Chika Hile Yapmak İstiyor

Problem Adı	Hile
Girdi Dosyası	Etkileşimli görev
Çıktı Dosyası	Etkileşimli görev
Zaman limiti	2 saniye
Hafıza limiti	512 megabytes

Chika'nın çeşitli pozitif tam sayılarla numaralandırılmış q iskambil kartından oluşan bir destesi var. Bu kartları kullanarak Shuchi'in Akademisi öğrenci konseyinden arkadaşlarıyla bazı oyunlar oynamak istiyor, ama aynı zamanda kazanmak da istiyor, bu yüzden destesindeki kartların arkasını gizlice işaretlemeye karar veriyor.

Kartların tümü kare şeklindedir ve 2×2 boyutundadır. Sol alt köşenin koordinatları $(0, 0)$'dır ve sağ üst köşenin koordinatları $(2, 2)$'dir. Chika her kartın arkasına belirli bir desen çizer, böylece daha sonra desene bakarak kartın ön yüzünde hangi sayının yazılı olduğunu bilecektir. Desenleri çizmek için aşağıdaki prosedürü kullanır: İstediği kadar (muhtemelen 0 kere), kartın sol alt köşesine göre tamsayı koordinatları olan iki ayrı A ve B noktası seçer ve aralarına bir **doğru parçası** çizer.

Chika yalnızca **geçerli** doğru parçaları çizecektir. A'dan B'ye çizilen bir doğru parçasının geçerli bir doğru parçası olması için bu doğru parçası üzerinde A ve B'den farklı ve tamsayı koordinatları olan bir C noktası olmamalıdır. Örneğin, $(0, 0)$ ile $(2, 2)$ arasına çizilecek bir doğru parçası $(1, 1)$ noktasını içerdiği için **geçersizdir**. Ancak $(0, 0)$ ile $(1, 1)$ arasına çizilecek bir doğru parçası, yada $(1, 1)$ ile $(2, 2)$ arasına çizilecek bir doğru parçası **geçerlidir**. Hatta Chika bu iki doğru parçasını birden bir deseni oluştururken çizebilir. Ayrıca, doğru parçalarının yönü olmadığına dikkat edin: A'dan B'ye çizilen bir doğru parçası hem kendisiyle hem de tersi yönde (B'den A'ya) çizilen doğru parçasıyla **özdeştir**.

Daha da önemlisi, Chika, nasıl döndürüldüklerine bakılmaksızın kartlarını tanıyacağından emin olmak istiyor. Bir kart, orijinal yönüne göre saat yönünün tersine 0, 90, 180 veya 270 derece döndürülebilir.

Sizin göreviniz Chika'nın destesindeki q kartının desenlerini tasarlamasına ve sonrasında bu kartları tanımasına yardım etmektir.

Gerçekleştirme

Bu, iki aşamalı etkileşimli bir görevdir. ***her aşama programınızın ayrı bir çalışmasını içerir***. İki fonksiyonu kodlamanız gerekmektedir:

- `BuildPattern`: Verilen bir kartın arkasına çizilecek deseni döner. Bu fonksiyon ilk aşamada q kere çağrılacak.
- `GetCardNumber`: İlk aşamada çizilen bir deseni taşıyan (muhtemelen döndürülmüş) bir kartın numarasını döner. Bu fonksiyon ikinci aşamada q kez çağrılacak.

İlk fonksiyon,

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

parametre olarak tek bir n tamsayısını alır. Bu sayı kartın ön yüzünde yazılı olan sayıdır. Chika'nın kartın arka yüzüne çizdiği doğru parçalarını içeren bir `std::vector` dönmelisiniz. Bir doğru parçası noktaların bir `std::pair`'ı ile temsil edilir. $0 \leq x, y \leq 2$ olmak üzere kartın sol alt kösesine göre tamsayı koordinatlı bir (x, y) noktası ise (x, y) `std::pair`'ı ile temsil edilir. Chika'nın çizdiği tüm doğru parçalarının geçerli olması ve ikili olarak özdeş olmaması gerekir. `BuildPattern` fonksiyonuna yapılacak q çağrının her birinde parametre olarak farklı bir n tamsayısının verileceği garanti edilmiştir.

q kartın herbirinin desenlerini aldıktan sonra, değerlendirici her bir desen üzerinde aşağıdaki işlemlerden herhangi birini herhangi bir sayıda yapabilir:

- Tüm deseni saat yönünün tersine 0, 90, 180 veya 270$ derece döndürmek.
- Desenin `std::vector` gösterimindeki doğru parçalarının sırasını değiştirmek.
- Desendeki bir doğru parçasının uç noktalarının sırasını değiştirmek. (A'dan B'ye çizilen bir doğru parçasını B'den A'ya çizilen özdeş doğru parçasına dönüştürmek.)

İkinci fonksiyon,

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

tek bir p parametresi alır. Bu parametre sizin `BuildPattern` fonksiyonunuzun daha önceki bir çağrıya döndüğü değere bağlı olarak Chika'nın kartın arka yüzüne çizdiği desendeki doğru parçalarını temsil eden bir `std::vector`'dür. Fonksiyon kartın ön yüzündeki n tamsayısını dönmelidir. p'nin, `BuildPattern` tarafından dönülen orijinal biçimde olması gerekmediğini hatırlayın; yukarıda bahsedilen üç işleme bağlı olarak değiştirilmiş olabilir. Kartların sırasının ilk aşamada verildiklerinden farklı olması da mümkündür. Ancak her bir kartın tam olarak bir kere kullanılacağı garanti edilmektedir.

