Задача A. Обобщенные числа Фибоначчи

Имя входного файла:	`fib.in`
Имя выходного файла:	`fib.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Обобщенными числами Фибоначчи $F_n^{(k)}$ называют следующую последовательность: $F_1^{(k)} = F_2^{(k)} = \ldots = F_k^{(k)} = 1$, $F_i^{(k)} = \sum_{j=1}^{k} F_{i-j}^{(k)}$, $i > k$.

Ваша задача — вычислить остаток от деления $F_n^{(k)}$ на p.

Формат входного файла

Входной файл содержит три целых числа: n, k и p ($1 \le n, k \le 10^6$, $2 \le p \le 10^9$).

Формат выходного файла

В выходной файл выведите $F_n^{(k)} \bmod p$.

Примеры

fib.in	fib.out
3 2 10	2
10 2 10	5

Задача B. Преобразователь строк

Имя входного файла:	`morpher.in`
Имя выходного файла:	`morpher.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Преобразователь строк — специальная программа, получающая на вход строку и выдающая на выход также строку. В процессе обработки к некоторым подстрокам исходной строки применяются операции двух типов:

- разворот (обозначается символом `R`) — из строки $s_1 s_2 \dots s_{n-1} s_n$ получается строка $s_n s_{n-1} \dots s_2 s_1$;
- сортировка (обозначается символом `S`) — из строки $s_1 s_2 \dots s_{n-1} s_n$ получается строка, содержащая те же символы, но отсортированные в порядке неубывания их номеров по алфавиту. Например, из строки `hello` получится строка `ehllo`.

Задана строка и последовательность операций над некоторыми ее подстроками. Необходимо определить результат применения этих операций.

Формат входного файла

Первая строка входного файла содержит исходную строку. Она не пуста, состоит только из строчных букв латинского алфавита. Ее длина m не превышает 200 символов. Вторая строка входного файла содержит количество операций n ($1 \le n \le 200$).

Каждая из последующих n строк содержит описание одной операции. Описание операции имеет формат `OP L R`, где `OP` — символ, обозначающий операцию, `L` — позиция первого символа подстроки, к которой применяется операция, `R` — позиция ее последнего символа ($1 \le L \le R \le m$). Если до применения операции обрабатываемая строка имела вид $s_1 s_2 \dots s_m$, то после применения операции она будет иметь вид $s_1 \dots s_{L-1} OP(s_L \dots s_R) s_{R+1} \dots s_m$, где $OP(s_L \dots s_R)$ — результат применения описываемой операции к строке $s_L \dots s_R$.

Формат выходного файла

В выходной файл выведите результат применения к исходной строке всех перечисленных во входном файле операций.

Примеры

morpher.in	morpher.out
helloworld 2 S 1 5 R 6 10	ehllodlrow

Задача C. Оптимизация сепарабельной функции

Имя входного файла:	`optimize.in`
Имя выходного файла:	`optimize.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Функцию n переменных $f(x_1, x_2, \ldots, x_n)$ назовем *сепарабельной*, если она представима как сумма следующего вида $f(x_1, x_2, \ldots, x_n) = f_1(x_1) + f_2(x_2) + \ldots + f_n(x_n)$, где каждая f_i зависит только от x_i.

В рассматриваемой задаче каждая из функций f_i является многочленом второй степени от x_i.

Задача минимизации заданной функции на гиперпараллелепипеде формулируется следующим образом: «Задана функция $f(x_1, x_2, \ldots, x_n)$ и множество ограничений $a_1 \le x_1 \le b_1$, $a_2 \le x_2 \le b_2$, …, $a_n \le x_n \le b_n$. Необходимо найти точку $(x_1^*, x_2^*, \ldots, x_n^*)$ такую, что она удовлетворяет всем ограничениям, и значение функции в ней минимальное среди всех возможных».

Ваша задача состоит в том, чтобы решить задачу минимизации на гиперпараллелепипеде для заданной сепарабельной функции, состоящей из многочленов второй степени.

Формат входного файла

Первая строка входного файла содержит целое число n ($1 \le n \le 1000$). Каждая из последующих n строк содержит по пять целых чисел: a_i, b_i, p_i, q_i, r_i — они соответствуют ограничению $a_i \le x_i \le b_i$ и функции $f_i(x_i) = p_i \cdot x_i^2 + q_i \cdot x_i + r_i$ ($-1000 \le a_i \le b_i \le 1000$, $-10 \le p_i, q_i, r_i \le 10$, $p_i \ne 0$).

