Çika hiylə işlətmək istəyir

Məsələnin adı	Hiylə
Giriş faylı	İnteraktiv sual
Çıxış faylı	İnteraktiv sual
Zaman limiti	2 saniyə
Yaddaş limiti	512 MB

Çikanın q sayda oyun kartı var və bu kartların üzərində müxtəlif müsbət tam ədədlər var. O, Shuchi'in Akademiyasından olan dostları ilə bu kartlardan istifadə edərək bəzi oyunlar oynamaq istəyir, amma eyni zamanda udmaq da istəyir, buna görə də qərara gəlir ki gizlincə kartların arxa tərəflərinə işarə qoysun.

Kartların hər biri 2×2 ölçülü kvadrat formasındadır. Kartın sol alt küncünün koordinatları $(0, 0)$, sağ üst küncünün koordinatları isə $(2, 2)$-dir. Çika hər kartın arxasına naxış çəkir ki, sonradan həmin naxışlara baxanda kartın üstündə hansı ədəd yazıldığını bilsin. O, naxışları bu prosedurdan istədiyi qədər (0 dəfə də olar) istifadə edərək çəkir: İki müxtəlif A və B nöqtələri götürür. Bu nöqtələr kartın sol alt küncünə nəzərən tam koordinatlıdır. Daha sonra o nöqtələr arasında **düz xətt seqmenti** çəkir.

Bunu edərkən Çika heç bir **səhv** seqment çəkmək istəmir. Əgər A və B nöqtələri arasında çəkilmiş seqment üzərində koordinatları tam olan başqa C nöqtəsi varsa, o zaman bu seqment səhv sayılır. Məsələn, $(0, 0)$ və $(2, 2)$ nöqtələri arasında çəkilmiş seqment səhvdir, çünki bu seqment üzərində $(1, 1)$ nöqtəsi var. Lakin, $(0, 0)$ və $(1, 1)$ nöqtələri arasında çəkilmiş seqment, və ya $(1, 1)$ və $(2, 2)$ nöqtələri arasında çəkilmiş seqment səhv deyil, və Çika bu seqmentləri hətta eyni kart üzərində çəkə bilər. Həmçinin nəzərə alın ki, seqmentlərin istiqaməti yoxdur: Yəni A-dan B-yə çəkilmiş seqment, həm özü ilə, həm də özünün tərsi olan B-dən A-ya çəkilmiş seqment ilə **eynidir**.

Vacibdir ki, Çika kartları istiqamətlərindən asılı olmadan tanıya bilsin. Kartlar ilkin istiqamətinə nəzərən saat əqrəbinin əksi istiqamətində 0, 90, 180 və ya 270 dərəcə fırladıla bilər.

Sizin tapşırığınız Çikaya q sayda kartı üçün naxışlar yaratmaq, daha sonra isə həmin naxışları tanımaqdır.

İmplementasiya detalları

Bu tapşırıq interaktivdir və iki addımdan ibarətdir. **Hər addım üçün proqram ayrı şəkildə çalışdırılacaq** Aşağıdakı iki funksiyanı implement etməlisiniz:

- Kartın arxasında çəkilməli olan naxışı qaytaran `BuildPattern` funksiyası. Bu funksiya ilk addımda q dəfə çağırılacaq.
- Kartın (fırladılmış ola bilər) üstündə yazılmış olan ədədi tapıb qaytaran `GetCardNumber` funksiyası. Bu funksiya ikinci addımda q dəfə çağırılacaq.

Birinci funksiya

```
std::vector<std::pair<std::pair<int, int>, std::pair<int, int>>> BuildPattern(int n);
```

yeganə parameter olan n ədədini, yəni kartın üstündə yazılan ədədi götürür. Siz, Çikanın kartın arxasına çəkməli olduğu seqmentləri `std::vector` vasitəsi ilə qaytaracaqsınız. Seqmentlər nöqtələrin `std::pair`-i olaraq, nöqtələr isə `std::pair` (x, y) olaraq ifadə olunub. Bu koordinatlar kartın sol alt küncünə nəzərən tam ədədlərdir və $0 \leq x, y \leq 2$. Çikanın çəkdiyi seqmentlər bir-birindən fərqli olmalı və səhv olmamalıdırlar. Zəmanət verilir ki, q sorğunun hər birində parameter olaraq müxtəlif n göndəriləcək.

q sayda kartın hər biri üçün naxışları aldıqdan sonra qreyder aşağıdakı əməliyyatlardan hər hansı birini istədiyi sayda hər bir naxış üçün tətbiq edəcək:

- Bütün naxışı 0, 90, 180 və ya 270 dərəcə saat əqrəbinin əksi istiqamətində fırlatmaq;
- `std::vector` içində naxışı göstərən seqmentlərin sırasını dəyişmək;
- Naxışdakı seqmentlərin uz nöqtələrini dəyişmək. (A-dan B-yə çəkilmiş seqmenti B-dən A-ya çəkilmiş kimi göstərə bilər)

İkinci funksiya

```
int GetCardNumber(std::vector<std::pair<std::pair<int, int>, std::pair<int, int>>> p);
```

yeganə parameter olan p-ni, yəni Çikanın kartın arxasına çəkdiyi naxışı göstərən seqment `std::vector`-nu alır. Bu `std::vector` əvvəlki funksiyadan qaytarılan cavaba əsaslanır. Funksiya kartın üstündə yazılan n ədədini qaytarmalıdır. Xatırladaq ki, p naxışı `BuildPattern` funksiyasından qaytarılmış original cavab olmaya bilər. Birinci addımdan sonra həmin cavab üzərində yuxarıda təyin olunmuş üç əməliyyat tətbiq oluna bilər. Həmçinin, kartların soruşulma sırası da birinci funksiyaya verildiyindən fərqli ola bilər.

