Задача A. Задачка по геометрии

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

Петя учится в третьем классе. Его учитель геометрии часто задает ему сложные домашние задания.

На последнем уроке Петя проходил окружности. Он научился рисовать окружность с помощью циркуля.

Выполнив большую часть домашнего задания, Петя столкнулся со следующей проблемой: ему дано два отрезка. Требуется провести окружность, которая пересекла бы внутреннюю часть каждого отрезка ровно один раз.

При этом окружность должна именно *пересекать внутреннюю часть* каждого отрезка, не разрешается, чтобы она касалась отрезка или проходила через его конец.

Помогите Пете доделать его домашнее задание.

Формат входного файла

Входной файл содержит несколько тестовых наборов. Каждый тестовый набор записан на двух строках.

Первая строка тестового набора содержит четыре целых числа $x_{11}, y_{11}, x_{12}, y_{12}$ — координаты концов первого отрезка. Вторая строка содержит четыре целых числа $x_{21}, y_{21}, x_{22}, y_{22}$ — координаты концов второго отрезка.

Входной файл завершается парой строк, каждая из которых содержит по четыре нуля. Эти строки не следует обрабатывать.

Координаты не превышают 100 по модулю.

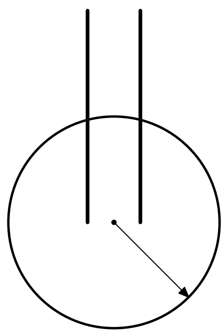
Формат выходного файла

Для каждого тестового набора выведите три вещественных числа — координаты центра и радиус окружности. Все числа в выходном файле не должны по модулю превышать 10^{10}. Жюри выполняет все сравнения вещественных чисел с точностью 10^{-4}.

Пример

geometry.in	geometry.out

`0 0 0 4`
`1 0 1 4`
`0 0 0 0`
`0 0 0 0`
	`0.5 0 2`



Задача B. Ничего и не терялось

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

Разработка новой поисковой системы, которая ведется группой компаний *Giggle*, проходит под лозунгом «Ничего и не терялось». Вы работаете в отделе передовых разработок, и на сегодняшний день вашей задачей является разработка тестовой версии поискового «движка».

Тестовая версия реализует лишь часть функциональности полной версии. В частности, отсутствуют такие функции, как использование логических выражений в запросах, перевод найденных страниц с одного языка на другой и т.д.

Возможности тестовой версии поисковой системы ограничиваются обработкой трех видов запросов: запросов на добавление, на удаление и на поиск. Система работает следующим образом. В любой момент времени существует множество известных системе *сайтов*, причем для каждого сайта известно множество *ключевых слов*, встречающихся на нем.

Запрос на добавление содержит ключевое слово и название сайта. При его выполнении ключевое слово добавляется в множество ключевых слов, присутствующих на данном сайте. Если этого слова в соответствующем множестве еще нет, то результатом запроса является `OK`, в противном случае — `Already exists`.

Запрос на удаление содержит ключевое слово и название сайта. При его выполнении ключевое слово исключается из множества ключевых слов, присутствующих на данном сайте. Если этого слова в соответствующем множестве нет, то результатом запроса является `Not found`, в противном случае — `OK`.

Запрос на поиск содержит только ключевое слово. Результатом запроса является лексикографически отсортированный список сайтов, содержащих данное ключевое слово. При этом в результат выводятся только первые 10 сайтов из этого списка.

Задан список запросов. Необходимо вывести результат их последовательного выполнения.

Формат входного файла

Первая строка входного файла содержит целое число n — количество запросов, которые необходимо обработать ($0 \le n \le 2500$). Каждая из последующих n строк содержит запрос.

Запрос на добавление имеет следующий формат:

`Add keyword “<keyword>” to <site>`, где `<keyword>` — ключевое слово `<site>` — название сайта, на котором появилось это ключевое слово.

Запрос на удаление имеет следующий формат:

`Remove keyword “<keyword>” from <site>`, где `<keyword>` — ключевое слово `<site>` — название сайта, на котором появилось это ключевое слово.

