Задача A. Сушка

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

Тетя Люба только что постирала все белье и теперь перед ней стоит непростая задача — как его высушить, чтобы ни одна вещь не успела испортиться. Сразу после стирки, i-я постиранная вещь имеет влажность w_i. Если она сушится на веревке, то за минуту ее влажность уменьшается на 1, а если на батарее — то на r (если влажность была меньше r, то она становится равной 0). Причем веревок у тети Любы много (хватает для одновременной сушки всех вещей), а батарея только одна, причем такая маленькая, что на ней нельзя сушить две вещи одновременно. i-я вещь испортится, если не высохнет за время d_i. Помогите тете Любе составить план, когда какую вещь повесить на батарею.

Формат входного файла

Первая строка входного файла содержит целые числа n ($1 \le n \le 10^5$) — количество мокрых вещей, и r ($1 \le r \le 10^9$). Следующие n строк содержат описания постиранных вещей — пары чисел w_i и d_i ($1 \le w_i, d_i \le 10^9$).

Формат выходного файла

Выведите план сушки в виде пар целых чисел t_i и k_i, где t_i — (измерено время в минутах от начала сушки, а k_i — номер вещи, которую нужно повесить на батарею в этот момент. Выводите пары в порядке увеличения t_i. Пар не должно быть больше 100000. Не выводите числа больше 10^9.

Если высушить все вещи невозможно, выведите слово «`Impossible`».

Пример

drying.in	drying.out
3 3 2000 1000 2000 2000 2500 1500	0 3 500 1 1000 3
3 3 2000 1000 2000 1000 2000 1000	Impossible

Задача B. Халява

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

Гриша очень любит газировку *PupsiCola*. Однажды он узнал, что, собрав несколько крышек со звездочками, можно получить футболку. Гриша нашел a крышек с одной звездочкой, b крышек с двумя звездочками и c крышек с тремя звездочками. На футболку можно обменять набор крышек, общее количество звездочек на которых не меньше k.

Помогите Грише узнать, сколько футболок он может получить.

Формат входного файла

Входной файл содержит целые числа a, b, c и k ($0 \le a, b, c \le 100$, $1 \le k \le 1000$).

Формат выходного файла

Выведите максимальное количество футболок, которые может получить Гриша.

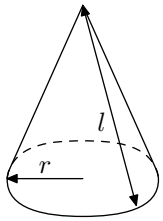
Пример

free.in	free.out
2 2 2 4	3
0 0 4 4	2

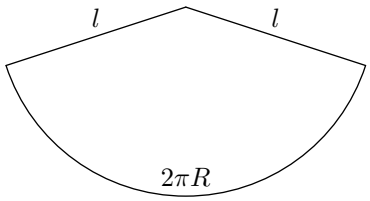
Задача C. Упаковка подарка

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

Для упаковки подарка Роме нужно сделать из фольги конус с нижним основанием радиуса r и длиной образующей l.



Для этого ему нужен кусок фольги такой формы:



У Ромы есть рулон фольги шириной d, от которого он может отрезать кусок произвольной длины. Помогите ему узнать минимальную длину куска, из которого можно вырезать нужную фигуру.

Формат входного файла

Во входном файле содержатся три целых числа: r, l и d (все числа не превосходят 10^6, $l > r$). Гарантируется, что существует способ вырезать нужную фигуру.

Формат выходного файла

Выведите в выходной файл минимально возможную длину куска фольги, из которого Рома сможет сделать нужную упаковку.

Выведите не менее 6 знаков после точки.

Пример

present.in	present.out
1 2 2	4

Задача D. Дороги

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

Мэр города Гадюкино решил проверить состояние дорог после только что проведенного капитального ремонта. Для этого он хочет проехать по каждой дороге в обоих направлениях.

Помогите мэру составить кратчайший маршрут, проходящий по каждой дороге в каждом направлении хотя бы один раз.

Формат входного файла

В городе Гадюкино n перекрестков и m дорог, каждая из которых соединяет два различных перекрестка. Между двумя перекрестками может быть не более одной дороги.

Известно, что по дорогам от каждого перекрестка можно доехать до любого другого.

