

Elevator

Zaman sınırı: 10 saniye

Bellek sınırı: 64 MiB (her işlem için)

Bu bir iletişim problemidir.

Katları 0 ile N arasında numaralandırılmış bir bina vardır. Kat 0 zemin kattır ve kat N 'in üzerinde çatı bulunmaktadır. Dolayısıyla çatı da dahil olmak üzere binada $N + 2$ seviye vardır.

Her kat f 'de ($0 \leq f \leq N$) tam olarak bir kişi yaşamaktadır ve bu katta yalnızca o kişinin bildiği gizli bir v_f ($0 \leq v_f \leq 3$) tam sayısı vardır. Ayrıca bu binanın çatısında yaşayan Karlsson vardır. Amaç, Karlsson'ın $v_0, v_1, \dots, v_N$ değerlerinden mümkün olduğunca çoğunu belirlemesidir; binada yaşayan herkes, Karlsson'ın belirleyebileceği değer sayısını en üst düzeye çıkarmak için **iş birliği yapmaktadır**.

İletişim kurmak için bina sakinleri asansörü aşağıdaki şekilde kullanırlar.

Asansör kat 0'dan başlar ve yalnızca yukarı doğru hareket eder. Asansörün içinde 1 ile N arasındaki katlarla etiketlenmiş düğmeler bulunmaktadır. Başlangıçta hiçbir düğmeye basılmamıştır. Bir düğmeye basıldıktan sonra, o düğme kalıcı olarak basılı kalır. Asansör, $f = 0$ ise veya kat f 'ye karşılık gelen düğmeye daha önceki bir durakta basılmışsa, kat f 'de durur ve kapılarını açar. Düğmesine basılmış en yüksek kat ziyaret edildikten sonra (veya hiçbir düğmeye hiç basılmadıysa kat 0'dan sonra), asansör kalan katlarda durmadan doğrudan çatıya gider.

Asansör kat f 'de durduğunda aşağıdakiler gerçekleşir:

1. Kat sakini, o anda basılı olan düğmelerin tamamını görür.
2. Yalnızca bu bilgiye ve v_f değerine dayanarak, henüz basılmamış ve kendi katının **kesinlikle üzerinde** bulunan katlara karşılık gelen düğmelerden istediği herhangi bir alt kümeyi seçebilir. Kat sakini bu düğmelere basar.
3. Asansör, düğmesine basılmış ve henüz ziyaret edilmemiş olan en düşük numaralı kata gider (veya düğmesine basılmış tüm katlar ziyaret edilmişse çatıya gider).

Asansör bir katta durmazsa, o kattaki kişi hiçbir düğmeye basamaz.

Asansör çatıya ulaştığında Karlsson yalnızca basılmış düğmelerin son kümesini görür. Bu bilgiyi kullanarak $v_0, v_1, \dots, v_N$ değerlerinden mümkün olduğunca çoğunu belirlemeye çalışır.

v değerleri programınız başlamadan önce sabitlenir ve süreç boyunca değişmez.

Uygulama detayları

T adet test durumu olacaktır.

Bu iki fonksiyonu gerçekleştiren tek bir dosya göndermelisiniz.

Gerçekleştirmeniz gereken ilk fonksiyon aşağıdaki gibidir:

```
std::vector<int> press_buttons(int subtask, int N,  
                               int f, int v, std::vector<int> p)
```

- $subtask$: bu fonksiyonun çağrıldığı alt görev, burada $0 \leq subtask \leq 4$;
- N : son katın numarası;
- f : mevcut kat, burada $0 \leq f \leq N$;
- v : kat f üzerinde yazılı olan v_f değeri;
- p : artan sırada verilmiş, o anda basılı olan düğmeler.

Bu fonksiyon, asansör kat f 'de kapılarını açtığında çağrılır (yani $f = 0$ ise veya kat f 'ye karşılık gelen düğmeye daha önceki bir durakta basılmışsa). İlgili düğmeye daha önce basılmamışsa, bu fonksiyon o kat için çağrılmaz.

Dönüş değeri, yeni basılacak düğmelerin listesidir. Döndürülen dizideki değerlerin sırası önemli değildir. Basılacak her x düğmesi aşağıdaki koşulları sağlamalıdır:

- $f < x \leq N$;
- x , zaten p içinde bulunmamalı ve döndürülen dizide tam olarak bir kez yer almalıdır.

Gerçekleştirmeniz gereken ikinci fonksiyon aşağıdaki gibidir:

```
std::vector<int> answer(int subtask, int N, std::vector<int> p)
```

- $subtask$: bu fonksiyonun çağrıldığı alt görev, burada $0 \leq subtask \leq 4$;
- N : son katın numarası;
- p : artan sırada verilmiş, basılmış düğmelerin son kümesi.

Bu fonksiyon, asansör çatıya ulaştığında her test durumu için tam olarak bir kez çağrılır.

Döndürülen dizinin uzunluğu $N + 1$ olmalıdır; dizinin i 'inci elemanı v_i değerine eşit olmalı, Karlsson'ın **belirleyemediği** değerlerin yerine ise -1 yazılmalıdır.

