Kereta

Pada tahun 2992, sebagian besar pekerjaan sudah diambil alih oleh robot. Oleh karena itu, orang-orang memiliki banyak waktu luang, tak terkecuali keluarga Anda, yang baru saja memutuskan untuk melakukan perjalanan antarbintang!

Terdapat N planet yang dapat dikunjungi yang dinomori dari 0 hingga $N-1$ dan M rute kereta antarbintang. Rute kereta i ($0 \le i < M$) berangkat dari Planet $X[i]$ pada waktu $A[i]$, tiba di Planet $Y[i]$ pada waktu $B[i]$, dan dengan biaya $C[i]$. Kereta adalah satu-satunya moda perjalanan antar planet, jadi Anda hanya dapat turun dari kereta di planet tujuannya dan harus naik kereta selanjutnya di planet yang sama (pindah kereta tidak memakan waktu). Secara formal, sebuah urutan kereta $q[0], q[1], ..., q[P]$ dikatakan valid untuk diambil jika dan hanya jika untuk setiap $1 \le k \le P$, $Y[q[k-1]] = X[q[k]]$ dan $B[q[k-1]] \le A[q[k]]$.

Karena perjalanan antarbintang memakan waktu, Anda menyadari bahwa tidak hanya tarif kereta, tapi harga makanan juga perlu diperhatikan. Untungnya, **kereta antarbintang menyediakan makanan yang tak terbatas secara gratis**. Artinya, jika Anda memutuskan untuk menggunakan rute kereta i, maka kapan saja di antara $A[i]$ dan $B[i]$ **(inklusif)** Anda dapat mengambil sebanyak-banyaknya makanan tanpa biaya. Namun, ketika keluarga Anda sedang menunggu kereta selanjutnya pada suatu Planet i, Anda harus membayar setiap makanan dengan harga $T[i]$.

Keluarga Anda memerlukan W makanan, dan makanan ke-i ($0 \le i < W$) bisa dimakan kapan saja pada interval waktu $L[i]$ dan $R[i]$ **(inklusif)**. Makanan akan dimakan **dengan sekejap**.

Sekarang pada waktu 0, keluarga Anda berada di Planet 0. Anda harus mencari biaya minimum untuk tiba di Planet $N-1$. Apabila Anda tidak bisa mencapainya, jawaban Anda haruslah -1.

Detail Implementasi

Anda harus mengimplementasikan fungsi berikut:

```
long long solve(int N, int M, int W, std::vector<int> T,
                std::vector<int> X, std::vector<int> Y,
                std::vector<int> A, std::vector<int> B, std::vector<int> C,
                std::vector<int> L, std::vector<int> R);
```

- N: Banyaknya planet.
- M: Banyaknya rute kereta antarbintang.

- W: Banyaknya makanan.
- T: *Array* dengan panjang N. $T[i]$ merepresentasikan harga seporsi makanan di planet i.
- X, Y, A, B, C: Lima *array* dengan panjang M. Tupel $(X[i], Y[i], A[i], B[i], C[i])$ mendeskripsikan rute Kereta i.
- L, R: Dua *array* dengan panjang W. Pasangan $(L[i], R[i])$ mendeskripsikan interval waktu untuk makan makanan ke-i.
- Fungsi ini harus mengembalikan biaya minimum untuk tiba di Planet $N-1$ dari Planet 0 apabila Anda bisa tiba di Planet $N-1$, atau -1 apabila sebaliknya.
- Untuk setiap kasus uji, fungsi ini dipanggil tepat satu kali.

Contoh

Contoh 1

Perhatikan pemanggilan berikut:

```
solve(3, 3, 1, {20, 30, 40}, {0, 1, 0}, {1, 2, 2},
        {1, 20, 18}, {15, 30, 40}, {10, 5, 40}, {16}, {19});
```

Salah satu cara untuk tiba di Planet $N-1$ adalah dengan naik Kereta 0 lalu Kereta 1, dengan harga 45 (detail perhitungannya adalah sebagai berikut).

Waktu	Aksi	Biaya (jika ada)
1	Naik Kereta 0 di Planet 0	10
15	Tiba di Planet 1	
16	Makan makanan ke-0 di Planet 1	30
20	Naik Kereta 1 di Planet 1	5
30	Tiba di Planet 2	

Cara yang lebih murah untuk tiba di Planet $N-1$ adalah dengan naik Kereta 2 saja, dengan harga 40 (detail perhitungannya adalah sebagai berikut).

Waktu	Aksi	Biaya (jika ada)
18	Naik Kereta 2 di Planet 0	40
19	Makan makanan ke-0 di Kereta 2	
40	Tiba di Planet 2	

Dengan cara untuk tiba di Planet $N-1$ ini, valid juga untuk makan makanan ke-0 pada waktu 18.

Oleh karena itu, fungsi harus mengembalikan 40.

Contoh 2

Perhatikan pemanggilan berikut:

```
solve(3, 5, 6, {30, 38, 33}, {0, 1, 0, 0, 1}, {2, 0, 1, 2, 2},
        {12, 48, 26, 6, 49}, {16, 50, 28, 7, 54}, {38, 6, 23, 94, 50},
        {32, 14, 42, 37, 2, 4}, {36, 14, 45, 40, 5, 5});
```

Jalur optimalnya adalah dengan naik Kereta 0 dengan biaya 38. Makanan ke-1 dapat dimakan secara gratis di Kereta 0. Makanan ke-0, 2, dan 3 harus dimakan di Planet 2 dengan biaya $33 \times 3 = 99$. Makanan ke-4 dan 5 harus dimakan pada Planet 0 dengan biaya $30 \times 2 = 60$. Total biayanya adalah $38 + 99 + 60 = 197$.

Oleh karena itu, fungsi harus mengembalikan 197.

Batasan

- $2 \le N \le 10^5$.
- $0 \le M, W \le 10^5$.
- $0 \le X[i], Y[i] < N, X[i] \ne Y[i]$.
- $1 \le A[i] < B[i] \le 10^9$.
- $1 \le T[i], C[i] \le 10^9$.
- $1 \le L[i] \le R[i] \le 10^9$.

Subsoal

1. (5 poin): $N, M, A[i], B[i], L[i], R[i] \le 10^3$ dan $W \le 10$.
2. (5 poin): $W = 0$.
3. (30 poin): Tidak ada dua interval waktu makan yang tumpang tindih. Secara formal, untuk waktu z yang memenuhi $1 \le z \le 10^9$, terdapat paling banyak satu i ($0 \le i < W$) yang memenuhi $L[i] \le z \le R[i]$.
4. (60 poin): Tidak ada batasan tambahan.

Contoh *Grader*

Contoh *grader* membaca masukan dengan format berikut:

- Baris 1: $N\ M\ W$
- Baris 2: $T[0]\ T[1]\ T[2]\ \cdots\ T[N-1]$
- Baris $3 + i$ ($0 \le i < M$): $X[i]\ Y[i]\ A[i]\ B[i]\ C[i]$
- Baris $3 + M + i$ ($0 \le i < W$): $L[i]\ R[i]$

Contoh *grader* mencetak keluaran dengan format berikut:

- Baris 1: nilai kembalian dari `solve`