문제 1. 두 안테나

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

1번부터 N번까지의 번호가 붙어있는 N개의 안테나가 일렬로 놓여 있다. 각 안테나는 다른 연속된 안테나와 1km 떨어져 있다. i번 ($1 \leq i \leq N$) 안테나의 높이는 H_i이다. i번 안테나는 자신으로 부터 $A_i$km 이상 $B_i$km 이하 떨어져 있는 안테나에게만 정보를 보낼 수 있다. 만약 x번 안테나와 y번 안테나가 ($1 \leq x < y \leq N$) 서로 정보를 주고 받을 수 있다면, 이 둘은 통신할 수 있고, 통신 비용은 $|H_x - H_y|$이다.

JOI 공화국의 수상 K씨는 시민들로부터 연결상태에 관한 불만 Q개를 들었다. 조사 결과 j 번째 ($1 \leq j \leq Q$) 불만은, $L_j,\ L_j + 1, \cdots, R_j$번 안테나 중 무언가가 이상이 있는것으로 밝혀졌다. 당신은, 이 안테나들중 서로 통신할 수 있는 안테나 쌍이 있는지, 만약 있다면 그 중 가장 통신 비용이 높은 쌍의 통신 비용은 얼마인지 알아보는 일을 맡았다.

안테나의 정보와 불만의 정보가 주어졌을 때, $L_j,\ L_j + 1, \cdots, R_j$번 안테나 중 서로 통신할 수 있는 쌍이 있는지, 있다면 통신 비용의 최댓값은 얼마인지를 알려주는 프로그램을 작성하여라.

입력 형식

표준 입력에서 다음과 같은 형식으로 주어진다. 모든 값은 정수이다.

N

$H_1\ A_1\ B_1$

$\vdots$

$H_N\ A_N\ B_N$

Q

$L_1\ R_1$

$\vdots$

$L_Q\ R_Q$

출력 형식

표준 출력으로 Q개의 줄을 출력하여라. j번째 ($1 \leq j \leq Q$)줄은 $L_j,\ L_j + 1, \cdots, R_j$번 안테나 중 서로 통신할 수 있는 쌍이 없으면 `-1`, 있다면 통신 비용의 최댓값이어야 한다.

제한

- $2 \leq N \leq 200\ 000$.
- $0 \leq H_i \leq 1\ 000\ 000\ 000$ ($1 \leq i \leq N$).
- $1 \leq A_i \leq B_i \leq N - 1$ ($1 \leq i \leq N$).
- $1 \leq Q \leq 200\ 000$.
- $1 \leq L_j < R_j \leq N$ ($1 \leq j \leq Q$).

서브태스크 1 (2 점)

- $N \leq 300$
- $Q \leq 300$

서브태스크 2 (11 점)

- $N \leq 2\ 000$

서브태스크 3 (22 점)

- $Q = 1$
- $L_1 = 1$
- $R_1 = N$

서브태스크 4 (65 점)

추가 제한조건이 없다.

예제

standard input	standard output
5 10 2 4 1 1 1 2 1 3 1 1 1 100 1 1 5 1 2 2 3 1 3 1 4 1 5	-1 1 8 8 99

1번 안테나와 2번 안테나는 서로 통신할 수 없으므로, 첫 번째 불만에 대한 답은 -1이다.

두 번째, 세 번째, 네 번째, 다섯 번째 불만에 대한 최대 통신 비용을 가진 안테나 쌍은 각각 (2, 3), (1, 3), (1, 3), (4, 5) 이다.

standard input	standard output
20 260055884 2 15 737689751 5 5 575359903 1 15 341907415 14 14 162026576 9 19 55126745 10 19 95712405 11 14 416027186 8 13 370819848 11 14 629309664 4 13 822713895 5 15 390716905 13 17 577166133 8 19 195931195 10 17 377030463 14 17 968486685 11 19 963040581 4 10 566835557 1 12 586336111 6 16 385865831 8 9 1 1 20	806460109

이 입력은 서브태스크 3의 조건을 만족한다.

