Содержание

Вы не умеете читать/выводить данные, открывать файлы? Воспользуйтесь примерами.

В некоторых задачах большой ввод и вывод. Пользуйтесь быстрым вводом-выводом.

В некоторых задачах нужен STL, который активно использует динамическую память (set-ы, map-ы) переопределение стандартного аллокатора ускорит вашу программу.

Обратите внимание на компилятор `GNU C++11 5.1.0 (TDM-GCC-64) inc`, который позволяет пользоваться дополнительной библиотекой. Под ним можно сдать вот это.

Разделяй и властвуй

Задача A. Деление многочленов [1 sec, 256 mb]

Даны два многочлена с коэффициентами из $\mathbb{Z}/7\mathbb{Z}$. Старший коэффициент обоих не равен нулю. Нужно поделить их с остатком.

Формат входных данных

Каждая из двух строк задаёт описание многочлена. Многочлен $a_k x^k + \cdots + a_2 x^2 + a_1 x + a_0$ описывается числом k ($0 \leqslant k \leqslant 50\,000$) и $k+1$ числами от 0 до 6: $a_k, \ldots, a_2, a_1, a_0$.

Формат выходных данных

На первой строке многочлен-частное. На второй строке многочлен-остаток. Выводите многочлены в том же формате. Если многочлен – тождественный ноль, для него $k = 0$.

Примеры

divpoly.in	divpoly.out
3 1 1 1 1 1 1 1	2 1 0 1 0 0
3 1 1 3 1 2 1 1 1	1 1 0 1 2 1
8 2 1 2 1 2 1 2 1 2 4 1 2 3 4 5	4 2 4 2 5 2 3 3 1 3 6

Суффиксный автомат

Задача B. Суффиксный автомат [1 sec, 256 mb]

Дана строка. Постройте её суффиксный автомат.

Формат входных данных

Строка длины от 1 до 100 000, состоящая из маленьких латинских букв.

Формат выходных данных

На первой строке число состояний автомата и число рёбер. Следующие строки содержат рёбра в формате “откуда” “куда” “символ на ребре”. Далее число терминальных состояний и строка, содержащая все терминальные состояния в произвольном порядке. Начальным состоянием автомата должно быть состояние номер один.

Примеры

automaton.in	automaton.out
ababb	7 9 1 2 a 1 7 b 2 3 b 3 4 a 3 6 b 4 5 b 5 6 b 7 4 a 7 6 b 3 6 7 1

Задача C. Помогите, спасите! [1 sec, 256 mb]

Дана строка. Найдите для каждого её префикса количество различных подстрок в нём.

Формат входных данных

В единственной строке входных данных содержится непустая строка S, состоящая из N ($1 \le N \le 2 \cdot 10^5$) маленьких букв английского алфавита.

Формат выходных данных

Выведите N строк, в i-й строке должно содержаться количество различных подстрок в i-м префиксе строки S.

Примеры

keepcounted.in	keepcounted.out
aabab	1 2 5 8 11
atari	1 3 5 9 14

Задача D. Подстроки-4 [1 sec, 256 mb]

Даны K строк из маленьких латинских букв. Найдите их наибольшую общую подстроку.

Формат входных данных

В первой строке число K ($1 \leqslant K \leqslant 10$). Далее K строк длины от 1 до 200 000.

Формат выходных данных

Наибольшая общая подстрока.

Примеры

substr4.in	substr4.out
3 abacaba mycabarchive acabistrue	cab

Задача E. LZSS encoding [2 sec, 256 mb]

Алиса хочет отправить сообщение Бобу. Она хочет зашифровать сообщение, используя оригинальный метод шифрования. Сообщение – строка S, состоящая из N строчных английских букв.

