B. Dark Ride (Karanlık Yolculuk)

Problem Adı	Dark Ride (Karanlık Yolculuk)
Zaman Limiti	1 saniye
Hafıza Limiti	1 gigabyte

Erika, yakın zamanda Bonn yakınlarındaki Phantasialand eğlence parkında bir yaz işi bulmuştur. Erika, bu parkta yolculuk aracının geçtiği odalardaki ışıkları kontrol etmekle görevlendirilmiştir.

Yolculuk aracı, 0 ile $N-1$ arasında numaralandırılmış N tane odadan geçmektedir. Bu yolculukta odalar, 0 odasından başlanarak $N-1$ odasında sona erecek şekilde sırayla geçilmektedir. Odalardaki ışıklar, her oda için bir tane olmak üzere N tane anahtarla kontrol edilmektedir. Bu anahtarlar aynı zamanda 0 ile $N-1$ arasında numaralandırılmıştır. s anahtarı (burada $0 \leq s < N$) p_s odasındaki ışığı kontrol etmektedir.

Erika'nın patronu, Erika'dan ilk ve son odadaki ışıkları açmasını ve diğerlerini kapatmasını ister. Kulağa kolay geliyor, değil mi? Erika'nın tek yapması gereken, A ve B anahtarlarını açmaktır öyle ki $p_A = 0$ ve $p_B = N-1$ (ya da $p_B = 0$ ve $p_A = N-1$). Ne yazık ki Erika, patronu ona yapılması gerekenleri anlatırken tam olarak dikkat etmemiştir ve dolayısıyla p **dizisini, yani hangi anahtarın hangi odayı kontrol ettiğini** hatırlamamaktadır.

Erika, patronu fark etmeden önce bunu çözmeli. Her yolculuktan önce Erika tüm ışıkları kapatır ve anahtarların bir kısmını açabilir. Yolculuk odadan odaya geçerken, aydınlık bir odadan aydınlık olmayan bir odaya veya tam tersi yönde geçtiğinde, Erika yolcuların heyecanla çığlık attığını duyacaktır. Yolculuğun hızı değişebileceğinden, Erika hangi odaların aydınlık olduğunu doğrudan çıkaramaz ama en azından çığlık sayısını duyacaktır. Yani, yolculuğun aydınlık bir odadan aydınlık olmayan bir odaya veya aydınlık olmayan bir odadan aydınlık bir odaya kaç kez geçtiğini öğrenecektir.

Erika'nın patronu fark etmeden önce ilk ve son odaların ışıklarını hangi iki anahtarın kontrol ettiğini bulmasına yardım eder misin? En fazla 30 tane yolculuk kullanabilirsin.

Etkileşim

Bu interaktif bir problemdir.

- Programınız karanlık yolculuktaki oda sayısını belirten N tamsayısını içeren bir satırı okuyarak başlamalıdır.
- Daha sonra programınız grader ile etkileşime geçmelidir. Bir yolculuğu başlatmak için, soru işaretiyle "`?`" başlayan bir satır ve ardından `0` (kapalı) ve `1` (açık) olmak üzere N uzunluğunda bir dizgi (String) yazdırmalısınız. Bu dizgi, N adet anahtarı nasıl ayarladığınızı gösterir. Sonrasında, programınız Erika'nın yolcuların çığlıklarını kaç kez duyduğunu gösteren tek bir tam sayı ℓ ($0 \leq \ell < N$) okumalıdır.
- Cevap vermek istediğinizde, ünlem işareti " `!` " ve ardından iki tam sayı A ve B ($0 \leq A, B < N$) içeren bir satır yazdırın. Cevabınızın kabul edilebilmesi için, bunların iki uç odayı kontrol eden anahtarların indeksleri olmalıdır. Bu indeksker herhangi bir sırayla yazılabilir. Bundan sonra programınız kapanmalıdır.

Grader adaptif değildir, yani gizli dizi p etkileşim başlamadan önce belirlenir.

Her yolculuğun cevabından sonra standart çıktıyı flush ettiğinizden emin olun, aksi takdirde programınız Zaman Sınırı Aşıldı (Time Limit Exceeded) olarak değerlendirilebilir. Python'da, satırları okumak için `input()` kullandığınız sürece bu otomatik olarak gerçekleşir. C++'da, `cout << endl;` yeni bir satır yazdırmanın yanı sıra flush yapar; printf kullanıyorsanız, `fflush(stdout)` kullanın.

Kısıtlar ve Puanlama

- $3 \leq N \leq 30\,000$.
- En fazla 30 yolculuk hakkı verebilirsiniz (son cevabı yazdırmak yolculuk olarak sayılmaz). Bu sınırı aşarsanız, "Yanlış Cevap" (Wrong Answer) kararı alırsınız.

Çözümünüz, her biri belirli bir puan değerinde olan bir dizi test grubunda test edilecektir. Her test grubu, test case'ler içerir. Bir test grubunun puanını almak için, test grubundaki tüm test case'leri çözmeniz gerekir.

Grup	Puan	Sınırlar
1	9	$N = 3$
2	15	$N \leq 30$
3	17	$p_0 = 0$, yani, 0 anahtarı 0 odasını kontrol eder
4	16	N çifttir ve baş veya son odaların anahtarlarından biri ilk yarıdadır ($0 \leq A < \frac{N}{2}$) ve diğeri ikinci yarıdadır ($\frac{N}{2} \leq B < N$).
5	14	$N \leq 1000$
6	29	Ek kısıt yoktur

Test Araçları

Çözümünüzün test edilmesini kolaylaştırmak için indirebileceğiniz basit bir araç sağladık. Kattis problem sayfasının altındaki "ekler" (attachments) bölümüne bakın. Bu aracı opsiyoneldir. Resmi Kattis graderin, sağlanan test aracından farklı olduğunu unutmayın.

