Задача 4. Путешествие в Метрополис

Имя входного файла:	`trains.in`
Имя выходного файла:	`trains.out`
Ограничение по времени:	4 секунды
Ограничение по памяти:	512 мегабайт

Путешествие по стране никогда не бывает простым, особенно когда не существует прямого сообщения между городами. Группа туристов хочет добраться в город Метрополис, используя сеть железных дорог, которая соединяет n городов, пронумерованных от 1 до n. Город, из которого выезжает группа, имеет номер 1, Метрополис имеет номер n.

На железной дороге постоянно функционируют m маршрутов поездов. Каждый маршрут определяется последовательностью городов, перечисленных в том порядке, в каком их проезжает поезд, обслуживающий этот маршрут. В каждом маршруте для каждой пары соседних городов задано время, за которое поезд этого маршрута проезжает перегон между этими городами. При этом поезда разных маршрутов могут проезжать один и тот же перегон за разное время.

По пути в Метрополис группа может садиться на поезд и сходить с поезда в любом городе маршрута, не обязательно в начальном или конечном. При этом, можно сойти с поезда маршрута i, пересесть на поезд маршрута j, возможно сделать еще несколько пересадок, а потом вновь сесть в поезд того же маршрута i.

Туристы предъявляют высокие требования к выбору способа проезда в Метрополис.

Во-первых, суммарное время, проведенное в поездах, должно быть минимальным.

Во-вторых, среди всех способов с минимальным временем нахождения в поездах предпочтительным является тот способ, для которого сумма квадратов промежутков времени, непрерывно проведенных в поезде между двумя пересадками, максимальна. Назовём эту сумму *качеством путешествия*.

Время, проведенное вне поездов, не учитывается.

Требуется написать программу, которая по описаниям имеющихся маршрутов поездов определит минимальное время, которое группе туристов придется провести в поездах, а также максимальное качество путешествия с таким временем.

Формат входных данных

В первой строке входных данных заданы два целых числа ($2 \leqslant n \leqslant 10^6$, $1 \leqslant m \leqslant 10^6$) — количество городов и количество маршрутов соответственно.

Далее в m строках содержится описание маршрутов.

Описание каждого маршрута начинается с целого числа s_i — количество перегонов в маршруте с номером i ($1 \leqslant s_i \leqslant 10^6$). Далее следуют $(2s_i + 1)$ целых чисел, описывающих города маршрута и время проезда перегона между соседними городами маршрута, в следующем порядке: $v_{i,1}$, $t_{i,1}$, $v_{i,2}$, $t_{i,2}$, $v_{i,3}$, …, t_{i,s_i}, v_{i,s_i+1}, где $v_{i,j}$ — номер j-го города маршрута, $t_{i,j}$ — время проезда перегона между j-м и $(j + 1)$-м городом ($1 \leqslant v_{i,j} \leqslant n$, $1 \leqslant t_{i,j} \leqslant 1000$).

Гарантируется, что $s_1+s_2+\ldots+s_m \leqslant 10^6$. Никакие два города в описании маршрута не совпадают. Гарантируется, что с помощью имеющихся маршрутов можно добраться из города с номером 1 в город с номером n.

Формат выходных данных

Выходные данные должны содержать два целых числа — минимальное суммарное время, которое придется провести в поездах, и максимальное качество пути с таким временем.

Примеры

trains.in	trains.out
2 1 1 1 3 2	3 9
5 2 4 1 3 2 3 3 5 5 10 4 3 4 2 2 1 3 4 1	9 35
5 2 3 1 1 2 2 3 3 4 3 2 2 3 3 4 4 5	10 82

Замечание

В первом примере группа туристов отправится прямым маршрутом в Метрополис.

Во втором примере не оптимально проехать напрямую по первому маршруту, так как время в поезде при этом не будет минимальным возможным. Поэтому они отправятся на поезде по маршруту 1 из города 1 в город 2, затем на поезде по маршруту 2 из города 2 в город 3, а затем снова на поезде по маршруту 1 из города 3 в город 5. При этом сумма квадратов промежутков времени, проведенных в поездах между пересадками, равна $3^2 + 1^2 + 5^2 = 35$.

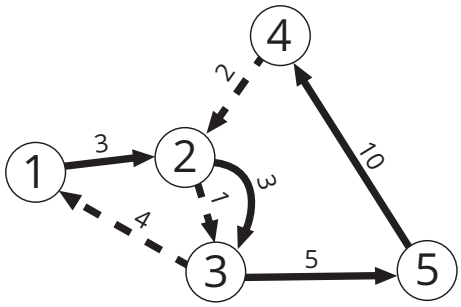

Иллюстрация ко второму примеру.

В третьем примере добраться из города 1 в город 4 за минимальное время можно, пересаживаясь с маршрута 1 на маршрут 2 в любом из городов 2, 3 или 4. Максимальное качество путешествия достигается при пересадке в городе 2: $1^2 + 9^2 = 82$.

Обратите внимание, что второй и третий примеры не удовлетворяют ограничениям первой и второй подзадачи, решение будет протестировано на этих подзадачах, если оно пройдет первый тест из примера. Все тесты из примера подходят под ограничения подзадач 3 – 7, решение будет проверяться на тестах этих подзадач только в случае прохождения всех тестов из примера.

Система оценивания

Подзадача	Баллы	Ограничения		Необходимые подзадачи	Результаты во время тура
		n	s_i		
1	10	$n \leqslant 10$	$\sum s_i \leqslant 20,\ s_i = 1$		Потестовые
2	10	$n \leqslant 1\,000$	$\sum s_i \leqslant 1\,000,\ s_i = 1$	1	Потестовые
Подзадачи, начиная с 3, тестируются только при прохождении *всех* тестов из примера					
3	17	$n \leqslant 1\,000$	$\sum s_i \leqslant 1\,000$	1, 2	Потестовые
4	17	$n \leqslant 1\,000$	$\sum s_i \leqslant 100\,000$	1 – 3	Потестовые
5	19	$n \leqslant 10\,000$	$\sum s_i \leqslant 200\,000$	1 – 4	Баллы
6	19	$n \leqslant 200\,000$	$\sum s_i \leqslant 200\,000$	1 – 5	Баллы
7	8	$n \leqslant 10^6$	$\sum s_i \leqslant 10^6$	1 – 6	Баллы