

Sınıf Oyunu

Taşkent'teki önde gelen bir lise, öğrenciler ve öğretmenler arasında düzenlenen bir oyunla kuruluş yıl dönümünü kutluyor. Sınıfta N öğrenci oturmaktadır ve bu öğrencilerin numaraları 0 ile $N - 1$ arasındadır. Her öğrenci, bir tamsayı dizisi içeren bir kağıt tutmaktadır. Başlangıçta her kağıt boştur, yani boş bir dizi içerir.

Öğrenciler, M öğretmenle (numaraları 0 ile $M - 1$ arasındadır) ve okul müdürüyle oyun oynamaktadır. Oyun, M adımda oynanır ve bu adımların numaraları da 0 ile $M - 1$ arasındadır. j 'inci adımında, öğretmen j sınıfa girer ve aşağıdaki olaylar gerçekleşir:

- Bazı (sıfır da olabilir) öğrenciler elini kaldırır. Her adımda en az bir öğrencinin elini **kaldırmayacağı** garanti edilir ve her öğrenci oyun boyunca **en fazla bir kez** elini kaldırabilir.
- Öğretmen, öğrencilerin o anda ellerinde tuttuğu kağıtlara bakar ve bu adımda elini **kaldırmayan** öğrencilerin elindeki kağıtlarda **değişiklik yapabilir**. Öğretmen, değişiklik yapabildiği öğrenci için kağıdının içeriğini yeni (boş dizi de olabilir) bir tamsayı dizisiyle değiştirebilir. Her tamsayı, 0 ile 63 aralığında (bu değerler dahil) olmalıdır ve dizi en fazla 63 eleman içerebilir.
- Bu değişikliklerin ardından, öğretmen j odadan çıkar ve öğrenciler gizli bir P permütasyonuna göre kağıtlarını değiştirirler. Yani, her öğrenci i ($0 \leq i < N$) kağıdını $P[i]$ öğrencisine verir; burada $P[0], P[1], \dots, P[N - 1]$, 0 ile $N - 1$ aralığında (bu değerler dahil), N **farklı** tamsayıdan oluşan bir dizidir. P dizisinin değerleri oyunun tüm aşamaları boyunca **sabit** kalır, ancak öğretmenler ve okul müdürü bu değerleri bilmez.

M adımların tümü tamamlandıktan ve P permütasyonuna göre son değişim gerçekleştirildikten sonra, okul müdürü odaya girer. Okul müdürü her i öğrencisi ($0 \leq i < N$) için, sadece öğrencilerin elindeki kağıtlara bakarak, öğrenci i 'nin elini kaldırdığı adım numarasını olan j 'yi belirlemeli ya da öğrenci i 'nin hiç elini kaldırmadığı sonucuna varmalıdır.

Öğretmenler, kağıtların üzerine yazılmış tamsayı dizileri dışında, diğer öğretmenlerle veya okul müdürüyle iletişim kuramazlar. Her öğretmen, sınıfa girdiği adımı bilir.

Göreviniz, öğretmenler ve okul müdürü için, her öğrencinin elini kaldırıp kaldırmadığını ve ne zaman kaldırdığını doğru bir şekilde belirleyen bir strateji tasarlamak ve uygulamaktır. Puanınız, kağıda yazılan herhangi bir dizinin maksimum uzunluğuna bağlı olacaktır: maksimum uzunluk ne kadar kısa olursa, puanınız o kadar yüksek veya aynı olur.

Kodlama Detayları

Biri öğretmenler, diğeri okul müdürü için olmak üzere iki prosedür kodlamalısınız.

Öğretmenler için kodlamanız gereken prosedür şudur:

```
std::vector<std::vector<int>> process_step(  
    int N, int M, int R,  
    std::vector<int> T,  
    std::vector<std::vector<int>> A)
```

- N : öğrenci sayısı.
- M : adım sayısı ve aynı zamanda öğretmen sayısı.
- R : mevcut adım numarası (0'ten $M - 1$ 'e kadar).
- T : bu adımda elini kaldıran öğrencilerin indislerinden oluşan, artan sırada sıralanmış (boş olabilen) bir dizi.
- A : kağıtları tanımlayan, uzunluğu N olan bir dizi; burada $A[i]$ ($0 \leq i < N$), adımın başında i numaralı öğrencinin elindeki kağıtta bulunan tamsayılar dizisidir.
- Bu prosedür, her oyun için tam olarak M kez, $R = 0, 1, \dots, M - 1$ sırasına göre çağrılır.

