Задача A. Сумма двух

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мебибайт

Найдите сумму двух заданных чисел.

Формат входных данных

В первой строке входных данных расположены два целых числа A и B, не превосходящие 1000 по модулю.

Формат выходных данных

Ваша программа должна выдавать одно число — сумму чисел A и B.

Примеры

стандартный ввод	*стандартный вывод*
2 3	5
17 -18	-1

Задача B. Сумма двух наоборот

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мебибайт

Конечно, вы уже сталкивались с такой задачей: даны два числа, требуется вычислить их сумму. Решите ту же задачу, только наоборот. А именно: дана сумма двух целых чисел, найдите два числа, дающие такую сумму.

Формат входных данных

В первой строке входных данных задано одно целое число c.

Формат выходных данных

Выведите два целых числа a и b, разделив их пробелом. Выведенные числа не должны превышать по модулю 10^9. Кроме того, для них должно выполняться равенство $a + b = c$. Гарантируется, что заданное c таково, что существует ответ, удовлетворяющий этим ограничениям. Если ответов несколько, разрешается выводить любой из них.

Пример

стандартный ввод	*стандартный вывод*
1	3 -2

Задача C. Одно число на месте

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мебибайт

Как известно, последовательность из n целых чисел называется перестановкой, если каждое из чисел 1, 2, …, n встречается в ней ровно один раз. Например, последовательности 1 2 3 и 2 3 4 1 — перестановки, а последовательности 1 1 и 2 3 4 — нет.

Число в перестановке считается стоящим на своём месте, если номер этого числа, считая с начала перестановки, совпадает с ним самим. Например, в перестановке 1 2 3 каждое число стоит на своём месте, а в перестановке 2 3 4 1 ни одно число не стоит на своём месте: 1 стоит на четвёртом месте, 2 — на первом, 3 — на втором, а 4 — на третьем.

Для заданного натурального числа n выведите любую перестановку длины n, в которой на своём месте стоит ровно одно число, или выясните, что такой перестановки не существует.

Формат входных данных

В первой строке входного файла задано натуральное число n ($1 \leqslant n \leqslant 27$).

Формат выходных данных

Выведите в выходной файл ровно n целых чисел, образующих перестановку. В выведенной перестановке ровно одно число должно стоять на своём месте. При выводе числа следует разделять пробелами.

Если требуемая перестановка не существует, выведите в выходной файл вместо перестановки одно число `-1`. Если правильных ответов несколько, можно вывести любой из них.

Примеры

стандартный ввод	*стандартный вывод*
4	2 3 1 4
3	3 2 1

Пояснения к примерам

В первом примере четвёрка стоит на четвёртом месте, остальные числа — не на своих местах. Это не единственный правильный ответ: например, в перестановке 1 3 4 2 на своём месте тоже стоит ровно одно число.

Во втором примере двойка стоит на втором месте, остальные числа — не на своих местах. Это снова не единственный правильный ответ: например, в перестановке 2 1 3 на своём месте тоже стоит ровно одно число.

Задача D. Сумма трёх кубов

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мебибайт

Петя увлекается математикой. Ему нравится складывать и перемножать числа, возводить их в квадрат, в куб и подмечать закономерности в отношениях между числами.

На прошлой неделе Петя занимался квадратами натуральных чисел. Он придумал себе такое занятие: взять некоторое число n, возвести его в квадрат, а затем искать два других числа a и b таких, чтобы сумма их квадратов была равна квадрату числа n. Для некоторых n найти такие a и b получилось: например, $5^2 = 3^2 + 4^2$, $26^2 = 10^2 + 24^2$. Для других, например, для $n = 6$ или 7, выяснилось, что таких чисел a и b не существует.

На этой неделе Петя решил сделать то же самое с кубами натуральных чисел: по выбранному n искать такие a и b, что $n^3 = a^3 + b^3$. Однако, как он ни старался, ни одного такого равенства найти не получилось. Отчаявшись, Петя обратился к учителю математики Виктору Георгиевичу, и тот рассказал, что равенство $n^3 = a^3 + b^3$ невозможно ни для каких трёх натуральных чисал a, b и n. Это следует из великой теоремы Ферма, которая гласит, что уравнение $n^p = a^p + b^p$ не имеет натуральных решений n, a и b не только для $p = 3$, но и вообще для любого натурального $p > 2$.