$S[a \dots b]$ означает подстроку S от $S[a]$ до $S[b]$ $(0 \leqslant a \leqslant b < N)$. Если первые i букв уже зашифрованы, Алиса найдёт такие (j, k): $s[j..j+k] = s[i..i+k], k \geqslant 0, 0 \leqslant j < i, k = \max$. Если несколько j дают максимальное k, Алиса выберет минимальное j. Если $k > 0$ Алиса добавит пару $\langle j, k \rangle$ в шифр и увеличит i на k, иначе Алиса добавит -1 и ASCII код буквы $S[i]$ в шифр и увеличит i на 1.

Очевидно шифр начнёт с -1, далее будет ASCII код символа $S[0]$. Помогите Алисе реализовать её метод шифрования.

Формат входных данных

Перрвая строка ввода содержит количество тестов T $(1 \leqslant T \leqslant 50)$. Следующие T строк содержат сообщения для шифровки, каждое длины от 1 до 10^5, состоящие из строчных английских букв. Гарантируется, что суммарная длина всех сообщений не превосходит $2 \cdot 10^6$.

Формат выходных данных

Для каждого теста на отдельной строке выведите "`Case #X:`", где X – номер теста, нумерация с 1. Далее выведите шифр, в каждой строке по два целых числа через пробел.

Примеры

lzss.in	lzss.out
2 aaaaaa aaaaabbbbbaaabbc	Case #1: -1 97 5 0 Case #2: -1 97 4 0 -1 98 4 5 5 2 -1 99

Паросочетания

Задача F. Рабочее расписание [0.25 sec, 256 mb]

Дан неориентированный граф из N вершин и нескольких рёбер.
Найти максимальное паросочетание.
Здесь можно прочесть оригинальную легенду.

Формат входных данных

На первой строке число вершин N ($1 \leqslant N \leqslant 222$).
Далее строки, содержащие рёбра.

Формат выходных данных

На первой строке число вершин, покрытых максимальным паросочетанием.
На следующих строках рёбра паросочетания.
Если есть несколько максимальных паросочетаний, выведите любое.

Примеры

schedule.in	schedule.out
3 1 2 2 3 1 3	2 1 2

Подсказка по решению

Алгоритм Эдмондса сжатия соцветий. Реализация Габова.

Линейное программирование

Задача G. Простая задача [0.25 sec, 256 mb]

Найдите оптимальное решение задачи линейного программирования.

$$\begin{cases} a_{11}x_1 + \ldots + a_{1n}x_n \leqslant b_1 \\ a_{21}x_1 + \ldots + a_{2n}x_n \leqslant b_2 \\ \ldots \\ a_{m1}x_1 + \ldots + a_{mn}x_n \leqslant b_m \\ x_1 \geqslant 0 \\ \ldots \\ x_n \geqslant 0 \\ c_1x_1 + \ldots + c_nx_n \to \max \end{cases}$$

Формат входных данных

Первая строка входного файла содержит два целых числа: n и m — количество переменных и количество уравнений ($1 \leqslant n, m \leqslant 5$). Следующие m строк содержат по $n+1$ целому числу: $a_{i1}, \ldots, a_{in}, b_i$. Следующая строка содержит n целых чисел: $c_1, \ldots, c_n$. Все числа во входном файле не превышают 100 по модулю.

Формат выходных данных

Выведите одно число: максимальное значение $c_1x_1+\ldots+c_nx_n$. Гарантируется, что решение существует и максимальное значение достигается.

Пример

simple.in	simple.out
2 2 1 2 3 2 1 3 1 1	2.0
1 1 1 2 -2	0.0
1 1 4 5 3	3.75

Задача H. Простая задача [0.25 sec, 256 mb]

Найдите оптимальное решение задачи линейного программирования.

$$\begin{cases} a_{11}x_1 + \ldots + a_{1n}x_n \leqslant b_1 \\ a_{21}x_1 + \ldots + a_{2n}x_n \leqslant b_2 \\ \ldots \\ a_{m1}x_1 + \ldots + a_{mn}x_n \leqslant b_m \\ x_1 \geqslant 0 \\ \ldots \\ x_n \geqslant 0 \\ c_1x_1 + \ldots + c_nx_n \to \max \end{cases}$$

Формат входных данных

Первая строка входного файла содержит два целых числа: n и m — количество переменных и количество уравнений ($1 \leqslant n, m \leqslant 5$). Следующие m строк содержат по $n+1$ целому числу: $a_{i1}, \ldots, a_{in}, b_i$. Следующая строка содержит n целых чисел: $c_1, \ldots, c_n$. Все числа во входном файле не превышают 100 по модулю.

