Задача A. Годовой баланс

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

В конторе «Рога и Копыта» подходит время подведения годового баланса. В бухгалтерию поступили сведения о том, что, согласно документам, суммарный расход составил a рублей, а суммарный приход — b рублей. Поскольку с реальным положением дел эти цифры все равно не имеют ничего общего, бухгалтер решил реализовать следующую свою идею. Как известно, при наборе чисел на компьютере люди часто вводят цифры в неправильном порядке. Поэтому бухгалтер хочет найти такой способ переставить цифры в числах a и b, чтобы в результате разность $a-b$ (и, соответственно, количество денег, которые он положит к себе в карман), была максимальна, а в случае чего можно будет сослаться на ошибку секретаря. При этом нельзя забывать о знаке чисел и о том, что ноль не может быть первой цифрой числа. Напишите программу, которая поможет бухгалтеру.

Формат входного файла

Во входном файле задаются два цеылх числа — a и b на отдельных строках. По модулю числа не превосходят $2 \cdot 10^9$.

Формат выходного файла

В выходной файл выведите одно целое число — наибольшую разность чисел, первое из которых может быть получено перестановкой цифр a, а второе — перестановкой цифр b.

Примеры

balance.in	balance.out
18 10	71
1 -23	33

Задача B. Электронная почта

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

Современный мир немыслим без Интернета и электронной почты. Для того, чтобы людям было проще ориентироваться в потоке поступащих писем, были созданы специальные программы — почтовые клиенты. Фирма *TIRLABS* занимается разработкой почтового клиента *The Bar!*.

Недавно программисты компании завершили разработку очередной, 366239-ой, версии этого почтового клиента. Менеджеры по продажам и рекламе уже готовы вовсю рекламировать и продавать эту новую программу. Однако, генеральный директор компании *TIRLABS* считает, что любая программа должна быть подвергнута всестороннему тестированию, прежде чем она будет продаваться. «Да и не работающую программу, скорее всего, никто не купит!» - сказал он.

Одним из видов тестирования сложных программ является так называемое стресс-тестирование. При нем программа тестируется в экстремальных условиях, часто даже в таких, на какие она не рассчитана. Для тестирования *The Bar! ver. 366239* был выбран такой метод: программа запускается на n компьютерах, стоящих в одной комнате, после чего с компьютеров друг на друга посылается m писем. При этом никакие два события (отправление или прием письма) на происходят одновременно, а сеть настолько надежна, что все письма доходят до адресата. Адресат у каждого письма при этом только один.

Почтовый клиент *The Bar!* в процессе работы ведет протокол, в который заносятся идентификаторы отправленных и полученных писем в том порядке, в котором они были обработаны почтовым клиентом. При этом при оценке результатов тестирования в расчет принимаются только события, отраженные в этом протоколе.

Программа считается *правильно работающей по результатам тестирования*, если всем событиям, указанным в протоколах, можно сопоставить моменты времени таким образом, что никакие два события не происходят одновременно, и каждое из писем отправляется до того, как получается. При этом, разумеется, внутри каждого из протоколов порядок событий должен остаться прежним.

Даны протоколы работы почтового клиента на каждом из компьютеров. Напишите программу, проверяющую, можно ли по результатам этого тестирования признать программу правильно работающей.

Формат входного файла

Входной файл содержит несколько наборов входных данных. Первая строка входного файла содержит t — число наборов входных данных. Оставшиеся строки входного файла содержат эти наборы входных данных.

Описание каждого набора начинается со строки, содержащей два целых числа $n(1 \le n \le 50000)$ и $m(1 \le m \le 100000)$. Далее следуют n строк, i-ая из которых содержит протокол работы почтового клиента на i-ом компьютере. Протокол работы состоит из целого числа $k_i(0 \le k_i \le 2m)$ и k_i чисел $a_{i,j}$, описывающих события. Если $a_{i,j} > 0$, то j-ым по счету событием на i-ом компьютере была посылка письма с идентификатором $a_{i,j}$, если же $a_{i,j} < 0$, то j-ым по счету событием на i-ом компьютере было получение письма с идентификатором $-a_{i,j}$. Нулю $a_{i,j}$ равно быть не может.

Идентификатор письма — это целое число от 1 до 10^6. Внутри одного набора входных данных все письма имеют различные идентификаторы. Гарантируется, что все письма, которые были отправлены, были кем-то приняты, то есть сумма всех всех k_i в одном наборе входных данных равна $2m$..

Сумма чисел n по всем наборам входных данных не превосходит 50000, сумма чисел m по всем наборам входных данных не превосходит 100000.

Формат выходного файла

Для каждого набора входных данных выведите ровно одну строку. Эта строка должна содержать слово `YES`, если программу можно считать правильно работающей по результатам тестирования, и `NO` — в противном случае.

