문제4 : 아이싱

수많은 중고등학생들에게 아이싱이라는 온라인 일대일 대전 게임이 유행하고 있다. 게임회사에서는 이 게임을 확장하여 팀 I와 팀 Sing 사이에 대전을 할 수 있는 팀플레이 전용 아이싱II를 개발하였다. 아이싱II는 게이머들의 기존 아이싱에서의 대전 이력을 활용하여 독특한 방식으로 팀을 구성하도록 설계되었다. 즉, 아이싱에서 일대일 대전을 한 두 게이머는 서로 다른 팀에 속해야 한다. 각 팀의 인원수는 1명 이상이며 두 팀의 인원수는 서로 달라도 된다.

예를 들어, 아래 표는 네 명의 게이머 A, B, C, D가 아이싱에서 일대일 대전을 한 기록을 이용하여 아이싱II의 팀을 구성한 예이다.

아이싱 대전 기록
A - B A - C B - D C - D

▶

아이싱II
팀 I : A, D 팀 Sing : B, C

<예시 1>

그런데, 이제 3분 후면 아이싱II 서버 오픈인데 치명적인 문제가 발견되었다. 예를 들어, 게이머 A, B, C가 모두 서로 대전한 적이 있다면 각각 다른 팀에 들어가야 하므로 팀 I와 팀 Sing 구성이 불가능한 것이다.

이 문제를 해결하기 위해 다음과 같은 해법을 적용하려 한다. 바로 아이싱 대전 기록 중 최소 개수의 대전 기록을 지우는 것이다. 예를 들어, 아래 그림과 같이 네 명의 게이머 A, B, C, D의 아이싱에서 일대일 대전 기록이 주어질 때(왼쪽)를 생각해 보자. 이 경우 최소 하나의 기록을 지워야 하며, 대전 기록 중 A - B를 지운다면(가운데) 두 팀 구성이 가능하다(오른쪽).

아이싱 대전 기록
A - B A - C A - D B - C

▶

최소로 삭제한 후 아이싱 대전 기록
A - C A - D B - C

▶

아이싱II
팀 I : A, B 팀 Sing : C, D

<예시 2>

다른 예로, 다음과 같은 아이싱 대전 기록을 생각해 보자. 이 경우 최소 두 개의 대전 기록을 지워야 하며, A - B와 C - D를 지운다면 아래와 같이 두 팀을 구성할 수 있다.

아이싱 대전 기록
A - B A - C A - D B - C B - D C - D

▶

최소로 삭제한 후 아이싱 대전 기록
A - C A - D B - C B - D

▶

아이싱II
팀 I : A, B 팀 Sing : C, D

<예시 3>

하지만, 게임의 런칭이 3분 밖에 남지 않았기 때문에 지울 수 있는 기록의 개수는 **최대 두 개로 한정된다**. 만약에 3개 이상의 기록을 지워야 한다면 런칭을 지연하는 것이 더 좋을 수도 있다.

이 와중에 게임회사 대표는 두 팀을 구성하기 위해 지울 수 있는 최소 크기의 대전 기록 집합이 얼마나 다양하게 존재하는지 궁금해졌다. 즉, 최소로 지워야 하는 대전 기록의 개수가 K일 때, 크기 K인 지울 수 있는 대전 기록의 부분집합의 경우의 수를 세어야 한다.

<예시 2>에서는 {A – B}, {A – C}, 또는 {B – C}를 지우면 최소 개수의 대전 기록을 지우면서 두 팀을 구성할 수 있다. 이 경우의 답은 3이다. <예시 3>에서는 {A – B, C – D}, {A – C, B – D}, 또는 {A – D, B – C}를 지우면 최소 개수의 대전 기록을 지우면서 두 팀을 구성할 수 있다. 이 경우의 답은 3이다.

