Задача A. Ферзя в угол

Имя входного файла:	`queen1.in`
Имя выходного файла:	`queen1.out`
Ограничение по времени:	1 секунда
Ограничение по памяти:	64 мегабайта

64 мегабайта

В левом нижнем углу доски $M \times N$ стоит ферзь. Двое игроков по очереди ходят ферзем, перемещая его на любое число клеток по вертикали вверх, по горизонтали вправо, или по диагонали вправо-вверх. Выигрывает тот, кто поставит ферзя в правый верхний угол доски. Определите, какой из игроков имеет выигрышную стратегию. Гарантируется, что нужно сделать хотя бы один ход.

Формат входных данных

На вход программе подается два натуральных числа M и N, не превосходящих 100.

Формат выходных данных

Программа должна вывести номер игрока (1 или 2), который имеет выигрышную стратегию.

Пример

queen1.in	queen1.out
3 4	1

Задача B. Функция Гранди

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

256 мегабайт

Дан ориентированный ациклический граф. Посчитайте функцию Гранди для каждой стартовой вершины.

Формат входных данных

На первой строке будут даны числа n и m — количество вершин и рёбер в графе ($1 \leqslant n, m \leqslant 100\,000$). На следующих m строках содержится по два числа x и y ($1 \leqslant x, y \leqslant n$).

Учтите, что в графе могут быть кратные рёбра.

Формат выходных данных

Выведите n чисел — значение функции Гранди для каждой стартовой вершины.

Примеры

стандартный ввод	стандартный вывод
3 3 1 2 2 3 1 3	2 1 0
2 1 2 1	0 1

Задача C. Малыш и Карлсон

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

На свой День рождения Малыш позвал своего лучшего друга Карлсона. Мама испекла его любимый пирог прямоугольной формы $a \times b \times c$ сантиметров. Карлсон знает, что у Малыша еще есть килограмм колбасы. Чтобы заполучить ее, он предложил поиграть следующим образом: они по очереди разрезают пирог на две ненулевые по объему прямоугольные части с целыми измерениями и съедают меньшую часть (в случае, когда части равные, можно съесть любую). Проигрывает тот, кто не может сделать хода (то есть когда размеры будут $1 \times 1 \times 1$). Естественно, победителю достается колбаса.

Малыш настаивает на том, чтобы он ходил вторым.

Помогите Карлсону выяснить, сможет ли он выиграть, и если сможет — какой должен быть его первый ход для этого.

Считается, что Малыш всегда ходит оптимально.

Формат входных данных

Во входном файле содержится 3 целых числа a, b, c ($1 \leqslant a, b, c \leqslant 5\,000$) — размеры пирога.

Формат выходных данных

В случае, если Карлсон не сможет выиграть в Малыша, выведите `NO`. В противном случае в первой строке выведите `YES`, во второй — размеры пирога после первого хода Карлсона в том же порядке, что и во входном файле.

Примеры

karlsson.in	karlsson.out
1 1 1	NO
1 2 1	YES 1 1 1
1 1 10	YES 1 1 7

Задача D. Произведение графов

Имя входного файла:	`graphprod.in`
Имя выходного файла:	`graphprod.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

256 мегабайт

Пусть дан ориентированный ациклический граф. Стандартная игра на графе заключается в следующем: изначально на одной из вершин графа (называемой начальной позицией) стоит фишка. Двое игроков по очереди двигают её по рёбрам. Проигрывает тот, кто не может сделать ход.

В теории игр часто рассматриваются более сложные игры. Например, прямое произведение двух игр на графах. Прямое произведение игр — это следующая игра: изначально на каждом графе в начальной позиции стоит по фишке. За ход игрок двигает обе фишки по рёбрам (каждую фишку двигает в собственном графе). Проигрывает тот, кто не может сделать ход. То есть тот, кто не может сделать ход хотя бы в одной игре.

Ваша задача — опеределить, кто выиграет при правильной игре.

Формат входных данных

На первой строке будут даны числа N_1 и M_1 — количество вершин и рёбер в первом графе ($1 \leqslant N_1, M_1 \leqslant 100\,000$). На следующих M_1 строках содержится по два числа x и y ($1 \leqslant x, y \leqslant N_1$).

В следующих $M_2 + 1$ строках задан второй граф в том же формате.

