Задача A. Жестокая задача

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

Штирлиц и Мюллер стреляют по очереди. В очереди n человек, стоящих друг за другом. Каждым выстрелом убивается один из стоящих. Кроме того, если у кого-то из стоящих в очереди убиты все его соседи, то этот человек в ужасе убегает. Проигрывает тот, кто не может ходить. Первым стреляет Штирлиц. Требуется определить, кто выиграет при оптимальной игре обеих сторон, и если победителем будет Штирлиц, то найти все возможные первые ходы, ведущие к его победе.

Формат входных данных

Входной файл содержит единственное число n ($2 \leqslant n \leqslant 5\,000$) — количество человек в очереди.

Формат выходных данных

Если выигрывает Мюллер, выходной файл должен состоять из единственного слова `Mueller`. Иначе в первой строке необходимо вывести слово `Schtirlitz`, а в последующих строках — номера людей в очереди, которых мог бы первым ходом убить Штирлиц для достижения своей победы. Номера необходимо выводить в порядке возрастания.

Примеры

cruel.in	cruel.out
3	Schtirlitz 2
4	Mueller
5	Schtirlitz 1 3 5

Задача B. 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.

Примеры

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 C. Игры на графе

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

Противник начал e2–e4. Я
проанализировал его архитектуру и сдался

Из мемуаров 20-го чемпиона мира Фрица Рыбкина

Прибыв на место, Ааз тут же потребовал организовать совещание букмекеров, на котором он изложит свой план.

— Главная задача, — начал Ааз своё выступление перед букмекерами, — научиться использовать достижения прогресса. Мы планируем запуск множества новых видов соревнований, что — вполне возможно — приведёт к тому, что появятся какие-то игры по правилам, придуманным не нами. А значит, необходимо уметь быстро выяснять, насколько эти правила могут быть нам полезны.

— А можно ли хотя бы в общем пояснить, как это будет делаться? — последовал вопрос из зала.

— Вот пример задачи, решив которую, мы сможем разобраться с целым классом игр. Дан ориентированный граф некоторой игры для двух игроков и начальная позиция в ней. Напомним, что в игре на графе игрок имеет право походить из позиции в любую позицию, в которую есть ребро из текущей. Игроки ходят по очереди; проигрывает тот, кто не может сделать ход. Требуется проверить, верно ли, что при любой игре сторон всегда выигрывает первый игрок.

Формат входных данных

Во входном файле содержится описание одного или нескольких тестов. В первой строке каждого теста заданы число вершин V и число рёбер E ($1 \leqslant V \leqslant 100\,000$, $1 \leqslant E \leqslant 100\,000$), а также номер начальной позиции a ($1 \leqslant a \leqslant V$). Далее следуют E строк — описания рёбер в формате u_i v_i ($1 \leqslant u_i, v_i \leqslant V$), что означает наличие ребра, направленного из вершины u_i в вершину v_i. Файл завершается тремя нулями. Сумма всех E по всем тестам не превосходит $100\,000$, количество тестов в файле не превосходит 1000.

Формат выходных данных

Следуйте формату примера максимально точно — проверка производится автоматически.

Примеры

gg.in
3 2 1 1 2 1 3 1 1 1 1 1 0 0 0

gg.out
First player always wins in game 1. Players can avoid first player winning in game 2.

Задача D. Шоколадка-революция (задача 3)

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

Аким получил золотую медаль на IOI, за что от правительства ему положена благодарность в виде шоколадки. На данный момент имеется N видов шоколадок, которые производит государственная шоколадная фабрика. У каждой шоколадки есть стоимость. Кроме того, у каждого вида, кроме шоколадки «Юлька», есть вид-прародитель. Известно, что прородителем может быть только шоколадка, которая была выпушена хронологически раньше. Исторически самый первый вид — «Юлька». Чиновник и Аким выбирают шоколадку Акиму в подарок. При этом цель первого — сэкономить, второго — получить настолько дорогую шоколадку, насколько это возможно. Начинается выбор с «Юльки». Аким за ход либо берет текущую шоколадку, либо меняет свой выбор на шоколадку-потомка (если это возможно). Чиновнику же кажется, что позже выпущенные виды хуже и дешевле, поэтому он меняет выбор на шоколадку-потомка (если это возможно), либо покупает Акиму текущую шоколадку (если потомков нет). Первым ходит Аким. Определите, на шоколадку какой стоимости может претендовать Аким.

Формат входных данных

Виды шоколадок занумерованы в некотором произвольном порядке, «Юлька» имеет номер 1. В первой строке входного файла находится одно число N — количество видов шоколадок ($0 < N \leqslant 200\,000$). Далее следуют N строк, в каждой из которых заданы два числа P_i и C_i — номер сорта-прародителя i-ого сорта и стоимость i-ого сорта ($0 \leqslant P_i \leqslant N$, $0 < C_i \leqslant 1\,000\,000\,000$; для «Юльки», у которой нет прародителя, указано $P_1 = 0$);

Формат выходных данных

Выведите в выходной файл одно число — стоимость шоколадки, которую получит Аким при правильных действиях как чиновника, так и своих.

Примеры

chocorev.in	chocorev.out
8 0 4 5 3 1 2 5 1 1 5 4 8 3 6 3 7	6

Задача E. Дровосек

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

Двое играют в следующую игру: имеется дерево с отмеченной вершиной (корнем). Игроки ходят по очереди. За ход игрок разрубает ветку (стирает ребро), причем из двух получившихся компонент связности остается только та, которая содержит корень — остальная отваливается и больше в игре не участвует. Проигрывает тот, кто не может сделать ход.

Определите, может ли выиграть первый игрок, и если да, то укажите любой из его выигрышных ходов.