Формат выходных данных

Выведите одно число: максимальное значение $c_1x_1+\ldots+c_nx_n$. Если решения нет, выведите "`No solution`". Если можно получить сколь угодно большое значение, выведите "`Unbounded`".

Пример

simple2.in	simple2.out
2 2 1 2 3 2 1 3 1 1	2.0
2 1 -1 -1 0 1 1	Unbounded
2 1 1 1 -1 1 1	No solution

Задача I. Гиперраздление [0.5 sec, 256 mb]

Вам даны N d-мерных точек. Нужно разделить их $d-1$ мерной плоскостью, проходящей через точку $(0, \ldots, 0)$. Гарантируется, что решение всегда существует.

Формат входных данных

В первой строке N ($1 \leqslant N \leqslant 5\,000$), d ($1 \leqslant d \leqslant 10$). Далее N строк содержат по $d+1$ чисел. Первые d чисел — вещественные числа, координаты очередной точки $x_{j,1}, x_{j,2}, \ldots, x_{j,d}$. Последнее число — целое число y_j ($|y_j| = 1$). Число y_j определяет положение точки относительно плоскости, читайте дальше для более точного понимания.

Координаты точек по модулю не превосходят 100.0. Все вещественные числа содержат не более 6 знаков после десятичной точки.

Формат выходных данных

Выведите d вещественных чисел $n_1, n_2, \ldots, n_d$ — нормаль плоскости. Нормаль — вектор произвольной **положительной** длины. Для каждой точки $sign(\sum_{i=1}^{d} x_{j,i} \cdot n_i)$ должен быть равен ее y_j. Если решений несколько, можно вывести любое.

Ответ будет считаться корректным, если для каждой точки $|\sum_{i=1}^{d} x_{j,i} \cdot n_i| \geqslant 10^{-4}|n|$, где $|n|$ — длина вектора нормали.

Пример

pam.in	pam.out
2 1 1 -1 -1 1	-1.00000
4 3 -99.749 12.71 -61.33 1 61.7 17.00 -4.0 -1 -29.94 79.192 64.56 1 49.320 -65.178 71.788 -1	-0.891651465 0.415648308 -0.179427280

Задача J. Road times [0.5 sec, 256 mb]

Дорожная сеть страны – ориентированный граф. У каждого ребра есть длина, целое число от 1 до 1000, и ограничение на максимальную скорость, вещественное число от 30 до 60, в километрах в час. Длины вам известны, а ограничения – нет. Известно, что когда водитель такси берётся доставить пассажира из вершины a в вершину b, он осуществляет перевозку строго по кратчайшему пути между вершинами. Длина пути – сумма длин рёбер. Также известно, что $\forall a, b, \exists$ *единственный* кратчайший путь из a в b.

Ваша задача – по длинам рёбер и уже сделанным поездкам a_i b_i $time_i$ оценить минимальное и максимальное время в пути между вершинами c_j d_j

Формат входных данных

На первой строке число вершин n ($1 \leqslant n \leqslant 30$). Вершины нумеруются числами от 0 до $n-1$. Следующие n строк содержат матрицу $n \times n$ длин дорог. Дороги двусторонние, поэтому матрица симметричная. Отсутствие дорог обозначено числом -1, длины дорог целые от 1 до 1 000. Всего не более 100 дорог.

Далее идёт число r ($1 \leqslant r \leqslant 100$) и r уже известных маршрутов a_i b_i $time_i$, $time_i$ вещественное. Затем число запросов q ($1 \leqslant q \leqslant 100$) и q строк c_j d_j.

Формат выходных данных

Для каждого запроса выведите четыре числа – откуда, куда, min время, max время.