여러분은 주어진 아이싱 대전 기록을 이용하여 두 팀을 구성할 수 있도록 지울 수 있는 **최소 개수의 대전 기록을 선택하는 서로 다른 방법의 개수**를 구하기 위해서 다음 함수를 구현해야만 한다. 이때 지울 대전 기록의 선택 순서는 중요하지 않다. **또한, 2개 이하의 대전 기록을 지워서 게임을 진행할 수 없다면, 0을 반환한다.** 만약, 아무 기록도 지워야 할 필요가 없다면, 아무 것도 안 지우는 것이 가장 빠른 유일한 방법이니 방법의 개수는 1이다.

- `long long count_ways( int N, vector<int> U, vector<int> V ) ;` N은 게이머 수, U와 V는 크기 M인 vector로 U_i와 V_i 사이에 아이싱 대전 기록이 존재한다. 이를 이용하여 지울 최소 크기의 대전 기록을 선택하는 서로 다른 방법의 개수를 구하여 return한다. 단, 3개 이상의 기록을 지워야만 한다면 0을 return한다.

구현 세부사항

여러분은 ising.cpp라는 이름을 가진 하나의 파일을 제출해야만 한다. 이 파일에는 다음의 함수가 구현되어 있어야 한다.

- `long long count_ways( int N, vector<int> U, vector<int> V ) ;`

이 함수는 위에서 설명한 것과 같이 동작하여야 한다. 물론, 다른 함수들을 만들어서 내부적으로 사용할 수 있다. 제출한 코드는 입출력을 수행하거나 다른 파일에 접근하여서는 안된다.

grader 예시

주어지는 grader는 다음과 같은 형식으로 입력을 읽는다.

- line 1: N M
  (N: 게이머 수, M: 아이싱 대전 기록 개수)
- line 2+i ($0 \le i \le M-1$): U_i V_i
  (게이머 U_i와 게이머 V_i 사이의 아이싱 대전 기록)

주어지는 grader는 여러분의 코드가 `count_ways()` 함수에서 리턴한 값을 출력한다.

제한 조건

- $3 \le N \le 250{,}000$
- $N-1 \le M \le 250{,}000$
- $1 \le U_i, V_i \le N$ $(0 \le i \le M-1)$
- $U_i \ne V_i$
- $U_i = U_j$이고 $V_i = V_j$이면 $i = j$임

  ($U_i = V_j$이고 $V_i = U_j$인 경우는 없음)
- 어떤 두 게이머도 대전 기록을 통해 서로 연관됨

  (직접 대전을 했거나, 대전한 다른 게이머의 대전 관계를 반복적으로 통하여)

서브태스크 1 [10 points]

- $N \le 200$, $M \le 200$

서브태스크 2 [23 points]

- $N \le 5{,}000$, $M \le 5{,}000$

서브태스크 3 [11 points]

- $N \le 500$

서브태스크 4 [25 points]

- $N \le 5{,}000$

서브태스크 5 [31 points]

- $M - N \le 200$

서브태스크 6 [50 points]

- 추가 제한이 없다.

[입력 예 1]

```
4 4
1 2
1 3
2 4
3 4
```

[출력 예 1]

```
1
```

아래는 예 1에 대해 함수 호출 및 그 결과를 보여준다.

함수호출	결과 값
count(4, {1, 1, 2, 3}, {2, 3, 4, 4})	1

[입력 예 2]

```
4 4
1 2
1 3
1 4
2 3
```

[출력 예 2]

```
3
```

[입력 예 3]

```
4 6
1 2
1 3
1 4
2 3
2 4
3 4
```

[출력 예 3]

```
3
```

[입력 예 4]

```
5 10
1 2
1 3
1 4
1 5
2 3
2 4
2 5
3 4
3 5
4 5
```

[출력 예 4]

```
0
```

[입력 예 5]

```
12 16
1 2
2 3
3 4
4 1
5 6
6 7
7 8
8 9
9 10
10 11
11 12
12 5
1 5
2 7
3 9
4 11
```

[출력 예 5]

```
2
```