Заканчивается входной файл списком пар начальных вершин, для которых нужно решить задачу. На первой строке задано число T ($1 \leqslant T \leqslant 100\,000$) — количество пар начальных вершин. В следующих T строках указаны пары вершин v_1 и v_2 ($1 \leqslant v_1 \leqslant N_1$, $1 \leqslant v_2 \leqslant N_2$).

Учтите, что в графах могут быть кратные рёбра.

Формат выходных данных

На каждую из T пар начальных вершин выведите строку “`first`”, если при правильной игре выиграет первый, и “`second`”, если второй.

Пример

graphprod.in	graphprod.out
3 2 1 2 2 3 2 1 1 2 2 2 1 3 2	first second

Задача E. Конфетки

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

После разгромной победы Директора ЛКШ руководитель физической школы сдался и отпустил Деда Мороза к нам. Но ради интереса предложил сразиться ЛФШатам и ЛКШатам в еще одной непростой игре.

В каждой игре участвует один из вас и один ЛФШонок, а ходите вы по очереди. В кучку перед вами кладется N вкусных конфеток. На каждом ходе игрок может съесть от 1 до K конфеток (больше нельзя — много сладкого вредно даже в Новый Год), но при этом не больше, чем взял его противник на предыдущем ходе (не будем жадничать, мы же добрые). Второго ограничения нет лишь для первого хода каждой игры. Проигрывает тот, кому не осталось конфеток.

У нас возникли подозрения, что директор ЛФШ специально подобрал такие N и K, чтобы ЛКШата никогда не смогли выиграть. Мы надеемся, что это не так, и очень просим вас проверить это.

Формат входных данных

Во входном файле записаны через пробел два целых числа — N ($1 \leqslant N \leqslant 500$) и K ($1 \leqslant K \leqslant 100$).

Формат выходных данных

В выходной файл выведите минимальное число конфет, которое должен съесть ЛКШонок первым ходом, чтобы выиграть при оптимальной игре ЛФШонка, либо 0, если даже самый умный из нас не сможет одолеть идеального играющего противника.

Пример

sweets.in	sweets.out
7 3	1

Задача F. Огромный ним

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

Петя и Вася играют в ним, но не простой, а просто огромный. У них есть очень много кучек камней. Кучки разделены на n групп. Группа i состоит из кучек размеров от l_i до r_i включительно.

Помогите ребятам понять, кто выиграет при оптимальной игре

Формат входных данных

Первая строка входного файла содержит число n ($1 \leqslant n \leqslant 10^5$), следующие n строк содержат пары чисел l_i, r_i ($1 \leqslant l_i \leqslant r_i \leqslant 10^{18}$).

Формат выходных данных

Если первый игрок проигрывает, выведите `Lose`, если выигрывает — выведите в первой строке `Win`, а во второй строке — любой выигрышный ход для первого игрока. Ход задается размером кучки до хода и после него.

Примеры

big-nim.in	big-nim.out
1 1 10	Win 8 3
2 2 5 2 5	Lose

Задача G. Ретроанализ для маленьких

Имя входного файла:	`retro.in`
Имя выходного файла:	`retro.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Дан ориентированный весёлый граф из n вершин и m ребер. Оля и Коля играют в игру. Изначально фишка стоит в вершине i. За ход можно передвинуть фишку по любому из исходящих ребер. Тот, кто не может сделать ход, проигрывает. Ваша задача — для каждой вершины i определить, кто выиграет при оптимальной игре обоих.

Формат входных данных

Входные данные состоят из одного или нескольких тестов. Каждый тест содержит описание весёлого ориентированного графа. Граф описывается так: на первой два целых числа n ($1 \leqslant n \leqslant 300\,000$) и m ($1 \leqslant m \leqslant 300\,000$). Следующие m строк содержат ребра графа, каждое описывается парой целых чисел от 1 до n. Пара a b обозначает, что ребро ведет из вершины a в вершину b. В графе могут быть петли, могут быть кратные ребра. Сумма n по всем тестам не превосходит $300\,000$, сумма m по всем тестам также не превосходит $300\,000$.

Формат выходных данных

Для каждого теста выведите для каждой вершины `FIRST`, `SECOND` или `DRAW` в зависимости от того, кто выиграет при оптимальной игре из этой вершины. Ответы к тестам разделяйте пустой строкой.

