Задача A. Калила и Димна на лесозаготовках

Имя входного файла:	`lumber.in`
Имя выходного файла:	`lumber.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Спонсор сегодняшней задачи — codeforces round 189. Codeforces — мечты сбываются!

Калила и Димна — два шакала. Они живут в огромных джунглях. Однажды шакалы решили устроиться на завод лесозаготовки и подработать.

Управляющий завода хочет, чтобы они отправились в джунгли и срубили n деревьев высотой $a_1, a_2, \ldots, a_n$. Для этого Калила и Димна купили цепную пилу в магазине. Каждый раз, когда они используют пилу на дереве номер i, они уменьшают высоту этого дерева на единицу. Каждый раз Калила и Димна должны заправить пилу для использования. Цена заправки зависит от того, какие деревья полностью спилены (дерево считается полностью спиленным, если его высота равна 0). Если максимальный идентификатор полностью срубленного дерева равняется i (первоначально это дерево имело высоту a_i), то цена заправки пилы равняется b_i. Если ни одно дерево не срублено полностью, то заправлять пилу запрещается. Изначально пила заправлена. Известно, что для каждого $i < j$, $a_i < a_j$ и $b_i > b_j$, а также $b_n = 0$ и $a_1 = 1$.

Калила и Димна хотят полностью срубить все деревья с минимальными затратами. Они ждут Вашей помощи! Поможете?

Формат входных данных

В первой строке записано целое число n ($1 \leqslant n \leqslant 10^5$). Во второй строке записано n целых чисел $a_1, a_2, \ldots, a_n$ ($1 \leqslant a_i \leqslant 10^9$). В третьей строке записано n целых чисел $b_1, b_2, \ldots, b_n$ ($0 \leqslant b_i \leqslant 10^9$).

Гарантируется, что $a_1 = 1$, $b_n = 0$, $a_1 < a_2 < \cdots < a_n$ и $b_1 > b_2 > \cdots > b_n$.

Формат выходных данных

В единственной строке должна быть записана минимальная стоимость вырубания всех деревьев.

Примеры

lumber.in	lumber.out
5 1 2 3 4 5 5 4 3 2 0	25
6 1 2 3 10 20 30 6 5 4 3 2 0	138

Задача B. Ciel и гондолы

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	4 секунды
Ограничение по памяти:	256 мегабайт

Лиса Ciel зашла в парк аттракционов. И вот, она в очереди на колесо обозрения. В очереди стоит n людей (хотя нет, скорее лис): мы будем считать, что первая лиса стоит в начале очереди, а n-я лиса стоит в хвосте очереди.

Всего имеется k гондол, мы распределяем лис по гондолам следующим образом:

- Когда подплывает первая гондола, q_1 лис переходят из начала очереди в подплывшую гондолу.
- Затем, когда подплывает вторая гондола, q_2 лис из начала оставшейся очереди переходит в эту гондолу.

  ...
- Оставшиеся q_k лис идут с последнюю (k-ю) гондолу.

Обратите внимание, что числа q_1, q_2, ..., q_k должны быть положительные. Из условия следует, что $\sum_{i=1}^{k} q_i = n$ и $q_i > 0$.

Вы знаете как лисам не хочется задерживаться в гондолах с незнакомцами. Итак, Ваша задача — найти оптимальный способ размещения (то есть определить оптимальную последовательность q), чтобы угодить всем. Для каждой пары лис i и j задано значение u_{ij}, обозначающее степень незнакомости. Можете считать, что $u_{ij} = u_{ji}$ для всех i, j ($1 \leqslant i, j \leqslant n$) и что $u_{ii} = 0$ для всех i ($1 \leqslant i \leqslant n$). Тогда значение незнакомости в гондоле определяется как сумма значений незнакомости между всеми парами лис, которые находятся в этой гондоле. Общее значение незнакомости определяется как сумма значений незнакомости по всем гондолам.

Помогите лисе Ciel найти минимальное возможное значение общей незнакомости при некотором оптимальном распределении лис по гондолам.