Prosedür, öğretmenin değişikliklerinden sonraki kağıtları temsil eden, N uzunluğunda bir B dizisini döndürmelidir; bu dizi, A ile aynı formatta olmalıdır.

- Elini kaldıran her öğrenci i için (yani, i , T içinde yer alıyorsa), kağıt değiştirilmemelidir; dolayısıyla $B[i] = A[i]$.
- $0 \leq i < N$ olan her i için, $B[i]$ 'in uzunluğu 63'ü aşmamalı ve $B[i]$ 'deki her tamsayı 0 ile 63 arasında (bu sınırlar dahil) olmalıdır.

Müdür için kodlamanız gereken prosedür şudur:

```
std::vector<int> determine_steps(  
    int N, int M, std::vector<std::vector<int>> A)
```

- N , M : yukarıdakiyle aynı.
- A : N uzunluğunda bir dizi; burada $A[i]$, tüm M adımdan sonra i öğrencisinin elindeki kağıtta bulunan tamsayı dizisidir.
- Bu prosedür, her oyun için tam olarak bir kez, `process_step` işlevine yapılan son çağrıdan sonra çağrılır.

Prosedür, uzunluğu N olan bir D dizisi döndürmelidir. $0 \leq i < N$ olan her i için aşağıdakilerden biri geçerli olmalıdır:

- $D[i] = j$ eğer öğrenci i , j 'inci adımda elini kaldırıyorsa, veya
- $D[i] = -1$ eğer öğrenci i , oyun boyunca hiç elini kaldırmadıysa.

Programınız, `process_step` işlevinin dönüş değeri dışında, `process_step` işlevine yapılan bir çağrıdan diğerine herhangi bir bilgiyi depolayamaz veya aktaramaz. Programınız bunu yapmaya çalışırsa, CMS'deki puanınız yanlış olabilir ve yarışma bittikten sonra düşürülebilir. Değerlendiricinin programınızın birden fazla servis birimini (instance) aynı anda çalıştırabileceğini unutmayın. Bu, aynı oyundan gelen `process_step` ve `determine_steps` çağrılarının farklı servis birimlerinde, farklı oyunlardan gelen `process_step` ve `determine_steps` çağrılarının ise aynı servis birimlerinde çalıştırılabileceği anlamına gelir. Her test durumu en fazla 5 oyun içerir.

Kısıtlamalar

- $2 \leq N \leq 63$
- $1 \leq M \leq 63$
- Her öğrenci, oyunun tamamı boyunca **en fazla bir kez** elini kaldırır. Başka bir deyişle, her öğrenci i 'nin elini kaldırdığı en fazla bir j adımı vardır.
- Her adımda, en az bir öğrenci elini **kaldırmaz**.

Alt Görevler ve Puanlama

Alt Görev	Puan	Ek Kısıtlamalar
1	4	$M = 1$
2	6	$N = 2$
3	9	$P[i] = i$ tüm $0 \leq i \leq N - 1$ için.
4	25	Her adımda en fazla bir öğrenci elini kaldırır.
5	56	Ek kısıtlama yoktur.

Her test durumu için, `process_step` prosedürüne yapılan herhangi bir çağrının dönüş değeri geçersizse veya `determine_steps` prosedürüne yapılan herhangi bir çağrının dönüş değeri yanlışsa, puanınız 0 olur (CMS'de `Output isn't correct` olarak bildirilir).

Aksi takdirde, `process_step` tarafından döndürülen **herhangi bir** B içindeki **herhangi bir** dizinin maksimum uzunluğuna C diyelim. Bu durumda, puanı S olan bir alt görevdeki test durumunun puanı, $S \cdot X$ olarak hesaplanır; burada X değeri, C değerine aşağıdaki tablodaki gibi bağlıdır:

Koşul	X
$C \leq 2$	1.00
$C = 3$	0.75
$C = 4$	0.55
$5 \leq C \leq 13$	$0.50 - 0.03 \cdot (C - 5)$
$14 \leq C \leq 63$	$0.19 \cdot \frac{64-C}{64-14} + 0.04$

Örnek

$N = 4$ öğrenci, $M = 2$ öğretmen ve $P = [0, 3, 1, 2]$ permütasyonu içeren bir senaryo düşünün. Başlangıçta tüm kağıtlar boştur.

