

문제 1. 항공 노선도

시간 제한: 2초
메모리 제한: 1024MB

Alice는 JOI 왕국에 살고 있다. 그는 IOI 공화국에 사는 Bob을 초대하게 되어 JOI 왕국 내의 항공 노선도를 Bob에게 보내게 되었다. JOI 왕국은 0번 섬부터 $N - 1$ 번 섬까지 총 N 개의 섬으로 이루어진 섬나라이다. JOI 왕국에는 M 개의 항공 노선이 있어서 $i + 1$ 번째 ($0 \leq i \leq M - 1$) 항공 노선은 A_i 번 섬과 B_i 번 섬을 양방향으로 잇는다. 같은 쌍의 섬을 잇는 서로 다른 두 개의 항공 노선은 존재하지 않는다. JOI 왕국에서 IOI 공화국으로 정보를 보내기 위해서는, JOI 왕국이 운영하는 전보를 써야 한다. 하지만 전보를 쓸 때, 정점 번호와 간선 번호가 무작위로 섞일 것이다.

정확히 전보는 다음과 같은 순서로 보내지게 된다. G 를 Alice가 보내는 그래프라고 하자. (V 를 G 의 정점 개수, U 를 G 의 간선 개수라고 하자.)

- Alice는 G 의 정점 개수 V 와 간선 개수 U 를 지정한다. 또한, G 의 각 정점에 $0, 1, \dots, V - 1$ 의 번호를 G 의 각 간선에 $0, 1, \dots, U - 1$ 의 번호를 각각 붙인다.
- Alice는 인자 $C_0, C_1, \dots, C_{U-1}$ 및 $D_0, D_1, \dots, D_{U-1}$ 을 지정한다. 이 인자는 G 의 간선을 의미한다. 즉, 각 j ($0 \leq j \leq U - 1$) 에 대해, G 의 간선 j 는 정점 C_j 와 D_j 를 연결한다.
- JOI 왕국에 의해 G 의 정점 번호가 무작위로 섞인다. 우선 JOI 왕국은 $0, 1, \dots, V - 1$ 로 이루어진 순열 $p[0], p[1], \dots, p[V - 1]$ 을 생성한다. 다음, $C_0, C_1, \dots, C_{U-1}$ 을 $p[C_0], p[C_1], \dots, p[C_{U-1}]$ 로, $D_0, D_1, \dots, D_{U-1}$ 을 $p[D_0], p[D_1], \dots, p[D_{U-1}]$ 로 바꾼다.
- 이어서, JOI 왕국에 의해 G 의 간선 번호가 무작위로 섞인다. 우선 JOI 왕국은 $0, 1, \dots, U - 1$ 로 이루어진 순열 $q[0], q[1], \dots, q[U - 1]$ 을 생성한다. 다음, $C_0, C_1, \dots, C_{U-1}$ 을 $C_{q[0]}, C_{q[1]}, \dots, C_{q[U-1]}$ 로, $D_0, D_1, \dots, D_{U-1}$ 을 $D_{q[0]}, D_{q[1]}, \dots, D_{q[U-1]}$ 로 바꾼다.
- V, U 의 값과 바뀐 후의 인자 $C_0, C_1, \dots, C_{U-1}$ 및 $D_0, D_1, \dots, D_{U-1}$ 가 Bob에게 전달된다.

단, 이 전보로 보낼 수 있는 그래프는 단순 그래프 뿐이다. 단순 그래프는 다중간선이나 루프가 없는 그래프이다. 즉, 각 i, j ($0 \leq i < j \leq U - 1$)에 대해 $(C_i, D_i) \neq (C_j, D_j)$ 와 $(C_i, D_i) \neq (D_j, C_j)$ 를 만족하며 또한 모든 i ($0 \leq i \leq U - 1$) 에 대해 $C_i \neq D_i$ 를 만족하는 그래프를 보내는 것이 가능하다.

Alice는 가능한 한 적은 정점 수의 그래프로 Bob에게 JOI 왕국 내의 항공 노선도를 보내고 싶다.

Alice와 Bob의 통신을 구현하기 위해 다음 2개의 프로그램을 작성하여야.

