Problem 2. Ленивый художник

Input file:	`lazy.in`
Output file:	`lazy.out`
Time limit:	2 seconds
Memory limit:	64 megabytes

Говорят, преклонных лет художник Калевич невероятно обленился! Ему лень даже передвигать руку для рисования картин! Да что говорить: ему лень поднять руку от холста! Однако, при всем при этом Калевич не прекращает рисовать картины, и он уже задумал новый шедевр. Он решил нарисовать на холсте N отрезков! Гениально! Восхитительно! Современное искусство еще не знало такого полета мысли!

Художник уже придумал как конкретно будут расположены на холсте отрезки. Единственное что его смущает, это тот объем работы, который ему предстоит. Объем работы, по мнению мастера, равен минимальному количеству раз отрыва руки от холста, чтобы нарисовать картину, если он хочет, чтобы каждая линия на картине была прорисована ровно K раз. Заметим, что отрезки могут накладываться и их пересечение рассматривается как одна линия.

Input

В первой строке входного файла записана пара натуральных чисел N, K ($1 \le N \le 64, 1 \le K \le 10^6$). Далее в файле заданы N отрезков парами координат своих концов. Отрезки не могут вырождаться в точки. Все координаты целые, не превосходящие 10^6 по абсолютной величине.

Output

Выведите минимальному количеству раз отрыва руки от холста, необходимых для того, чтобы каждая линия на картине была прорисована ровно K раз.

Example

lazy.in	lazy.out
2 1 0 -10 0 10 -1 -1 1 1	1
2 1 0 -10 0 10 0 -5 0 5	0

Problem 3. Американские горки

Input file:	`innovate.in`
Output file:	`innovate.out`
Time limit:	2 секунды
Memory limit:	64 megabytes

— Что это? — перед Иваном Ивановичем на столе лежал лист ватмана, густо исчерченный разнообразными прямыми и закругленными линиями.

— Ну как же, Иван Иванович, вы же сами вчера на банкете приказали к утру приготовить проект новой трассы американских горок. Это он и есть — проект трассы, — ответил первый зам Ивана Иваныча. — Наш web-дизайнер всю ночь рисовал.

— Да? — Иван Иванович смущенно рассматривал ватман. — Господи, сколько же тут поворотов... А длина трассы какая?

— Вы же сами вчера сказали — надо 10 километров. Так и нарисовали. Самая длинная в Европе! 238 поворотов, 56 туннелей. Каждому посетителю в дорогу выдается сухой паёк...

— Паёк-то зачем? — Иван Иванович с трудом вспоминал вчерашнее... Кажется, он действительно что-то такое говорил. — Они же за пять минут ее проедут? Или нет?

Иван Иванович посмотрел на зама, тот в недоумении пожал плечами:

— Дизайнер про это ничего не сказал. Длину-то он худо-бедно посчитал, а вот время... Он же дизайнер, а не математик.

Иван Иванович опытным чутьём руководителя со стажем понял, что задача эта по силам только Настоящему Программисту. Он набрал номер отдела кадров:

«Лидочка, Петров сейчас не в отпуске?»

«Нет, не в отпуске. Должен быть на месте.»

«Давай его ко мне, срочно.» - сказал Иван Иванович и положил трубку. - «Пусть подсчитает время, зря, что ли, его столько лет *считать* учили. В рамках, так сказать, конверсии...»

Через пять минут вошёл программист Петров, с чёрными кругами под глазами, явно невыспавшийся. Было видно, что ночная партия в Star Wars с коллегами, приехавшими из Новосибирска делиться опытом эксплуатации ЕС ЭВМ, отняла у него много сил.

— Вот, — сказал Иван Иванович, — это схема новой трассы американских горок. У технологов возьмете паспорт на вагончик. Через пять часов скажете мне, сколько времени этот вагончик потратит на всю трассу. Меня интересует минимально возможное время. Сначала вагончик стоит вот тут, — Иван Иванович ткнул карандашом в начало трассы, — а в конце должен стоять тут, — и он показал самый ее конец. — Все ясно?

