문제 1. 특별관광도시

입력 파일:	`standard input`
출력 파일:	`standard output`
시간 제한:	2 seconds
메모리 제한:	512 megabytes

JOI나라에는 N개의 도시가 있다. 이 도시들은 1번부터 N번까지 번호가 붙어있다. 이 도시에는 $N-1$개의 도로가 있고, 1번부터 $N-1$번까지의 번호가 붙어있다. i번 $(1 \leq i \leq N-1)$ 도로는 노선이 두개가 있다. 한 노선은 A_i번 도시에서 B_i번 도시로 향하는 노선이고, 다른 노선은 B_i번 도시에서 A_i번 도시로 향하는 노선이다. 즉, 모든 도로는 양방향이다. 어떤 두 도시간에도 몇개의 도로를 사용해서 이동하는 것이 가능하다.

처음에 모든 노선들은 정비되어있지 않다. 각 도로의 각 노선에 대해, 우리는 노선을 정비하는 비용을 알고 있다. i번 $(1 \leq i \leq N-1)$ 도로의 A_i번 도시에서 B_i번 도시로 향하는 노선을 정비하는 비용은 C_i이고, B_i번 도시에서 A_i번 도시로 향하는 노선을 정비하는 비용은 D_i이다.

JOI나라의 장관인 K이사장은 몇몇 도시를 돌라 그 도시를 **특별관광도시**로 만들것이다. x번 $(1 \leq x \leq N)$을 특별관광도시로 만들 때, 각 도로 $i(1 \leq i \leq N-1)$에 대해, 다음 일이 일어날 것이다.

> A_i번과 B_i번 도시 중에서 x번 도시에 가까운 도시는 a번 도시이고, 먼 도시는 b번 도시라고 하자. 여기서, 가까운 도시라고 함은 x번 도시에 가기 위해 사용해야 하는 도로의 수가 더 적은 도시를 말한다. 이 때, b번 도시에서 a번 도시로 향하는 노선이 정비되지 않은 상태라면 정비된다.

특별관광도시를 만들기 위해 노선을 정비하는 비용은 세금으로 충당되지만, 특별관광도시가 만들어 진 이후에 남은 도로를 정비하는 비용은 K이사장의 개인 자금에서 나간다.

K이사장이 계획한 Q개의 계획이 있다. j 번째 $(1 \leq j \leq Q)$ 계획에서는, 그는 특별관광도시가 없고 모든 노선이 정비되지 않은 상태에서 시작해서 정확히 E_j개의 도시를 특별관광도시로 만들것이다. 하지만, 어떤 도시들이 특별관광도시가 될지는 계획되지 않았다. 그는 개인 자금에서 나가는 도로 정비 비용을 최소로 하고 싶다.

JOI나라의 도시 수, 도로의 정보와 계획의 정보가 주어졌을 때, 각 계획마다 K이사장의 개인 자금에서 나가는 도로 정비 비용을 최소로 하는 프로그램을 작성하여라.

입력 형식

표준 입력에서 다음과 같은 형식으로 주어진다. 모든 값은 정수이다.

N

$A_1\ B_1\ C_1\ D_1$

$\vdots$

$A_{N-1}\ B_{N-1}\ C_{N-1}\ D_{N-1}$

Q

E_1

$\vdots$

E_Q

출력 형식

표준 출력으로 Q개의 줄을 출력하여라. j 번째 $(1 \leq j \leq Q)$줄은 j 번째 계획에서 이사장의 개인 자금에서 나가는 도로 정비 비용의 최솟값이어야 한다.

제한

- $2 \leq N \leq 200\ 000$.

- $1 \leq A_i \leq N$ $(1 \leq i \leq N)$.
- $1 \leq B_i \leq N$ $(1 \leq i \leq N)$.
- $A_i \neq B_i$ $(1 \leq i \leq N)$.
- 어떤 두 도시간에도 몇개의 도로를 사용해서 이동하는 것이 가능하다.
- $0 \leq C_i \leq 1\ 000\ 000\ 000$ $(1 \leq i \leq N)$.
- $0 \leq D_i \leq 1\ 000\ 000\ 000$ $(1 \leq i \leq N)$.
- $1 \leq A_i \leq B_i \leq N - 1$ $(1 \leq i \leq N)$.
- $1 \leq Q \leq N$.
- $1 \leq E_j \leq N$ $(1 \leq j \leq Q)$.

