Problem A. Dangerous Skating

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	256 megabytes

JOI군은 대자연의 광대한 스케이트링크에서 아이스스케이트를 타는 취미가 있다.

스케이트링크는 남북으로 R칸, 동서로 C칸으로 이루어진 직사각형으로 표현된다. 북쪽에서 r번째, 서쪽에서 c번째의 칸을 (r, c)라고 표시하자. 각각의 격자는 JOI군이 통과 할 수 있지만, 얼음덩어리가 있어 통과 할 수 없는 칸이 몇 개 있다. 또한, 스케이트 링크의 외벽의 격자에는 모두 얼음덩어리가 있어, 링크 밖으로 나가는것도 불가능 하다. 즉, 격자 $(i, 1), (i, C) (1 \leq i \leq R)$ 과, 격자 $(1, j), (R, j) (1 \leq j \leq C)$에는 얼음덩어리가 있다.

JOI군은 스케이트를 잘 타지 못한다. JOI군이 스케이트 링크 위에서 이동할 때는, 동서남북중 한 방향을 정해, 현재 JOI군이 있는 격자에서 미끄러져, 얼음 덩어리에 부딪히기 바로 전 격자까지 달려가 멈춘다. 링크에서 미끄러져 멈출때 까지를 1번의 이동으로 센다. 이동하고자 하는 칸에 얼음덩어리가 있을 때는, 그 방향으로 이동할 수 없다.

여느날, JOI군이 스케이트를 즐기고 있는 동안, JOI군이 링크에서 이동하면, 그 자리에 얼음덩어리가 생기는 것을 깨달았다. 미끄러진 지점 이외에 얼음 덩어리가 생기는 일은 없다. 이 상황에서 스케이트를 계속 타는것은 위험하므로, JOI군은 최대한 빨리 스케이트 링크를 탈출하고 싶다.

JOI군은, 현재 격자 (r_1, c_1)에 있다. 이 스케이트 링크에서 탈출하기 위해서는 출구의 격자 (r_2, c_2)에서 멈출 필요가 있다. JOI군이 안전하게 스케이트링크에서 탈출 할 수 있도록, 현재 지점에서 이동을 시작해 출구에서 멈추기 위해서는, 적어도 몇번의 이동이 필요한가 계산하는 프로그램을 작성했으면 한다. 스케이트 링크의 상태는, JOI군이 현재 지점에서 출구 격자에 멈추는 것이 불가능 할 수도 있다. JOI군의 이동 중에 출구 격자를 지나친것 만으로는 스케이트링크를 탈출 할 수 없다는 것에 주의해라.

스케이트링크 위의 얼음덩어리의 정보와, JOI군의 현재 위치, 출구 격자의 위치가 주어질 때, JOI군이 현재 격자에서 이동을 시작해서, 출구 격자에 멈추는 것이 가능한지 판단하고, 가능 할 경우, 필요한 이동의 최소횟수를 구하는 프로그램을 작성하여라.

Input

표준 입력으로, 다음의 데이터가 들어온다.

- 첫째 줄에는, 정수 R, C가 공백으로 구분되어 들어온다. 이는, 스케이트 링크의 크기가 남북으로 R칸, 동서로 C칸임을 의미한다.
- 다음 R개의 줄은 각각, C문자로 된 문자열이 들어온다. 각 문자는 '`.`' 혹은 '`#`'이다. 이 R행 중 r번째 $(1 \leq r \leq R)$에서, 왼쪽에서 c번째 문자 $(1 \leq c \leq C)$는, 스케이트 링크의 격자 (r, c)의 초기상태를 의미한다. 이 문자가 '`.`'인 경우에는, 그 격자를 지나갈 수 있다는 것을 의미하고, 이 문자가 '`#`'인 경우에는 그렇지 않다는 것을 의미한다.
- 다음 1개의 줄에는 정수 r_1, c_1이 공백으로 구분되어 들어온다. 이것은, JOI군이 현재 위치가 격자 (r_1, c_1)이라는 것을 의미한다.
- 다음 1개의 줄에는 정수 r_2, c_2가 공백으로 구분되어 들어온다. 이것은, 스케이트링크의 출구가 격자 (r_2, c_2)에 위치함을 의미한다.