Değerlendirici önce şunu çağırır:

```
process_step(4, 2, 0, [1], [[], [], [], []])
```

1 numaralı öğrenci elini kaldırmıştır, bu nedenle kağıdı değiştirilemez. 0 numaralı öğretmen, 0 numaralı öğrencinin kağıdına $[10]$ yazmaya, 3 numaralı öğrencinin kağıdına $[1, 63, 4]$ yazmaya ve 2 numaralı öğrencinin kağıdını boş bırakmaya karar verebilir. Bunu yapmak için, prosedür $B = [[10], [], [], [1, 63, 4]]$ değerini döndürmelidir.

0 numaralı öğretmen odadan çıktıktan sonra, öğrenciler P permütasyonundaki talimatlara göre kağıtlarını değiştirirler. Değişimden sonra kağıtlar $A = [[10], [], [1, 63, 4], []]$ şeklindedir.

Değerlendirici şimdi şu çağrılarını yapar:

```
process_step(4, 2, 1, [0, 3], [[10], [], [1, 63, 4], []])
```

0 ve 3 numaralı öğrenciler ellerini kaldırmışlardır, bu nedenle kağıtları değiştirilemez. 1 numaralı öğretmen, 1 numaralı öğrencinin kağıdına $[0, 40]$ yazmaya ve 2 numaralı öğrencinin kağıdına $[1, 50]$ yazmaya karar verebilir. Bunun için prosedürün $B = [[10], [0, 40], [1, 50], []]$ değerini döndürmesi gerekir.

1 numaralı öğretmen odadan çıktıktan sonra, kağıtlar P 'ye göre tekrar değiştirilir. Değişimden sonra kağıtlar $A = [[10], [1, 50], [], [0, 40]]$ şeklindedir.

Son olarak, değerlendirici şu çağırışı yapar:

```
determine_steps(4, 2, [[10], [1, 50], [], [0, 40]])
```

Bu prosedür, $D = [1, 0, -1, 1]$ dizisini döndürmelidir; çünkü öğrenciler 0 ve 3, 1 adımında el kaldırmış, öğrenci 1, 0 adımında el kaldırmış ve öğrenci 2 ise hiç el kaldırmamıştır. Bu örnekte, $C = 3$.

Örnek Değerlendirici

Girdi biçimi:

```
N M
Q[0] Q[1] ... Q[N-1]
P[0] P[1] ... P[N-1]
```

Burada, Q , uzunluğu N olan bir dizidir; burada, öğrenci i , adım j sırasında elini kaldırırsa $Q[i] = j$, öğrenci i oyun boyunca hiç elini kaldırmazsa $Q[i] = -1$ olur.

Çıktı biçimi:

`process_step` işlevinin her çağrılmasından sonra, örnek değerlendirici kağıtların içeriklerini çıktı olarak verir. B dizisinin uzunluğu K ve $B[i]$ dizisinin uzunluğu $L[i]$ olsun (her bir $0 \leq i < K$ için). Örnek değerlendirici, B 'yi aşağıdaki biçimde yazdırır ve sonrasında **boş bir satırla takip eder**:

```
K
L[0] B[0][0] B[0][1] ... B[0][L[0]-1]
L[1] B[1][0] B[1][1] ... B[1][L[1]-1]
:
L[K-1] B[K-1][0] B[K-1][1] ... B[K-1][L[K-1]-1]
```

Örnek değerlendirici daha sonra kağıtları P permütasyonuna göre değiştirir. $K \neq N$ ise, örnek değerlendirici bir hata mesajı yazdırır ve programı sonlandırır.

`determine_steps` işlevini çağırdıktan sonra, örnek değerlendirici şu çıktıyı verir:

```
C H
D[0] D[1] ... D[H-1]
```

Burada, H , `determine_steps` tarafından döndürülen D dizisinin uzunluğudur.