Запрос на поиск имеет следующий формат:

`Search “<keyword>”`, где `<keyword>` — ключевое слово.

Все ключевые слова (`<keyword>`) состоят из строчных букв латинского алфавита. Длины ключевых слов не превосходят 30 символов.

Все названия сайтов (`<site>`) состоят из строчных букв латинского алфавита, символов «косая черта» («/») и точек («.»). Длины названий сайтов не превосходят 100 символов.

Формат выходного файла

Для каждого запроса выведите его результат. При этом придерживайтесь формата, приведенного в примере.

Результаты запросов разделяйте строкой из пяти символов «равно» («=»).

Примеры

giggle.in

```
12
Add keyword "olympiads" to neerc.ifmo.ru/school/io
Add keyword "neerc" to neerc.ifmo.ru
Search "olympiads"
Search "neerc"
Add keyword "olympiads" to neerc.ifmo.ru
Search "olympiads"
Add keyword "olympiads" to neerc.ifmo.ru/school/io
Remove keyword "olympiads" from neerc.ifmo.ru/school/io
Search "olympiads"
Remove keyword "olymp" from neerc.ifmo.ru
Remove keyword "olympiads" from neerc.ifmo.ru
Search "olympiads"
```

giggle.out

```
OK
=====
OK
=====
Results: 1 site(s) found
1) neerc.ifmo.ru/school/io
=====
Results: 1 site(s) found
1) neerc.ifmo.ru
=====
OK
=====
Results: 2 site(s) found
1) neerc.ifmo.ru
2) neerc.ifmo.ru/school/io
=====
Already exists
=====
OK
=====
Results: 1 site(s) found
1) neerc.ifmo.ru
=====
Not found
=====
OK
=====
Results: 0 site(s) found
```

giggle.in
12 Add keyword "keyword" to site01 Add keyword "keyword" to site02 Add keyword "keyword" to site03 Add keyword "keyword" to site04 Add keyword "keyword" to site05 Add keyword "keyword" to site06 Add keyword "keyword" to site07 Add keyword "keyword" to site08 Add keyword "keyword" to site09 Add keyword "keyword" to site10 Add keyword "keyword" to site11 Search "keyword"
giggle.out
OK ===== OK ===== OK ===== OK ===== OK ===== OK ===== OK ===== OK ===== OK ===== OK ===== OK ===== Results: 11 site(s) found 1) site01 2) site02 3) site03 4) site04 5) site05 6) site06 7) site07 8) site08 9) site09 10) site10

Задача C. Главные дороги

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

Город, в котором живет Гоша, имеет n перекрестков, некоторые из которых соединены дорогами. По каждой дороге разрешено движение в обоих направлениях.

Каждый день Гоша ездит на машине из дома на работу и обратно. Но дороги в городе, где живет Гоша, не очень хорошие, они часто портятся и приходится их ремонтировать. Гоша заметил, что когда некоторые дороги закрывают на ремонт, часто он все равно может доехать из дома до работы за то же время, что в случае, когда ни одна дорога на ремонт не закрыта.

С другой стороны, встречаются дороги, такие что когда они закрываются на ремонт, время, которое требуется Гоше, чтобы добраться от дома до работы, увеличивается. А иногда Гоша просто не может доехать от дома до работы на машине. Такие Гоша называет *главными*.

Помогите Гоше найти все главные дороги в городе.

Формат входного файла

Первая строка входного файла содержит n и m — количество перекрестков и дорог в городе, соответственно ($2 \le n \le 20\,000$, $1 \le m \le 100\,000$). Гоша живет около перекрестка 1, а работает у перекрестка n.

Следующие m строк содержат информацию дорогах. Каждая дорога описывается двумя перекрестками, которые она соединяет, и временем, которое необходимо Гоше, чтобы проехать по этой дороге от одного конца до другого. Время проезда по дороге — положительное целое число, не превышающее 100 000. Между парой перекрестков может быть несколько дорого, но никакая дорога не соединяет перекресток с самим собой.

Гарантируется, что если все дороги доступны, Гоша может добраться от дома до работы.