Output

표준출력에, JOI군이 현재 위치에서 시작해서, 출구의 격자에 멈추기 위해 필요한 이동횟수의 최솟값을 의미하는 정수를 첫째 줄에 출력하여라. 단, JOI군이 어떻게 해도 출구의 격자에 멈출 수 없을 경우, `-1`을 출력하여라.

Constraints

모든 입력데이터는 다음의 조건을 만족한다.

- $3 \leq R \leq 1\ 000$
- $3 \leq C \leq 1\ 000$
- $1 \leq r_1 \leq R$
- $1 \leq c_1 \leq C$
- $1 \leq r_2 \leq R$
- $1 \leq c_2 \leq C$
- 스케이트링크의 외벽의 칸은 모두 얼음이 있다. 즉, 격자 $(i, 1), (i, C)(1 \leq i \leq R)$과, 격자 $(1, j), (R, j)(1 \leq j \leq C)$에는 얼음덩어리가 있다.
- 격자 (r_1, c_1)과 격자 (r_2, c_2)에는 얼음덩어리가 없다.

Subtask 1 (13 points)

다음의 조건을 만족한다.

- $R \leq 10$
- $C \leq 10$
- JOI군이 현재 위치에서 시작해, 출구가 있는 격자로 나갈 수 있는 경우에는, 필요한 이동의 횟수가 10 번 이내이다.

Subtask 2 (65 points)

다음의 조건을 만족한다.

- $R \leq 200$
- $C \leq 200$

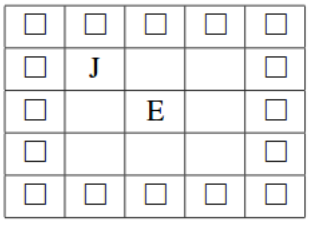
Subtask 3 (22 points)

추가 제한조건이 없다.

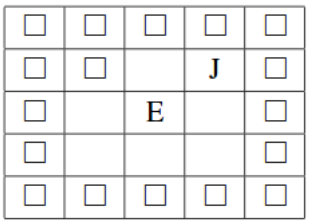
Examples

standard input	standard output
5 5 ##### #...# #...# #...# ##### 2 2 3 3	4

입력 예제 1에서는, 스케이트 링크의 초기상태가 다음과 같다. 흰색 사각형이 적힌 격자는 얼음덩어리를, J가 적힌 격자는 JOI군의 현재 위치를, E가 적힌 격자는 출구를 의미한다.



일단, JOI군이 동쪽 방향으로 이동한 후에, 스케이트링크의 상태는 다음과 같이 된다.



그 후 JOI군이 서쪽으로, 남쪽으로, 북쪽으로 차례로 이동한 경우, 총 4계의 이동으로 출구에서 멈추는 것이 가능하다. 3번 이하로 이동해서 출구에서 멈추는 것이 불가능 하므로, 4를 출력한다.

standard input	standard output
8 6 ###### #..#.# ##...# #....# #.#..# #....# ##...# ###### 4 3 6 4	5
5 5 ##### #.#.# #.#.# #.#.# ##### 2 2 4 4	4
3 3 ### #.# ### 2 2 2 2	0

입력 예제 4에서는, JOI군이 현재 위치가 출구이므로, 필요한 이동 횟수는 0이다.

This page is intentionally left blank

Problem B. Snowy Roads

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

러시아의 많은 지점은 눈이 쌓여있다. 러시아에는 N개의 도시가 있고, 0부터 $N-1$까지의 번호가 붙어있다. 러시아에는 $N-1$개의 도로가 있고, 0부터 $N-2$까지의 번호가 붙어있다. 도로 i $(0 \leq i \leq N-2)$는, 2개의 서로 다른 도시 $A_i, B_i (0 \leq A_i < B_i \leq N-1)$를 양방향으로 잇고 있다. 러시아의 어떤 2개의 다른 도시도, 몇개의 도로를 경유하면 이동하는 것이 가능하다.

