문제 1. 카멜레온의 사랑

시간 제한: 2 second
메모리 제한: 512 megabytes

JOI 동물원에는 1번부터 $2N$번까지의 번호가 붙어있는 $2N$ 마리의 카멜레온이 있다. 그 중, N 마리의 카멜레온은 성별이 X이다. 다른 N 마리의 카멜레온은 성별이 Y이다.

각 카멜레온에는 **원본 색**이 있다. 카멜레온의 원본 색에 대해서는 다음과 같은 사실이 알려져 있다.

- 성별 X인 카멜레온 N 마리의 원본 색은 모두 다르다.
- 성별이 X인 각 카멜레온에 대해, 원본 색이 같은 성별이 Y인 카멜레온이 유일하게 존재한다.

지금 JOI 동물원에는 새로운 사랑이 싹트는 계절이다. 각 카멜레온은 다른 카멜레온을 **사랑**한다. 카멜레온의 사랑에 대해서는 다음과 같은 사실이 알려져 있다.

- 각 카멜레온은 성별이 자신과 다른 카멜레온을 정확히 한 마리 사랑한다.
- 어떤 카멜레온과 그 카멜레온이 사랑하는 카멜레온의 원본 색은 다르다.
- 같은 카멜레온을 사랑하는 서로 다른 두 카멜레온은 존재하지 않는다.

당신은 카멜레온 몇 마리를 모아서 미팅을 주선하려고 한다. 미팅에 참석한 카멜레온 s에 대해, s가 카멜레온 t를 사랑한다고 하자. s의 **피부색**은 다음과 같이 정해진다.

- t가 미팅에 참석하면, s의 피부색은 t의 원본 색이다.
- t가 미팅에 참석하지 않으면, s의 피부색은 s의 원본 색이다.

카멜레온의 피부색은 어떤 미팅에 참석하냐에 따라 바뀔 수 있다. 당신이 주선하는 미팅에 대해, 미팅에 참석한 카멜레온의 피부색이 총 몇 종류인지 알 수 있다.

당신은 미팅을 20 000번 이하로 주선해서 원본 색이 같은 카멜레온 쌍을 모두 알고 싶다.

카멜레온의 수가 주어질 때, 미팅을 20 000번 이하로 주선해서 원본 색이 같은 카멜레온 쌍을 모두 정하는 프로그램을 작성하여라.

구현 명세

당신은 파일 하나를 제출해야 한다.

이 파일의 이름은 `chameleon.cpp`이다. 파일은 다음 함수를 구현해야 한다. 또한, `chameleon.h`를 include 해야 한다.

- `void Solve(int N)`

  이 함수는 각 테스트 케이스마다 정확히 한 번 불린다.

  - 인자 `N`은 성별이 X인 카멜레온의 수 N을 나타낸다.

  당신의 프로그램은 다음 함수를 호출 할 수 있다.

  - `int Query(const std::vector<int> &p)`
    당신은 이 함수를 호출함으로써 미팅을 주선할 수 있다.
    * 인자 `p`는 미팅에 참여하는 카멜레온의 목록이다.
    * 이 함수의 반환값은 미팅에 참석한 카멜레온의 피부색이 몇 종류인지이다.
    * `p`에 있는 각 원소는 1 이상 $2N$ 이하의 정수여야 한다. 이를 만족하지 않은 경우에는 **오답** [1]이 된다.

  - p에 있는 각 원소는 서로 달라야 한다. 이를 만족하지 않은 경우에는 **오답** [**2**]이 된다.
  - 당신은 이 함수를 20 000번 이상 호출해서는 안 된다. 이를 만족하지 않은 경우에는 **오답** [**3**]이 된다.