У этой, на первый взгляд, простой теоремы — удивительная история. Она была сформулирована французским математиком Пьером Ферма ещё в 1637 году, но доказательства он не оставил. Долгое время эта теорема была самой известной из нерешённых проблем в математике. Более трёхсот лет математики со всего мира пытались доказать её. Окончательно доказана она была лишь в 1995 году Эндрю Джоном Уайлсом. За доказательство он был, в частности, посвящён в рыцари Британской Империи.

Итак, Петя вынужден был найти себе другое занятие. Он подумал, что, хотя из двух кубов не получается собрать третий, может быть, из трёх кубов можно собрать четвёртый? И действительно, почти сразу он нашёл четыре подходящих числа: $6^3 = 216 = 27 + 64 + 125 = 3^3 + 4^3 + 5^3$. Обрадованный, Петя стал искать другие такие n, что их кубы представимы в виде $n^3 = a^3 + b^3 + c^3$ для некоторых трёх натуральных чисел a, b и c. Но числа очень быстро стали большими, и Петя стал ошибаться в арифметических действиях.

Помогите Пете! Напишите программу, которая по заданному натуральному числу n находит такие три натуральных числа a, b и c, что $a^3 + b^3 + c^3 = n^3$, или выясняет, что для данного n таких чисел не существует.

Формат входных данных

В первой строке входного файла задано натуральное число n ($1 \leqslant n \leqslant 100$).

Формат выходных данных

Если n^3 можно представить в виде суммы кубов трёх натуральных чисел a, b и c, выведите в выходной файл эти три числа через пробел. В противном случае выведите в выходной файл одно число -1.

Порядок чисел a, b и c не имеет значения. Если правильных ответов несколько, можно вывести любой из них.

Примеры

стандартный ввод	*стандартный вывод*
6	3 4 5
2	-1

Задача E. Сумма девяти квадратов

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мебибайт

Как известно, целое число a называется точным квадратом, если существует такое целое число b, что a является квадратом b, то есть $b^2 = b \cdot b = a$. Например, 16 — точный квадрат, поскольку $4 \cdot 4 = 16$. Напротив, 10 — не точный квадрат, так как равенство $b^2 = 10$ неверно ни для какого целого числа b.

Представьте заданное во входном файле натуральное число n в виде суммы девяти точных квадратов неотрицательных целых чисел.

Формат входных данных

В первой строке входного файла задано натуральное число n ($1 \leqslant n \leqslant 1\,000\,000\,000$).

Формат выходных данных

Выведите в выходной файл ровно девять неотрицательных целых чисел: a_1, a_2, …, a_9. Эти числа должны быть такими, что

$$a_1^2 + a_2^2 + a_3^2 + a_4^2 + a_5^2 + a_6^2 + a_7^2 + a_8^2 + a_9^2 = n.$$

При выводе числа следует разделять пробелами. Порядок чисел не имеет значения. Если правильных ответов несколько, можно вывести любой из них.

Примеры

стандартный ввод	*стандартный вывод*
5	1 2 0 0 0 0 0 0 0
9	1 1 1 1 1 1 1 1 1

Пояснения к примерам

В первом примере $1^2+2^2+0^2+0^2+0^2+0^2+0^2+0^2+0^2 = 1+4 = 5$. Это не единственный правильный ответ: например, ответ `1 1 1 1 1 0 0 0 0` тоже подходит.

Во втором примере $1^2+1^2+1^2+1^2+1^2+1^2+1^2+1^2+1^2 = 9 \cdot 1 = 9$. Это не единственный правильный ответ: например, ответ `0 0 0 0 0 0 0 0 3` тоже подходит.

Задача F. Целые точки

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мебибайт

Точку на координатной плоскости будем называть целой, если обе её координаты — целые числа. К примеру, точки $(0, 0)$ и $(-4, 7)$ — целые, а точки $(-1, 0.5)$ и $(\frac{1}{3}, \sqrt{2})$ — нет.

Сколько целых точек содержит заданный отрезок на плоскости?