각 도로의 강설 상황은 매일 바뀐다. 어떤 날짜에 대해서도, 도로의 강설 상황은 눈이 내리거나, 눈이 내리지 않는것 중 하나이다. 하루 안에 강설상황이 변하는 일은 없다.

Anya와 Boris는 러시아의 통신국에서 일하고있다. Anya는 도로정보 관리부서, Boris는 시민에서 질문을 대답하는 부서에 소속되어 있다. 시민이 하는 질문은, 러시아의 수도인 도로 0부터 다른 어떤 도시까지 이동하기 위해서는, 눈이 내리는 도로를 최소 몇개 거쳐야 하는가 이다. Boris는 보통 질문을 받은 후, Anya와 상호작용을 하여 시민에게 답변 한다.

여기서는, 러시아에서 Q일간 프로그래밍 컨테스트 세계대회가 개최되었다. 대회기간 중에, 통신회선이 혼잡해서, Anya와 Boris가 직접 상호작용하는 것이 어려울 것이라 예상된다. 그래서, Anya와 Boris는 다음의 방법으로 상호작용을 하는것으로 했다.

- Anya는, 하루의 시작에 그 날의 강설정보를 모아, 중간 서버에 데이터를 전송한다.
- Boris는, 시민으로부터 질문을 받아, 중간 서버와 상호작용 하는 것으로 답을 한다.

단, Anya 와 Boris가 상호작용을 하는데에는 제약이 있다.

- 중간 서버의 용량은 $L = 1000$비트이다. Anya는 중간 서버에 최대 L비트의 정보만 저장 할 수 있다.
- 중간 서버에 저장된 데이터는, 하루의 시작에 모두 0으로 초기화 된다.
- Boris는 중간서버와 1번 주고받기를 할 때, 지정한 1비트의 정보를 읽는 것이 가능하다.
- 질문에 대답하기 위해 Boris가 중간 서버와 상호작용 할 수 있는 것은, 각 질문에 대회 최대 20번 까지이다.

통신국장과 아는 사이인 당신은, Anya와 Boris에게 전략을 알려주게 되었다.

시민에게서 받은 질문을 바르게 대답하도록 Anya와 Boris의 전략을 구현한 프로그램을 작성하여라.

Interaction Protocol

당신은, 같은 프로그래밍 언어로 2개의 파일을 제출해야 한다.

첫번째 파일은 `Anya.c`혹은 `Anya.cpp`라는 이름이다. 이 파일은 Anya의 전략이 구현된 파일이고, 다음의 2개의 함수가 구현되어 있어야한다. 프로그램은 `Anyalib.h`를 `include`해야 한다.

- `void InitAnya(int N, int A[], int B[])`

  이 함수는, 각 테스트케이스에 대해 한 번만 호출된다.

  - 인자 `N`은 도시의 갯수를 의미한다.
  - 인자 `A[]`와 `B[]`는 각각의 길이 $N-1$인 배열이고, 도로의 연결정보를 의미한다. `A[i]`와 `B[i]` $(0 \leq i \leq N-2)$는 도로 i가 도시 `A[i]`와 `B[i]`를 양방향으로 잇는 것을 의미하는 정수이고, $0 \leq$ `A[i]` ¡ `B[i]` $\leq N-1$을 만족한다.

- `void Anya(int N, int C[])`

  이 함수는, `InitAnya`가 호출 된 후, Q번 호출된다. 이 함수는, 하루의 시작에 도로의 강설 상황을 갱신한 후, Anya가 중간 서버에 저장할 비트를 정하는 것에 대응된다.

  - 인자 `C[]`의 길이는 $N-1$이고, 도로의 강설정보를 의미한다. `C[i]`$(0 \leq i \leq N-2)$는 도로 i의 강설정보를 의미하는 0이나 1인 숫자이고, `C[i]` $= 1$이면, 도로 i에 눈이 내린 것을, `C[i]` $= 0$이면, 도로 i에 눈이 내리지 않은 것을 의미한다.

    함수 `Anya`의 중에 이하의 함수를 호출할 수 있다.

