D. Nüfus Sayımı (census)

Zaman Limiti: 1 saniye
Bellek Limiti: 128 MiB

Cesenatico hakkında az bilinen bir gerçek, buranın N bilişimciden oluşan gizli bir topluluğa ev sahipliği yapmasıdır. Bu topluluk gerçekten çok gizli; hiçbir üye bir diğerini tanımıyor. Her üyenin benzersiz bir kimlik numarası (ID) var: negatif olmayan bir tamsayı I.

Üyeler arasındaki tek iletişim dolaylı yoldan, şehrin farklı yerlerine tebeşirle karalanan sayılar aracılığıyla gerçekleşiyor. Topluluk her 100 yılda bir, üyelerini saymak için bir nüfus sayımı gerçekleştirir. Sayım tamamlandıktan sonra, her üye topluluktaki toplam üye sayısını bilmelidir.

Sayım birden fazla gün sürer. Her gün, sürece hala dahil olan her üye tam olarak bir eylem seçip gerçekleştirir: **okumak**, **yazmak** veya sürece dahil olmaktan **vazgeçmek**.

- Bir üye **okumayı** seçerse, bir P konumu seçer. Gün boyunca P konumunu ziyaret eder ve orada yazılı olan sayıyı okur.
- Bir üye **yazmayı** seçerse, bir P konumu ve bir V sayısı seçer. Akşam geç saatlerde P konumunu ziyaret eder ve orada yazılı olan sayıyı V olarak değiştirir. Hava zaten karardığı için, yeni sayıyı yazmadan önce eski sayıyı okuyamaz.
- Bir üye **vazgeçmeyi** seçerse, sonraki günlerde artık hiçbir eylemde bulunmaz.

Eğer bir üye diğerinin bir sayı yazdığını görürse, onu tanıyabilir. Bu yüzden, aynı gün aynı konuma iki veya daha fazla üyenin yazması kesinlikle yasaktır. (Okuma için böyle bir kısıtlama yoktur, çünkü bu gizlice yapılabilir.)

Eğer bir veya daha fazla üye, başka bir üyenin aynı gün yazmak istediği bir konumdan okuma yaparsa, tüm okumalar yazma işleminden önce gerçekleşir.

Topluluk, herkes doğru üye sayısını öğrenene kadar geçen gün sayısını en aza indirmek için sayım sürecini nasıl planlamalıdır?

Implementasyon

⇨ Bu etkileşimli (interaktif) bir problemdir; programınızın bilinmeyen sayıda örneği ($1 \leq N \leq 100$) aynı anda çalıştırılacaktır. Her örnek, topluluğun bir üyesini simüle eder.

10^{18} tane konum vardır. Konumun P numarası $0 \leq P < 10^{18}$ şartını sağlamalıdır. Başlangıçta, tüm konumlarda yazılı olan değer $V = 0$'dır.

Bir konuma yazılan yeni V değeri her zaman $0 \leq V \leq 10^9$ olacak şekilde bir tamsayı olmalıdır. Çoğu alt görevde V sadece 0 veya 1 olabilir. Daha fazla ayrıntı için Puanlama bölümüne bakın.

Programınızın bir örneği başladığında, önce iki tamsayı içeren bir satır okumalıdır: I ve M ($0 \leq I \leq M - 1$): bu örnek tarafından temsil edilen üyenin benzersiz kimlik numarası ve toplam olası kimlik sayısı. Her test durumu içinde, tüm örnekler aynı M değerini ve farklı I değerlerini alacaktır. Hiçbir üyeye atanmamış kimlik numaraları olabileceğini unutmayın.

Sonrasında, sayım sürecindeki her gün için programınız gerçekleştirmek istediği eylemi seçmeli ve buna uygun bir satır yazdırmalıdır:

Eylem	Anlamı
`r` P	Konum P'yi **oku**. Bu satırı yazdırdıktan sonra, programınız P'de yazılı olan mevcut değeri içeren bir satır okumalıdır.
`w` P V	Konum P'ye yeni V değerini **yaz**. Eğer aynı gün birden fazla örnek aynı P konumuna yazmaya çalışırsa, *Yanlış cevap* (Not correct) kararı alırsınız. Örnekler ve alt görev 3 hariç, $0 \leq V \leq 1$ yazmalısınız; Puanlama bölümüne bakın.
`!` N	**Cevapla ve dur**: N üye olduğunu bildir ve sayıma katılmayı bırak. Cevap verdikten sonra, **programınız normal şekilde sonlanmalıdır**. (Diğer örneklerin cevap verip çıkmadan önce ek günler boyunca çalışmaya devam edebileceğine dikkat edin.)