- `void Answer(int a, int b)`
  이 함수를 사용하여, 같은 원본 색을 가진 카멜레온의 쌍을 답할 수 있다.
  - 인자 a와 b는 a 번째 카멜레온과 b 번째 카멜레온의 원본 색이 같다는 것을 의미한다.
  - $1 \leq a \leq 2N$과 $1 \leq b \leq 2N$을 만족해야 한다. 이를 만족하지 않을 경우 **오답** [**4**]이 된다.
  - a와 b로 주어진 수는 2번 이상 나타나서는 안 된다. 이를 만족하지 않은 경우에는 **오답** [**5**]이 된다.
  - 서로 원본 색을 가진 카멜레온을 지정할 경우 **오답** [**6**]이 된다.
  - 함수 Answer는 정확히 N번 호출될 필요가 있다. 함수 Solve의 실행 종료 시에 함수 Answer의 호출 횟수가 N번이 아닐 경우 **오답** [**7**]이 된다.

참고 사항

- 당신의 프로그램은 내부에서 사용할 목적으로 함수나 전역변수를 사용할 수 있다.
- 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.cpp이다. 당신의 프로그램을 테스트 하기 위해서, grader.cpp, chameleon.cpp, chameleon.h를 같은 디렉토리 안에 놓고, 컴파일하기 위해 다음 커맨드를 실행하여라.

- `g++ -std=gnu++14 -O2 -o grader grader.cpp chameleon.cpp`

컴파일이 성공적이면, 파일 grader가 생성된다.

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

입력 형식

샘플 그레이더는 표준 입력에서 다음과 같은 형식으로 입력받는다.

N

$Y_1 \cdots Y_{2N}$

$C_1 \cdots C_{2N}$

$L_1 \cdots L_{2N}$

Y_i는 $(1 \leq i \leq 2N)$ i 번째 카멜레온의 성별을 나타내고, 0 혹은 1이다. 0이면 성별이 X이고, 1이면 성별이 Y이다.

C_i는 $(1 \leq i \leq 2N)$ i 번째 카멜레온의 원본 색을 나타내고, 1 이상 N 이하의 정수이다.

L_i는 $(1 \leq i \leq 2N)$ i 번째 카멜레온이 사랑하는 카멜레온의 번호이다.

출력 형식

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

- 정답으로 판단된 경우, Query함수의 호출 횟수를 "`Accepted: 100`"과 같은 형식으로 출력한다.

- 오답으로 판단된 경우, 오답의 종류를 "`Wrong Answer [1]`"과 같은 형식으로 출력한다.

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

제한

모든 입력 데이터는 다음의 조건을 만족한다. Y, C, L의 의미는 입력 형식을 참고하여라.

- $2 \leq N \leq 500$.
- $0 \leq Y_i \leq 1$ $(1 \leq i \leq 2N)$.
- 각 j $(1 \leq j \leq N)$ 에 대해, $Y_i = 0$과 $C_i = j$를 만족하는 유일한 i가 $(1 \leq i \leq 2N)$ 존재한다.
- 각 j $(1 \leq j \leq N)$ 에 대해, $Y_i = 1$과 $C_i = j$를 만족하는 유일한 i가 $(1 \leq i \leq 2N)$ 존재한다.
- $1 \leq L_i \leq 2N$ $(1 \leq i \leq 2N)$.
- $Y_i \leq Y_{L_i}$ $(1 \leq i \leq 2N)$.
- $C_i \leq C_{L_i}$ $(1 \leq i \leq 2N)$.
- $L_k \neq L_l$ $(1 \leq k < l \leq 2N)$.

서브태스크 1 (4 점)

- $L_{L_i} = i$. $(1 \leq i \leq 2N)$.

서브태스크 2 (20 점)

- $N \leq 7$.

서브태스크 3 (20 점)

- $N \leq 50$.

서브태스크 4 (20 점)

- $Y_i = 0$ $(1 \leq i \leq N)$.

서브태스크 5 (36 점)

추가 제한조건이 없다.

예제

이 함수는 그레이더의 예제 입력과 해당하는 함수 호출을 보여준다.