    * `void Save(int place, int bit)`

      이 함수는 Anya가 중간 서버의 비트를 저장하는 작업을 의미한다.

      · 인자 `place`는 비트를 쓸 위치를 의미한다. `place`는 0 이상 $L-1$이하의 정수여야 한다. 이 범위 외의 값을 정해서 함수를 호출 할 경우, **오답** [1]이 된다. 또, 이 함수는 각 `Anya`의 호출에 대해, 같은 인자 `place`를 2회 이상 호출할 수 없다. 같은 인자로 2회 이상 호출한 경우 **오답** [2]가 된다.

      · 인자 `bit`는, 써 넣은 비트를 의미하는 정수이고, 0이나 1이어야 한다. 이 범위 외의 값을 정해서 함수를 호출 할 경우, **오답** [3]이 된다.

    `Save`를 호출 한 후, 중간서버의 `place`번째 비트는 `bit`이 된다. `Save`를 호출 하는 중에 오답이 된 경우, 그 시점에서 프로그램을 종료한다.

    함수 `Anya`가 호출 되기 직전에, 반드시, 중간서버의 비트는 0으로 초기화된다. 즉, 함수 `Save`를 통해 저장되지 않은 장소에는, 함수 `Anya`가 종료 될 때, 0이 적혀 있다.

두번째 파일은 `Boris.c`혹은 `Boris.cpp`라는 이름이다. 이 파일은 Anya의 전략이 구현된 파일이고, 다음의 2개의 함수가 구현되어 있어야한다. 프로그램은 `Borislib.h`를 `include`해야 한다.

- `void InitBoris(int N, int A[], int B[])`

  이 함수는, 각 테스트케이스에 대해 한 번만 호출된다.

  - 인자 `N`은 도시의 갯수를 의미한다.
  - 인자 `A[]`와 `B[]`는 각각의 길이 $N-1$인 배열이고, 도로의 연결정보를 의미한다. `A[i]`와 `B[i]` $(0 \leq i \leq N-2)$는 도로 i가 도시 `A[i]`와 `B[i]`를 양방향으로 잇는 것을 의미하는 정수이고, $0 \leq$ `A[i]` ¡ `B[i]` $\leq N-1$을 만족한다.

- `void Boris(int city)`

  이 함수는, `InitAnya`가 호출 된 후, 몇번이든 호출 된다. 이함수는 시민의 질문에 대응하는 Boris의 행동에 대응된다.

- 인자 `city`는 시민의 질문을 의미한다. `city`는 1 이상 $N-1$이하의 정수이다. 이것은 도시 0 부터 도시 `city`까지 이동하려면, 눈이 쌓인 도로를 최소 몇 개 지나야 하는지를, 시민이 질문하고 있는 것을 의미한다.

- 함수 `Boris`는, 시민의 질문에 대한 답을 0 이상 $N-1$이하의 정수로 반환해야 한다. 이 범위 외의 값을 반환 할 경우, **오답** [4]가 된다. 답이 올바르지 않을 경우 **오답** [7]이 된다.

  함수 `Boris`의 중에 이하의 함수를 호출할 수 있다.

- `int Ask(int place, int bit)`
  이 함수는 Boris가 중간 서버에서 비트를 읽는 작업을 의미한다.
  * 인자 `place`는 비트를 읽을 위치를 의미한다. `place`는 0 이상 $L-1$이하의 정수여야 한다. 이 범위 외의 값을 정해서 함수를 호출 할 경우, **오답 [5]**가 된다.

  함수 `Ask`의 반환값은, 중간 서버의 `place`번째 비트를 의미하는 정수이고, 0 혹은 1이다. 또, 함수 `Ask`는 함수 `Boris`의 **각 호출에 대해, 최대 20번 만 호출 할 수 있다.** 20번을 초과해 호출을 한 경우, **오답 [6]**이 된다.

`Ask`를 호출 하는 중에 오답이 된 경우, 그 시점에서 프로그램을 종료한다.

채점은 이하의 순서로 진행된다. 오답이라고 판정된 경우에는, 그 시점에 프로그램이 종료된다.