Примеры

roadtimes.in	roadtimes.out
3 0 50 -1 55 0 40 -1 40 0 1 0 2 120 3 0 1 1 2 1 0	0 1 50.0 80.0 1 2 40.0 70.0 1 0 55.0 110.0

Планарные графы

Задача K. Проверка на планарность [0.5 sec, 256 mb]

Дано множество рёбер на n вершинах, проверьте сколько максимум первых рёбер можно взять, чтобы получился планарный граф.

Формат входных данных

На первой строке число вершин n ($1 \leqslant n \leqslant 100$) и рёбер m ($0 \leqslant m \leqslant 3n$). Следующий m строк содержат рёбра графа. Вершины нумеруются от 0 до $n-1$. Петель и кратных рёбер в графе нет.ы

Формат выходных данных

Выведите одно число – такое максимальное i, что первые i рёбер, перечисленных во вводе, образуют планарный граф.

Примеры

planarity.in	planarity.out
6 13 2 1 5 1 5 3 5 4 3 1 4 1 4 0 5 2 5 0 4 2 4 3 3 0 2 0	12

Рандомизированные алгоритмы

Задача L. Сторожевая башня [1.5 sec, 256 mb]

Король, мудро правящий Берляндией, обеспокоен! На страну ополчились соседние племена варваров и они совершают набеги на города этой страны! Но, к счастью, при дворе Берляндии служит великий волшебник, который может построить башню, которая защитит города от будущих набегов. Единственная проблема заключается в том, что строительство башни требует больших ресурсов, а, как следствие, денег. Разумеется, король хочет минимизировать свои затраты на строительство башни. Чем больше башня, тем больше расходы, поэтому нужно построить башню наименьшей высоты, которая сможет защитить все города от набегов.

Для простоты будем считать, что города и башня представляют собой точки на плоскости. Чтобы город оказался под защитой башни, необходимо, чтобы ее высота была не меньше расстояния до этого города. Помогите королю защитить своих подданых, а заодно и его казну! Для этого всего лишь надо найти такое положение и высоту башни, которое минимизирует расходы.

Формат входных данных

В первой строке ввода задано целое число n ($1 \leqslant n \leqslant 1\,000\,000$). Следующие n строк содержат координаты городов x_i, y_i ($|x_i|, |y_i| \leqslant 1\,000\,000$) Разумеется, не может быть такого, что два города находятся в одной точке.

Формат выходных данных

В первой строке выведите вещественное число h — минимально возможную высоту башни. Во второй строке выведите координаты места, где будет построена башня.

Ваш ответ будет считаться правильным, если его абсолютная или относительная погрешность не превышает 10^{-6}

towers.in	towers.out
4 1 0 0 1 1 2 2 1	1 1 1

Геометрия многоугольников

Задача M. Самая дальняя [2 sec, 256 mb]

Даны N точек на плоскости, нужно уметь обрабатывать следующие запросы:

- `get a b` — возвращает максимум по всем точкам величины $ax + by$.
- `add x y` — добавить точку в множество.

Формат входных данных

Число N ($1 \leqslant N \leqslant 10^5$) и N точек. Далее число M ($1 \leqslant M \leqslant 10^5$ — количество запросов и собственно запросы. Формат запросов можно посмотреть в примере. Все координаты точек и числа `a`, `b` — целые числа, по модулю не превосходящие 10^9.

Формат выходных данных

На каждый запрос вида `get` выведите одно целое число — максимум величины $ax + by$.

Пример

mostfar.in	mostfar.out
3 0 0 1 0 0 1 10 get 1 1 get -1 -1 get 1 -1 get -1 1 add 2 2 add -2 -2 get 1 1 get -1 -1 get 1 -1 get -1 1	1 0 1 1 4 4 1 1

Замечание

Здесь можно обойтись без динамической выпуклой оболочки... помните, что такое “пополняемые структуры данных”? А вот обычную выпуклую оболочку придётся построить. При поиске экстремальной точки убедитесь постарайтесь использовать функцию `lower_bound`.