Формат входных данных

В первой строке даны два целых числа n и k ($1 \leqslant n \leqslant 4000$ and $1 \leqslant k \leqslant min(n, 800)$) — количество лис в очереди и количество гондол. В следующих n строках записано по n целых чисел — матрица u, ($0 \leqslant u_{ij} \leqslant 9$, $u_{ij} = u_{ji}$ и $u_{ii} = 0$).

Пожалуйста, используйте методы быстрого чтения (например, для Java используйте BufferedReader вместо Scanner).

Формат выходных данных

Выведите целое число — минимальное возможное значение общей незнакомости.

Примеры

стандартный ввод	стандартный вывод
5 2 0 0 1 1 1 0 0 1 1 1 1 1 0 0 0 1 1 0 0 0 1 1 0 0 0	0
8 3 0 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 0	7

Замечание

В первом примере можно распределить лис вот так: 1, 2 идут в одну гондолу, 3, 4, 5 идут в другую гондолу.

Во втором примере оптимальное распределение таково: 1, 2, 3 | 4, 5, 6 | 7, 8.

Задача C. Коды, сохраняющие порядок

Имя входного файла:	`codes.in`
Имя выходного файла:	`codes.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Двоичный код — это код, где каждому символу сопоставляется последовательность из единиц и нулей. Код называется префиксным, если ни одно кодовое слово не является префиксом другого. Код называется сохраняющим порядок, если лексикографический порядок кодовых слов совпадает с алфавитным порядком символов.

Рассмотрим текст над алфавитом, содержащим n символов, в котором a_1 раз встречается первый символ, a_2 раз встречается второй символ, $\cdots$, a_n раз встречается n-й символ. Длина текста после кодирования его префиксным кодом, где первому символу сопоставлена строка длины l_1, второму — строка длины l_2, и т. д., будет равна $a_1 \cdot l_1 + a_2 \cdot l_2 + \cdots + a_n \cdot l_n$.

Требуется найти сохраняющий порядок префиксный код, минимизирующий длину закодированного текста.

Формат входных данных

Первая строка содержит число n — число символов в алфавите ($2 \leqslant n \leqslant 2000$). Следующая строка содержит n целых чисел — сколько раз каждый символ встречается в тексте: $a_1, a_2, \cdots, a_n$. Числа положительные и не превосходят 10^9.

Формат выходных данных

Выведите n двоичных последовательностей — искомый код.

Примеры

codes.in	codes.out
5 1 8 2 3 1	00 01 10 110 111

Задача D. Серверы

Имя входного файла:	`server.in`
Имя выходного файла:	`server.out`
Ограничение по времени:	2.5 секунд
Ограничение по памяти:	256 мегабайт

Компьютерная сеть в некотором доме строилась по принципу присоединения нового компьютера к последнему из уже подключенных. Никакие два компьютера, будучи подключенными в сеть, между собой дополнительно никак не связывались. Таким образом, в сеть были объединены последовательно N компьютеров. Соседи обменивались информацией между собой, но в какой-то момент поняли, что им нужны прокси-серверы. Компьютерное сообщество дома решило установить прокси-серверы ровно на K компьютеров. Осталось только решить, какие именно компьютеры выбрать для этой цели. Главным критерием является ежемесячная стоимость обслуживания серверами всех компьютеров.

Для каждого компьютера установлен тариф его обслуживания, выраженный в рублях за метр провода. Стоимость обслуживания одного компьютера каким-то сервером равна тарифу компьютера, умноженному на суммарную длину провода от этого компьютера до сервера, которым он обслуживается.

Ваша задача написать программу, которая выберет такие K компьютеров, чтобы установить на них прокси-серверы, что общие затраты на обслуживание всех компьютеров были бы минимальными.

Формат входных данных

В первой строке входного файла записано два целых числа N и K — количество компьютеров в сети и количество прокси-серверов, которые нужно установить ($1 \leqslant K \leqslant N \leqslant 2000$).

Все компьютеры в сети пронумерованы числами от 1 до N по порядку подключения.

Во второй строке записано одно целое число T_1 — тариф обслуживания первого компьютера.

В следующих $N - 1$ строках записано через пробел по два целых неотрицательных числа L_i, T_i — информация об остальных компьютерах в сети по порядку номеров. L_i — длина провода, соединяющего i-й компьютер с соседним с меньшим номером, T_i — тариф обслуживания данного компьютера ($2 \leqslant i \leqslant N$). Все L_i и T_i от 0 до 10^6.