예제 입력	예제 함수 호출		
	호출	호출	반환값
4	Solve(4)		
1 0 1 0 0 1 1 0		Query([])	0
4 4 1 2 1 2 3 3		Query([6, 2])	2
4 3 8 7 6 5 2 1		Query([8, 1, 6])	2
		Query([7, 1, 3, 5, 6, 8])	4
		Query([8, 6, 4, 1, 5])	3
		Answer(6, 4)	
		Answer(7, 8)	
		Answer(2, 1)	
		Answer(3, 5)	

대회 홈페이지의 아카이브에서 받을 수 있는 파일 중, sample-02.txt는 서브태스크 1의 조건을, sample-03.txt는 서브태스크 4의 조건을 만족한다.

문제 2. 조이터에서 친구를 만드는건 재밌어

입력 파일:	`standard input`
출력 파일:	`standard output`
시간 제한:	3 seconds
메모리 제한:	1024 megabytes

조이터는 당신의 추억을 친구와 남길 수 있는 소셜 미디어 서비스이다.

조이터에서 당신은 다른 사용자를 **팔로우**할 수 있다. 사용자 a가 사용자 b를 팔로우하면, 사용자 a는 사용자 b의 게시글을 자신의 타임라인에서 읽을 수 있다. 이 경우에 사용자 b가 사용자 a를 팔로우할 수도, 아닐 수도 있다. 하지만, 사용자 a가 특정 사용자 b를 한 번 넘게 팔로우할 수는 없다.

1번, 2번, $\cdots$, N번 사용자 총 N명의 사용자가 조이터를 시작했다. 처음에는 아무도 서로를 팔로우하고 있지 않다.

이제부터 총 M일 동안 i 번째 $(1 \leq i \leq M)$ 날에 A_i번 사용자가 B_i번 사용자를 팔로우한다.

조이터는 **교류 이벤트**를 M일 동안 열려고 한다. 교류 이벤트는 다음과 같은 방법으로 이루어진다.

1. 사용자 한 명을 고른다. 이 사용자를 x라고 하자.
2. 사용자 x가 현재 팔로우하고 있는 사용자 중 한 명을 고른다. 이 사용자를 y라고 하자.
3. 다음 조건을 만족하도록 사용자 z를 고른다: z는 x와 다르며, x가 z를 팔로우하고 있지 않고, y가 z를 팔로우하고, z가 y를 팔로우한다.
4. x가 z를 팔로우한다.
5. 조건을 만족하는 (x, y, z)가 없을 때까지 위 과정을 반복한다.

교류 이벤트가 언제 진행될지는 결정되지 않았다. 그래서 각 M 개의 날에 대해 어떤 사용자가 다른 사용자를 팔로우한 이후 교류 이벤트가 개최되었을 때, 교류 이벤트가 종료된 시점에 N명이 팔로우하고 있는 사람 수의 총합의 최댓값을 구하고 싶다. 단, 교류 이벤트는 시작한 날에 종료되는 것으로 한다.

사용자의 수와 M일간의 팔로우 정보가 주어졌을 때, 각 일에 교류 이벤트가 열린 직후에 N명이 팔로우하는 사람 수의 총합의 최댓값을 구하는 프로그램을 작성하여라.

입력 형식

표준 입력에서 다음과 같은 형식으로 주어진다. 모든 값은 정수이다.

N M

A_1 B_1

$\vdots$

A_M B_M

출력 형식

M 개의 줄을 표준 출력으로 출력하여라. i 번째 $(1 \leq i \leq M)$ 줄에는, i 번째에 어떤 사용자가 다른 사용자를 팔로우한 이후 교류 이벤트가 개최되었을 때, 교류 이벤트가 종료된 시점에 N명이 팔로우하는 사람 수의 총합의 최댓값을 출력하여라.

제한

- $2 \leq N \leq 100\ 000$.
- $1 \leq M \leq 300\ 000$.
- $1 \leq A_i \leq N$ $(1 \leq i \leq M)$.