Петров обреченно посмотрел на ватман... Похоже поспать ему сегодня не удастся... Впрочем, он вспомнил, что кроме него есть ещё Настоящие Программисты. Или по крайней мере ученики Настоящих Программистов...

Мы надеемся, что вы не станете на бумажке подсчитывать время на каждый из 238 поворотов и 239 прямых участков трассы. Напишите программу, которая по заданной трассе и паспорту на вагончик сама подсчитает время на всю трассу.

Вам повезло — дизайнер забыл на чертеже указать высоты горок, поэтому можете считать, что вся трасса плоская. В основном трасса состоит из прямолинейных участков, причем начало трассы и конец приходятся как раз на них. Никакие два последовательных прямолинейных участка не имеют одинакового направления. Между двумя последовательными отрезками трасса должна идти по дуге окружности так, чтобы эти отрезки являлись касательными к окружности.

Да, кстати, дизайнер запросто мог ошибиться и нарисовать подряд два отрезка, которые на самом деле нельзя сопрячь никакой дугой. В таких случаях надо продлить один из этих отрезков, чтобы их все-таки можно было соединить дугой. Гарантируется, что это всегда возможно. Кроме того, гарантируется, что вам не придется соединять прямые участки дугами нулевого или бесконечного радиуса.

Паспорт на вагончик аттракциона “Американские горки”
Максимальная разрешённая скорость: 10 м/с Максимальное разрешённое ускорение: - вдоль трассы: 1 м/с ^{2} - центростремительное: 1 м/с^2 Максимальное разрешённое ускорение торможения совпадает с максимальным разрешённым ускорением разгона.

Для тех, кто забыл, центростремительное ускорение вычисляется как V^2/R, где V — скорость движения, а R — радиус поворота. Для тех, кто не забыл, впрочем, оно тоже этому равняется.

Input

В первой строке находится единственное число N ($1 \leqslant N \leqslant 239$) — количество прямых отрезков трассы. В следующих N строках идут координаты концов этих отрезков, по четыре числа в строке. Все числа целые, разделены пробелами и/или переводами строк, и не превосходят по модулю 1000. Все отрезки ненулевой длины.

Output

Необходимо вывести минимальное время на преодоление трассы, с учетом остановки в самом конце, в секундах с точностью 3 знака после десятичной точки. Кроме того, дизайнер гарантирует, что ответ совершенно точно не будет превосходить 500000 секунд (≈ 5.8 суток).

Example

innovate.in	innovate.out
1 0 0 4 0	4.000
2 -2 4 0 4 4 0 4 -2	7.142

Problem 5. Монумент.

Input file:	`oir.in`
Output file:	`oir.out`
Time limit:	2 секунды
Memory limit:	64 megabytes

После триумфа ЦСКА было принято решение воздвигнуть в честь первого триумфа российской команды в еврокубках памятную стелу в Москве. Естественно, стелу было доверено изготовить известному московскому скульптору. Стела должна была отражать путь команды по турнирной сетке вверх, к результатам, и вширь, к совершенству. Под влиянием гравюр не менее известного голландского художника, скульптор взял за основу эмблемы фрагмент периодической решётки, которая представляет собой поверхность фигуры, состоящей из единичных кубиков с координатами $(2k, 2l, 2m)$, $(2k+1, 2l, 2m)$, $(2k, 2l+1, 2m)$, $(2k, 2l, 2m+1)$ для всех целочисленных троек (k, l, m). После того, как внешние очертания эмблемы приобрели достойный, с точки зрения скульптора, вид (а главное — размер), внутри сооружения осталось большое пространство, заполненное элементами этой решётки (каждая её грань — это квадрат со стороной 1 метр).