Aracı kullanmak için, "sample1.in" gibi bir input dosyası oluşturun. Bu dosya N sayısıyla başlamalı ve ardından gizli permütasyonu belirten $p_0, p_1, ..., p_{N-1}$ satırı gelmelidir. Örneğin:

```
5
2 1 0 3 4
```

Python programları için örnek olarak `solution.pyiçin (nromal olarakpypy3 solution.py`` şeklinde çalışır) şunu çalıştırın:

```
python3 testing_tool.py pypy3 solution.py < sample1.in
```

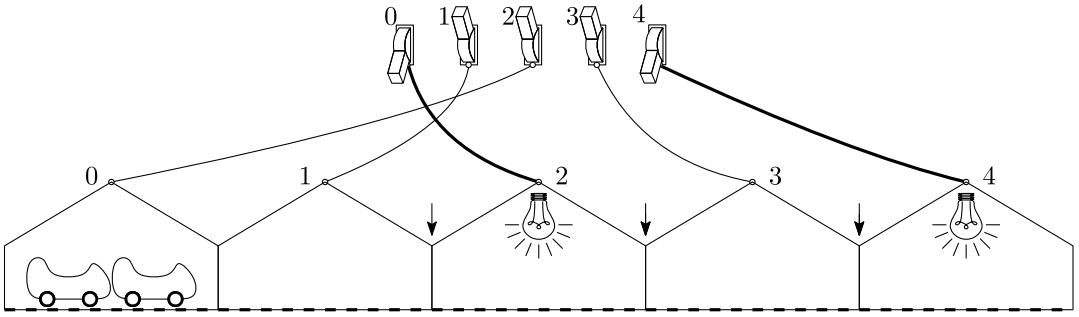
C++ programları için ilk olarak şu şekilde derleyin: (örnek `g++ -g -O2 -std=gnu++23 -static solution.cpp -o solution.out`)

ve sonra çalıştırın:

```
python3 testing_tool.py ./solution.out < sample1.in
```

Örnek

İlk örnekte gizli permütasyon $[p_0, p_1, p_2, p_3, p_4] = [2, 1, 0, 3, 4]$ şeklindedir. Bu, 2, 5 ve 6 numaralı test gruplarının kısıtlamalarını karşılar. İlk olarak, program $N = 5$ tamsayısını okur. Daha sonra, program iki adet anahtar açıkken bir yolculuk ister: anahtar 4 ve anahtar 0 . Bu $p_4 = 4$ ve $p_0 = 2$ odalarını kontrol eder; aşağıdaki çizime bakın. Erika 3 çığlık duyar (şekilde oklarla işaretlenmiştir): ilk olarak yolculuk aydınlatılmamış oda 1 'den aydınlatılmış oda 2 'ye geçerken; ikinci olarak aydınlatılmış oda 2 den aydınlatılmamış oda 3 e geçerken; ve üçüncü olarak aydınlatılmamış oda 3'ten aydınlatılmış oda 4 e geçerken. Program daha sonra p_0, p_2 ve p_3 odalarının aydınlatıldığı başka bir yolculuk ister ve Erika 3 çığlık duyar. Son olarak, program $A = 2$ ve $B = 4$ ile cevap verir ki bu doğrudur çünkü bunlar ilk ve son odaları kontrol eder ($p_2 = 0$ ve $p_4 = 4$). $A = 4$ ve $B = 2$ nin de doğru cevap olacağını unutmayın.



İkinci örnekte gizli permütasyon $[p_0, p_1, p_2] = [2, 0, 1]$ şeklindedir. Bu, 1, 2, 5 ve 6 numaralı test gruplarının kısıtlamalarını karşılar. Program, üç anahtarın da açık olduğu bir yolculuk ister. Bu, tüm odaların aydınlatılmış olduğu anlamına geldiğinden, Erika hiçbir çığlık duymayacaktır. İkinci yolculukta, 1 ve 0 anahtarları açıktır, $p_1 = 0$ ve $p_0 = 2$ odaları yanarken, 1 odası yanmaz. Erika iki çığlık duyar: yolculuk 0 odasından (aydınlık) 1 odasına (aydınlıksız) ve 1 odasından (aydınlıksız) 2 odasına (aydınlık) geçtiğinde. Son yolculukta, hiçbir anahtar açık değildir, bu da üç odanın da aydınlatılmış olmadığı ve yine Erika'nın hiçbir çığlık duymadığı anlamına gelir. Program daha sonra 1 ve 0 anahtarlarıyla yanıt verir, ki bunlar gerçekten de ilk ve son odaları kontrol eder. Hem "`! 0 1`" hem de "`! 1 0`" kabul edilen yanıtlardır.

Üçüncü örnekte gizli permütasyon $[p_0, p_1, p_2, p_3] = [0, 1, 2, 3]$ tür. Bu, 2, 3, 4, 5 ve 6 numaralı test gruplarının kısıtlamalarını karşılar.

İlk Örnek

grader çıktısı	sizin çıktınız
5	
	? 10001
3	
	? 10110
3	
	! 2 4

İkinci Örnek

grader çıktısı	sizin çıktınız
3	
	? 111
0	
	? 110
2	
	? 000
0	
	! 1 0

Üçüncü Örnek

grader çıktısı	sizin çıktınız
4	
	? 1010
3	
	! 0 3