- $1 \leq B_i \leq N$ $(1 \leq i \leq M)$.
- $A_i \neq B_i$ $(1 \leq i \leq M)$.
- $(A_i, B_i) \neq (A_j, B_j)$ $(1 \leq i < j \leq M)$.

서브태스크 1 (1 점)

- $N \leq 50$.

서브태스크 2 (16 점)

- $N \leq 2\ 000$.

서브태스크 3 (83 점)

추가 제한조건이 없다.

예제

standard input	standard output
4 6 1 2 2 3 3 2 1 3 3 4 4 3	1 2 4 4 5 9

- 첫째 날에 1번 사용자가 2번 사용자를 팔로우한다. 이후 교류 이벤트를 개최해도, 새롭게 누군가가 다른 누군가를 팔로우하는 일이 없이, 팔로우 수 합계는 1이다.
- 둘째 날에 2번 사용자가 3번 사용자를 팔로우한다. 이후 교류 이벤트를 개최해도, 새롭게 누군가가 다른 누군가를 팔로우하는 일이 없이, 팔로우 수 합계는 2이다.
- 셋째 날에 3번 사용자가 2번 사용자를 팔로우한다. 이후 교류 이벤트를 개최 하면, 예를 들면 1번 사용자가 3번 사용자를 팔로우 할 수 있다. 이때 팔로우 수 합계는 4가 되고, 이 값이 최댓값이다.
- 넷째 날에 1번 사용자가 3번 사용자를 팔로우한다. 이후 교류 이벤트를 개최해도, 새롭게 누군가가 다른 누군가를 팔로우하는 일이 없이, 팔로우 수 합계는 4이다.
- 다섯째 날에 3번 사용자가 4번 사용자를 팔로우한다. 이후 교류 이벤트를 개최해도, 새롭게 누군가가 다른 누군가를 팔로우하는 일이 없이, 팔로우 수 합계는 5이다.
- 여섯째 날에 4번 사용자가 3번 사용자를 팔로우한다. 이후 교류 이벤트를 개최 하면, 예를 들면 1번 사용자가 3번 사용자를 팔로우하고, 2번 사용자가 4번 사용자를 팔로우하고, 4번 사용자가 2번 사용자를 팔로우한다. 이때 팔로우 수 합계는 9가 되고, 이 값이 최댓값이다.

standard input	standard output
6 10 1 2 2 3 3 4 4 5 5 6 6 5 5 4 4 3 3 2 2 1	1 2 3 4 5 7 11 17 25 30

문제 3. 유적 3

입력 파일:	`standard input`
출력 파일:	`standard output`
시간 제한:	4 seconds
메모리 제한:	512 megabytes

JOI 교수는 IOI 왕국 역사 연구의 일인자이다. IOI 왕국에 관련된 신사를 조사하는 중 신사에 건설된 N 개의 기둥을 발견했다. 또한, IOI 왕국의 고대인들이 쓴 것으로 추정되는 고문서를 발견했다. 이 문서에는 기둥에 대한 정보가 쓰여 있었다. 문서에 쓰인 내용은 다음과 같다.

- 기둥 건설 직후에 $2N$개의 기둥이 있었고, 1번부터 $2N$번까지의 번호가 붙어있다.
- 기둥 건설 직후에 각 k에 $(1 \leq k \leq N)$ 대해, 높이가 k인 기둥이 정확히 두 개 있었다.
- 지진이 N번 일어났다. 지진 이후에 몇몇 기둥은 무너졌고 높이가 1 감소했다. 다른 기둥들은 고대인들이 보호했고 기둥이 무너지지 않았기 때문에 높이도 변하지 않았다.
- 지진이 일어났을 때 각 k에 $(1 \leq k \leq N)$ 대해, 높이가 k인 기둥이 정확히 하나가 보호되었다. 만약에 높이가 k인 기둥이 두 개 이상 있었을 경우, 가장 번호가 높은 기둥이 보호되었다. 다른 말로, 지진 전의 i번째 $(1 \leq i \leq 2N)$ 기둥의 높이를 h_i라 할 때, $h_i \geq 1$이고, 모든 $j > i$에 대해 $h_j \neq h_i$를 만족한 경우 i 번째 기둥이 보호되었다.
- N번의 지진이 일어난 이후 N개의 기둥이 남았다. (즉 기둥의 높이가 1 이상인 기둥이 정확히 N개 있다.)