Пример

retro.in	retro.out
5 5 1 2 2 3 3 1 1 4 4 5 2 1 1 2 4 4 1 2 2 3 3 1 1 4	DRAW DRAW DRAW FIRST SECOND FIRST SECOND FIRST FIRST SECOND SECOND

Задача H. Бинарная игра

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	2 секунды
Ограничение по памяти:	512 мегабайт

Искандер и Оля любят придумывать ребусы. Но больше, чем придумывать ребусы, они любят придумывать какие-нибудь игры на строках. Вот и сейчас им в голову пришла забавная игра со следующими правилами:

- Выбирается какой-то набор *запрещённых* двоичных (состоящих из нулей и единиц) строк $f_1, f_2, \ldots, f_n$.
- Выбирается некоторая стартовая бинарная строка s, такая что ни одна из запрещённых строк не входит в неё как подстрока.
- Игроки по очереди дописывают в конец строки s по одному символу «`0`» или «`1`». Оля ходит первой.
- Проигрывает тот, после чьего хода хотя бы одна из запрещённых строк $f_1, f_2, \ldots f_n$ входит в s как подстрока.
- В случае если при оптимальной игре обоих игроков игра может продолжаться сколь угодно долго, то объявляется ничья.

Вы обожаете портить другим людям их любимые развлечения, поэтому решили написать программу, которая будет определять исход игры по заданному набору запрещённых строк и стартовой строке s.

Формат входных данных

В первой строке входных данных записаны два целых числа n и m ($0 \leqslant n \leqslant 100\,000$, $0 \leqslant m \leqslant 1\,000\,000$) — количество запрещённых строк и изначальная длина строки s.

В каждой из последующих n строк содержится одна запрещённая строка. Гарантируется, что все эти строки непусты, состоят из символов «`0`» и «`1`» и никакая из них не является подстрокой строки s. Дополнительно гарантируется, что **суммарная длина** всех запрещённых строк не превосходит $1\,000\,000$.

В последней строке входных данных записана стартовая строка s длины m, состоящая только из символов «`0`» и «`1`». Обратите внимание, строка s может быть пустой, в этом случае соответствующая строка входных данных отсутствует (в том числе символ перевода строки). Длина s не превосходит $1\,000\,000$.

Формат выходных данных

В зависимости от результата игры при оптимальной игре обоих игроков выведите:

- «`Olya`» (без кавычек), если Оля может победить вне зависимости от того как будет играть Искандер. Напомним, что Оля ходит первой.
- «`Iskander`» (без кавычек), если Искандер может победить не зависимо от ходов Оли.
- «`Friendship`» (без кавычек), если при оптимальной игре обоих игроков игра будет продолжаться бесконечно долго.

Примеры

стандартный ввод	стандартный вывод
1 0 1	Friendship
3 1 000 001 011 0	Olya
2 3 1001 000 100	Iskander

Замечание

Если вы не слушали вводную лекцию про анализ игр на ациклических и циклических графах, рекомендую первую главу данной статьи:

https://ejudge.lksh.ru/archive/2014/07/A/games.pdf

Задача I. Peep Blue

Имя входного файла:	`peep.in`
Имя выходного файла:	`peep.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

256 мегабайт

Суперкомпютер Peep Blue умеет играть в игры. Однако Серёжа не побоялся принять его вызов.

Правила, предложенные компьютером, таковы. Есть n кучек камней. Игроки ходят по очереди, начинает Серёжа. На первом ходу каждому из игроков разрешается убрать любой набор кучек (в том числе пустой, но не все). На втором и следующих ходах каждый из игроков может взять любое ненулевое число камней из любой кучки. Проигрывает тот, кто не может сделать ход.

Ваша задача — определить, может ли Серёжа одержать победу, а также какой первый ход ему следует сделать. Среди всех первых ходов, приносящих победу, выберите такой, который оставит в игре как можно больше камней: это должно усложнить задачу Peep Blue.

Формат входных данных

В первой строке ввода записано целое число n — количество кучек камней ($1 \leqslant n \leqslant 100\,000$). В следующей строке записаны n натуральных чисел, a_i, не превышающих $2^{30}-1$, — количества камней в кучках.

Формат выходных данных

Если Серёжа не может одержать победу, выведите единственное число -1. Иначе выведите максимально возможное суммарное число камней, которое можно оставить.

Пример

peep.in	peep.out
4 1 2 3 4	9