Beachte, dass das Lesen tagsüber stattfindet, vor dem Schreiben am Abend. Daher sieht ID 3 immer noch eine 0.

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

Am dritten Tag lesen beide Ort 2, an dem eine 1 geschrieben steht.

Am vierten Tag antwortet ID 0, dass es 2 Mitglieder gibt (korrekt), während ID 3 die 1 an Ort 1 liest. ID 0 beendet sich sofort danach und nimmt an den kommenden Tagen nicht mehr teil.

Schliesslich, an Tag $D = 5$, antwortet auch das verbleibende Mitglied korrekt mit $N = 2$.

Testen

Um das Testen deiner Lösung zu erleichtern, stellen wir ein einfaches Tool zur Verfügung, das du von CMS herunterladen kannst. Das Tool ist optional zu verwenden. Beachte, dass der offizielle Grader auf CMS ein anderer ist als das Testwerkzeug.

Um das Tool zu verwenden, benötigst du eine Eingabedatei. Du kannst die bereitgestellten Beispiel-Eingaben `census.input0.txt` und `census.input1.txt` verwenden oder eigene erstellen. Die Eingabedatei sollte mit der Anzahl der Mitglieder N und den möglichen IDs M beginnen, gefolgt von einer Zeile mit N Ganzzahlen, die die IDs der Gesellschaftsmitglieder spezifizieren.

Für Python-Programme, sagen wir `census.py` (normalerweise ausgeführt als `pypy3 census.py`), führe das Testwerkzeug wie folgt aus:

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

Für C++-Programme, kompiliere zuerst deine Lösung:

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

und führe dann das Testwerkzeug aus:

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

Beachte, dass bei diesem Problem die Standardausgabe für die Kommunikation mit dem Grader verwendet wird, sie sollte daher nicht zum Debuggen verwendet werden. Verwende stattdessen die Standardfehlerausgabe (stderr). In C++ kannst du `cerr << msg << endl;` nutzen. In Python kannst du `print(msg, file=sys.stderr)` verwenden.

Das Testwerkzeug wird diese stderr-Nachrichten lesen und zusammen mit den Abfragen präsentieren, die von all deinen Programm-Instanzen durchgeführt wurden. Beachte, dass sie aus technischen Gründen möglicherweise leicht asynchron zueinander angezeigt werden.