1. 도로정보를 알리기 위해, `InitAnya`와 `InitBoris`가 1번씩 순서대로 호출된다.
2. $i = 1, \cdots, Q$에 대해, 순서대로 이하의 작업을 한다.
   (a) 함수 `Anya`를 1회 부른다. 이것은, 하루의 시작에 도로의 강설정보를 갱신 한 후에, Anya가 중간 서버에 저장하는 비트열을 정하는 것에 대응된다.
   (b) 함수 `Boris`를 D_i회 부른다. D_i는 i일에 대해 시민이 한 질문의 횟수 이다. 그 중 j번째 $(1 \leq j \leq D_i)$ 호출을 의미하는 수는 R_{ij} $(1 \leq R_{ij} \leq N-1)$이다. j번째 호출에 대해, 함수 `Boris`의 반환값이, 도시 0부터 도시 R_{ij}까지 이동하기 위해 건너야 하는 눈이 쌓인 길의 최솟값과 일치하지 않은 경우, 오답이 된다.
3. 한번도 오답이라고 판정되지 않은 경우, 정답이 된다.

- 실행시간 계산 ·사용 메모리 계산의 대상은, 채점의 순서의 1, 2번이다. 정답인 경우에는, 순서 2에서 `Anya`가 Q번, $Boris$가 합계 $D_1 + \cdots + D_Q$번 호출된다.
- Anya나 Boris는, $D_1, \cdots, D_Q$에 대해서 알 수 없다.
- 함수 `Boris`는, 어떤 시점에 함수 `Anya`가 중간 서버에 존재하고 있는 데이터를 갱신했는가에 대한 정보를 모른다.
- 내부 사용을 위해서 다른 함수를 구현하거나, 글로벌 변수를 선언하는것은 자유이다. 하지만, 당신이 제출한 2개의 프로그램은 링크되어 하나의 실행파일이 되므로, 각 파일의 모든 글로벌 변수와 내부 함수(`InitAnya, Anya, InitBoris, Boris`를 제외)를 `static`으로 선언하여, 다른 파일과의 충돌을 막을 필요가 있다. 채점시에는, 이 프로그램을 Anya측, Boris측의 2개의 프로세스로 하여 실행하므로, Anya측과 Boris측이 프로그램 중에 글로벌 변수를 공유하는 것은 불가능하다.
- 당신의 제출은 표준 입출력이나, 다른 함수에 접근하면 안 된다.

작성한 프로그램을 테스트하기 위한 채점 프로그램 샘플이, 콘테스트 사이트에서 다운로드 받을 수 있는 아카이브 안에 있다. 이 아카이브는, 제출해야하는 파일의 샘플도 들어있다.

채점 프로그램 샘플은 1개의 파일이다. 이 파일은 `grader.c` 혹은 `grader.cpp`이다. 작성한 프로그램 `Anya.c`와 `Boris.c`, 혹은 `Anya.cpp`와 `Boris.cpp`라고 할 때, 작성 한 프로그램을 테스트 하기 위해서는, 다음의 커맨드를 실행한다.

- C의 경우 `gcc -std=c11 -O2 -o grader grader.c Anya.c Boris.c -lm`
- C++의 경우 `g++ -std=c++11 -O2 -o grader grader.cpp Anya.cpp Boris.cpp`

컴파일에 성공하면, `grader`라는 이름의 파일이 생성된다.

실제의 채점 프로그램은, 채점 프로그램 샘플과는 다르므로 주의한다. 채점 프로그램의 샘플은 단일 프로세스로 실행된다. 이 프로그램은, 표준입력에서 입력을 받아서, 표준출력으로 결과를 출력한다.

채점 프로그램의 샘플은, 채점의 순서에 따라 함수를 호출한다. 다음이 실제 채점 프로그램과 다름을 주의하라.

- **채점 프로그램의 샘플은, 오답[7]을 판정하지 않는 대신에, 각 질문에 대한 함수 `Boris`의 반환값을 출력한다.**

Input

채점 프로그램 샘플은, 표준입력에서 다음과 같은 데이터를 읽는다.