$2N$개 기둥의 높이를 복원할 수 있다면 세기의 대발견이 될 수 있다고 생각한 JOI 교수는 기둥을 좀 더 면밀히 조사했다. 그는 N번의 지진이 일어난 이후로 남은 기둥의 번호가 $A_1, A_2, \cdots, A_N$번이라는 것을 알아냈다.

JOI 교수는 기둥이 건설되었을 때 $2N$개 기둥의 높이로 가능한 경우가 몇 가지인지를 알고 싶다. 당신은 JOI 교수의 제자로 경우의 수를 계산하는 프로그램을 작성하는 요구를 받았다.

N번의 지진 이후에 남은 기둥의 번호가 주어졌을 때, $2N$개의 기둥의 높이로 가능한 경우의 수를 1 000 000 007로 나눈 나머지를 구하여라.

입력 형식

표준 입력에서 다음과 같은 형식으로 주어진다. 모든 수는 정수이다.

N

$A_1 \cdots A_N$

출력 형식

표준 출력에 하나의 줄을 출력하여라. 이 줄은 답을 1 000 000 007로 나눈 나머지이다.

제한

- $1 \leq N \leq 600$.
- $1 \leq A_i \leq 2N$ $(1 \leq i \leq N)$.
- $A_i < A_{i+1}$ $(1 \leq i \leq N-1)$.

서브태스크 1 (6 점)

- $N \leq 13$.

서브태스크 2 (52 점)

- $N \leq 60$.

서브태스크 3 (42 점)

추가 제한조건이 없다.

예제

standard input	standard output
3 3 4 6	5

이 예제에서, 예를 들어 기둥의 높이가 (2, 2, 3, 3, 1, 1)이었다고 하자. 각 k에 $(1 \leq k \leq 3)$ 대해 높이가 k인 기둥이 정확히 두 개 존재하므로 고문서에 쓰인 것과 일치한다.

- 첫 번째 지진 이후 2, 4, 6번 기둥이 고대인들에 의해 보호되었다. 높이는 (1, 2, 2, 3, 0, 1)이 되었다.
- 두 번째 지진 이후 3, 4, 6번 기둥이 고대인들에 의해 보호되었다. 높이는 (0, 1, 2, 3, 0, 1)이 되었다.
- 세 번째 지진 이후 3, 4, 6번 기둥이 고대인들에 의해 보호되었다. 높이는 (0, 0, 2, 3, 0, 1)이 되었다.

세 지진 이후 3, 4, 6번 기둥이 남았고 입력에 주어진 정보와 같다. 또한, 기둥의 높이가 (2, 3, 2, 3, 1, 1), (2, 3, 3, 2, 1, 1), (3, 2, 2, 3, 1, 1), (3, 2, 3, 2, 1, 1)인 경우도 가능하다.

그러므로, 총 5가지의 기둥 높이가 고문서에 쓰인 것과 입력으로 주어진 정보와 일치한다.

standard input	standard output
1 1	0

이 예제에서, (1, 1)이 고문서와 일치하는 유일한 높이이다. 첫 번째 지진 이후 기둥의 높이는 (0, 1)이 된다.

그러므로 고문서에 쓰인 것과 입력을 모두 만족 시키는 높이는 존재하지 않는다.

standard input	standard output
10 5 8 9 13 15 16 17 18 19 20	147003663

건설되었을 때 가능한 $2N$개의 기둥 높이로 111 147 004 440가지가 가능하다. 111 147 004 440을 1 000 000 007로 나눈 나머지는 147 003 663이다. 그러므로 147 003 663을 출력한다.