문제 1. 케이크 3

입력 파일:	`standard input`
출력 파일:	`standard output`
시간 제한:	4초
메모리 제한:	256MB

오늘은 IOI양의 생일이다. 이날을 위해 JOI군은 생일 케이크를 예약했다. 원형 케이크 하나를 통째로 예약할 생각이었지만 착오가 있어서 N 조각의 케이크를 예약해 버렸다. 각 조각에는 1번부터 N번까지 번호가 붙어 있고, i 번 $(1 \leq i \leq N)$ 조각의 가치는 V_i이며 색의 짙음은 C_i이다.

JOI군은 서로 다른 M개의 케이크를 골라 원하는 순서대로 배열해 합쳐서 원형 케이크를 만들기로 결심했다. 케이크 조각들이 k_1번, $\cdots$, k_M번 조각의 순서로 나열되어 있을 때, 이 케이크의 아름다움은

$$\sum_{j=1}^{M} V_{k_j} - \sum_{j=1}^{M} |C_{k_j} - C_{k_{j+1}}|$$

으로 정의된다. (단, $k_{M+1} = k_1$이다.) 즉, 아름다움은 사용된 케이크 조각의 가치의 합으로 부터 인접한 두 케이크의 색의 짙음에 차의 절댓값의 합계로 정의된다. JOI군은 되도록 원형 케이크의 아름다움을 최대로 하고 싶다.

케이크 조각의 개수, 각 케이크 조각의 가치와 색의 짙음, 원형 케이크를 만들기 위해 필요한 조각의 개수가 주어졌을 때, JOI군이 만들 수 있는 원형 케이크의 아름다움의 최댓값을 구하는 프로그램을 작성하여라.

입력 형식

표준 입력에서 다음과 같은 형식으로 주어진다. 모든 값은 정수이다.

N M

V_1 C_1

$\vdots$

V_N C_N

출력 형식

표준 출력으로 한 개의 줄에 하나의 수를 출력하여라. 이는 JOI군이 만들 수 있는 원형 케이크의 아름다움의 최댓값이다.

제한

- $3 \leq N \leq 200\ 000$.
- $3 \leq M \leq N$.
- $1 \leq V_i \leq 1\ 000\ 000\ 000$ $(1 \leq i \leq N)$.
- $1 \leq C_i \leq 1\ 000\ 000\ 000$ $(1 \leq i \leq N)$.

서브태스크 1 (5 점)

- $N \leq 100$.

서브태스크 2 (19 점)

- $N \leq 2\ 000$.

서브태스크 3 (76 점)

추가 제한조건이 없다.

예제

standard input	standard output
5 3 2 1 4 2 6 4 8 8 10 16	6

JOI군이 1번, 3번, 2번 조각을 골라서 순서대로 붙이면, 조각의 가치의 합은 $2 + 6 + 4 = 12$이고, 케이크의 짙음의 차이 합은 $|1 - 4| + |4 - 2| + |2 - 1| = 6$이다. 그래서 원형 케이크의 아름다움은 $12 - 6 = 6$이다.

또한, 2번, 3번, 4번 조각을 골라서 순서대로 붙여서 아름다움이 6인 원형 케이크를 만들 수도 있다.

더 아름다움이 큰 원형 케이크를 만들 수는 없기 때문에, 6을 출력해야 한다.

standard input	standard output
8 4 112103441 501365808 659752417 137957977 86280801 257419447 902409188 565237611 965602301 689654312 104535476 646977261 945132881 114821749 198700181 915994879	2323231661

문제 2. 합병

입력 파일:	`standard input`
출력 파일:	`standard output`
시간 제한:	3초
메모리 제한:	256MB

JOI 합중국에는 N개의 도시가 있어서, 1번부터 N번까지의 번호가 붙어있다. 또한, JOI 합중국에는 $N-1$개의 국도가 있다. i 번째 $(1 \leq i \leq N-1)$ 국도는 A_i번 도시와 B_i번 도시를 양방향으로 잇는다. 어떤 두 도시에 대해서도, 국도 몇 개를 이용하면 서로 오가는 것이 가능하다.

현재 JOI 합중국은 1번부터 K번까지의 번호가 붙어있는 K개의 주로 나뉘어 있다. j번 $(1 \leq j \leq N)$도시는 S_j번 주에 속한다. 모든 주에는 적어도 하나의 도시가 속해 있다.

JOI 합중국의 대통령인 K 이사장은 이 나라가 분열하지 않을까 걱정이 되었다. 다음 조건을 모두 만족하도록 모든 도시를 2개의 그룹 X, Y로 나누는 것이 가능할 때, JOI 합중국은 **분열 가능**한 상태라고 말한다.