서브태스크 1 (16 점)

- $N \leq 16$

서브태스크 2 (7 점)

- $Q = 1$
- $E_1 = 1$

서브태스크 3 (9 점)

- $Q = 1$
- $E_1 = 2$

서브태스크 4 (17 점)

- $N \leq 2000$

서브태스크 5 (17 점)

- $Q = 1$

서브태스크 6 (44 점)

추가 제한조건이 없다.

예제

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

첫 번째 계획은 정확히 하나의 도시를 특별관광도시로 만드는 것이다. 만약 1번 도시를 특별관광도시로 지정한다면, 1번 도로의 2번 도시에서 1번 도시로 향하는 노선, 2번 도로의 3번 도시에서 1번 도시로 향하는 노선, 3번 도로의 1번 도시에서 3번 도시로 향하는 노선이 정비될 것이다. 이 때 정비되지 않고 남는 노선은 1번 도로의 1번 도시에서 2번 도시로 향하는 노선, 2번 도로의 1번 도시에서 3번 도시로 향하는 노선, 3번 도로의 1번 도시에서 4번 도시로 향하는 노선이다. 이 노선들을 정비하는 비용은 1+3+5=9이다. 이보다 더 낮은 비용으로 특별관광도시를 지정하는 방법은 없다. 그래서 답은 9이다.

두번째 계획은 정확히 두 개의 도시를 특별관광도시로 만드는 것이다. 만약 3번 도시와 4번 도시를 특별관광도시로 지정했다면, 도로의 1번 도시에서 2번 도시로 향하는 노선만 정비되지 않았을 것이다. 이 노선을 정비하는데에 드는 비용은 1이다. 이보다 더 낮은 비용으로 두 특별관광도시를 지정하는 방법은 없다. 그래서 답은 1이다.

standard input	standard output
5 1 3 13 6 5 1 17 8 5 2 6 10 1 4 16 11 1 1	36

이 입력은 서브태스크 2의 조건을 만족한다.

standard input	standard output
6 1 6 6 12 6 2 5 16 1 4 13 4 5 1 19 3 3 1 9 13 1 2	14

이 입력은 서브태스크 3의 조건을 만족한다.

문제 2. 램프

입력 파일:	`standard input`
출력 파일:	`standard output`
시간 제한:	1 second
메모리 제한:	256 megabytes

긴 복도에 N개의 램프가 일렬로 나열되어 있다. 램프는 왼쪽부터 차례로 1번부터 N번까지의 번호가 붙어있다. 각 램프는 off또는 on중 하나의 상태이다.

램프의 상태를 바꾸는 특별한 기작이 있어서, 한 번의 작업으로 다음 셋 중 한 가지 동작을 할 수 있다.

- $1 \leq p \leq q \leq N$을 만족하는 정수 p와 q를 골라서 $p, p+1, \cdots, q$를 off 상태로 만든다.
- $1 \leq p \leq q \leq N$을 만족하는 정수 p와 q를 골라서 $p, p+1, \cdots, q$를 on 상태로 만든다.
- $1 \leq p \leq q \leq N$을 만족하는 정수 p와 q를 골라서 $p, p+1, \cdots, q$의 상태를 바꾼다. (off를 on으로, on을 off로)

처음에 램프의 상태는 길이 N의 문자열 A로 표현된다. A의 i 번째 $(1 \leq i \leq N)$ 문자가 `0`이면 i 번째 램프가 off 상태인 것이고, `1`이면 on 상태인 것이다. 우리는 만들고 싶은 상태가 길이 N의 문자열 B로 표현 되어 있고, 작업의 수를 최소한으로 하여 만들고 싶다. B의 i 번째 $(1 \leq i \leq N)$ 문자가 `0`이면 i 번째 램프가 off 상태인 것이고, `1`이면 on 상태인 것이다.

램프의 수와, 현재 상태와 만들고 싶은 상태가 주어졌을 때, 만들고 싶은 상태로 바꾸는 데에 드는 연산의 수의 최솟값을 출력하여라.

입력 형식

표준 입력에서 다음과 같은 형식으로 주어진다.

N

A

B

출력 형식

표준 출력으로 한 개의 줄을 출력하여라. 이는 원하는 상태를 만들기 위한 연산의 수의 최솟값이다.