Монтаж сооружения осуществлялся рядом с ипподромом днём и ночью. Рабочие, которые были приглашены на стройку, с помощью специальных подвесных устройств «Муха-2005» могли перемещаться по граням сооружения изнутри. По замыслу скульптора, на каждой грани изнутри должно было быть выгравировано «ЦСКА-чемпион», для каждой грани — особым шрифтом. И вот сооружение закончено. К уставшим рабочим подошёл известный в прошлом тренер многократных чемпионов страны, а ныне безработный. Он был слегка навеселе — отмечал свою очередную отставку. Тренер спросил монтажников, что это за сооружение. Ему объяснили. Тогда он спросил, как это они умудряются перемещаться по граням сооружения, даже «по стенам и потолку», как мухи. А главное — зачем? Ему объяснили про надписи и показали подвесные устройства. И тут он нацепил одно из них и полез внутрь со словами «Я вам устрою чемпиона!»... Прицепившись к некоторой грани, он начал движение. Маршрут тренера состоял из переползаний с грани на соседнюю грань, каждому из которых предшествовал поворот направо или налево, либо его отсутствие. Так как тренер был «под мухой», то новые ощущения ему понравились, и он ушёл куда-то далеко, мелом делая исправления к надписям.

А тем временем уже настало утро... очередной команде, оказавшейся в кризисе, потребовался «тренер с именем». Звонок по мобильному показал всю сложность ситуации — оказалось, что он не только не помнит дороги, но и не представляет, далеко ли он ушёл. Известно только, что он ни разу не выполз на внешнюю поверхность эмблемы (снаружи шёл дождь, и тренер тут же протрезвел бы). Руководство РФПЛ поставило перед Вами задачу — подсчитать, какое наименьшее число переползаний нужно совершить тренеру, чтобы вернуться в ту точку, где расположен выход из сооружения (то есть грань, на которой он забрался в монумент). Возвращаться он должен из того же положения (включая направление) в котором он оказался в момент звонка.

Input

Строка, состоящая из символов F, L, R, задающая маршрут тренера внутри сооружения. (R — повернуть направо и переползти на соседнюю грань, L — повернуть налево и переползти на соседнюю грань, F — переползти на соседнюю грань, никуда не сворачивая). Длина строки не больше 1000.

Output

Целое число — минимальная длина маршрута, который должен по тем же правилам пройти тренер, чтобы вернуться в точку старта.

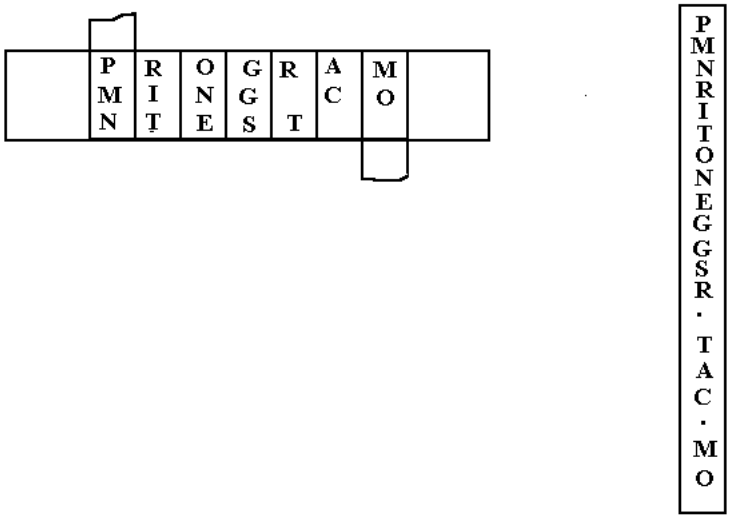
Example

oir.in	oir.out
FF	2
RL	4
RFFF	0

Problem 7. Дисквалификация тренера.