Формат входных данных

В первой строке входного файла заданы два числа x_1 и y_1 — координаты одного конца отрезка. Во второй строке заданы два числа x_2 и y_2 — координаты другого конца отрезка. Числа в каждой строке разделены пробелами. Все заданные координаты — целые числа, не превосходящие по модулю $1\,000\,000\,000$. Гарантируется, что заданные две точки не совпадают.

Формат выходных данных

Выведите в выходной файл количество целых точек на заданном отрезке. Обратите внимание, что концы отрезка тоже учитываются.

Примеры

стандартный ввод	*стандартный вывод*
2 1 4 1	3
0 0 5 7	2

Пояснения к примерам

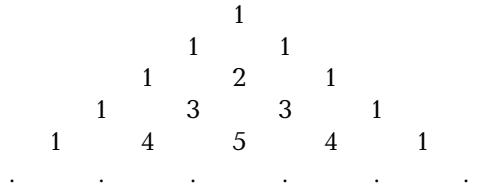
В первом примере целые точки — $(2, 1)$, $(3, 1)$ и $(4, 1)$.

Во втором примере целые точки — только концы отрезка $(0, 0)$ и $(5, 7)$.

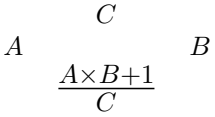
Задача G. Треугольная таблица

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мебибайт

Рассмотрим следующую бесконечную треугольную таблицу чисел:



По краям в каждой строке стоят единицы, а остальные числа получаются по «правилу ромба»: число снизу равно произведению двух чисел над ним на предыдущей строке, увеличенному на 1 и разделённому на число, находящееся на две строки выше.



Например, в пятой строке второе, третье и четвёртое числа равны соответственно $\frac{1\times3+1}{1} = 4$, $\frac{3\times3+1}{2} = \frac{10}{2} = 5$ и $\frac{3\times1+1}{1} = 4$.

Может показаться удивительным, но, несмотря на деление, все числа в таблице будут целыми.

По заданным n и k найдите k-е число в n-й строке этой таблицы. Строки, а также числа в каждой строке, нумеруются с единицы.

Формат входных данных

В первой строке входного файла заданы два числа n и k ($1 \leqslant k \leqslant n \leqslant 1\,000\,000$). Числа разделены одним пробелом.

Формат выходных данных

Выведите в выходной файл число, стоящее на k-м месте в n-й строке таблицы.

Примеры

стандартный ввод	*стандартный вывод*
5 2	4
6 3	7

Пояснения к примерам

В первом примере $\frac{1\times3+1}{1} = 4$.

Во втором примере $\frac{4\times5+1}{3} = \frac{21}{3} = 7$.

Задача H. Друзья и враги

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мебибайт

На острове Дженту живут n крокодилов. Каждый крокодил имеет ровно одного друга и ровно одного врага среди остальных крокодилов на острове. Отношения дружбы и вражды симметричны: если крокодил $\mathcal{A}$ — друг крокодила $\mathcal{B}$, то крокодил $\mathcal{B}$ — друг крокодила $\mathcal{A}$; если же крокодил $\mathcal{A}$ — враг крокодила $\mathcal{B}$, то и крокодил $\mathcal{B}$ — враг крокодила $\mathcal{A}$. Никакой крокодил не является одновременно врагом и другом другого крокодила. Кроме того, никакой крокодил не может быть ни другом, ни врагом самому себе.

Крокодилы острова Дженту весьма эмоциональны. Когда встречаются два врага, они со страшной силой колотят по земле хвостами, а затем происходит Битва Крокодилов. Когда встречаются два друга, они исполняют разрушительный Танец Дружбы Крокодилов.

Недавно танцы и битвы разбудили вулкан, находящийся под островом, и остров раскололся на две части. Вулкан успокоился, но крокодилы опасаются, что дальнейшие танцы и битвы разбудят его снова, и обе половины острова зальёт лава.

Теперь крокодилы хотят расселиться по двум половинам острова таким образом, чтобы на каждой половине оказалось *нейтральное* множество крокодилов — такое множество, что никакие два крокодила в нём не являются ни друзьями, ни врагами. Расселившись так, крокодилы не будут устраивать танцы и битвы, а значит, можно надеяться, что вулкан не будет извергаться снова.