- 모든 도시는 그룹 X 혹은 그룹 Y에 속한다.
- 그룹 X에는 적어도 하나의 도시가 속해 있다.
- 그룹 Y에는 적어도 하나의 도시가 속해 있다.
- 모든 주에 대해서, 그 주에 있는 모든 도시는 모두 같은 그룹에 속해 있다.
- 그룹 X에 속한 어떤 두 도시에 대해서도, 그룹 X에 속한 도시만을 경유해서 서로 오가는 것이 가능하다.
- 그룹 Y에 속한 어떤 두 도시에 대해서도, 그룹 X에 속한 도시만을 경유해서 서로 오가는 것이 가능하다.

K 이사장은 JOI 합중국이 분열 가능하지 않은 상태를 만들기 위해 주를 합병하려고 한다. 한 번의 합병은 두 개의 주를 골라서 합치는 것을 말한다. 새로운 주는 기존의 두 주에 속해 있던 도시가 속해있다. K 이사장은 주를 최소한의 횟수만큼 합병하여 JOI 합중국을 분열 가능하지 않은 상태로 만들고 싶어 한다.

도시와 국도의 위치, 현재 어떤 도시가 어떤 주에 속해있는가에 대한 상태가 주어졌을 때, JOI 합중국이 분열 가능하지 않은 상태로 만들기 위한 합병의 최소 횟수를 구하는 프로그램을 작성하여라.

입력 형식

표준 입력에서 다음과 같은 형식으로 주어진다. 모든 값은 정수이다.

N K

A_1 B_1

$\vdots$

A_{N-1} B_{N-1}

S_1

$\vdots$

S_N

출력 형식

JOI 합중국을 분열 가능하지 않은 상태로 만들기 위한 합병의 최소횟수를 표준 출력의 첫째 줄에 출력하여라.

제한

- $1 \leq N \leq 500\ 000$.
- $1 \leq K \leq N$.

- $1 \leq A_i \leq N$ $(1 \leq i \leq N-1)$.
- $1 \leq B_i \leq N$ $(1 \leq i \leq N-1)$.
- 어떤 두 도시에 대해서도, 국도 몇 개를 이용하면 서로 오가는 것이 가능하다.
- $1 \leq S_j \leq K$ $(1 \leq j \leq N)$
- 모든 k $(1 \leq k \leq N)$에 대해, $S_j = k$를 만족하는 j $(1 \leq j \leq N)$가 존재한다.

서브태스크 1 (10 점)

- $N \leq 100$.
- $K \leq 7$.

서브태스크 2 (24 점)

- $N \leq 3\ 000$.

서브태스크 3 (14 점)

- $N \leq 100\ 000$.
- $K \leq 50$.

서브태스크 4 (22 점)

- $N \leq 100\ 000$.
- 처음 상황에서, 모든 주에 대해, 그 주에 속한 어떤 두 도시에 대해서도, 국도 100개 이하를 이용하면 서로 오가는 것이 가능하다.

서브태스크 5 (30 점)

추가 제한조건이 없다.

예제

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

이 예제에서는, 처음의 상태는 분열 가능하다. 예를 들면, 1번, 2번, 3번, 4번 도시를 그룹 X라 하고, 5번 도시를 그룹 Y라고 하면 된다.

3번 도시와 4번 도시를 합병할 경우 분열 가능하지 않은 상태가 된다. 그러므로 답은 1이다.

standard input	standard output
5 4 1 2 2 3 3 4 4 5 1 2 3 4 1	0

이 예제에서는, 처음의 상태는 분열 가능하지 않다. 그러므로 답은 0이다.

standard input	standard output
2 2 1 2 1 2	1

문제 3. 광물

시간 제한: 1 second
메모리 제한: 256 megabytes

JOI교수의 연구실에서는 N 종류의 광물을 연구하고 있다. 연구실에는 각 종류의 광물이 두 조각씩 있다. 조각은 총 $2N$ 개가 존재하고, 각 조각에는 1번부터 $2N$번까지의 번호가 붙어있다.

하지만 어느 날, 조수 비타로는 $2N$ 개의 광물을 포함한 박스를 떨어뜨려 어떤 조각과 어떤 조각이 같은지 모르게 되어 버렸다.

연구실은 0 개 이상 $2N$ 개 이하의 조각을 넣으면, 광물이 흡수하는 빛의 파장을 측정하는 것으로, 기계 안에 몇가지 종류의 광물이 들어있는가를 판단하는 장치가 있다. 비타로는 이 장치를 사용해서 $2N$ 개의 조각에서 같은 종류의 광물 짝 N 쌍을 알고싶어 한다. 비타로는 처음에 기계에 광물을 넣지 않은 상태로부터 시작해서 다음 동작 중 하나를 반복한다.

- 기계에 새로운 조각을 하나 집어넣어서, 기계 안에 몇 가지 종류의 광물이 있는지를 알아낸다.
- 기계에서 새로운 조각을 하나 빼서, 기계 안에 몇 가지 종류의 광물이 있는지를 알아낸다.