Input file:	`coach.in`
Output file:	`coach.out`
Time limit:	2 секунды
Memory limit:	64 megabytes

После того, как главный тренер ФК «Чукотка» Море Жозинью в интервью обвинил судей в том, что они перед перерывом перемигивались с главным тренером команды оппонентов, после чего во втором тайме был удалён ключевой игрок «Чукотки», было проведено расследование. Выяснилось, что судьи не могли перемигиваться, так как весь матч провели с повязкой на глазах. После чего Жозинью за ложные обвинения был дисквалифицирован на два матча. То есть на его общение с командой во время матча был наложен запрет. Максимум, что он мог сделать — сидеть на трибунах, как простой болельщик. Однако Море нашёл выход из положения. Если полоску бумаги намотать спиралью на палочку и написать на этой бумаге вдоль палочки текст, то, после снятия полоски, буквы на ней располагаются так, что первоначальный текст сообщения прочитать без ключа невозможно. Ключом является сама палочка нужного диаметра. (этот шифр придуман древними спартанцами и называется «скитала» по названию стержня, на который наматывались свитки папируса).



Тренер написал распоряжения относительно изменений в тактике команды таким образом, а затем просто бросил полоску бумаги, как обычный «серпантин», часто бросаемый болельщиками. Сидевший на скамейке его помощник как бы случайно решил поднять полоску... Видеокамера зафиксировала этот момент, и после матча записка была найдена. Выяснили, что размер всех букв в записке одинаков, то есть расшифровка текста сводится к поиску числа букв, которые размещаются на длине окружности стержня. Задачу облегчает тот факт, что известно, что в тексте записки содержится фамилия игрока, который сразу же после этого был заменён, или хотя бы первые несколько букв из этой фамилии.

Ваша задача — написать программу, которая найдет ключ шифровки — число букв на окружности стержня. К сожалению, конец ленты с последним символом размок, но в результате экспертизы стало известно, что буквы там не было, хотя точка, соответствующая пробелу, могла там находиться.

Input

В первой строке исходного файла содержится текст тайного сообщения. Длина строки не превышает 10 000 символов. Следующая строка содержит имя заменённого футболиста. Длина имени — не более 100 символов (имена африканских футболистов часто бывают достаточно причудливы). Зашифрованный текст, а также имя футболиста записаны только большими латинскими буквами

(и знаком точки вместо пробела). Знак точки, соответствующей последнему пробелу, может отсутствовать.

Output

Вы должны вывести два целых числа n и m: n — число символов текста, размещаемых на длине окружности стержня, при котором в тайнописи содержится максимально длинное начало имени заменённого игрока (известно, что главный тренер — большой педант и сокращений скорее не любит), m — соответствующая максимальная длина подстроки ($m \geq 2$). Если существует несколько решений, Вы должны вывести минимально возможное значение n, при котором m имеет максимальное значение из всех возможных. Если решения нет, Вы должны вывести число -1 вместо каждого из значений n и m.

Example

coach.in	coach.out
PMNRITONEGGSR.TAC.MO GRAMMA	3 5
PMNRITONEGGSR.TAC.MO NZNR	-1 -1

Problem 11. Захват власти

Input file:	`infiltr.in`
Output file:	`infiltr.out`
Time limit:	2 секунды
Memory limit:	64 megabytes

А тем временем Саруман планирует хитростью захватить власть над соседней Ристанией.

В Ристании N поселений, пронумерованных от 1 до N. Между некоторыми парами поселений проложены двунаправленные дороги, общее количество которых составляет $N-1$. Дороги пересекаются только в поселениях. Из любого поселения можно добраться по дорогам в любое другое.

Исследования, проведенные Саруманом, показали, что любое поселение Ристании можно захватить, направив туда "`гнилоуста`", т.е. человека, который будет ходить по улицам поселения и убеждать людей, что им будет выгоднее делать то, что им скажет мудрый маг Саруман. Так как ристанийцы храбры, но простодушны, то через конечное время они поверят словам "`гнилоуста`" и без сопротивления подчинятся Саруману. Однако поселение поселению рознь - в одних словам "`гнилоуста`" верят быстрее, в других - медленнее. Для каждого поселения известно время его захвата, т.е. время, которое пройдет между моментом, когда Саруман направил в поселение "`гнилоуста`", и моментом, когда поселение подчинилось.