Входной файл содержит целые числа n и m ($1 \le n \le 10^4$, $1 \le m \le 10^5$), и далее m пар целых чисел a_i и b_i — номера перекрестков, которые соединяет i-я дорога.

Формат выходного файла

Выведите число s — минимальную длину пути и далее $s+1$ число — номера перкрестков в том порядке, в котором их нужно проезжать.

Пример

roads.in	roads.out
3 3 1 2 2 3 1 3	6 1 2 3 1 3 2 1

Задача E. Зарплата

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

В одном государственном учреждении работают n сотрудников. Их фактические зарплаты равны d_i, но по закону всем сотрудникам положено платить одинаково, поэтому им всем официально выплачивают среднюю зарплату d, такую, чтобы общая сумма оставалось такой же, а потом сотрудники сами перераспределяют полученные деньги.

Короче, творится полный бардак. И чтобы этот бардак уменьшить, Самый Главный Начальник решил использовать недавно принятый закон «о материальной помощи», который позволяет любому сотруднику часть своей зарплаты (целое число рублей от 1 до $d-1$) передавать другому сотруднику в виде материальной поддержки. Однако по закону, сотрудник таким образом может «помогать» только одному «малоимущему».

Например, если Петя и Вася получают по 100 рублей и Петя напишет заявление на передачу 30% своей зарплаты Васе, то Петя будет получать 70 рублей, а Вася — 130.

Теперь Самый Главный Начальник хочет узнать: кто, кому и сколько должен передавать, чтобы в результате все получали ровно столько, сколько нужно. Помогите ему это сделать.

Формат входного файла

Входной файл содержит целые числа n и d ($1 \le n \le 10^5$, $1 \le d \le 10^9$), и далее n целых чисел d_i ($1 \le d_i \le 10^9$). Сумма всех d_i равна nd.

Формат выходного файла

Выведите n пар целых чисел a_i и b_i, означающих, что сотрудник i передает сотруднику a_i чать своей зарплаты в размере b_i рублей. Если сатрудник i никому ничего не передает, выведите вместо a_i и b_i два нуля.

Пример

salary.in	salary.out
3 200 100 300 200	2 100 0 0 0 0
3 200 10 300 290	3 190 0 0 2 100

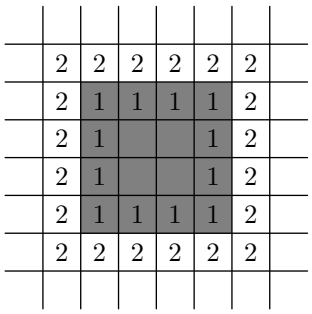
Задача F. Квадрат

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

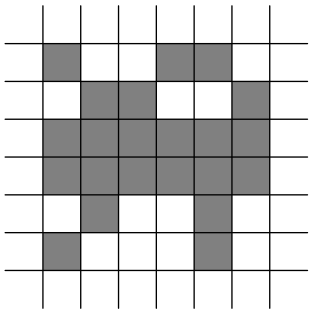
На сайте сотового оператора BeepLine сделали защиту от роботов, рассылающих SMS-сообщения: прежде, чем отправить SMS, пользователь должен написать, какую фигуру он видит в специальном окошке: квадрат или круг. Причем, для усиления защиты, в рисунок внесены небольшие помехи.

Коле срочно нужно разослать всем друзям сообщение, поэтому он просит Вас написать программу, распознающую изображение.

Экспрементально установлено, что система рисует квадрат с помехами следующим образом: сначала на белом фоне рисуется черный квадрат $k \times k$ клеток ($k \geq 3$), затем некоторые клетки на границе квадрата (на рисунке обозначены цифрой 1) закрашиваются белым, а некоторые клетки, граничащие с квадратом (на рисунке обозначены цифрой 2), закрашиваются черным.



Например, квадрат 4×4 после нанесения помех может выглядеть так:



Формат входного файла

Первая строка входного файла содержаит целые числа n и m — размеры экрана ($1 \leq n, m \leq 1000$). Следующие n строк, по m символов в каждой, содержат описание картинки. Черные клетки обозначены символом «`*`», а белые — символом «`.`».