Önemli: Binadaki kişiler, basılan düğmeler dışında hiçbir başka yolla iletişim kuramazlar. Bunu sağlamak için programınız $N + 2$ ayrı işlem olarak çalıştırılacaktır: her kat için bir işlem ve çatı için bir işlem. Katlara ait her işlemde en fazla T adet `press_buttons` fonksiyon çağrısı, çatı işleminde ise tam olarak T adet `answer` fonksiyon çağrısı olacaktır. Bu nedenle programınız herhangi bir

paylaşılan durumun (örneğin global değişkenlerin) varlığını varsaymamalıdır. Her işlem, diğer işlemlerin belleğinden ayrı olan 64 MiB belleğe sahip olacaktır ve tüm bu çağrılar (en fazla $(N + 1) \times T$ adet `press_buttons` ve tam olarak T adet `answer`) 10 saniye içinde tamamlanması gerekmektedir.

Kısıtlar

- $N = 60$
- $T \leq 10\,000$
- her $0 \leq i \leq N$ için $0 \leq v_i \leq 3$

Örnek

Örnek test, aşağıdaki kat değerlerine sahip 1 test durumu içermektedir:

```
1 1 0 1 1 1 1 1 1 0 1 1 1 0 1 0 1 1 0 1 1 0 1 0 1 0 1 1 1
1 0 1 1 0 1 1 1 0 1 0 1 0 1 1 0 1 1 1 1 0 1 1 1 1 0 1 1
```

Aşağıda örnek bir etkileşim verilmiştir:

Katılımcı programı	Jüri programı
	<code>press_buttons(0, 60, 0, 1, {})</code>
<code>return {2}</code>	
	<code>press_buttons(0, 60, 2, 0, {2})</code>
<code>return {13, 42}</code>	
	<code>press_buttons(0, 60, 13, 0, {2, 13, 42})</code>
<code>return {}</code>	
	<code>press_buttons(0, 60, 42, 1, {2, 13, 42})</code>
<code>return {}</code>	
	<code>answer(0, 60, {2, 13, 42})</code>
<code>return {1, -1, 0, -1, -1, ..., -1}</code>	

Bu etkileşim, 0'ıncı ve 2'nci değerleri doğru şekilde belirler. Karlsson diğer değerleri belirleyemediği için döndürülen diğer tüm değerler -1 'e eşittir.

Alt görevler

Alt görev	Puan	Ek kısıtlar	Tam puan için belirlenmesi gereken kat sayısı
0	0	Örnek.	-
1	15	$0 \leq v_i \leq 1$ $v_0 = v_N = 1$ Her $0 \leq i < N$ için $v_i = 0$ ise $v_{i+1} = 1$	60
2	35	$0 \leq v_i \leq 1$	40
3	15	$0 \leq v_i \leq 2$	30
4	35	$0 \leq v_i \leq 3$	25

Puanlama

Programınız bir alt görevin herhangi bir test durumunda yanlış belirlenmiş bir değer döndürür veya geçersiz bir işlem gerçekleştirirse (örneğin yanlış uzunlukta bir `vector` döndürürse), gönderiminiz o alt görevden sıfır puan alacaktır.

Bir test durumu için gönderiminizin *belirlenen değer sayısı*, döndürülen değeri -1 olmayan konumların sayısıdır. K , bir alt görevin tüm test durumları arasındaki en küçük belirlenen değer sayısı olsun (her alt görevde en fazla 10 000 test durumu içeren yalnızca 1 test vardır). Bu durumda her alt görev için alınacak puan aşağıdaki tabloya göre hesaplanır.

Alt görev	K aralığı	Puan
0	-	0
1	$K < 60$	$0.25K$
	$K \geq 60$	15
2	$K < 30$	$0.5K$
	$30 \leq K < 40$	$2K - 45$
	$K \geq 40$	35
3	$K < 30$	$0.5K$
	$K \geq 30$	15
4	$K < 20$	$0.7K$
	$20 \leq K < 25$	$4K - 66$
	$K \geq 25$	35

Örnek değerlendirici

Örnek değerlendirici, tüm test durumları için bütün fonksiyon çağrılarını tek bir çalıştırmada gerçekleştirir.

Girdi formatı aşağıdaki gibidir:

- satır 1: üç tam sayı – test durumu sayısı T , son katın numarası N ve alt görev numarası S ;
- satır $1 + i$: $N + 1$ adet tam sayı $v_0, v_1, \dots, v_N$, burada v_f , ilgili test durumu için kat f üzerinde yazılı olan değerdir.

Çıktı formatı aşağıdaki gibidir:

- satır i : bir tam sayı – test durumu i 'de doğru şekilde belirlenen v değerlerinin sayısı;
- satır $T + 1$: toplam puan.

Daha ayrıntılı geri bildirim almak için grader'ın ilk satırındaki `DETAILED` makrosunu `false` değerinden `true` değerine değiştirebilirsiniz.

Grader'ın sizin için test durumları (v_f değerleri) üretmesini sağlamak için grader'ın ikinci satırındaki `AUTO_GENERATE` makrosunu `false` değerinden `true` değerine değiştirebilirsiniz.