Məhdudiyyətlər

- $1 \leq q \leq 10\,000$.

- $1 \leq n \leq 67\,000\,000$, `BuildPattern` funksiyasına olan bütün çağırışlar üçün.
- Qeyd edək ki, $67\,000\,000$ müxtəlif kartı tanıya bilən alqoritm mövcuddur.

Qiymətləndirmə

- Alt tapşırıq 1 (2 bal): $n \leq 2$.
- Alt tapşırıq 2 (9 bal): $n \leq 25$.
- Alt tapşırıq 3 (15 bal): $n \leq 1\,000$ və qreyder iki addım arasında kartları **fırlatmayacaq**. (Qreyder digər iki əməliyyatı tətbiq edə **bilər**.)
- Alt tapşırıq 4 (3 bal): $n \leq 16\,000\,000$ və qreyder iki addım arasında kartları **fırlatmayacaq**. (Qreyder digər iki əməliyyatı tətbiq edə **bilər**.)
- Alt tapşırıq 5 (24 bal): $n \leq 16\,000\,000$.
- Alt tapşırıq 6 (18 bal): $n \leq 40\,000\,000$.
- Alt tapşırıq 7 (29 bal): Əlavə məhdudiyyət yoxdur.

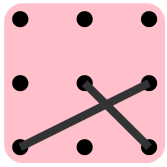
Nümunə interaksiya

Funksiya çağırışı	Qaytarılan cavab	İzah
Birinci addım başladı.	-	-
`BuildPattern(3)`	`{{{0, 0}, {2, 1}}, {{1, 1}, {2, 0}}}`	3 nömrəli ədəd üçün 2×2 kartın üzərində naxış yaratmalıyıq. 2 seqment çəkmək qərarına gəldik: - $(0, 0)$ və $(2, 1)$ arasında, - $(1, 1)$ və $(2, 0)$ arasında.
`BuildPattern(1)`	`{{{0, 1}, {0, 0}}}`	1 nömrəli ədəd üçün 2×2 kartın üzərində naxış yaratmalıyıq. 1 seqment çəkmək qərarına gəldik: - $(0, 1)$ və $(0, 0)$ arasında.
Birinci addım bitdi.	-	-
İkinci addım başladı.	-	-
`GetCardNumber({{{0, 0}, {0, 1}}})`	`1`	1 seqmentdən ibarət naxış aldıq: - $(0, 0)$ və $(0, 1)$ arasında. Bu aşağıdakı seqment ilə eynidir: - $(0, 1)$ və $(0, 0)$ arasında Bu da `BuildPattern` funksiyasına olan ikinci çağırışın cavabıdır və onunla eyni istiqamətdədir (0 dərəcə döndərilib). Buna görə də 1 qaytarırıq.
`GetCardNumber({{{1, 1}, {2, 2}}, {{1, 2}, {2, 0}}})`	`3`	2 seqmentdən ibarət naxış aldıq: - $(1, 1)$ və $(2, 2)$ arasında, - $(1, 2)$ və $(2, 0)$ arasında.

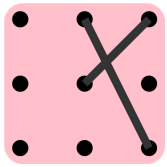
		Bu da `BuildPattern` funksiyasına olan birinci çağırışın cavabıdır və lakin 90 dərəcə saat əqrəbinin əksi istiqamətində döndərilib. Buna görə də 3 qaytarırıq.
İkinci addım bitdi.	-	-

Növbəti üç şəkil sırası ilə bunlar göstərir:

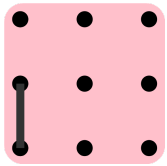
- `BuildPattern` funksiyasına ilk çağırışın cavabı olaraq qaytarılan naxış:



- `GetCardNumber` funksiyasına ikinci çağırışda ötürülən naxış, hansı ki birinci naxışın 90 dərəcə saat əqrəbi əksi istiqamətində dönməsi ilə yaranıb.



- `BuildPattern` funksiyasına ikinci çağırışa qaytarılan cavab, hansı ki `GetCardNumber` funksiyasına ilk çağırışda ötürülən naxışla eynidir.



Nümunə qreyder

`Cheat.zip` qoşmasında verilmiş nümunə qreyder `grader.cpp` standart girişdən bir tam ədəd q oxuyur və aşağıdakı addımları q dəfə yerinə yetirir:

- Standart girişdən n ədədini oxumaq.
- `BuildPattern(n)` funksiyasını çağırmaq və cavabı p dəyişənində saxlamaq.
- `GetCardNumber(p)` funksiyasını çağırmaq və qaytarılan cavabı standart çıxışa yazmaq.

Verilən qreyder üzərində lokal dəyişikliklər edə bilərsiniz.

Nümunə qreyderi öz həllinizlə bir yerdə kompayl etmək üçün aşağıdakı kodu terminalda yaza bilərsiniz:

```
g++ -std=gnu++11 -O2 -o solution grader.cpp solution.cpp
```

Burada `solution.cpp` sizin CMS-ə göndərəcəyiniz həll faylıdır. Proqramı qoşmada təqdim olunmuş nümunə giriş ilə işlətmək üçün aşağıdakı kodu terminalda yaza bilərsiniz:

```
./solution < input.txt
```

Qeyd edək ki, nümunə qreyderdən fərqli olaraq, CMS-dəki əsas qreyder birinci və ikinci addımı proqramları ayrı şəkildə işlədərək yerinə yetirəcək.