1. 첫 번째 프로그램은 JOI 왕국의 섬의 수 N , JOI 왕국의 항공 노선의 수 M , JOI 왕국의 항공 노선을 의미하는 수열 A, B 가 주어질 때, Alice가 보내는 그래프 G 의 정보를 만든다.
2. 두 번째 프로그램은 Bob이 받은 그래프 G 가 주어질 때, JOI 왕국의 항공 노선도를 복원한다.

구현 명세

당신은 파일 두 개를 제출해야 한다.

첫째 파일의 이름은 `Alice.cpp`이다. 이 파일은 Alice의 일을 나타내고 다음 함수를 구현해야 한다. 또한, `Alicelib.h`를 `include`해야 한다.

- `void Alice(int N, int M, int A[], int B[])`
이 함수는 각 테스트케이스마다 정확히 한 번 불린다.
 - 인자 N 은 JOI 왕국의 섬의 수를 의미한다.
 - 인자 M 은 JOI 왕국의 항공 노선의 수 M 을 의미한다.
 - 인자 $A[], B[]$ 는 길이 M 의 배열이며, JOI 왕국의 항공 노선을 의미한다.

함수 Alice 안에서는 다음의 함수를 호출해서 보낼 그래프 G 의 정보를 설정한다.

- void InitG(int V, int U)
이 함수를 호출하는 것으로 G 의 정점 개수와 간선 개수를 설정한다.
 - * 인자 V는 G 의 정점 개수를 의미한다. V는 1 이상 1500 이하의 정수여야 한다. 이 범위 밖의 수로 함수를 호출한 경우 **오답 [1]**이 된다.
 - * 인자 U는 G 의 간선 개수를 의미한다. U는 0 이상 $V(V-1)/2$ 이하의 정수여야 한다. 이 범위 밖의 수로 함수를 호출한 경우 **오답 [2]**이 된다.
- void MakeG(int pos, int C, int D)
이 함수는 G 의 간선을 설정한다.
 - * 인자 pos는 G 의 정점 번호를 의미한다. pos는 0 이상 $U-1$ 이하의 정수여야 한다. 이 범위 밖의 수로 함수를 호출한 경우 **오답 [3]**이 된다. 또한, 같은 인자 pos로 함수를 두 번 이상 호출할 수 없다. 같은 인자로 두 번 이상 호출한 경우 **오답 [4]**이 된다.
 - * 인자 C와 D는 pos번 간선의 양쪽의 정점을 의미한다. C와 D는 0 이상 $V-1$ 이하의 정수여야 한다. 또한, $C \neq D$ 를 만족해야 한다. C 혹은 D가 이 조건을 만족하지 않은 경우, **오답 [5]**가 된다.

여기서 U 와 V 는 initG에서 설정된 값이다.

함수 Alice 안에서는 함수 InitG를 한 번 호출한 후, 함수 MakeG를 정확히 U 번 호출해야 한다. 함수 InitG가 두 번 이상 호출된 경우 **오답 [6]**이 된다. 함수 InitG가 호출되기 전에 함수 MakeG가 호출된 경우 **오답 [7]**이 된다. 함수 Alice의 실행이 완료될 때 함수 InitG가 호출되지 않은 경우 혹은 MakeG의 호출 횟수가 U 가 아니었던 경우 **오답 [8]**이 된다. 함수 Alice의 실행이 종료될 때 지정한 그래프 G 가 단순 그래프가 아닌 경우, **오답 [9]**이 된다.

Alice의 호출이 오답으로 판정된 경우 그 시점에서 프로그램은 종료된다.

둘째 파일의 이름은 Bob.cpp이다. 이 파일은 Bob의 일을 나타내고 다음 함수를 구현해야 한다. 또한, Bob.h를 include해야 한다.

- void Bob(int V, int U, int C[], int D[])
이 함수는 각 테스트케이스마다 정확히 한 번 불린다.
 - 인자 V은 그래프 G 의 정점 개수를 의미한다.
 - 인자 U은 그래프 G 의 간선 개수를 의미한다.
 - 인자 C[], D[]는 길이 U 의 배열이며, 그래프 G 를 의미한다.

함수 Bob 안에서는 다음의 함수를 호출해서 복원한 JOI 왕국의 항공 노선도의 정보를 복원한다.