Eğer programınızın herhangi bir örneği yanlış N değerini cevaplarsa, protokolü ihlal ederse, 500'den fazla gün kullanırsa veya (işlem başına) süre/bellek sınırını aşarsa, gönderiniz ilgili test durumu için *Yanlış cevap* olarak değerlendirilecektir.

Aksi takdirde, programınız test durumu üzerinde *(Kısmen) Doğru* olacak ve D değerine göre puanlanacaktır: herhangi bir örneğin cevap vermek için harcadığı maksimum gün sayısı. Tam puan almak için her test durumunu $D \leq 61$ ve $V \leq 1$ ile çözmeniz gerekir. Daha fazla ayrıntı için Puanlama bölümüne bakın.

Temizleme (Flushing). Eğer sağlanan şablonları kullanmıyorsanız, her satırı yazdırdıktan sonra standart çıktıyı temizlediğinizden (flush) emin olun, aksi takdirde programınız *Yanlış cevap* olarak değerlendirilebilir. Python'da, satırları okumak için `input()` kullanırsanız bu otomatik olarak gerçekleşir. C++'da, `cout << endl;` yeni bir satır yazdırmanın yanı sıra temizleme de yapar; printf kullanıyorsanız, `fflush(stdout)` kullanın.

Kısıtlamalar

- $1 \leq N \leq 100$.
- $1 \leq M \leq 100\,000$.
- En fazla 500 gün kullanabilirsiniz.

Puanlama

Programınız, alt görevler halinde gruplandırılmış birkaç test durumu üzerinde test edilecektir. Bir alt görevin puanını almak için, içerdiği tüm testleri doğru şekilde çözmelisiniz.

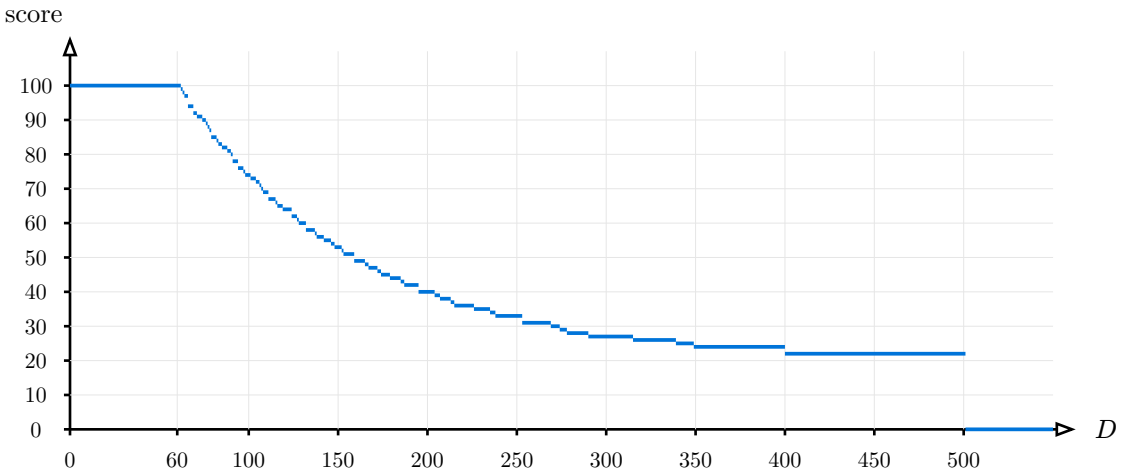
- **Alt görev 0 [0 puan]**: Örnekler (herhangi bir $0 \leq V \leq 1\,000\,000\,000$ tamsayısı yazabilirsiniz).
- **Alt görev 1 [11 puan]**: $M \leq 100$ ve N üyenin kimlik numaraları $0, 1, ..., N-1$.
- **Alt görev 2 [12 puan]**: $1 \leq N \leq 2$.
- **Alt görev 3 [22 puan]**: $M \leq 8000$ ve herhangi bir $0 \leq V \leq 1\,000\,000\,000$ tamsayısı yazabilirsiniz.
- **Alt görev 4 [55 puan]**: Ek kısıtlama yoktur.