Крокодилы острова Дженту мудры, но непрактичны. Они понимают, что при таком устройстве дружбы и вражды разделение крокодилов на два нейтральных множества всегда возможно, но не знают, как именно разделиться на такие множества.

Помогите им! Найдите такое разбиение крокодилов на два множества, что у каждого крокодила в своём множестве нет ни друзей, ни врагов.

Формат входных данных

В первой строке входного файла задано натуральное число n — количество крокодилов ($4 \leqslant n \leqslant 100$). Следующие n строк описывают крокодилов. В первой из них записаны два числа f_1 и e_1 через пробел — номер друга первого крокодила и номер его врага. Во второй записаны числа f_2 и e_2 — номера друга и врага второго крокодила, и так далее. В последней из этих строк записаны f_n и e_n — друг и враг крокодила с номером n. Крокодилы пронумерованы числами от 1 до n в том порядке, в котором они описываются во входном файле.

Все числа f_k и e_k целые и лежат в пределах от 1 до n, включительно. Отношения дружбы и вражды симметричны. Друг и враг каждого крокодила различны. Никакой крокодил не является ни другом, ни врагом самому себе.

Формат выходных данных

В первой строке выходного файла выведите n чисел через пробел. Каждое из этих чисел должно быть равно либо 1, либо 2. Если i-е и j-е числа равны, это означает, что крокодилы с номерами i и j оказались на одной половине острова. Если же эти числа различны, значит, крокодилы i и j оказались на разных половинах острова.

Если правильных ответов несколько, можно вывести любой из них.

Пример

стандартный ввод	*стандартный вывод*
4 2 4 1 3 4 2 3 1	1 2 1 2

Задача I. Вагонетки

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мебибайт

По бесконечной прямой ветке железной дороги n вагонеток перевозят грузы. Скорости движения всех вагонеток одинаковы и равны 1 км/ч, однако направления движения могут различаться: вагонетка может ехать по железной дороге слева направо, а может — справа налево.

Когда две вагонетки встречаются, им не обойти друг друга, так как железная дорога для движения в обоих направлениях — одна и та же. Поэтому они обмениваются грузами, и дальше обе вагонетки движутся в обратном направлении: та вагонетка, что ехала слева направо, теперь едет справа налево, а та, что ехала справа налево, теперь едет слева направо. Скорости вагонеток после встречи такие же, как до встречи — 1 км/ч. Время, которое требуется вагонеткам для остановки, обмена грузами и разгона, будем считать равным нулю; размером вагонеток также можно пренебречь.

Известны начальные положения вагонеток, а также время t в часах, прошедшее с момента, когда они были в таком положении. Вычислите координаты всех вагонеток в момент времени t и выведите их в неубывающем порядке.

Формат входных данных

В первой строке входного файла задано через пробел два натуральных числа n и t — количество вагонеток и время, прошедшее с начального момента ($1 \leqslant n \leqslant 100$, $1 \leqslant t \leqslant 1000$). Следующие n строк описывают вагонетки. В каждой из них записано по два целых числа x_k и v_k — начальная позиция вагонетки в километрах, отсчитываемых слева направо от некоторой точки железной дороги, и её начальная скорость в километрах в час ($1 \leqslant x_k \leqslant 1000$, $v_k = \pm 1$; положительная скорость означает движение слева направо, а отрицательная — справа налево). Гарантируется, что начальные позиции всех вагонеток различны.

Формат выходных данных

В первой строке выходного файла выведите n целых чисел через пробел — координаты вагонеток в момент времени t в той же системе координат, что и начальные позиции, заданные во входном файле. Числа следует выводить в неубывающем порядке.

Пример

стандартный ввод	*стандартный вывод*
4 6 2 1 14 -1 9 1 4 -1	-2 8 8 15

Задача J. Ординальные числа

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мебибайт

Ординальное число — это обобщение понятия натурального числа (здесь и далее в этой задаче 0 считается натуральным числом). Ординальные числа, соответствующие натуральным числам, можно определить рекурсивно:

> Ординальное число, соответствующее натуральному числу n — это множество, элементами которого являются ординальные числа, соответствующие натуральным числам 0, 1, 2, ..., $n-1$.