Формат выходного файла

Выведите l — количество главных дорог — на первой строк выходного файла. На второй строке выведите l чисел — номера главных дорог. Дороги пронумерованы от 1 до m в том порядке, в котором они заданы во входном файле.

Пример

important.in	important.out
6 7 1 2 1 2 3 1 2 5 3 1 3 2 3 5 1 2 4 1 5 6 2	2 5 7

Задача D. Мафия в городе

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

Об этом еще никто не знает, но многие догадываются — мафия уже в городе. Поговаривают, что в планах главы мафиозного клана захват контроля над всем городом, однако поначалу он решил ограничиться захватом основных линий связи города.

В городе находятся n базовых телефонных станций, некоторые пары которых соединены двусторонними каналами связи. Для удобства, занумеруем базовые станции целыми числами от 1 до n, канал связи в этом случае задается парой чисел (u, v) — номерами станций, которые он соединяет.

Будем говорить, что канал связи (u, v) *контролируется* мафией, если захвачена либо станция u, либо станция v (либо обе).

Глава мафиозного клана хочет контролировать все каналы связи, захватив при этом как можно меньше базовых станций. Ваша задача — помочь службе безопасности телефонной компании, составив возможный план захвата и определив количество таких планов.

Формат входного файла

Первая строка входного файла содержит два целых числа: n и m ($2 \le n \le 18$, $0 \le m$). Каждая из последующих m строк описывает один канал связи и содержит по два целых числа: u и v ($1 \le u, v \le n$, $u \ne v$) — номера базовых станций, соединенных этим каналом связи. Любая пара станций соединена не более, чем одним каналом.

Формат выходного файла

В первой строке входного файла выведите два числа: k и c — соответственно, минимальное количество базовых станций, которые необходимо захватить для того, чтобы контролировать все каналы связи, и число способов захватить такое количество станций, так чтобы контролировать все каналы связи.

Во второй строке входного файла выведите k чисел — номера базовых станций, соответствующих одному из способов захвата.

Примеры

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

В первом примере существует три способа захватить две станции так, чтобы контролировать все каналы связи: $\{1, 2\}$, $\{1, 3\}$, $\{2, 3\}$.

Задача E. Странный ним

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

Алиса и Боб играют в игру, которую они называют *странный ним*. На столе расположено три кучки камней, содержащие a, b и c камней, соответственно. Алиса ходит первой.

Тот игрок, который делает ход, выбирает одну из кучек и берет из нее несколько камней. При этом должно выполняться следующее условие: после хода каждого игрока на столе не должно быть двух кучек, содержащих равное количество камней. Забирать все камни из кучки разрешается.

Тот, кто берет последний камень, выигрывает.

Пусть, например, кучки содержат 1, 3 и 5 камней, тогда разрешены следующие ходы:

- взять 1 камень из первой кучки;
- взять 1 камень из первой кучки;
- взять 3 камня из первой кучки;
- взять 1 камень из первой кучки;
- взять 3 камня из первой кучки;
- взять 5 камней из первой кучки.

По заданным a, b и c, определите, кто выиграет, если оба игрока играют оптимально.

Формат входного файла

Входной файл содержит несколько тестовых примеров. Каждый тестовый пример состоит из трех целых чисел a, b и c, расположенных на одной строке ($1 \le a, b, c \le 10^9$, $a \ne b$, $a \ne c$, $b \ne c$). Последняя строка входного файла содержит три нуля, ее обрабатывать не требуется.

Формат выходного файла

Для каждого тестового примера выведите, кто выиграет в игре, если оба игрока играют оптимально.

Пример

nim.in	nim.out

`1 2 3`
`1 3 5`
`0 0 0`
	`Alice wins the game.`
	`Bob wins the game.`

Задача F. Числа

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

Рассмотри числа от 1 до n.

Требуется найти минимальное в лексикографическом порядке число, которое делится на k.

Формат входного файла

Входной файл содержит несколько тестовых примеров. Каждый тестовый пример расположен на одной строке и состоит из двух целых чисел n и k ($1 \le n \le 10^{18}$, $1 \le k \le n$). Последняя строка содержит два нуля. Их обрабатывать не требуется.