Однако ристанийцы могут заподозрить неладное, если одновременно в двух или более поселениях будут вести свою деятельность "`гнилоусты`" — ведь всё должно выглядеть, как случайность. Поэтому Саруман решил захватывать поселения по очереди. Как только захват очередного поселения осуществлен, "`гнилоуст`" сразу же направляется в следующее место, который нужно захватить.

Кроме того, с целью обеспечения маневренности своей армии, Саруман хочет, чтобы в любой момент времени между любыми двумя захваченными им поселениями существовал путь, все промежуточные поселения в котором также верны Саруману.

Захват Ристании начинается в момент времени 0, когда "`гнилоуст`" направляется в поселение 1. За захват каждого поселения "`гнилоуст`" получает премию. Если очередное поселение было захвачено в момент времени T, то премия равна T денежных единиц.

Саруман координирует все действия "`гнилоуста`". В частности, он решает после захвата очередного поселения, в какое следующее поселение направится "`гнилоуст`". Учиывая ограниченность ресурсов Ортханка по сравнению как с Раздолом, так и с Мордором, Саруман очень экономен. Поэтому он хочет управлять действиями "`гнилоуста`" так, чтобы суммарная заплаченная ему премия после захвата Ристании была минимальной возможной.

Ристания считается захваченной, когда все N посёлков в ней захвачены.

Временем перемещения "`гнилоуста`" между посёлками можно пренебречь.

Input

Входной файл содержит описание Ристании. В первой строке файла записано целое число N ($1 \le N \le 100000$), равное количеству посёлков в стране. Во второй строке записаны N натуральных чисел, не превышающих 100000, разделенных пробелами. i-е число во второй строке входного файла равно времени захвата посёлка i.

Следующие $N-1$ строк файла описывают дороги. Каждая из этих строк содержит два целых числа A и B ($1 \le A, B \le N$, $A \ne B$), задающих дорогу между посёлками A и B.

Output

Выходной файл должен содержать одно целое число, равное минимальной возможной суммарной премии, заплаченной "`гнилоусту`" после захвата Ристании.

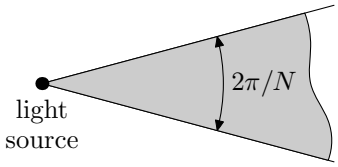
Example

infiltr.in	infiltr.out
7 1 5 10 10 10 1 1 1 2 1 3 2 4 2 5 3 6 3 7	121

Problem 13. Illumination

Input file:	`ill.in`
Output file:	`ill.out`
Time limit:	2 seconds
Memory limit:	64 megabytes

You are given N light sources on the plane, each of which illuminates the angle of $2\pi/N$ with the vertex in the source point (including its sides).



You must choose the direction of the illuminated angle for each of these sources, so that the whole plane is illuminated. It can be proved that this is always possible.

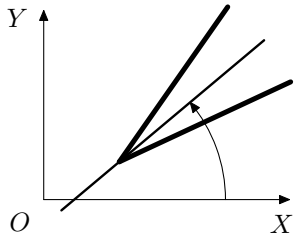
A light source itself casts no shadow and does not interfere with light beams from the other sources.

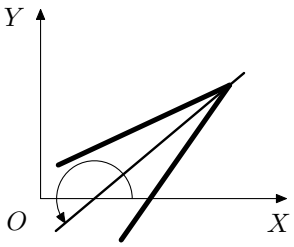
Input

The first line of the input file contains N — the number of light sources ($3 \le N \le 30$). Next N lines contain two integer numbers each — the coordinates of the light sources. All coordinates do not exceed 100 by their absolute value. No two sources coincide.