문제 2. 두 요리

입력 파일:	`standard input`
출력 파일:	`standard output`
시간 제한:	5 seconds
메모리 제한:	1024 megabytes

요리사 비타로는 요리 대회에 참여했다. 이 대회에서 참가자는 IOI 돈부리와 JOI 카레를 요리해야 한다.

IOI 돈부리를 요리하는 방법은 N단계로 이루어져 있다. i 번째 $(1 \leq i \leq N)$ 단계는 정확히 A_i분이 걸린다. 처음에, 그는 첫 번째 단계만 실행할 수 있다. i번째 $(2 \leq i \leq N)$ 단계를 실행하려면, $(i-1)$번째 단계를 끝마쳐야 한다.

JOI 카레를 요리하는 방법은 M단계로 이루어져 있다. j 번째 $(1 \leq j \leq M)$ 단계는 정확히 B_j분이 걸린다. 처음에, 그는 첫 번째 단계만 실행할 수 있다. j번째 $(2 \leq j \leq M)$ 단계를 실행하려면, $(j-1)$번째 단계를 끝마쳐야 한다.

각 단계를 집중해야 하기 때문에, 한 단계를 시작하면, 그 단계를 끝날 때 까지 다른 단계를 실행할 수 없다. 한 단계가 끝난 이후에는 다른 요리의 단계를 시작해도 상관 없다. 대회가 시작하면 두 요리가 끝나기 까지의 쉬는 시간은 없다.

이 대회에서는, 각 참가자는 **예술 점수**를 다음 기준에 따라 받는다.

- IOI 돈부리를 만드는 i번째 $(1 \leq i \leq N)$ 단계를 처음부터 S_i분 안에 끝냈을 경우 P_i점을 얻는다. P_i는 음수 일 수도 있다.
- JOI 카레를 만드는 j번째 $(1 \leq j \leq M)$ 단계를 처음부터 T_j분 안에 끝냈을 경우 Q_j점을 얻는다. Q_j는 음수 일 수도 있다.

비타로는 예술 점수를 최대화 하고 싶다.

요리 단계의 수와, 각 단계에 걸리는 시간과, 예술 점수의 정보가 주어졌을 때, 비타로가 얻을 수 있는 예술 점수의 최댓값을 구하여라.

입력 형식

표준 입력에서 다음과 같은 형식으로 주어진다. 모든 값은 정수이다.

$N\ M$

$A_1\ S_1\ P_1$

$\vdots$

$A_N\ S_N\ P_N$

$B_1\ T_1\ Q_1$

$\vdots$

$B_M\ T_M\ Q_M$

출력 형식

표준 출력으로 한 개의 줄을 출력하여라. 이는 비타로가 얻을 수 있는 예술 점수의 최댓값이다.

제한

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

- $1 \leq B_j \leq 1\ 000\ 000\ 000$ $(1 \leq j \leq M)$.
- $1 \leq S_i \leq 2\ 000\ 000\ 000\ 000\ 000 = 2 \times 10^{15}$ $(1 \leq i \leq N)$.
- $1 \leq T_j \leq 2\ 000\ 000\ 000\ 000\ 000 = 2 \times 10^{15}$ $(1 \leq j \leq M)$.
- $-1\ 000\ 000\ 000 \leq P_i \leq 1\ 000\ 000\ 000$ $(1 \leq i \leq N)$.
- $-1\ 000\ 000\ 000 \leq Q_j \leq 1\ 000\ 000\ 000$ $(1 \leq j \leq M)$.