В частности, пустое множество $\varnothing = \{\}$ — ординальное число, соответствующее натуральному числу 0, единице соответствует множество $\{\varnothing\} = \{\{\}\}$, двойке — множество $\{\varnothing, \{\varnothing\}\} = \{\{\}, \{\{\}\}\}$ и так далее. Удобно записывать ординальное число просто как соответствующее ему натуральное число, например, $4 = \{0, 1, 2, 3\}$. Вообще, $n = \{0, 1, \ldots, (n-1)\}$.

Такое определение удобно тем, что его можно расширить на бесконечные множества. К примеру, наименьшее бесконечное ординальное число ω определяется как множество, содержащее все конечные ординальные числа. Следующее за ним число (будем записывать его как $\omega + 1$) равно $\{\omega, 0, 1, \ldots\}$ или, другими словами, $\omega + 1 = \omega \cup \{\omega\}$, и так далее. В общем случае по любому ординальному числу α можно определить следующее ординальное число $\alpha + 1 = \alpha \cup \{\alpha\}$. Однако, предыдущее ординальное число определено не для всех чисел, например, для ω не существует такого γ, что $\omega = \gamma \cup \{\gamma\}$.

Игорь выписал на доске некоторое конечное ординальное число. Запись была корректной и состояла из открывающих и закрывающих фигурных скобок, а также запятых. Формат записи рекурсивно определяется так: запись каждого множества начинается с открывающей фигурной скобки, далее по одному разу следуют записи всех элементов этого множества в произвольном порядке, разделённые запятыми, а в конце стоит закрывающая фигурная скобка. К примеру, ординальное число 0 записывается как «{}», 1 — как «{{}}», 2 — как «{{},{{}}}» или «{{{}},{}}» и так далее.

Улучив момент, Владислав подкрался к доске и стёр один символ. Запись больше не является корректной! Это несколько расстраивает Игоря. Установив виновника, Владимир Михайлович попросил Владислава срочно исправить положение. Помогите Владиславу восстановить исходную запись.

Формат входных данных

В первой строке ввода задана запись ординального числа, в которой отсутствует ровно один символ. Длина этой записи — от 1 до $3 \cdot 10^6$ символов.

Формат выходных данных

Выведите одну строку — правильную запись ординального числа, полученную добавлением одного символа в любое место заданной записи. Если правильных ответов несколько, выведите любой из них.

Примеры

стандартный ввод	*стандартный вывод*
{{{}}{}}	{{{}},{}}
{{{}},}}	{{{}},{}}
{{{},{}}	{{{}},{}}

Пояснения к примерам

Во всех трёх тестах после добавления одного символа получается ординальное число 2.

В первом тесте нужно добавить запятую, во втором — открывающую фигурную скобку, в третьем — закрывающую фигурную скобку.

Задача K. Лыжная трасса

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мебибайт

Лыжница Аня находится на старте лыжной трассы. Трасса представляет собой участок плоскости, на котором введена декартова система координат. Координаты измеряются в метрах. Старт соответствует прямой $y = 0$, а финиш — прямой $y = n$. На прямых $y = 1$, $y = 2$, …, $y = n$ расположены ворота. Каждые ворота состоят из двух флажков, расстояние между которыми равно d метрам. Ворота характеризуются координатой левого края: на каждой прямой $y = k$ эта координата равна x_k.

Чтобы пройти трассу, Ане нужно последовательно пройти сквозь все ворота. Другими словами, пересекать каждую прямую вида $y = k$ можно лишь в точках (x, y) таких, что $x_k \leqslant x \leqslant x_k + d$. Движение между двумя соседними воротами ничем не ограничено.

Аня может начать движение из любой точки прямой $y = 0$ и закончить движение в любой точке прямой $y = n$, находящейся внутри последних ворот. Найдите минимальную длину маршрута, по которому можно пройти трассу. Размером Ани можно пренебречь и считать её точкой.