Output

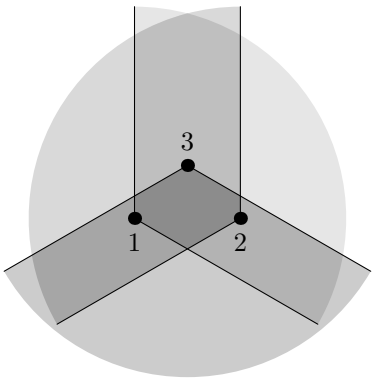
Print N real numbers — for each light source specify an angle that the bisector of the illuminated angle makes with OX axis, counterclockwise. Print at least six digits after the decimal point. No angle must exceed 100π by its absolute value.





Example

ill.in	ill.out
3 0 0 2 0 1 1	0.52359877559829887 2.61799387799149437 4.71238898038468986



Problem 17. Дорвались!

Input file:	input.txt
Output file:	output.txt
Time limit:	60 секунд
Memory limit:	64 megabytes

После того, как Тройка была изгнана, Кристобаль Хозевич Хунта в сопровождении Привалова и Амперяна проследовал в музей Колонии. Оказалось, что рачительный Зубо запаковал все ценные экспонаты в N пронумерованных коробок одинакового размера ($1\times1\times1$ м^3), но разной массы.

Судя по прилагавшейся инструкции, коробки нужно составить в один ряд вдоль стены длины L. Коробки можно ставить (переставлять) друг на друга, но верхняя коробка должна быть легче нижней. Коробки не должны быть подняты выше, чем на H метров.

Составленные коробки надо распаковать в порядке возрастания их номеров, начиная с первой. Для распаковки можно брать только коробки, на которых ничего не стоит, при этом распакованная коробка убирается. Также коробки, на которых ничего не стоит сверху, можно переставлять на свободные места вдоль стены или на другие коробки, соблюдая правила.

Так как тратить время на перекладывание коробок, осознавая, насколько интересные вещи могут оказаться внутри, глупо, то ваша задача — составить коробки таким образом, чтобы за минимальное количество перестановок можно было бы распаковать все коробки.

Input

В первой строке входного файла задаются три натуральных числа N, L, H ($1 \le N \le 16$, $1 \le L$, $H \le 4$).

В следующей строке задано N целых чисел, записанных через пробел — массы коробок в порядке возрастания номеров коробок, масса каждой коробки не превосходит 100. Массы всех коробок попарно различны.

Output

В выходной файл нужно вывести одно целое число — минимальное количество перестановок. Если расставить коробки невозможно, то вывести число **–1**.

Example

input.txt	output.txt
3 2 2 10 8 6	1

Problem 19. Canteen.

Input file:	`c.in`
Output file:	`c.out`
Time limit:	2 секунды
Memory limit:	64 megabytes

The canteen in some university is used by students as well as by the lecturers. Sometimes it is possible to view the situation when one person is more important than all the others. For example if all people standing in the queue are students and suddenly a lecturer comes, than he will be first admitted to the food window and will leave the queue first as the most important person. It is obvious, that different lecturers have different importance - masters are less important than doctors, and both doctors and masters are less important than professors. Seniority plays an important role here, both for lecturers and students. The longer some person works/studies, the more important he is (obviously title is dominating element here, so professor with 1-year seniority is more important than doctor with 50-years seniority). In case when two persons have equal importance, the person first entering the queue will first leave the queue.

Every day first and second dish are served in the canteen. These two types of dishes are served in different food windows, and each of these two windows have its own queue. If a man wants to eat both dishes than he should stand in the first queue, wait, get the first dish, sit down to some table, eat the first dish up, stand in the second queue, wait, get the second dish, eat the second dish up and immediately after this leave the canteen. Sometimes people want to eat only the first dish (then they should leave the canteen immediately after having eaten the first dish) or only the second dish (then they should stand in the second queue, skiping the first one). It sometimes happen that the activity of canteen is finished, but some people haven't eaten their dishes or even haven't started to eat. In this case all these people are enforced to leave the canteent. However, the situation when a person come to the canteen after it is already finished its activity does never happen. The movement of persons in the canteen is regulated by the following rules:

— if a queue is not empty than every second the most important person in the queue (in case when several persons have equal importance the person first entering the queue is chosen among them) comes to the food window, gets the food and leaves the queue;

— a person can potentially get the food at the same time, as it just stands in the queue - it happens if the queue was empty or the person is more important than all people standing in the queue;

— if two or more persons want to stand in some queue at the same moment, than person that came to the canteen earlier than others have the highest priority to stand in this queue;

— for every pair of persons it is possible to determine, which person came to the canteen earlier then other person, even if both persons came to the canteen in the same second (the doors of the canteen are so narrow that it is impossible for two persons to enter in the same moment);

— times that persons spend moving on the canteen are very small and shouldn't be taken into consideration (however times of standing in the queues and times of eating the dishes should).

You are given the time of opening the canteen and the list of persons visiting it. For every person the following data is known: title (if it has any), name, surname, seniority, time of coming to the canteen (calculated from the time of opening the canteen), time of eating the first dish (if it wants to eat the first dish) and time of eating the second dish (if it wants to eat the second dish). Your program should calculate the time of leaving the canteen for each person. All times are measured in seconds.

Input

In the first line of input file two integer numbers N and M are given ($1 \le N \le 50000, 1 \le M \le 10^9$). N is the number of persons visiting the canteen and M is the time when the canteen finishes its activity. The next N lines of the file describe the persons, visiting the canteen (in order of their appearence). Every person is described in single line by the following items (separated by single spaces): title (optional), name, surname, R, T_w, T_z, T_d. Title item is one of the following: $\le$<mgr» (master), $\le$<dr» (doctor),

≤<prof.» (professor). The absence of title means that the person is usual student. Name and surname are the strings composed from 2 to 100 english letters. First letter of name and surname is uppercase, all the other letters are lowercase. As it sometimes happen in real life, there can be two (or more) persons with the same name and surname (and even title). All the other items means correspondingly: seniority (in complete years), time of entering the canteen, time of eating the first dish, time of eating the second dish (times are given in seconds). There are the following constraints on these items: $0 \le R \le 50$, $0 \le T_w \le M$, $0 \le T_z \le 10^9$, $0 \le Td \le 10^9$. Keep attention, that $T_z = 0$ means that person doesn't want to eat the first dish and $T_d = 0$ means that person doesn't want to eat the second dish (both T_z and T_d can't be equal zero).

Output

For every person in the input file you should output its title, name, surname and time of leaving the canteen (in seconds, calculated from the time of opening the canteen). You should output the persons in the same order they are given in the input file.

Example

c.in	c.out
3 100 dr Ccc Ddd 0 0 0 111 mgr Aa Bb 11 22 33 44 prof. Prof Prof 30 30 30 30	dr Ccc Ddd 100 mgr Aa Bb 99 prof. Prof Prof 90
3 1000 Michal Kichal 1 10 15 20 prof. Huhu Ha 50 11 15 25 John Ixinski 1 25 0 22	Michal Kichal 45 prof. Huhu Ha 51 John Ixinski 49

Explanation of examples: The first example is trivial. In the second example Michal enters the canteen on 10th second and gets the first dish at the same time. He finishes eating the first dish after 15 seconds and stands in the second queue on 25th second. Exactle at the same time John enters the canteen and stands in the second queue too, but Michal has higher priority (because he came to the canteen earlier than John and both Michal and John doesn't have any titles and have the same seniority). So Michal gets the second dish exactly on 25th second. John is to wait a second, but exactly on 26th second prof. Huhu Ha finishes eating its first dish and stands on the second queue. He is more important than John, so he gets the food first. At least, on the 27th second poor John gets his second dish. After this nothing interesting happens in the canteen and the moments of leaving the canteen by the persons are calculated very simply.