서브태스크 1 (5 점)

- $N \leq 200\ 000$
- $M \leq 200\ 000$
- $S_1 = \cdots = S_N$
- $T_1 = \cdots = T_N$

서브태스크 2 (3 점)

- $N \leq 12$
- $M \leq 12$
- $P_i = 1$ $(1 \leq i \leq N)$
- $Q_j = 1$ $(1 \leq j \leq M)$

서브태스크 3 (7 점)

- $N \leq 2\ 000$
- $M \leq 2\ 000$
- $P_i = 1$ $(1 \leq i \leq N)$
- $Q_j = 1$ $(1 \leq j \leq M)$

서브태스크 4 (39 점)

- $N \leq 200\ 000$
- $M \leq 200\ 000$
- $P_i = 1$ $(1 \leq i \leq N)$
- $Q_j = 1$ $(1 \leq j \leq M)$

서브태스크 5 (11 점)

- $N \leq 200\ 000$
- $M \leq 200\ 000$

- $1 \leq P_i$ $(1 \leq i \leq N)$
- $1 \leq Q_j$ $(1 \leq j \leq M)$

서브태스크 6 (9 점)

- $1 \leq P_i$ $(1 \leq i \leq N)$
- $1 \leq Q_j$ $(1 \leq j \leq M)$

서브태스크 7 (17 점)

- $N \leq 200\ 000$
- $M \leq 200\ 000$

서브태스크 8 (9 점)

추가 제한조건이 없다.

예제

standard input	standard output
5 10 2 4 1 1 1 2 1 3 1 1 1 100 1 1 5 1 2 2 3 1 3 1 4 1 5	-1 1 8 8 99

이 입력은 서브태스크 2의 조건을 만족한다.

이 입력에서 비타로는 다음과 같은 방식으로 요리 할 수 있다.

1. JOI 돈부리의 첫 번째 단계를 요리한다. 그는 대회가 시작한 후 3분 후에 단계를 끝낸다. 6분 안이므로, 1점을 얻는다.
2. IOI 카레의 첫 번째 단계를 요리한다. 그는 대회가 시작한 후 5분 후에 단계를 끝낸다. 1분 밖이므로, 아무 점수도 얻지 못한다.
3. IOI 돈부리의 두 번째 단계를 요리한다. 그는 대회가 시작한 후 8분 후에 단계를 끝낸다. 8분 안이므로, 1점을 얻는다.
4. JOI 카레의 두 번째 단계를 요리한다. 그는 대회가 시작한 후 10분 후에 단계를 끝낸다. 11분 안이므로, 1점을 얻는다.
5. IOI 돈부리의 세 번째 단계를 요리한다. 그는 대회가 시작한 후 12분 후에 단계를 끝낸다. 13분 안이므로, 1점을 얻는다.
6. IOI 돈부리의 네 번째 단계를 요리한다. 그는 대회가 시작한 후 13분 후에 단계를 끝낸다. 13분 안이므로, 1점을 얻는다.

7. JOI 커리의 세 번째 단계를 요리한다. 그는 대회가 시작한 후 15분 후에 단계를 끝낸다. 15분 안이므로, 1점을 얻는다.

총 예술점수는 6점이다. 6점 보다 더 많은 점수를 얻을 수는 없으므로 6을 출력해야 한다.

standard input	standard output
5 7 16 73 16 17 73 10 20 73 1 14 73 16 18 73 10 3 73 2 10 73 7 16 73 19 12 73 4 15 73 15 20 73 14 15 73 8	63

이 입력은 서브태스크 1의 조건을 만족한다.

standard input	standard output
9 11 86 565 58 41 469 -95 73 679 28 91 585 -78 17 513 -63 48 878 -66 66 901 59 72 983 -70 68 1432 11 42 386 -87 36 895 57 100 164 10 96 812 -6 23 961 -66 54 193 51 37 709 82 62 148 -36 28 853 22 15 44 53 77 660 -19	99

문제 3. 두 운송수단

입력 파일:	`standard input`
출력 파일:	`standard output`
시간 제한:	1.5 seconds
메모리 제한:	256 megabytes