- void InitMap(int N, int M)
이 함수를 호출하는 것으로 JOI 왕국의 섬의 수와 JOI 왕국의 항공 노선의 수를 복원한다.
 - * 인자 N은 복원한 JOI 왕국의 섬의 수를 의미한다. N은 실제 JOI 왕국의 섬의 수와 일치해야 한다. 일치하지 않을 경우, **오답 [10]**이 된다.
 - * 인자 M은 복원한 JOI 왕국의 항공 노선의 수를 의미한다. N은 실제 JOI 왕국의 항공 노선의 수와 일치해야 한다. 일치하지 않을 경우, **오답 [11]**이 된다.
- void MakeMap(int A, int B)
이 함수를 호출하는 것으로 JOI 왕국의 항공 노선을 복원한다.
 - * 인자 A와 B는 JOI 왕국의 A번 섬과 B번 섬을 잇는 항공 노선이 존재한다는 것을 의미한다. A와 B는 0 이상 $N-1$ 이하의 정수여야 한다. 또한, $A \neq B$ 를 만족해야 한다. A 혹은 B가 이 조건을 만족하지 않은 경우 **오답 [12]**가 된다. JOI 왕국의 섬 A와 섬 B를 연결하는 항공 노선이 없을 경우 **오답 [13]**이 된다. 또한, 과거에 호출한 것과 같은 항공 노선을 호출해서는 안 된다. MakeMap(A, B)를 호출할 때 이미 MakeMap(A, B) 또는 MakeMap(B, A)를 호출한 경우 **오답 [14]**이 된다.

여기서 N 은 `initMap`에서 설정된 값이다.

함수 `Bob` 안에서는 함수 `InitMap`을 한 번 호출한 후, 함수 `MakeMap`을 정확히 M 번 호출해야 한다. 함수 `InitMap`이 두 번 이상 호출된 경우 **오답 [15]**이 된다. 함수 `InitMap`이 호출되기 전에, 함수 `MakeMap`이 호출된 경우 **오답 [16]**이 된다. 함수 `Bob`의 실행이 완료 될 때, 함수 `InitMap`이 호출되지 않은 경우 혹은 `MakeMap`의 호출 횟수가 M 이 아니었던 경우 **오답 [17]**이 된다. 여기서 M 은 `initMap`에서 설정된 값이다.

`Bob`의 호출이 오답으로 판정된 경우 그 시점에서 프로그램은 종료된다.

채점은 다음과 같은 방식으로 진행된다. 오답으로 판정된 경우 그 시점에서 프로그램은 종료된다.

- (1) JOI 왕국의 항공 노선도의 정보를 인자로 하여 함수 `Alice`를 한 번 호출한다.
- (2) 함수 `Alice`안에서 설정된 그래프를 G 라고 하자. G 의 정점 번호와 간선 번호를 뒤섞은 그래프를 의미하는 정보를 인자로 하여 함수 `Bob`을 한 번 호출한다.
- (3) 채점이 종료된다.

참고 사항

- 당신의 프로그램은 내부에서 사용할 목적으로 함수나 전역변수를 사용할 수 있다. 제출된 파일들은 같이 컴파일되어 하나의 실행 파일이 된다. 모든 글로벌 변수나 함수는 충돌을 피하기 위하여 `static`으로 선언되어야 한다. 채점될 때는, `Alice`과 `Bob`에 해당하는 두 프로세스로 나누어서 실행될 것이다. `Alice`의 프로세스와 `Bob`의 프로세스는 전역변수를 공유할 수 없다.
- 당신의 프로그램은 표준 입출력을 사용해서는 안 된다. 당신의 프로그램은 어떠한 방법으로도 다른 파일에 접근해서는 안 된다. 단, 당신의 프로그램은 디버그 목적으로 표준 에러출력에 출력할 수 있다.

당신은 대회 홈페이지의 아카이브에서 프로그램을 테스트하기 위한 목적의 샘플 그레이더를 받을 수 있다. 아카이브는 당신의 프로그램의 예제 소스 또한 첨부되어 있다. 샘플 그레이더는 파일 `grader.cpp`이다. 당신의 프로그램을 테스트하기 위해서, `grader.cpp`, `Alice.cpp`, `Bob.cpp`, `Alicelib.h`, `Boblib.h`를 같은 디렉토리 안에 놓고, 컴파일하기 위해 다음 커맨드를 실행하여야.

- `g++ -std=c++14 -O2 -o grader grader.cpp Alice.cpp Bob.cpp`

컴파일이 성공적이면, 파일 `grader`가 생성된다.