Формат выходных данных

В первую строку выходного файла необходимо вывести одно целое число — минимальную стоимость обслуживания всех компьютеров всеми серверами. Во второй строке должны быть записаны через пробел K номеров компьютеров, на которые необходимо установить серверы (номера разрешается выводить в любом порядке). При существовании нескольких вариантов размещения разрешается вывести любой.

Примеры

server.in	server.out
3 1 10 2 2 3 3	19 1
3 2 10 2 2 3 3	4 1 3

Задача E. Петя и прямоугольники

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Маленький Петя очень любит прямоугольники. Петя дал маме список прямоугольников, которые он хочет получить в подарок на Новый Год. Каждый прямоугольник характеризуется w и высотой h. Мама хочет сделать Пете приятное и купить все прямоугольники из его списка. Мама отправилась в магазин и узнала, что цена одного прямоугольника равна его площади. К ее счастью, в магазине действует предновогодняя акция, позволяющая покупать прямоугольники не по одному, а сразу наборами. Стоимость одного набора равна ширине самого широкого прямоугольника, умноженной на высоту самого высокого прямоугольника из этого набора. Обратите внимание, что поворачивать прямоугольники (тем самым меняя местами ширину и высоту) нельзя. Помогите маме Пети купить все прямоугольники из списка ее сына, потратив на это наименьшее количество денег.

Формат входных данных

В первой строке записано число N ($1 \leqslant N \leqslant 200\,000$) — количество прямоугольников в списке Пети. В каждой из следующих N строк записаны по 2 целых положительных числа, не превышающих 10^6 — ширина и высота очередного прямоугольника.

Формат выходных данных

Выведите одно число — наименьшее количество денег, которое может потратить мама чтобы купить Пете все прямоугольники из его списка.

Примеры

стандартный ввод	стандартный вывод
4 100 1 15 15 20 5 1 100	500
5 1 10 2 20 3 30 4 40 10 1	170

Задача F. Кафе на набережной

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Вдоль набережной одного города расположено N кафе. Они пронумерованы от 1 до N с запада на восток, а расстояние между кафе i и $i+1$ равно D_i.

У Саши есть K различных купонов на посещение кафе. Кафе под номером i готово в обмен на купон j дать блюдо *аппетитностью* $A_{i,j}$. Каждый купон можно использовать ровно один раз, но в одном кафе можно использовать сколько угодно купонов.

Саша хочет использовать все K купонов, начав в любом из кафе. Он планирует ходить по кафе и использовать оставшиеся купоны, приятно проводя вечер. Его итоговое *счастье* от проведенного вечера можно измерить как: «Суммарная аппетитность съеденных блюд» минус «Пройденное за вечер расстояние».

Помогите Саше вычислить, какое максимально возможное счастье он может получить за вечер, начав свой путь в любом из кафе.

Формат входных данных

В первой строке находятся два числа: N ($2 \leqslant N \leqslant 5000$) — число кафе на набережной и K ($1 \leqslant K \leqslant 200$) — число купонов.

Во второй строке перечислено $N-1$ число D_i — расстояние от кафе i до кафе $i+1$ ($1 \leqslant D_i \leqslant 10^9$).

В каждой из следующих N строк перечислены K чисел $A_{i,j}$ — аппетитности блюд в $i-$ом кафе ($1 \leqslant A_{i,j} \leqslant 10^9$).

Формат выходных данных

Выведите единственное число — максимально возможное счастье за вечер.

Примеры

стандартный ввод	стандартный вывод
3 4 1 10 3 1 10 1 1 5 3 7 1 2 3 4	24
6 3 1 2 3 4 5 10 2 2 2 2 2 2 10 2 2 2 2 2 2 10 2 2 2	20

Замечание

В первом примере можно съесть в первом кафе блюда по купонам 1 и 3, а во втором кафе блюда по купонам 2 и 4, тогда счастье будет равно $(3 + 10 + 5 + 7) - (1) = 24$.