제한

- $1 \leq N \leq 1\ 000\ 000$.
- A와 B는 길이 N의 문자열이다.
- A와 B를 이루는 문자들은 `0` 혹은 `1`이다.

서브태스크 1 (6 점)

- $N \leq 18$

서브태스크 2 (41 점)

- $N \leq 2000$

서브태스크 3 (4 점)

- A를 이루는 각 문자는 0이다.

서브태스크 4 (49 점)

추가 제한조건이 없다.

예제

standard input	standard output
8 11011100 01101001	4

이 입력에서 우리는 원하는 상태를 다음과 같은 방법으로 네 번의 작업으로 만들 수 있다.

1. 1, 2, 3, 4번 램프의 상태를 바꾼다. 램프의 상태는 00101100이 된다.
2. 2번 램프를 off 상태로 만든다. 램프의 상태는 01101100이 된다.
3. 6, 7, 8번 램프의 상태를 바꾼다. 램프의 상태는 01101011이 된다.
4. 6, 7번 램프를 on 상태로 만든다. 램프의 상태는 01101001이 된다.

네 번보다 더 적은 작업으로 원하는 상태를 만들 수 있는 방법은 없으므로, 4를 출력한다.

standard input	standard output
13 1010010010100 0000111001011	3
18 001100010010000110 110110001000100101	5

문제 3. 시간을 달리는 비타로

입력 파일:	`standard input`
출력 파일:	`standard output`
시간 제한:	3 seconds
메모리 제한:	512 megabytes

비버랜드에는 N개의 도시가 있다. 이 도시들은 1번부터 N번까지 번호가 붙어있다. i번째 $(1 \leq i \leq N-1)$ 도로는 i번 도시와 $i+1$번 도시를 양방향으로 잇는다. 또한, 비버랜드의 하루는 1 000 000 000 000개의 단위시간으로 분열되어 있고, 이 단위시간을 쵸라고 부른다. 하루가 시작하고 나서 x쵸가 지난 시간을 시각 x라 부른다. 한 도로를 통과하는 데에는 1쵸가 걸리고, i번째 도로는 시각 L_i와 시각 R_i 사이에만 통과할 수 있다. 구체적으로, i번째 도로를 통과하기 위해서 우리는 도시 i나 $i+1$을 $L_i \leq x \leq R_i - 1$ 을 만족하는 시각 x에 떠나야 하고, 다른 도시에 시각 $x+1$에 도착해야 한다.

비타로는 비버랜드에 사는 평범한 비버다. 아니, 비버였다 라고 하는게 옳은 것일까. 지각을 자주한 비타로는 이를 개선하려고 한 결과로 시간을 거슬러 올라가는게 가능해 졌다. 이 능력을 한 번 사용하면 1쵸 뒤로 갈 수 있다. 하지만, 어제로 갈 수는 없다. 만약 그가 능력을 시각 0과 시각 1 사이에 사용했다면, 그는 시각 0으로 돌아갈 것이다. 그는 이 기술을 도시에 있을 때 사용할 수 있다. 비타로의 위치는 능력을 사용해도 변하지 않는다.

비타로는 기술을 사용하면 피곤해 진다. 최소한의 기술을 사용하여 이동하는 방법을 찾기 위한 비타로는 Q개의 사고실험을 진행했다. 사고 실험의 j 번째 단계에서는, 그는 다음 중 한 행동을 한다:

- P_j 번째 도로가 여행될수 있는 시각을 바꾼다. 바뀐 이후에는, 시각 S_j와 시각 E_j 사이에만 P_j 번째 도로를 통과할 수 있다.
- 그가 A_j번 도시, 시각 B_j에 있다고 할 때, C_j번 도시, 시각 D_j로 이동하기 위해 사용해야하는 능력의 수의 최솟값을 구하여라.

그는 사고실험의 결과를 궁금해한다.

비버랜드의 도시의 수, 도로의 정보, 사고실험의 방법이 주어졌을 때, 사고 실험의 결과를 계산하는 프로그램을 작성하여라.

입력 형식

표준 입력에서 다음과 같은 형식으로 주어진다. 모든 값은 정수이다.