JOI 나라에는 0부터 $N-1$까지의 번호가 붙은 도시 N개가 있다. JOI 나라에는 A개의 전철노선이 있고, 0번 부터 $A-1$번 까지의 번호가 붙어 있다. i번 $(0 \leq i \leq A-1)$ 전철 노선은 U_i번 도시와 V_i번 도시를 양방향으로 잇는다. 이 노선의 운임은 C_i이다. 서로 다른 전철 노선은 서로 다른 쌍의 도시를 잇는다. 또한, B개의 버스노선이 있고, 0번 부터 $B-1$번 까지의 번호가 붙어 있다. j번 $(0 \leq j \leq B-1)$ 버스 노선은 S_j번 도시와 T_j번 도시를 양방향으로 잇는다. 이 노선의 운임은 D_j이다. 서로 다른 버스 노선은 서로 다른 쌍의 버스를 잇는다. 하지만, 전철 노선과 버스 노선은 같은 쌍의 도시를 이을수도 있다. 전철 혹은 버스를 사용하면 어떠한 쌍의 도시도 오갈 수 있다.

Azer는 0번 도시로 부터 각 도시까지 가기 위한 최소 운임을 알고 싶다. Azer는 전철 노선만 알고 있기 때문에, 그는 버스 노선만 아는 Baijan과 협력해야 한다.

그들은 서로 문자 0 혹은 1을 주고 받으면서 통신한다. 보내는 총 문자의 갯수는 58 000개 이하여야 한다.

전철노선의 정보가 주어진 Azer과, 버스노선의 정보가 주어진 Baijan 사이에서 서로 통신하고 Azer가 0번 도시부터 각 도시까지 가기 위한 최소 운임을 출력하여라.

구현 명세

당신은 파일 두 개를 제출해야 한다.

첫째 파일의 이름은 `Azer.cpp`이다. 이 파일은 Azer의 일을 나타내고, 다음 함수를 구현해야 한다. 또한, `Azer.h`를 include해야 한다.

- `void InitA(int N, int A, std::vector<int> U, std::vector<int> V, std::vector<int> C)`

  이 함수는 프로그램 시작시에 정확히 한 번 불린다.

  - 인자 `N`은 도시의 수 N을 나타낸다.
  - 인자 `A`는 전철노선의 수 A를 나타낸다.
  - 인자 `U`, `V`는 길이 A의 배열이다. `U[i]`와 `V[i]`는 전철노선이 연결하는 두 도시 U_i번과 V_i번을 나타낸다. $(0 \leq i \leq A-1)$
  - 인자 `C`는 길이 A의 배열이다. `C[i]`는 i번 전철노선의 운임 C_i를 나타낸다. $(0 \leq i \leq A-1)$

- `void ReceiveA(bool x)` 이 함수는 Baijan으로 부터 문자를 받았을 때 마다 호출 된다.

  - 인자 `x`는 Baijan으로부터 받은 문자를 나타낸다. `true`이면 1, `false`이면 0을 받은 것이다.

- `std::vector<int> Answer()` 이 함수는 보내진 모든 문자를 받았을 때 실행된다. 이 함수는 0번 도시로부터 각 도시간의 최단거리를 담은 배열 `Z`를 반환해야 한다.

  - 배열 `Z`는 길이가 N이어야 한다. 길이가 N이 아닌 경우에, 프로그램은 **오답 [1]**이 된다. `Z[k]` $(0 \leq k \leq N-1)$은 0번 도시 부터 k번 도시까지 가는데 필요한 운임의 최솟값이어야 한다. 특히, `Z[0]` $= 0$ 이어야 함에 유의하여라.

이 프로그램은 다음 함수를 호출 할 수 있다.

- `void sendA(bool y)`

  Baijan에게 문자를 보내려면 이 함수를 사용해야 한다.

  - 인자 `y`는 Baijan에게 보내는 문자를 나타낸다. `true`이면 1, `false`이면 0을 보낸 것이다.

둘째 파일의 이름은 `Baijan.cpp`이다. 이 파일은 Baijan의 일을 나타내고, 다음 함수를 구현해야 한다. 또한, `Baijan.h`를 include해야 한다.

- `void InitB(int N, int B, std::vector<int> S, std::vector<int> T, std::vector<int> D)`

  이 함수는 프로그램 시작시에 정확히 한 번 불린다.