1, 2 ve 4 numaralı alt görevlerde, her Yazma eyleminde sadece $V = 0$ veya $V = 1$ yazabilirsiniz.

puan$_s$ maksimum puanı temsil etsin (yukarıda gösterilen), D_s ise s alt görevindeki bir testte programlarınızdan herhangi birinin kullandığı en büyük gün sayısı olsun. O zaman:

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

puan_s değeri her alt görev için en yakın tamsayıya yuvarlanır ve toplam puanınız bunların toplamıdır. Görevden tam puan almak için her test durumunda $D \leq 61$ ve $V \leq 1$ koşulunu sağlamanız gerekir.



Şekil 1: Toplam puan, her alt görevin aynı maksimum D ile çözüldüğü varsayılmıştır.

Örnekler

İlk Örnek. Sütun çiftlerinin her biri, değerlendirici ve bir örnek arasındaki iletişimi gösterir.

Dğr.	Örn. 0	Dğr.	Örn. 1	Dğr.	Örn. 2	Dğr.	Örn. 3	Dğr.	Örn. 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				

İkinci Örnek.

Değerlendirici	Örnek 0
0 8000	
	w 0 0
	w 1 1
	r 2
1	
	! 2

Değerlendirici	Örnek 1
3 8000	
	w 2 1
	r 1
0	
	r 2
1	
	r 1
1	
	! 2

Açıklama

İlk Örnek. $N = 5$ üyemiz var, ardışık kimlik numaraları $0, 1, 2, 3, 4$ ve $M = 100$ (alt görev 1, 3 ve 4 için geçerlidir). Örnek i, ID'si i olan üyeye karşılık gelir. Yukarıdaki etkileşim, işlemlere ait sadece olası yasal bir dizidir ve verimli veya mantıklı bir strateji olması amaçlanmamıştır; sadece protokolün nasıl çalıştığını göstermek için sunulmuştur.

İkinci Örnek. $N = 2$ üyemiz var, ID'leri 0 ve 3 olan, $M = 8000$ (alt görev 2, 3 ve 4 için geçerlidir). Birinci gün, ID'si 0 olan üye 0 konumuna 0 yazar (değişiklik yok) ve ID'si 3 olan üye 2 konumuna 1 yazar.

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

İkinci gün, ID'si 0 olan üye 1 konumuna 1 yazar ve ID'si 3 olan üye aynı konumu okur. Okumanın gündüz, yazma işleminden önce gerçekleştiğine dikkat edin. Bu nedenle, ID'si 3 olan üye hala 0 görür.

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

Üçüncü gün, ikisi de 1 yazılı olan 2 konumunu okur.

Dördüncü gün, ID'si 0 olan üye 2 üye olduğunu doğru bir şekilde cevaplar, ID'si 3 olan üye ise 1 konumundaki 1'i okur. ID'si 0 olan üye bundan hemen sonra çıkar ve sonraki günlerde katılmaz.

Son olarak, $D = 5$. günde, kalan üye de doğru bir şekilde $N = 2$ cevabını verir.

Test Etme

Çözümünüzün test edilmesini kolaylaştırmak için, CMS'den indirebileceğiniz basit bir araç sunuyoruz. Bu aracı kullanmak isteğe bağlıdır. CMS'deki resmi değerlendiricinin (grader) test aracından farklı olduğuna dikkat edin.

Aracı kullanmak için bir girdi dosyasına ihtiyacınız var. Sağlanan örnek girdi dosyaları olan `census.input0.txt` ve `census.input1.txt` dosyalarını kullanabilir veya kendi dosyanızı oluşturabilirsiniz. Girdi dosyası, üye sayısı N ve olası kimlik sayısı M ile başlamalı, ardından toplum üyelerinin kimlik numaralarını belirten N sayı içeren bir satır gelmelidir.

Python programları için, örneğin `census.py` (normalde `pypy3 census.py` olarak çalıştırılır) test aracını aşağıdaki gibi çalıştırın:

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

C++ programları için, önce çözümünüzü derleyin:

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

ve ardından test aracını çalıştırın:

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

Bu problemde standart çıktının değerlendirici ile iletişim için kullanıldığına dikkat edin, bu yüzden hata ayıklama (debugging) için kullanılmamalıdır. Bunun yerine, standart hata çıktısını (stderr) kullanabilirsiniz. C++'da `cerr << msg << endl;` kullanabilirsiniz. Python'da `print(msg, file=sys.stderr)` kullanabilirsiniz.

Test aracı bu stderr mesajlarını okuyacak ve tüm program örnekleriniz tarafından gerçekleştirilen sorgularla birlikte sunacaktır. Teknik nedenlerden dolayı, bunların birbirleriyle biraz uyumsuz (zaman kaymalı) görünebileceğini unutmayın.