- 첫째 줄에는, 정수 N이 입력으로 들어온다. 이것은 도시의 수가 N개 있다는 것을 의미한다.
- 다음 $N-1$개의 줄의 $I+1$번째 줄 $(0 \leq i \leq N-2)$에는, 정수 A_i, B_i가 공백으로 구분되어 들어온다. 이것은 도로 i가 도시 A_i와 도시 B_i를 양방향으로 잇는 것을 의미한다.
- 다음 줄에는 정수 Q가 쓰여 있다. 이것은 대회가 Q일간 개최되었다는 것을 의미한다.
- 다음 Q개의 줄의 i번째 줄 $(1 \leq i \leq Q)$에는, 길이 $N-1$의 문자열 S_i와 D_i+1개의 정수가 공백으로 구분되어 들어온다. S_i는 i번째 날의 강설정보를 의미하고, S_i에서 왼쪽으로 $j+1$번째 $(0 \leq j \leq N-2)$ 문자가, '`1`'이면, 도로 j가 눈이 내린 것을, '`0`'이면, 도로 j에 눈이 내리지 않은 것을 의미한다. 다음 D_i+1개의 정수 중 가장 첫 정수는 D_i이다. 다음 D_i개의 정수는 $R_{i1}, \cdots, R_{iD_i}$이다. 이것은, i번째 날에 대해, j번째 $(1 \leq j \leq D_i)$ 질문이, 도시 0부터 도시 R_ij까지 이동하기 위해 눈이 쌓인 도로를 건너야 하는 수의 최소치에 관련이 있다는 것을 의미한다.

Output

채점 프로그램의 샘플은 표준 출력으로 이하의 정보를 출력한다. (따옴표는 실제로 출력되지 않는다.)

- 프로그램의 실행 중에 오답이라고 판단되는 경우, 오답의 종류가 "`Wrong Answer[1]`"처럼 출력되어, 실행이 종료된다.
- i번째 $(1 \leq i \leq Q)$의 `Anya`의 호출 후 호출된 D_i개의 `Boris`의 호출에 대해 모두 오답이라고 판된되지 않았을 경우, 그 시점에 한 줄에 D_i개의 정수를 공백으로 구분해서 출력한다. 출력된 정수 중 j번째 $(1 \leq j \leq D_i)$ 정수는, `Boris`(R_ij)의 반환값이다.

실행한 프로그램이 여러개의 오답을 일으켰을 경우, 표시되는 것은 그 중 하나 뿐이다.

Constraints

모든 입력데이터는 다음의 조건을 만족한다.

- $2 \leq N \leq 500$
- $1 \leq Q \leq 500$
- $0 \leq A_i < B_i \leq N-1$ $(0 \leq i \leq N-2)$
- $1 \leq D_j$ $(1 \leq j \leq Q)$
- $D_1 + \cdots + D_Q \leq 500$
- 어떤 2개의 서로 다른 도시 사이에도, 몇개의 도로를 경유하면 갈 수 있다.

Subtask 1 (15 points)

다음의 조건을 만족한다.

- $N \leq 20$

Subtask 2 (5 points)

다음의 조건을 만족한다.

- $N \leq 100$

Subtask 3 (35 points)

다음의 조건을 만족한다.

- $A_i = i$ $(0 \leq i \leq N-2)$
- $B_i = i+1$ $(0 \leq i \leq N-2)$

Subtask 4 (45 points)

추가 제한조건이 없다.