  - 인자 `N`은 도시의 수 N을 나타낸다.
  - 인자 `B`는 버스노선의 수 A를 나타낸다.
  - 인자 `S`, `T`는 길이 B의 배열이다. `S[j]`와 `T[j]`는 전철노선이 연결하는 두 도시 S_j번과 T_j번을 나타낸다. $(0 \leq j \leq B-1)$
  - 인자 `D`는 길이 A의 배열이다. `D[j]`는 i번 버스노선의 운임 D_j를 나타낸다. $(0 \leq j \leq B-1)$

- `void ReceiveB(bool y)` 이 함수는 Azer로 부터 문자를 받았을 때 마다 호출 된다.

  - 인자 `y`는 Azer로부터 받은 문자를 나타낸다. `true`이면 1, `false`이면 0을 받은 것이다.

이 프로그램은 다음 함수를 호출 할 수 있다.

- `void sendB(bool x)`

  Azer에게 문자를 보내려면 이 함수를 사용해야 한다.

  - 인자 `x`는 Azer에게 보내는 문자를 나타낸다. `true`이면 1, `false`이면 0을 보낸 것이다.

당신은 프로그램이 다음과 같은 방법으로 실행된다는 것을 가정해도 좋다. 각 테스트 케이스 마다 Azer가 보낸 문자들을 담는 큐 Q_Y와 Baijan이 보낸 문자들을 담는 큐 Q_X, 두 개의 큐가 준비된다. 처음에, `InitA`와 `InitB`가 실행되고, 보낸 문자들은 각각 큐에 push된다.

- Q_X나 Q_Y가 비어있지 않으면, 문자 하나가 비어있지 않은 큐로 부터 pop되고, 해당하는 `ReceiveA` 혹은 `ReceiveB`가 호출된다. 단, Q_X와 Q_Y가 모두 비어있지 않을 경우에, `ReceiveA`와 `ReceiveB`중 어느쪽이 호출되는지는 결정되어있지 않다.
- `ReceiveA`의 호출 도중에 `SendA`가 실행 된 경우, 보내진 문자는 Q_Y에 push된다.
- `ReceiveB`의 호출 도중에 `SendB`가 실행 된 경우, 보내진 문자는 Q_X에 push된다.
- 두 큐가 모두 빈 경우에는, `Answer`이 호출되고 프로그램이 종료된다.

Azer와 Baijan이 보낸 문자의 총 갯수는 58 000보다 작거나 같아야 한다. 만약 더 큰 경우에는, **오답 [2]**가 된다.

참고 사항

- 당신의 프로그램은 내부에서 사용할 목적으로 함수나 전역변수를 사용할 수 있다. 제출된 파일들은 같이 컴파일 되어 하나의 실행 파일이 된다. 모든 글로벌 변수나 함수는 충돌을 피하기 위하여 이름이 없는 namespace에 구현되어야 한다. 채점 될 때는, Azer과 Baijan에 해당하는 두 프로세스로 나누어서 실행 될 것이다. Azer의 프로세스와 Baijan의 프로세스는 전역변수를 공유할 수 없다.
- 당신의 프로그램은 표준 입출력을 사용해서는 안된다. 당신의 프로그램은 어떠한 방법으로도 다른 파일에 접근해서는 안된다. 단, 당신의 프로그램은 디버그 목적으로 표준 에러출력에 출력할 수 있다.

당신은 대회 홈페이지의 아카이브에서 프로그램을 테스트 하기 위한 목적의 샘플 그레이더를 받을 수 있다. 아카이브는 당신의 프로그램의 예제 소스 또한 첨부되어 있다. 샘플 그레이더는 파일 `grader.cpp`이다. 당신의 프로그램을 테스트 하기 위해서, `grader.cpp`, `Azer.cpp`, `Baijan.cpp`, `Azer.h`, `Baijan.h`를 같은 디렉토리 안에 놓고, 컴파일 하기 위해 다음 커맨드를 실행하여라.

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

컴파일이 성공적이면, 파일 `grader`가 생성된다.