실제 그레이더와 샘플 그레이더는 다름에 주의하여야. 샘플 그레이더는 하나의 프로세스에서 실행되며, 입력을 표준 입력으로부터 받고, 출력을 표준 출력에 출력한다.

입력 형식

샘플 그레이더는 표준 입력에서 다음과 같은 형식으로 입력받는다.

- 첫째 줄에는 정수 N , M 이 주어진다. 이는 JOI 왕국의 섬의 수가 N 이며, JOI 왕국의 항공 노선의 수가 M 이라는 것을 의미한다.
- 다음 M 개의 줄에는 M 개의 항공 노선의 정보가 주어진다. M 개의 줄 중 $i+1$ 번째 ($0 \leq i \leq M-1$) 줄에는 두 개의 정수 A_i , B_i 가 주어진다. 이는 JOI 왕국의 항공 노선의 정보를 의미한다.

출력 형식

프로그램이 정상적으로 종료되었다면, 샘플 그레이더는 다음과 같은 정보를 표준 출력에 출력한다. (따옴표는 출력하지 않는다.)

- 프로그램의 실행 중에 오답으로 판단된 경우, 오답의 종류를 “Wrong Answer [1]”과 같은 형식으로 출력하고, 실행이 종료된다.
- Alice와 Bob 모두 오답이 아닌 경우, “Accepted”를 출력하고 또한 V 의 값이 출력된다.

프로그램이 다양한 오답의 종류에 속해 있으면 샘플 그레이더는 그중 하나만 출력할 것이다.

제한

- $1 \leq N \leq 1\,000$.
- $0 \leq M \leq N(N-1)/2$.
- $0 \leq A_i \leq N-1$ ($0 \leq i \leq M-1$).
- $0 \leq B_i \leq N-1$ ($0 \leq i \leq M-1$).
- $A_i \neq B_i$ ($0 \leq i \leq M-1$).
- $(A_i, B_i) \neq (A_j, B_j)$ 이며, $(A_i, B_i) \neq (B_j, A_j)$ ($0 \leq i < j \leq M-1$).

서브태스크 1 (22 점)

- $N \leq 22$

서브태스크 2 (15 점)

- $N \leq 40$

서브태스크 3 (63 점)

추가 제한조건이 없다.

배점

- 서브태스크 1 혹은 2에서는, 각 서브태스크의 모든 테스트 케이스를 맞은 경우, 만점이다.
- 서브태스크 3에서는, 모든 테스트 케이스를 맞은 경우, $V - N$ 의 최댓값을 MaxDiff로 해서, 다음 기준으로 채점한다.
 - $101 \leq \text{MaxDiff}$ 이면, 0점.
 - $21 \leq \text{MaxDiff} \leq 100$ 이면, $13 + \left\lfloor \frac{100 - \text{MaxDiff}}{4} \right\rfloor$ 점. 여기서, x 를 넘지 않는 최대의 정수를 $\lfloor x \rfloor$ 로 표현한다.
 - $13 \leq \text{MaxDiff} \leq 20$ 이면, $33 + (20 - \text{MaxDiff}) \times 3$ 점.
 - $\text{MaxDiff} \leq 12$ 이면, 63점.

예제

이 함수는 그레이더의 예제 입력과 해당하는 함수 호출을 보여준다.

예제 입력 1	함수 호출의 예			
	호출	반환값	호출	반환값
4 3 0 1 0 2 0 3	Alice(...)			
			InitG(4,3)	
				(없음)
			MakeG(0,0,1)	
				(없음)
			MakeG(1,0,2)	
				(없음)
			MakeG(2,0,3)	
				(없음)
		(없음)		
	Bob(...)			
			InitMap(4,3)	
				(없음)
			MakeMap(0,1)	
				(없음)
			MakeMap(0,2)	
				(없음)
			MakeMap(0,3)	
				(없음)
		(없음)		

이 때, Alice(...), Bob(...)에 주어진 인자는 다음과 같다.

인자	Alice(...)	Bob(...)
N	4	
M	3	
V		4
U		3
A	{0,0,0}	
B	{1,2,3}	
C		{2,2,2}
D		{3,0,1}

예제 입력 2
5 7
0 1
0 2
1 3
1 4
3 4
2 3
2 4