Межзвёздные поезда

В 2992 году роботы взяли на себя большую часть работы. Поэтому у многих людей появилось свободное время, так что ваша семья решила отправить вас в межзвёздное путешествие!

Всего на данный момент есть N достижимых планет, пронумерованных последовательными целыми числами от 0 до $N-1$ и M маршрутов межзвёздных поездов. Маршрут поезда i ($0 \leq i < M$) начинается с планеты $X[i]$ в момент времени $A[i]$, прибывает на планету $Y[i]$ в момент времени $B[i]$ и стоит $C[i]$. Межзвёздные поезда являются единственным способом перемещения между планетами, таким образом вы можете сойти с поезда только на конечной планете и должны ехать следующим поездом с этой же планеты (пересадку можно совершить сколь угодно быстро). Формально, вы можете проехать на последовательности поездов $q[0], q[1], ..., q[P]$ тогда и только тогда, когда для любого $1 \leq k \leq P$, $Y[q[k-1]] = X[q[k]]$ и $B[q[k-1]] \leq A[q[k]]$.

Межзвёздные путешествия занимают много времени, так что вы обнаружили, что в дополнение к ценам на билеты, вы вынуждены тратить деньги на питание. К счастью **питание на борту межзвёзднного поезда является бесплатным**, поэтому если вы решили использовать маршрут поезда i, тогда в любое время между $A[i]$ и $B[i]$ **(включительно)** вы можете брать любое количество обедов бесплатно. Однако, если вы ожидаете следующего поезда на планете i, то стоимость одного обеда равна $T[i]$.

Вы должны пообедать W раз, и i-й ($0 \leq i < W$) обед происходит **мгновенно** в любой момент времени между моментами времени $L[i]$ и $R[i]$ **(включительно)**.

В момент времени 0 вы находитесь на планете 0. Вы должны найти минимальную цену, за которую вы сможете добраться до планеты $N-1$. Если до соответствующей планеты добраться невозможно, ответ должен быть -1. Обратите внимание, что необходимо будет пообедать ровно W раз, даже если вы добрались до планеты $N-1$ раньше, чем время начала какого-то обеда.

Детали реализации

Вы должны реализовать следующую функцию:

```
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: Количество планет.
- M: Количество маршрутов межзвёздных поездов.
- W: Количество раз, когда вы должны пообедать.
- T: Массив длины N. $T[i]$ задаёт стоимость обеда на планете i.
- X, Y, A, B, C: пять векторов, каждый из которых имеет длину M. Пятёрка чисел $(X[i], Y[i], A[i], B[i], C[i])$ задаёт маршрут i-го поезда.
- L, R: Два вектора длины W. Пара чисел $(L[i], R[i])$ задаёт интервал времени для i-го обеда.
- Функция должна вернуть минимальную стоимость путешествия на планету $N-1$ с планеты 0, если вы можете добраться до планеты $N-1$, и -1, если добраться невозможно.
- Для каждого тестового примера функция будет вызвана ровно один раз.

Примеры

Пример 1

Рассмотрим следующий вызов:

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

Один способ для того, чтобы добраться до планеты $N-1$ — поехать на поезде 0, а затем на поезде 1, что в итоге приведёт к затратам 45 (таблица расходов ниже).

Время	Действие	Затраты (если есть)
1	Поездка поездом 0 с планеты 0	10
15	Прибытие на планету 1	
16	Обед 0 на планете 1	30
20	Поездка поездом 1 с планеты 1	5
30	Прибытие на планету 2	

Более экономно, однако, прибыть на планету $N-1$ — поехать на поезде 2, потратив только 40 (таблица расходов ниже).

Время	Действие	Затраты (если есть)
18	Поездка поездом 2 с планеты 0	40
19	Обед 0 на поезде 2	
40	Прибытие на планету 2	

В этом способе прибыть на планету $N-1$ также возможно съесть обед 0 в момент времени 18.

Таким образом, функция должна вернуть 40.

Пример 2

Рассмотрим следующий вызов:

```
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});
```

Оптимальным будет поехать поездом 0 за 38. Обед 1 будет бесплатным на борту поезда 0. Обеды 0, 2 и 3 будут на планете 2 с затратами $33 \times 3 = 99$. Обеды 4 и 5 будут на планете 0 с затратами $30 \times 2 = 60$. Общая стоимость равна $38 + 99 + 60 = 197$.

Таким образом, функция должна вернуть 197.

Ограничения

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

Подзадачи

1. (5 баллов): $N, M, A[i], B[i], L[i], R[i] \leq 10^3$ и $W \leq 10$.
2. (5 баллов): $W = 0$.
3. (30 баллов): Времена на обеды не перекрываются по времени. Формально, для каждого момента времени z, где $1 \leq z \leq 10^9$, существует не более одного такого i ($0 \leq i < W$), что $L[i] \leq z \leq R[i]$.
4. (60 баллов): Нет дополнительных ограничений.

Грейдер для участника

Предоставляемый участнику грейдер принимает входные данные в следующем формате:

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

Предоставляемый участнику грейдер выводит данные в следующем формате:

- Строка 1: возвращаемое написанной участником функцией `solve` значение.