Example

다음은 채점 프로그램 샘플이 입력받은 입력 예제와, 그에 대응되는 함수 호출 예이다.

standard input	interaction			
	함수 호출	반환값	함수 호출	반환값
	`InitAnya(...)`			
		(없음)		
	`InitBoris(...)`			
		(없음)		
	`Anya(...)`			
			`Save(0, 1)`	
				(없음)
			`Save(1, 0)`	
				(없음)
			`Save(500, 1)`	
				(없음)
`5`		(없음)		
`0 1`	`Boris(2)`			
`1 2`			`Ask(0)`	
`1 4`				`1`
`2 3`			`Ask(1)`	
`2`				`0`
`0101 3 2 4 3`			`Ask(2)`	
`1110 1 4`				`0`
		`1`		
	`Boris(4)`			
		`0`		
	`Boris(3)`			
		`2`		
	`Anya(...)`			
		(없음)		
	`Boris(4)`			
			`Ask(0)`	
				`0`
		`2`		

이 예제의 함수 호출은, 반드시 의미 있는 호출은 아니라는 점에 주의하라. 이 때, `InitAnya(...)`, `InitBoris(...)`, 첫번째와 2번째의 `Anya(...)`에 전달되는 인수는 다음과 같다.

인자	InitAnya(...)	InitBoris(...)	Anya(...)	
			함수호출	반환값
N	5	5	1번째	2번째
A	{0, 1, 1, 2}	{0, 1, 1, 2}		
B	{1, 2, 4, 3}	{1, 2, 4, 3}		
C			{0, 1, 0, 1}	{1, 1, 1, 0}

Problem C. Worst Reporter 2

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 second
Memory limit:	256 megabytes

때는, 21XX년, Problem Solving은 마인드 스포츠의 하나로 널리 인지되어 있고, TV나 신문 등 미디아에 오르는 일도 많다. 당신은 JOI신문사의 기사이고, Problem Solving 기사를 담당하고 있다. 어느 날, N명의 선수가 참가하는 국제적인 Problem Solving 대회가 개최되었다. 이 대회에 대해 기사를 쓰기 위해, 당신은 다음의 정보를 알았다.

- 국제 정보올림피아드 처럼, 이 대회는 몇개의 나라에서 온 선수들이 참여한다. 나라에는 1부터 N까지의 번호가 붙어있다. 하나의 나라에서 여러명의 선수가 참가 할 수도 있다. 또한, 선수가 참가하지 않은 나라가 있을 수도 있다.
- 이 대회의 시간제한은 5시간이다.
- 대회 중에 선수가 획득한 점수는, 그 후 줄어들지 않는다.
- 대회 개최후 2시간이 경과한 시점에, 동점인 선수는 없었다. 그 시점의 순위표는 i등 $(1 \leq i \leq N)$인 선수는 A_i 나라 출신이고, 그 선수의 점수는 B_i점 이었다.
- 대회가 끝날 때, 동점의 선수는 없었다. 대회가 끝날 때 순위표에 따르면 i등 $(1 \leq i \leq N)$인 선수는 C_i 나라 출신이고, 그 선수의 점수는 D_i점 이었다.

그러나, 기사를 쓰려 보니, 순위표의 출신국의 표시에 결함이 있었다는 것이 밝혀졌다. 선수의 출신국의 정보가 잘못되었을 가능성이 있다. 표시된 선수의 점수는 올바르다는 것을 알고 있다.

여기서, 당신은, 알고 있는 정보로 최소한의 수정을 가해서, 순위표의 정보를 모순이 없게(같은 선수의 국적이 대회 기간동안 변하지 않고, 획득한 순서의 감소가 없게) 추측하기로 했다. 즉, $2N$개의 값 $A_1, \cdots, A_N, C_1, \cdots C_N$중 최대한 적은 갯수의 값을 변경해서, 다음을 만족하도록 하고 싶다.

- $1, 2, \cdots, N$의 순열 $x_1, x_2, \cdots, x_N$에 대해, 각 $i = 1, 2, \cdots, N$에 대해, $A_i = C_{x_i}$ 이고, $B_i \leq D_{x_i}$가 성립하는 것이 존재한다.

당신은, 알고 있는 정보로, 최소 몇번의 수정을 가해야 하는가.

대회의 참가자수와, 대회 개최후 2시간 경과한 시점과 대회 종료 시점의 순위표에 대한 정보가 주어졌을 때, 순위표에 모순이 없게 만드는데 필요한, 출신국 정보의 변경 횟수의 최솟값을 구하는 프로그램을 작성하여라.

Input

표준 입력으로 다음의 데이터가 들어온다.