Формат выходного файла

В первой строке выведите минимальное возможное значение $f(x_1^*, x_2^*, \ldots, x_n^*)$. Во второй строке выведите $x_1^*, x_2^*, \ldots, x_n^*$, на которых достигается указанное минимальное значение.

Ответ будет считаться правильным, если найденное минимальное значение отличается от правильного не более, чем на 10^{-3}.

Примеры

optimize.in	optimize.out
2 1 2 3 4 5 -1 1 1 2 3	14.0 1.0 -1.0
1 -1 1 1 0 0	0.0 0.0

Задача D. Перестановка

Имя входного файла:	`perm.in`
Имя выходного файла:	`perm.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Рассмотрим $\mathbb{Z}^+ = \{0, 1, 2, 3, \ldots\}$ — множество неотрицательных целых чисел. Пусть последовательность a_n, $n \in \mathbb{Z}^+$ определяется формулой $a_n = n + (-1)^n$. Таким образом последовательность a_n выглядит следующим образом: $1, 0, 3, 2, 5, 4, \ldots$.

Эта последовательность интересна тем, что в ней встречаются все числа из $\mathbb{Z}^+$, причем каждое — ровно один раз, то есть она является перестановкой чисел из $\mathbb{Z}^+$.

Ваша задача — найти такое i, что $a_i = x$.

Формат входного файла

Входной файл содержит целое неотрицательное число x ($0 \le x \le 10^9$).

Формат выходного файла

В выходной файл выведите ответ на задачу.

Примеры

perm.in	perm.out
0	1
1	0

Задача E. Космические путешествия

Имя входного файла:	`space.in`
Имя выходного файла:	`space.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

3141 год. Космические путешествия с использованием подпространственных телепортирующих туннелей уже давно не новинка. Однако еще не во всех частях Межгалактической Федерации Землян сеть таких туннелей достаточно развита.

В одной звездной системе n планет соединены туннелями так, что от каждой планеты можно добраться до каждой единственным способом, двигаясь только по туннелям. При этом перемещаться по гиперпространственному туннелю разрешается в обе стороны.

На каждой планете установлена подстанция, которая обеспечивает работу всех туннелей, которыми эта планета соединена с другими. Если эта подстанция перестает работать (например, из-за неполадок или из-за того, что ее закрывают на профилактический ремонт), то все туннели, одним из концов которых является эта планета, перестают работать. Вследствие этого для некоторых других планет может исчезнуть возможность добраться от одной до другой. Будем называть планеты, подстанции которых обладают описанным свойством, *важными*.

Поясним более формально. Планета u называется *важной*, если после того, как туннели, одним из концов которых является u, перестанут работать, появятся хотя бы две такие планеты v и w, что v нельзя добраться до w по оставшимся туннелям.

Задана схема гиперпространственных туннелей в рассматриваемой звездной системе. Ваша задача — написать программу, которая вычисляет количество важных планет в этой звездной системе.

Формат входного файла

Первая строка входного файла содержит n — количество ($1 \le n \le 100$) планет в звездной системе. Далее следуют $(n-1)$ строка, каждая из которых описывает один туннель и содержит два числа: u и v — номера планет, соединенных соответствующим туннелем ($1 \le u, v \le n$, $u \ne v$).

Планеты занумерованы натуральными числами от 1 до n.

Формат выходного файла

В выходной файл выведите ответ на задачу — количество важных планет.

Примеры

space.in	space.out
4 1 2 1 3 1 4	1
5 1 2 2 3 2 4 4 5	2

Задача F. Стабилизация последовательности

Имя входного файла:	`stable.in`
Имя выходного файла:	`stable.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Пусть x — натуральное число. Обозначим как $s(x)$ сумму цифр его делителей. Например, $s(6) = 1 + 2 + 3 + 6 = 12$, $s(10) = 1 + 2 + 5 + 1 + 0 = 9$.

Рассмотрим теперь последовательность $a_1 = x$, $a_2 = s(x)$, $a_3 = s(s(x))$, $\ldots$, $a_n = s(a_{n-1})$, $\ldots$. Скажем, что эта последовательность *стабилизируется*, если для некоторого i выполняется равенство $a_i = a_{i+1}$ (тогда это свойство верно и для любого $j > i$).

Задано число x. Необходимо выяснить, стабилизируется ли последовательность a_n, и найти минимальное i, для которого $a_i = a_{i+1}$.