N Q

L_1 R_1

$\vdots$

L_{N-1} R_{N-1}

(Query 1)

$\vdots$

(Query Q)

여기서, (Query j)는 공백으로 구분된 4개나 5개의 정수로 이루어져 있다. T_j가 첫번째 정수라고 하자. 그러면,

- $T_j = 1$인 경우, (Query j)는 4개의 정수 T_j, P_j, S_j, E_j로 이루어져 있다. 이것은, 사고 실험의 j번째 단계에서, P_j번째 도로를 지날수 있는 시간이 시각 S_j와 시각 E_j 사이로 바뀐다는 것을 의미한다.
- $T_j = 2$인 경우, (Query j)는 5개의 정수 T_j, A_j, B_j, C_j, D_j로 이루어져 있다. 이는, j번째 사고 실험에서, 당신의 프로그램이 비타로가 A_j번 도시, 시각 B_j에 있다고 할 때, C_j번 도시, 시각 D_j로 이동하기 위해 사용해야하는 능력의 수의 최솟값을 구해야 한다는 것을 의미한다.

출력 형식

$T_j = 2$인 각 단계에 대해서, 사용해야 하는 능력의 수의 최솟값을 한 줄에 하나씩 차례로 출력하여라.

제한

- $1 \leq N \leq 300\ 000$.
- $1 \leq Q \leq 300\ 000$.
- $0 \leq L_i < R_i \leq 999\ 999\ 999$ $(q \leq i \leq N-1)$.
- $1 \leq T_j \leq 2$ $(1 \leq j \leq Q)$.
- $1 \leq P_j \leq N-1$ $(1 \leq j \leq Q, T_j = 1)$.
- $1 \leq S_j \leq E_j \leq 999\ 999\ 999$ $(1 \leq j \leq Q, T_j = 1)$.
- $1 \leq A_j \leq N$ $(1 \leq j \leq Q, T_j = 2)$.
- $1 \leq B_j \leq 999\ 999\ 999$ $(1 \leq j \leq Q, T_j = 2)$.
- $1 \leq C_j \leq N$ $(1 \leq j \leq Q, T_j = 2)$.
- $1 \leq D_j \leq 999\ 999\ 999$ $(1 \leq j \leq Q, T_j = 2)$.

서브태스크 1 (4 점)

- $N \leq 1000$
- $Q \leq 1000$

서브태스크 2 (30 점)

- $T_j = 2$ $(1 \leq j \leq Q)$.

서브태스크 3 (66 점)

추가 제한조건이 없다.

예제

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

사고 실험의 첫 번째 단계에서, 비타로는 1번 도시에서 2번 도시로 1초만에 이동하고, 2번 도시에서 3번 도시로 1초만에 이동하여 3번 도시, 시각 5에 위치하여 있다. 능력을 두번 사용하면, 그는 3번 도시, 시각 3에 위치할 수 있다.

사고 실험의 두 번째 단계에서, 2번 도시를 통과할 수 있는 시간이 시각 0부터 시각 1까지로 바뀐다.

사고 실험의 세 번째 단계에서, 1번 도시에서 2번 도시로 1초 만에 이동하여, 2번 도시, 시각 4에 위치해 있다. 여기서 능력을 네 번 사용하면, 3번 도시로 1초만에 이동하여 2초를 기다리면 3번 도시, 시각 3에 위치할 수 있다.

standard input	standard output
5 5 3 5 4 8 2 6 5 10 2 5 3 1 10 2 2 6 5 6 1 3 4 6 2 3 3 4 3 2 4 5 1 5	4 3 2 3
7 7 112103440 659752416 86280800 902409187 104535475 965602300 198700180 945132880 137957976 501365807 257419446 565237610 2 4 646977260 7 915994878 2 1 221570340 6 606208433 2 7 948545948 4 604273995 2 7 247791098 5 944822313 2 7 250362511 2 50167280 2 3 364109400 4 555412865 2 7 33882587 7 186961394	145611455 0 447180143 0 207252171 0 0
7 7 535825574 705426142 964175291 996597835 481817391 649559926 4519006 410772613 74521477 274584126 256535565 899389890 1 6 511428966 602601933 1 1 69986642 201421232 2 3 636443425 4 625975977 1 6 235225515 405336399 2 3 866680458 3 701821857 1 6 180606048 900533151 1 6 612564160 720179605	10467449 164858601