Формат выходного файла

Если заданная картинка может быть квадратом после преобразований, описанных в условии, выведите слово «`SQUARE`», иначе выведите слово «`CIRCLE`».

Пример

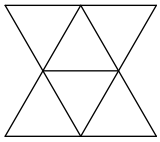
square.in	square.out
10 10 ....***..... ..***..... ..***.....	SQUARE
10 10 ....*..... ..*****... ..******.. .******... ..*****... ..*.*.*... ...*......	SQUARE
10 10 ....****.... .******... ******.**. ********.. ********.. .******... ..****.... ....*.....	CIRCLE
3 3	CIRCLE

Задача G. Треугольники

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

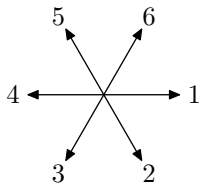
У Сени есть шоколадка, составленная из нескольких прилегающих друг к другу плиточек в форме правильных треугольников. Его брат Женя нашел эту шоколадку и решил сделать ее треугольной, съев все лишнее (ведь треугольные шоколадки намного вкуснее). Сколькими способами он может это сделать?

Например, из такой шоколадки:



можно сделать треугольную шоколадку со стороной 1 шестью способами или шоколадку со стороной 2 двумя способами. Итого восемь способов.

Формат входного файла



Форма шоколадки задается ее границей в порядке обхода по часовой стрелке. Первая строка содержит число n — количество отрезков на границе ($1 \le n \le 5000$). Далее n чисел от 1 до 6, задающих направление движения по границе (см. рис.).

Формат выходного файла

Выведите одно число — количество способов.

Пример

triangle.in	triangle.out
8 1 1 3 2 4 4 6 5	8

Задача H. Технология программирования

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

Толик придумал новую технологию программирования. Он хочет уговорить друзей использовать ее. Однако все не так просто. i-й друг согласится использовать технологию Толика, если его авторитет будет не меньше a_i (авторитет выражается целым числом). Как только он начнет ее использовать, к авторитету Толика прибавится число b_i (попадаются люди, у которых $b_i < 0$). Помогите Толику наставить на путь истинный как можно больше своих друзей.

Формат входного файла

На первой строке входного файла содержатся два числа: n ($1 \le n \le 1000$) — количество друзей у Толика, и первоначальный авторитет толика . Следующие n строк содержат пары чисел a_i и b_i. Все числа целые, по модулю не больше 10^6.

Формат выходного файла

Выведите в выходной файл m — максимальное число друзей, которых может увлечь Толик, и затем m чисел — номера друзей в том порядке, в котором их нужно агитировать.

Пример

tp.in	tp.out
5 1 1 3 6 -5 6 -4 2 2 2 -1	4 1 4 3 5

Задача I. НЛО

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

В маленьком городке М начала действоваь служба контроля за незаконными полетами НЛО. Первая задача службы — выяснить, сколько НЛО действует в окрестности города.

Агенты службы опросили множество свидетелей и составили список случаев встречи с НЛО, произощедших за одни сутки, с указанием места и времени наблюдения.

Теперь аналитики хотят понять, сколько же на самом деле было НЛО. Из данных разведки известна максимальная скорость, с которой может лететь НЛО. Аналитики просят вас узнать, какое минимальное количество НЛО могли наблюдать свидетели.

Формат входного файла

На первой строке входного файла содержатся целые числа n и v — количество случаев наблюдения и максимальная скорость НЛО ($1 \le n \le 100$, $1 \le v \le 10000$). Следующие n строк содержат описания случаев встречи с НЛО в формате «`ЧЧ:ММ x y`», где `ЧЧ:ММ` — время встречи, `x` и `y` — координаты места, в котором наблюдался НЛО (для простоты будем считать, что все встречи происходили на плоскости). Координаты по модулю не превышают 1000.

Скорость выражена в км/ч, координаты — в км.

Формат выходного файла

Выведите в выходной файл одно число — минимальное возможное количество НЛО.

Пример

ufo.in	ufo.out
4 1 12:00 0 0 13:10 0 1 14:00 1 0 15:00 1 1	2