Формат входного файла

Входной файл содержит целое число x ($1 \le x \le 10^9$).

Формат выходного файла

В первой строке выходного файла выведите искомое минимальное число i или -1, если оно превышает 1000. В первом случае выведите также во второй строке выходного файла первые i членов последовательности a_n через пробел.

Примеры

stable.in	stable.out
16	14 16 22 9 13 5 6 12 19 11 3 4 7 8 15

Задача G. Расширение Вселенных

Имя входного файла:	`universe.in`
Имя выходного файла:	`universe.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Астрономы и астрофизики Флатландии ведут активное исследование расширения Вселенной. Недавно они выяснили, что можно считать, что Вселенная представляет собой круг некоторого радиуса, причем этот радиус увеличивается с постоянной скоростью.

Кроме этого, из результатов последних исследований стало ясно, что существует как минимум две Вселенных. Так как они обе расширяются, то рано или поздно у них будет хотя бы одна общая точка.

В модели Флатландских ученых первая Вселенная представляет собой круг с центром в точке (x_1, y_1), радиус которого возрастает со временем как $r_1 + v_1 \cdot t$, а вторая Вселенная — круг с центром в точке (x_2, y_2), радиус которого возрастает со временем как $r_2 + v_2 \cdot t$.

Ваша задача состоит в том, чтобы написать программу, которая по описаниям Вселенных найдет ближайший к начальному момент времени, в который у двух Вселенных будет общая точка.

Формат входного файла

Первая строка входного файла содержит описание первой Вселенной. Оно содержит 4 целых числа: x_1, y_1, r_1, v_1 — соответственно координаты ее центра, ее радиус в начальный момент времени и скорость расширения. Вторая строка входного файла содержит описание второй Вселенной в аналогичном формате (x_2, y_2, r_2, v_2).

Все числа во входном файле не превышают 10^4 по абсолютному значению, скорости расширения и начальные радиусы строго положительны.

В начальный момент времени Вселенные не пересекаются и не касаются друг друга.

Формат выходного файла

В выходной файл выведите ответ на задачу. При проверке будет игнорироваться абсолютная погрешность, не превосходящая 10^{-6}.

Примеры

universe.in	universe.out
0 0 1 1 2 2 1 1	0.41421356237309515
324 3429 33 389 -134 3498 123 39	0.7176832614131544

Задача H. Победитель

Имя входного файла:	`winner.in`
Имя выходного файла:	`winner.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Интернет-олимпиады по информатике приобретают всю большую популярность. Уже нередко в одном туре участвуют несколько команд от одной школы. Разумеется, они соревнуются не только с другими командами, но и между собой. Причем победа во «внутришкольном» соревновании зачастую более важна, чем место, занятое в олимпиаде.

От одной школы города N-ска во всех одиннадцати базовых Интернет-олимпиадах этого сезона участвовало две команды. Теперь они хотят выяснить, кто из них победил в общем зачете.

Для каждой команды известно, какое место она заняла в каждой Интернет-олимпиаде. Общий зачет внутри этой школы ведется по следующим правилам. За каждую Интернет-олимпиаду команда, занявшая в ней более высокое место, получает 10 очков, а команда, занявшая более низкое место, получает 0 очков. При этом предполагается, что команды не могут занять одинаковое место.

Ваша задача — написать программу, которая по результатам этих команд в 11 Интернет-олимпиадах определит, кто из них победил в общем зачете внутри этой школы.

Формат входного файла

Первая строка входного файла содержит 11 чисел $a_1, a_2, \ldots, a_{11}$ — места, которые заняла первая из рассматриваемых команд в первой, второй, ..., одиннадцатой Интернет-олимпиадах.

Вторая строка содержит места $b_1, b_2, \ldots, b_{11}$, занятые второй командой, в аналогичном формате.

Все числа во входном файле целые, положительные и не превосходят 100. Для всех $i = 1 \ldots 11$ верно неравенство $a_i \neq b_i$.

Формат выходного файла

В выходной файл выведите слово `First`, если в общем зачете победила первая команда, или слово `Second`, если победила вторая команда.

Примеры

winner.in	winner.out
1 2 3 4 5 6 7 8 9 10 11 2 3 4 5 6 7 8 9 10 11 12	First
2 2 1 1 2 2 1 1 2 2 1 1 1 2 2 1 1 2 2 1 1 2	Second