Примеры

email.in	email.out
2 2 2 2 1 -2 2 2 -1 2 2 2 -2 1 2 -1 2	YES NO
1 2 3 2 1 -1 4 239 -239 366 -366	YES

Задача C. Неподвижные точки

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

Перестановкой $P[1..n]$ размера n называется набор чисел от 1 до n, расположенных в определенном порядке. При этом в нем должно присутствовать ровно один раз каждое из этих чисел. Примером перестановок являются $1, 3, 4, 5, 2$ (для $n = 5$) и $3, 2, 1$ (для $n = 3$), а, например, $1, 2, 3, 4, 5, 1$ перестановкой не является, так как число 1 встречается два раза.

Число i называется *неподвижной точкой для перестановки* P, если $P[i] = i$. Например, в перестановке $1, 3, 4, 2, 5$ ровно две неподвижных точки: 1 и 5, а перестановка $4, 3, 2, 1$ не имеет неподвижных точек.

Даны два числа: n и k. Найдите количество перестановок размера n с ровно k неподвижными точками.

Формат входного файла

Входной файл содержит два целых числа n ($1 \le n \le 9$) и k ($0 \le k \le n$).

Формат выходного файла

В выходной файл выведите ответ на задачу.

Примеры

fixpoint.in	fixpoint.out
5 2	20
9 6	168
2 1	0
9 0	133496

Задача D. Художник

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

Известный художник решил написать новый шедевр. После многих дней усердной работы он захотел исследовать свое творение.

Художник вспомнил, что картина писалась следующим образом. Сначала был взят белый холст, имеющий форму прямоугольника шириной w и высотой h. Затем художник нарисовал на этом холсте n прямоугольников с координатами углов (x_i^I, y_i^I), (x_i^I, y_i^{II}), (x_i^{II}, y_i^{II}), (x_i^{II}, y_i^I).

Помогите художнику определить площадь незакрашенной части холста.

Формат входного файла

Первая строка содержит два целых числа w и h $(1 \le w, h \le 100)$ — ширину и высоту холста соответственно. Вторая строка входного файла содержит целое число n $(0 \le n \le 5000)$ — количество прямоугольников. Следующие n строк содержат информацию о прямоугольниках. $i + 2$-ая строка содержит четыре целых числа x_i^I, y_i^I, x_i^{II}, y_i^{II} $(0 \le x_i^I < x_i^{II} \le w, 0 \le y_i^I < y_i^{II} \le h)$.

Формат выходного файла

В первой строке выходного файла выведите ответ на задачу.

Примеры

painter.in	painter.out
5 5 2 1 1 3 3 2 2 4 4	18
6 7 3 0 0 5 5 1 1 4 4 2 2 3 3	17

Задача E. SMS

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

В наше время непросто найти человека, который ни разу в жизни не использовал мобильный телефон для отправления текстовых сообщений. Старые телефоны поддерживали только один способ набора текста, упрощенно описывающийся следующими правилами: Каждой из восьми кнопок от «2» до «9» ставится в соответствие несколько букв, а кнопка «1» отводится для знаков препинания. Для ввода первого из соответствующих кнопке символов, ее надо нажать один раз, для ввода второго — два раза и так далее. Если два подряд идущие символа сопоставлены одной кнопке, то после ввода первого из них можно либо подождать, либо нажать на кнопку перемещения курсора (второй вариант оказывается быстрее). Для переключения регистра букв используется кнопка «#». Кроме того, для удобства, после ввода вопросительного и восклицательного знаков, а так же точки (если на этот момент включен нижний регистр), активируется режим «первой заглавной буквы». При этом следующая буква автоматически печатается заглавной, после чего опять включается нижний регистр. Пробел ставится нажатием кнопки «0». В последнее время в Интернете стали появляться стали результаты различных исследований, доказывающих неэффективность обычной раскладки телефонной клавиатуры, в которой буквы сопоставляются цифрам в алфавитном порядке. Составьте программу, которая по заданному сопоставлению букв кнопкам, будет находить минимальное количество нажатий, необходимых для ввода данного текста.

Формат входного файла

Строки с первой по девятую входного файла задают символы, сопоставленные соответствующим кнопкам. Следующая строка содержит сообщение длиной от 1 до 1000 символов.

Формат выходного файла

В выходной файл выведите единственное число - минимальное количество нажатий на кнопки, требующееся для ввода сообщения.

Примеры

sms.in
.?! abc def ghi jkl mno pqrs tuv wxyz Hello. How do you do? i hope everything is fine See ya!
sms.out
116

Задача F. Преобразователь строк

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

Преобразователь строк — это специальная компьютерная программа. Она получает на вход строку S и набор правил преобразования строки. Каждое из правил имеет вид $c_1c_2 \to \varepsilon$, где c_1 и c_2 — маленькие буквы латинского алфавита. Символом ε здесь обозначена пустая строка. При этом каждый символ присутствует не более, чем в одном правиле.