Задача G. Путешествие в Метрополис

Имя входного файла:	`trains.in`
Имя выходного файла:	`trains.out`
Ограничение по времени:	2 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 предпочти- тельным является тот способ, для которого сумма квадратов промежутков времени, непрерывно проведенных в поезде между двумя пересадками, максимальна. Назовём эту сумму качеством пу- тешествия. 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 ($2 \leqslant n \leqslant 10^4, 1 \leqslant m \leqslant 2 \cdot 10^5$) — 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 vi, j — номер j-го города маршрута, ti, 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 $s_1 + s_2 + ... + s_m \leqslant 10^6$. 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 $3 \cdot 2 + 1 \cdot 2 + 5 \cdot 2 = 35$. 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: $1c \cdot 2 + 9 \cdot 2 = 82$.

Задача H. Жоские диски

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

У вас есть кластер. В нем есть n пар жестких дисков (HDD). Каждый диск может быть представлен как точка с целой координатой на бесконечной прямой. В каждой паре один из HDD главный, а второй — запасной.

Вы хотите настроить k компьютеров, которые тоже представляются как точки с целыми координатами на той же прямой, после чего соединить все диски с компьютерами при помощи проводов. В конечном итоге, основной и запасной диски из одной пары должны быть соединены с одним и тем же компьютером (и только с ним), но каждый компьютер может быть соединен с любым количеством дисков (возможно с нулем). Каждый провод соединяет один HDD с одним компьютером, и длина провода равна расстоянию между соответствующими точками на прямой. Необходимо найти минимальную суммарную длину всех проводов.

Формат входных данных

В первой строке содержатся два целых числа n и k — количество пар дисков и количество компьютеров, соответственно ($2 \leqslant k \leqslant n \leqslant 100\,000$; $4 \leqslant k \times n \leqslant 100\,000$). Каждая из следующих n строк содержат по два целых числа a_i и b_i — координаты основного и запасного дисков ($-10^9 \leqslant a_i, b_i \leqslant 10^9$).

Формат выходных данных

Вывод должен содержать ровно одно число — ответ на задачу.

Примеры

стандартный ввод	стандартный вывод
5 2 6 7 -1 1 0 1 5 2 7 3	13

Замечание

В примере оптимальные позиции компьютеров — 0 и 6. Соединим вторую и третью пары дисков с первым компьютером, а остальные со вторым. Тогда суммарная длина проводов, ведущих к первому компьютеру, будет 3, и 10 для второго, что в сумме дает 13.

Задача I. Оптимальное бинарное дерево поиска

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Рассмотрим множество $S = \{e_1, e_2, \ldots, e_n\}$, состоящее из n различных элементов таких, что $e_1 < e_2 < \ldots < e_n$. Рассмотрим бинарное дерево поиска, состоящее из элементов S. Чем чаще производится запрос к элементу, тем ближе он должен располагаться к корню. Стоимостью $cost$ доступа к элементу e_i из S в дереве будем называть значение $cost(e_i)$, равное числу ребер на пути, который соединяет корень с вершиной, содержащей элемент. Имея частоту запросов к элементам из S — $f(e_1), f(e_2), \ldots, f(e_n)$ — определим общую стоимость дерева следующим образом:

$$\sum_{i=1}^{n} f(e_i) \cdot cost(e_i)$$

Дерево, имеющее наименьшую стоимость, считается наилучшим для поиска элементов из S. Именно поэтому оно называется Оптимальным Бинарным Деревом Поиска. Ваша задача — найти стоимость Оптимального Бинарного Дерева Поиска.

Формат входных данных

Состоит из нескольких тестов, каждый из которых расположен в отдельной строке. Первое число в строке $n (1 \leqslant n \leqslant 5000)$ указывает на размер множества S. Следующие n неотрицательных целых чисел описывают частоты запросов элементов из S: $f(e_1), f(e_2), \ldots, f(e_n)$. Известно, что $0 \leqslant f(e_i) \leqslant 100$. Сумма n по всем тестам не больше 5000.

Формат выходных данных

Для каждого теста в отдельной строке выведите стоимость Оптимального Бинарного Дерева Поиска.

Пример

стандартный ввод	стандартный вывод
1 5 3 10 10 10 3 5 10 20 6 1 3 5 10 20 30	0 20 20 63