Kısıtlar

- $1 \leq q \leq 10\,000$.
- `BuildPattern` fonksiyonuna bütün çağrılar için $1 \leq n \leq 67\,000\,000$.
- $67\,000\,000$ farklı kartı tanıyabilecek şekilde desenler oluşturmak için algoritmalar olduğunu unutmayın.

Puanlama

- Altgörev 1 (2 puan): $n \leq 2$.
- Altgörev 2 (9 puan): $n \leq 25$.
- Altgörev 3 (15 puan): $n \leq 1\,000$ ve değerlendirici 1. ve 2. aşamalar arasında desenleri **döndürmez**. (Değerlendirici diğer iki işlemi **yapabilir**.)
- Altgörev 4 (3 puan): $n \leq 16\,000\,000$ ve değerlendirici 1. ve 2. aşamalar arasında desenleri **döndürmez**. (Değerlendirici diğer iki işlemi **yapabilir**.)
- Altgörev 5 (24 puan): $n \leq 16\,000\,000$.
- Altgörev 6 (18 puan): $n \leq 40\,000\,000$.
- Altgörev 7 (29 puan): Ek kısıt yok.

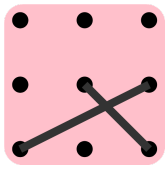
Örnek Etkileşim

Fonksiyon Çağrısı	Dönen değer	Açıklama
İlk aşama başlar.	-	-
`BuildPattern(3)`	`{{{0, 0}, {2, 1}}, {{1, 1}, {2, 0}}}`	3 sayısı için 2×2'lik kart üzerinde bir desen oluşturmalıyız. 2 tane doğru parçası çizmeye karar veriyoruz: - $(0, 0)$ ve $(2, 1)$ arasında, - $(1, 1)$ ve $(2, 0)$ arasında.
`BuildPattern(1)`	`{{{0, 1}, {0, 0}}}`	1 sayısı için 2×2'lik kart üzerinde bir desen oluşturmalıyız. 1 tane doğru parçası çizmeye karar veriyoruz: - $(0, 1)$ ve $(0, 0)$ arasında.
İlk aşama biter.	-	-
İkinci aşama başlar.	-	-
`GetCardNumber({{{0, 0}, {0, 1}}})`	`1`	1 doğru parçasından oluşan bir desen alırız. - $(0, 0)$ ve $(0, 1)$ arasında. Bu, aşağıdaki doğru parçasını çizerek elde edeceğimiz desenin aynısıdır. - $(0, 1)$ ve $(0, 0)$ arasında. Bu `BuildPattern` fonksiyonunun ikinci çağrılışında döndüğümüzle aynı

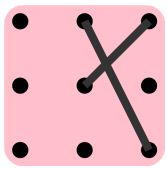
		oryantasyona sahip (0 derece döndürülmüş), tam olarak aynı desendir. Bu nedenle 1 döneriz.
`GetCardNumber( {{{1, 1}, {2, 2}}, {{1, 2}, {2, 0}}})`	`3`	2 doğru parçasından oluşan bir desen alırız. - $(1, 1)$ ve $(2, 2)$ arasında, - $(1, 2)$ ve $(2, 0)$ arasında. Bu `BuildPattern` fonksiyonunun ilk çağrılışında döndüğümüz desenin 90 derece saatin tersi yönde (counter-clockwise) döndürülmüş halidir. Bu nedenle 3 döneriz.
İkinci aşama biter.	-	-

Aşağıdaki 3 figür sırasıyla şunları temsil eder:

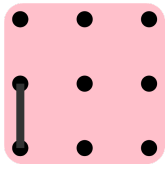
- `BuildPattern` fonksiyonunun ilk çağrılışında çıktı olarak dönen desen:



- `GetCardNumber` fonksiyonunun ikinci çağrılışında parametre olarak aldığı desen. Bu, 90 derece saatin tersi yönde (counter-clockwise) döndürüldükten sonraki ilk desendir.



- `BuildPattern` fonksiyonunun ikinci çağrılışında çıktı olarak dönen desen. Bu aynı zamanda `GetCardNumber` fonksiyonunun ilk çağrılışında argüman olarak aldığıyla aynı desendir.



Örnek Değerlendirici

`Cheat.zip` görev ekinde verilen `grader.cpp` örnek değerlendirici, standart girdiden bir tamsayı q okur, ve sonrasında aşamaları q defa yapar:

- Standart girdiden bir tamsayı n okur.
- `BuildPattern(n)` fonksiyonunu çağırır ve dönen değeri p değişkeninde tutar.

- `GetCardNumber(p)` fonksiyonunu çağırır ve dönen değeri standart çıktıya yazar.

İsterseniz değerlendiricinizi kendi bilgisayarınızda (lokal) değiştirebilirsiniz.

Örnek değerlendiriciyi çözümünüzle beraber derlemek için terminalde aşağıdaki komutu yazabilirsiniz:

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

Burada `solution.cpp`, CMS'ye gönderilecek çözüm dosyanızdır. Programı ekte sağlanan örnek girdiyle çalıştırmak için terminalde aşağıdaki komutu yazın:

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

Lütfen unutmayın, örnek değerlendiriciden farklı olarak, CMS'deki gerçek değerlendirici, birinci aşamayı ve ikinci aşamayı programınızın ayrı çalıştırmalarında (execution) gerçekleştirecektir.