Преобразователь строк работает по шагам. За один шаг он находит в текущей строке подстроку, совпадающую с правой частью одного из правил, после чего эта подстрока удаляется из текущей строки (этот процесс называется применением правила). Этот процесс продолжается до тех пор, пока существует правило, которое можно применить. Строка, которая остается к моменту, когда нельзя применить никакое правило, считается результатом T работы преобразователя строк.

Пусть, например, набор правил таков: $\{ab \to \varepsilon, cd \to \varepsilon\}$, а исходная строка $S = aabbccdba$. Тогда работа преобразователя будет выглядеть так: $aabbccdba \to abccdba \to ccdba \to cba$, и результатом T работы преобразователя будет строка cba.

Ваша задача состоит в том, чтобы написать программу, моделирующую работу преобразователя строк с заданным набором правил на заданной строке S.

Формат входного файла

Первая строка входного файла содержит целое число n ($0 \le n \le 13$) — количество правил. Далее идут n строк, каждая из которых описывает одно из правил. Гарантируется, что каждая из маленьких букв латинского алфавита присутствует не более, чем в одном правиле.

Последняя, $(n+2)$-ая строка входного файла содержит исходную строку S. Она не пуста и состоит только из маленьких букв латинского алфавита. Длина S не превосходит 100000.

Формат выходного файла

В выходной файл выведите строку T — результат работы преобразователя на строке S.

Примеры

string.in	string.out
2 ab cd aabbccdba	cba
0 abcdefghijklmnopqrstuvwxyz	abcdefghijklmnopqrstuvwxyz

Задача G. Симметрия

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

Многие из вас, вероятно, знакомы с понятием *симметрии* относительно прямой. Пусть на плоскости расположена прямая L и точка A. Точка B называется *симметричной* точке A относительно прямой L, если отрезок AB перпендикулярен прямой L и делится пополам точкой пересечения с ней. В частности, если точка A лежит на прямой L, то точка B совпадает с точкой A.

Задана прямая L, параллельная одной из осей координат, и точка A. Найдите точку B, симметричную A относительно L.

Формат входного файла

Первая строка входного файла содержит 4 числа: $x_{L1}, y_{L1}, x_{L2}, y_{L2}$ — координаты двух различных точек, через которые проходит прямая L. Вторая строка входного файла содержит 2 числа x_A и y_A — координаты точки A.

Все числа во входном файле целые и не превосходят 10^8 по модулю.

Формат выходного файла

В выходной файл выведите два числа: x_B и y_B — координаты точки B.

Примеры

symmetry.in	symmetry.out
0 0 0 1 10 10	-10 10
0 0 1 0 10 10	10 -10

Задача H. Игра в слова

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

Существует множество игр, которые называются «игра в слова». В одной из вариаций правила следующие: выбирается длинное слово, после чего два игрока пытаются вспомнить все слова, которые можно составить, использовав некоторые буквы загаданного слова. После этого они по очереди называют придуманные слова (называть одно слово два раза запрещается). Если первый игрок назвал больше слов, то он побеждает, если у обоих игроков слова закончились одновременно, засчитывается ничья, в оставшемся случае побеждает второй.

Оказывается, в этой игре важно не только знать много слов, но и придерживаться правильной стратегии. Напишите программу, которая будет узнавать, кто победит, если оба игрока будут играть идеально, если первый игрок будет играть оптимально, а второй — наихудшим образом, и если первый игрок будет поддаваться. При этом можно считать, что, придумав слова, игроки записывают их на бумагу, и каждый видит записи предыдущего. Если на ходу того игрока, который поддается, у него остаются не упоминавшиеся ранее ранее слова, он обязан назвать одно из них.

Формат входного файла

В первой строке входного файла записано загаданное слово. Затем описываются слова, которые знает первый игрок — на отдельной строке целое число n_1 ($0 \le n_1 \le 10000$), за которым следуют n_1 слов по одному на строке. После этого задается число n_2 слов, известных второму игроку, и описываются сами эти слова в таком же формате. Все слова состоят из маленьких букв латинского алфавита, а их длины не превосходят 100 символов.

Формат выходного файла

В выходной файл выведите три числа по одному на строке — ответы для случаев, когда оба игрока играют оптимально и когда поддаются первый и второй игроки соответственно: номер выигрывающего игрока и 0 в случае ничьи.

Примеры

words.in	words.out
internationalization 6 zone oil rent impression noir trail 7 teal creativity rent rain oil zealot zone	0 2 1