Формат выходного файла

Для каждого тестового примера выведите одно целое число — наименьшее в лексикографическом порядке число не превышающее n, которое делится на k.

Пример

numbers.in	numbers.out
2000 17 2000 20 2000 22 0 0	1003 100 1012

Задача G. Строки

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

Определим *расстояние* между равными по длине строками S_A и S_B (обозначим $d(S_A, S_B)$) как сумму для всех $1 \le i \le |S_A|$ кратчайших расстояний между буквами $S_A(i)$ и $S_B(i)$ в циклически замкнутом латинском алфавите (т.е. после буквы «a» идет буква «b», …, после буквы «z» идет «a»). Например $d(aba, aca) = 1$, а $d(aba, zbz) = 2$.

Напомним, что *циклическим сдвигом* строки S называется строка (обозначамим, как $S \hookrightarrow k$) $S_k S_{k+1} S_{k+2} ... S_{|S|} S_1 S_2 ... S_{k-1}$ для некоторого k, где $|S|$ — длина строки S.

Степенью циклического расстояния между строками S_A и S_B ($|S_A| = |S_B|$) называется сумма:

$$\sum_{i=1}^{|S_A|} \sum_{j=1}^{|S_B|} d(S_A \hookrightarrow i, S_B \hookrightarrow j)$$

Ваша задача посчитать *степень циклического расстояния* заданных строк S_A и S_B.

Формат входного файла

В первой и второй строке расположены две строки равной длины, не превышающей 100000 символов. Строки состоят только из маленьких букв латинского алфавита.

Формат выходного файла

В выходной файл выведите ответ на задачу.

Примеры

strings.in	strings.out
a b	1
ab ac	8

Разберем второй пример. Все циклические сдвиги строки «ab» — «ab» и «ba». Все циклические сдвиги строки «ac» — «ac» и «ca». Искомое значение равно $d(ab, ac) + d(ab, ca) + d(ba, ac) + d(ba, ca) = 1 + 3 + 3 + 1 = 8$

Задача H. Сумма или произведение

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

Федя недавно начал учиться математике. Он научился складывать и умножать и был немедленно потрясен фактом, что $2 + 2 = 2 \times 2$. Теперь он хочет найти еще примеры таких соотношений.

Петя называет набор чисел *интересным*, если произведение чисел из этого набора равно их сумме. Например, наборы $\{2, 2\}$, $\{5\}$, $\{1, 2, 3\}$ — интересные, а набор $\{2, 3\}$ — нет.

По заданному n, Федя хочет найти количество интересных наборов, содержащих n чисел. Помогите ему!

Формат входного файла

Первая строка входного файла содержит n ($2 \le n \le 500$).

Формат выходного файла

Выведите одно число — количество интересных наборов, содержащих n чисел.

Пример

sump.in	sump.out
2	1
5	3

В последнем примере интересные наборы следующие: $\{1, 1, 1, 2, 5\}$, $\{1, 1, 1, 3, 3\}$ и $\{1, 1, 2, 2, 2\}$.

Задача I. Поезда

Имя входного файла:	`trains.in`
Имя выходного файла:	`trains.out`
Ограничение по времени:	2 seconds
Ограничение по памяти:	64 megabytes

Во Флатландии расположено n городов. Недавно король Флатландии побывал в Японии и был поражен скоростными поездами *Shinkansen*, сеть которых связывает всю страну. Теперь он решил построить сеть высокоскоростных поездов во Флатландии.

Каждый маршрут скоростного поезда будет соединять ровно два города во Флатландии. Хотя реальной необходимости в скоростных поездах во Флатландии нет, король повелел, чтобы хотя бы одна линия скоростного поезда была в каждом городе.

Министр транспорта Флатландии сказал королю, что существует несколько сетей маршрутов поездов, удовлетворяющих данному свойству. Король был поражен этим фактом и попросил министра подсчитать количество возможных сетей. Помогите ему!

Формат входного файла

Входной файл содержит целое число n ($2 \le n \le 100$).

Формат выходного файла

Выведите количество сетей скоростных поездов, которые удовлетворяют требованиям короля.

Пример

trains.in	trains.out
4	41