실제 그레이더와 샘플 그레이더는 다름에 주의하여라. 샘플 그레이더는 하나의 프로세스에서 실행 되며, 입력을 표준 입력으로 부터 받고, 출력을 표준 출력에 출력한다.

입력 형식

샘플 그레이더는 표준 입력에서 다음과 같은 형식으로 입력받는다.

N A B

U_0 V_0 C_0

$\vdots$

U_{A-1} V_{A-1} C_{A-1}

S_0 T_0 D_0

$\vdots$

S_{A-1} T_{A-1} D_{A-1}

출력 형식

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

- 오답[1] 혹은 오답 [2]로 판단 된 경우, 오답의 종류를 "`Wrong Answer [1]`"과 같은 형식으로 표준 에러에 출력한다. 표준 출력에는 아무것도 출력되지 않는다.
- 아닌 경우, 보낸 문자의 총 갯수 L을 "`Accepted: L`"과 같은 형식으로 표준 에러에 출력한다. 또한, 표준 출력에 답 `Z`를 다음과 같은 형식으로 출력한다:

  ```
  Z[0]
  ⋮
  Z[N - 1]
  ```

  샘플 그레이더는 `Z`가 올바른지 검사하지 않는다.

프로그램이 다양한 오답의 종류에 속해 있을 경우, 샘플 그레이더는 그 중 하나만 출력 할 것이다.

제한

- $3 \leq N \leq 2\ 000$.
- $0 \leq A \leq 500\ 000$.
- $0 \leq B \leq 500\ 000$.
- $0 \leq U_i \leq N-1$. $(0 \leq i \leq A-1)$
- $0 \leq V_i \leq N-1$. $(0 \leq i \leq A-1)$
- $U_i \neq V_i$. $(0 \leq i \leq A-1)$
- $(U_{i_1},\ V_{i_1}) \neq (U_{i_2},\ V_{i_2})$ 이고, $(U_{i_1},\ V_{i_1}) \neq (V_{i_2},\ U_{i_2})$ 이다. $(0 \leq i_1 < i_2 \leq A-1)$
- $0 \leq S_j \leq N-1$. $(0 \leq j \leq B-1)$

- $0 \leq T_j \leq N-1$. $(0 \leq j \leq B-1)$
- $S_j \neq T_j$. $(0 \leq j \leq B-1)$
- $(S_{j_1},\ T_{j_1}) \neq (S_{j_2},\ T_{j_2})$ 이고, $(S_{j_1},\ T_{j_1}) \neq (T_{j_2},\ S_{j_2})$ 이다. $(0 \leq j_1 < j_2 \leq B-1)$
- 전철 혹은 버스를 사용하면 어떠한 쌍의 도시도 오갈 수 있다.
- $1 \leq C_i \leq 500$. $(0 \leq i \leq A-1)$
- $1 \leq D_j \leq 500$. $(0 \leq j \leq B-1)$

서브태스크 1 (6 점)

- $A = 0$

서브태스크 2 (8 점)

- $B \leq 1\ 000$

서브태스크 3 (8 점)

- $A + B = N - 1$

서브태스크 4 (38 점)

- $N \leq 900$

서브태스크 5 (14 점)

- $N \leq 1\ 100$

서브태스크 6 (10 점)

- $N \leq 1\ 400$

서브태스크 7 (16 점)

추가 제한조건이 없다.

예제

이 함수는 그레이더의 예제 입력과 해당하는 함수 호출을 보여준다.

예제 입력	예제 함수 호출		
	호출	호출	반환값
4 3 4 0 1 6 2 1 4 2 0 10 1 2 3 3 1 1 3 2 3 3 0 7	`InitA(4, 3, {0,2,2}, {1,1,0}, {6,4,10})`		
		`SendA(true)`	
		`SendA(false)`	
	`InitB(4, 4, {1,3,3,3}, {2,1,2,0}, {3,1,3,7}`		
	`ReceiveB(true)`		
		`SendB(true)`	
	`ReceiveA(true)`		
	`ReceiveB(false)`		
	`Answer()`		{0,6,9,7}