시간이 너무 오래 걸리면 JOI교수에게 들켜버리기 때문에, 비타로는 기계를 최대 1 000 000번만 사용할 수 있다.

광물의 종류가 주어졌을 때, 기계를 사용하여 모든 쌍의 광물을 알아내는 프로그램을 작성하여라.

구현 명세

당신은 파일 하나를 제출해야 한다.

이 파일의 이름은 `minerals.cpp`이다. 파일은 다음 함수를 구현해야 한다. 또한, `minerals.h`를 include해야 한다.

- `void Solve(int N)`

  이 함수는 각 테스트 케이스마다 정확히 한 번 불린다.

  - 인자 `N`은 광물의 수 N을 의미한다.

  당신의 프로그램은 다음 함수를 호출 할 수 있다.

  - `int Query(int x)`

    이 함수는 지정된 조각의 번호에 대해서 이 조각이 이미 기계 안에 들어 있으면 기계에서 빼고, 그렇지 않으면 이 조각을 기계에 넣는다.

    * 당신은 조각의 번호 x를 인자 `x`를 사용해서 나타내어야 한다. 이 번호는 $1 \leq x \leq 2N$을 만족해야 한다. 아닌 경우에는 **오답** [**1**]이 된다.
    * 당신은 이 함수를 1 000 000번 이상 호출해서는 안된다. 호출 한 경우에는 **오답** [**2**]이 된다.

  - `void Answer(int a, int b)`

    이 함수를 사용하여, 같은 종류의 광물의 쌍을 답할 수 있다.

    * 인자 `a`와 `b`는 a 번째 광물과 b 번째 광물이 같은 종류라는 것을 의미한다. 이 수는 $1 \leq a \leq 2N$과 $1 \leq b \leq 2N$을 만족해야 한다. 이것을 만족하지 않을 경우 **오답** [**3**]이 된다. 만약 a와 b로 주어진 수가 2번 이상 나타날 경우 **오답** [**4**]이 된다. 서로 다른 종류의 광물을 지정할 경우 **오답** [**5**]이 된다.

    함수 Answer는 정확히 N 번 호출될 필요가 있다. 함수 `solve`의 실행 종료 시에 함수 `Answer`의 호출 횟수가 N 번이 아닐 경우 **오답** [**6**]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.cpp`이다. 당신의 프로그램을 테스트하기 위해서, `grader.cpp`, `minerals.cpp`, `minerals.h`를 같은 디렉토리 안에 놓고, 컴파일하기 위해 다음 커맨드를 실행하여라.

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

컴파일이 성공적이면, 파일 `grader`가 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

X_1 Y_1

$\ldots$

X_N Y_N

X_i와 Y_i $(0 \leq i \leq N-2)$는 X_i번 조각과 Y_i번 조각이 같은 종류의 광물임을 의미한다.

출력 형식

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

- 정답으로 판단된 경우, `Query` 함수의 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. 샘플 그레이더의 X_i와 Y_i의 정의에 따라서

- $1 \leq N \leq 43\,000$.
- $1 \leq X_i \leq 2N$ $(1 \leq i \leq N)$.
- $1 \leq Y_i \leq 2N$ $(1 \leq i \leq N)$.
- $X_i \neq X_j$ $(1 \leq i < j \leq N)$.
- $Y_i \neq Y_j$ $(1 \leq i < j \leq N)$.
- $X_i \neq Y_j$ $(1 \leq i \leq N, 1 \leq j \leq N)$.

서브태스크 1 (6 점)

- $N \leq 100$.

서브태스크 2 (25 점)

- $N \leq 15\ 000$.
- $1 \leq X_i \leq N$ $(1 \leq i \leq N)$.
- $N + 1 \leq Y_i \leq 2N$ $(1 \leq i \leq N)$.

서브태스크 3 (9 점)

- $N \leq 15\ 000$

서브태스크 4 (30 점)

- $N \leq 38\ 000$

서브태스크 5 (5 점)

- $N \leq 39\ 000$

서브태스크 6 (5 점)

- $N \leq 40\ 000$

서브태스크 7 (5 점)

- $N \leq 41\ 000$

서브태스크 8 (5 점)

- $N \leq 42\ 000$

서브태스크 9 (10 점)

추가 제한조건이 없다.

예제

이 함수는 그레이더의 예제 입력과 해당하는 함수 호출을 보여준다.

예제 입력	예제 함수 호출		
	호출	호출	반환값
4 1 2 2 6 3 4 7 8	`Solve(4)`		
		`Query(1)`	1
		`Query(2)`	2
		`Query(5)`	2
		`Query(2)`	1
		`Answer(3, 4)`	(없음)
		`Answer(5, 1)`	(없음)
		`Answer(8, 7)`	(없음)
		`Answer(2, 6)`	(없음)