Формат входных данных

В первой строке входного файла заданы через пробел два целых числа n и d — количество ворот и ширина ворот в метрах ($1 \leqslant n \leqslant 100$, $2 \leqslant d \leqslant 10$). В следующей строке заданы n целых чисел x_1, x_2, …, x_n через пробел — координаты левого края первых, вторых, …, последних ворот трассы ($0 \leqslant x_k \leqslant 1000$).

Формат выходных данных

Выведите в выходной файл одно число — минимальную длину пути по трассе с точностью не менее шести знаков после запятой.

Примеры

стандартный ввод	*стандартный вывод*
3 2 1 2 1	3
5 2 0 2 5 7 10	9.94427190999916

Задача L. Цепная дробь

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мебибайт

Цепной дробью называется выражение вида

$$a_0 + \frac{1}{a_1 + \frac{1}{a_2 + \frac{1}{\ldots + \frac{1}{a_m}}}},$$

где все a_k целые, $a_k > 0$ при $k > 0$ и $a_m > 1$, если $m > 0$. Эти, на первый взгляд странные, правила позволяют каждому вещественному числу x единственным образом сопоставить цепную дробь так, чтобы выполнялось равенство

$$x = a_0 + \frac{1}{a_1 + \frac{1}{a_2 + \frac{1}{\ldots + \frac{1}{a_m}}}}.$$

Например, цепная дробь для числа $7/18$ выглядит как

$$0 + \frac{1}{2 + \frac{1}{1 + \frac{1}{1 + \frac{1}{3}}}}:$$

$$\begin{aligned} 7/18 &= 0 + 7/18 \\ &= 0 + 1/(18/7) \\ &= 0 + 1/(2 + 4/7) \\ &= 0 + 1/(2 + 1/(7/4)) \\ &= 0 + 1/(2 + 1/(1 + 3/4)) \\ &= 0 + 1/(2 + 1/(1 + 1/(4/3))) \\ &= 0 + 1/(2 + 1/(1 + 1/(1 + 1/3))). \end{aligned}$$

Цепную дробь можно записывать как $[\mathtt{a}_0; \mathtt{a}_1, \mathtt{a}_2, \ldots, \mathtt{a}_m]$; в этой записи $7/18 = [0; 2, 1, 1, 3]$.

Не все числа имеют конечную цепную дробь; так, для числа $\sqrt{2} \approx 1,41421356\ldots$ цепная дробь имеет вид $[1; 2, 2, 2, 2, 2, \ldots]$ — воспользовавшись равенством $\sqrt{2} - 1 = 1/(\sqrt{2} + 1)$, получаем:

$$\begin{aligned} \sqrt{2} &= 1 + (\sqrt{2} - 1) \\ &= 1 + 1/(\sqrt{2} + 1) \\ &= 1 + 1/(2 + (\sqrt{2} - 1)) \\ &= 1 + 1/(2 + 1/(\sqrt{2} + 1)) \\ &= \ldots \end{aligned}$$

Цепная дробь такого вида называется *периодической*; в данном случае предпериод — это число $a_0 = 1$, а период — повторяющееся бесконечное количество раз число 2. Для удобства записи период цепной дроби можно записывать в круглых скобках; для $\sqrt{2}$ сокращённая запись будет выглядеть как $[1; (2)]$. Можно доказать, что для всякого целого $N \geqslant 0$ цепная дробь числа $\sqrt{N}$ является конечной, если $\sqrt{N}$ — целое число, и периодической в противном случае.

По данному числу N выведите цепную дробь числа $\sqrt{N}$ в сокращённой записи.

Формат входных данных

В первой строке входного файла записано целое число N ($1 \leqslant N \leqslant 1000$).

Формат выходных данных

В первую и единственную строку выходного файла выведите сокращённую запись цепной дроби числа $\sqrt{N}$. Необходимо как можно более точно соблюдать такой же формат вывода, как в примерах. Гарантируется, что правильный ответ на каждый тест содержит не более 10 000 символов.

Примеры

стандартный ввод	***стандартный вывод***
1	[1]
2	[1;(2)]
3	[1;(1,2)]
4	[2]
5	[2;(4)]
6	[2;(2,4)]
7	[2;(1,1,1,4)]
76	[8;(1,2,1,1,5,4,5,1,1,2,1,16)]