- 첫째 줄에는, 정수 N이 들어온다. 이것은, 대회의 참가자 수가 N명이라는 것을 의미한다.
- 다음 N개의 줄의 i번째 줄$(1 \leq i \leq N)$에는, 정수 A_i, B_i가 공백으로 구분되어 들어온다. 이는, 대회 개최 후 2시간 경과한 시점의 순위표가 i등인 선수가 국가 A_i 출신으로 표시되었고, 획득 점수가 B_i 점이었다는 것을 의미한다.
- 다음 N개의 줄의 i번째 줄$(1 \leq i \leq N)$에는, 정수 C_i, D_i가 공백으로 구분되어 들어온다. 이는, 대회 종료 시점의 순위표가 i등인 선수가 국가 C_i 출신으로 표시되었고, 획득 점수가 D_i점이었다는 것을 의미한다.

Output

표준 출력에, 순위표에 모순이 없는 상황을 만들기 위해 필요한, 출신국 정보의 변경갯수의 최솟값을 첫째 줄에 출력하여라.

Constraints

모든 입력데이터는 다음의 조건을 만족한다.

- $2 \leq N \leq 200\ 000$
- $1 \leq A_i \leq N$ $(1 \leq i \leq N)$
- $0 \leq B_i \leq 1\ 000\ 000\ 000$ $(1 \leq i \leq N)$
- $B_i > B_{i+1}$ $(1 \leq i \leq N-1)$
- $1 \leq C_i \leq N$ $(1 \leq i \leq N)$
- $0 \leq D_i \leq 1\ 000\ 000\ 000$ $(1 \leq i \leq N)$
- $D_i > D_{i+1}$ $(1 \leq i \leq N-1)$
- $A_1, \cdots, A_N, C_1 \cdots, C_N$의 값을 몇개 수정하는 것으로, 순위표에 모순이 없는 상황을 만들 수 있다.

Subtask 1 (15 points)

다음의 조건을 만족한다.

- $N \leq 16$

Subtask 2 (15 points)

다음의 조건을 만족한다.

- $N \leq 50$

Subtask 3 (30 points)

다음의 조건을 만족한다.

- $N \leq 5\ 000$

Subtask 4 (40 points)

추가 제한조건이 없다.

Examples

standard input	standard output
3 3 500 2 200 1 100 1 1000 3 700 3 400	1

C_3의 값을 2로 수정하면, 다음과 같이 모순이 없는 순위표가 된다:

- 대회 개최후 2시간 경과한 시점에 500점으로 1위 였던 나라 3의 선수는, 대회 종료 후 700점으로 2위가 되었다.

- 대회 개최후 2시간 경과한 시점에 200점으로 2위 였던 나라 2의 선수는, 대회 종료 후 400점으로 3위가 되었다.
- 대회 개최후 2시간 경과한 시점에 100점으로 3위 였던 나라 1의 선수는, 대회 종료 후 1000점으로 1위가 되었다.

여기서, C_2의 값을 2로 수정한 경우, 나라 3 출신의 선수가 대회 개최후 2시간 경과한 시점에서 500점을 획득했음에도 불구하고, 대회 종료 시점에서 400점을 획득했기 때문에, 모순된 순서표가 된다. 1개 보다 더 작은 수정을 가해 모순이 없는 순위표를 만드는 것이 불가능 하므로, 1을 출력한다.

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

이 경우, 출신국 정보의 수정이 없어도 모순이 없는 순위표가 된다. 대회 개최후 2시간 경과시점의 점수에서 점수를 획득하지 못한 선수가 있을수도 있음에 주의하여라. 또, 순의표에 대해, 같은 나라의 전수가 여럿 있을 가능성이 있음에 주의하여라.

standard input	standard output
6 1 70 4 50 1 30 2 20 1 10 3 0 6 100 2 90 1 80 2 60 4 40 1 10	3

이 입력에 대해서는, A_1의 값을 2로 수정하고, A_6의 값을 4로 수정하고, C_1의 값을 4로 수정하는 것으로, 모순이 없는 순위표가 된다.

This page is intentionally left blank