Формат входных данных

В первой строке входного файла находится 2 числа N и R — количество вершин дерева и номер корня ($2 \leqslant N \leqslant 100\,000, 1 \leqslant R \leqslant N$). Далее следует $N - 1$ строк, в каждой из которых находятся два числа — номера вершин, которые соединяет очередное ребро.

Формат выходных данных

Выведите в выходной файл одно число: 1 или 2 — номер игрока, который выигрывает при правильной игре. Если выигрывает первый игрок, то выведите также любой его выигрышный ход, т.е. порядковый номер ребра во входном файле, которое ему достаточно разрубить первым ходом (число от 1 до $N - 1$).

Примеры

`woodcut.in`	`woodcut.out`
5 5 2 3 1 3 2 5 4 5	1 1

Задача F. Сумма игр

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

Пусть дан ориентированный 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, прямая сумма двух игр на графах. Прямая сумма игр — это следующая игра: изначально на каждом графе в начальной позиции стоит по фишке. За ход игрок выбирает любую фишку и двигает по ребру соответствующего графа. Проигрывает тот, кто не может сделать ход.

Ваша задача — определить, 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 ($1 \leqslant N_1, M_1 \leqslant 10\,000$). На следующих M_1 строках содержится по два числа x и y ($1 \leqslant x, y \leqslant N_1$). В следующей строке вводятся N_2 и M_2 — количество вершин и рёбер во втором графе соответственно. Далее в следующих M_2 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 ($1 \leqslant v_1 \leqslant N_1$, $1 \leqslant v_2 \leqslant N_2$).

Формат выходных данных

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

Примеры

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

Задача G. Странная игра

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

Двое играют в простую игру на доске $n \times n$. У первого игрока есть одна белая фишка, а у второго — одна чёрная. Игроки ходят по очереди, первым ходит первый игрок (белые).

Первый игрок имеет право двигать свою фишку на одну клетку в одном из четырёх основных направлений (влево, вправо, вверх, вниз). Второй игрок при своем ходе также выбирает одно из этих четырёх направлений, но может передвинуть свою фишку как на одну клетку в этом направлении, так и на две. Выигрывает тот, кто первым съедает фишку соперника.

Определите победителя и число ходов, требуемое для победы, при оптимальной игре сторон.

Формат входных данных

Во входном файле даны пять чисел — n ($2 \leqslant n \leqslant 20$), а также координаты белой и чёрной фишек. Фишки стоят в разных клетках доски.

Формат выходных данных

Выведите `WHITE` x, если выигрывают белые, `BLACK` x, если выигрывают чёрные, `DRAW`, если игра закончится вничью. Здесь x — число ходов обеих сторон (полуходов) до момента окончания игры.

Примеры

strange-game.in	strange-game.out
2 1 1 2 2	BLACK 2
2 2 2 1 2	WHITE 1
3 1 1 3 3	BLACK 6

Задача H. Вас снова замяукали!

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

Два котёнка попали в запутанный лабиринт со множеством комнат и переходов между ними. Котята долго по нему плутали, обошли все комнаты по много раз, нашли выход (да даже и не один, а несколько), в общем, изучили там всё, что смогли. Теперь этот лабиринт котята используют в своих играх.

Чаще всего котята играют в следующую игру: начиная в какой-то комнате лабиринта, котята поочерёдно выбирают, в какую из комнат им перейти. Котята изначально находятся в одной комнате и ходят вместе. Как только котёнок, который должен выбрать следующую комнату, не может этого сделать, он признаётся проигравшим. Обычно в таких играх выигрывающий игрок стремится выиграть как можно быстрее, а проигрывающий стремится как можно дольше оттянуть свое поражение. Но у котят свои представления о победе и поражении. Если котёнок знает, что, начиная из текущей комнаты, он выиграет (вне зависимости от действий другого котёнка), то он стремится играть как можно дольше, чтобы продлить себе удовольствие от выигрыша (естественно, при этом выигрывающий котёнок должен гарантировать себе, что будет постоянно уверен в выигрыше). Котёнок, который знает, что проиграет (при условии, конечно, что другой котёнок будет действовать оптимально), старается проиграть как можно быстрее, чтобы начать новую игру, в которой и взять реванш.

Если котята будут ходить бесконечно долго, но никто из них не сможет выиграть, то котята считают игру завершившейся вничью и замяукивают Вас.

Вас попросили для каждой комнаты в лабиринте узнать, выиграет или проиграет котёнок, начинающий ходить из данной комнаты. Если котёнок, начинающий из этой комнаты, выигрывает, требуется узнать максимальное количество ходов, которое он сможет играть, если же проигрывает — минимальное количество, которое ему придётся играть.

Формат входных данных

В первой строке ввода находятся два числа n и m — число комнат и переходов между комнатами в лабиринте ($1 \leqslant n \leqslant 100\,000$, $0 \leqslant m \leqslant 100\,000$). Далее следует m строк с описаниями переходов. Описание перехода состоит из двух чисел a и b, означающих, что котёнок, начинающий игру в комнате с номером a, может выбрать комнату b в качестве следующей.

Формат выходных данных

Выведите n строк — для каждой комнаты результат игры для котёнка, который начнет игру из этой комнаты. Если игра закончится вничью, выведите «`DRAW`». Если начинающий котёнок выиграет, выведите «`WIN K`», где `K` — количество ходов, которые сможет играть выигрывающий котёнок. Если котёнок сможет играть сколь угодно долго, сохраняя возможность в любой момент выиграть, выведите «`WIN INF`». Если котёнок, начинающий из этой комнаты, проиграет, выведите «`LOSE K`», где `K` — количество ходов, которые придется играть проигрывающему котёнку. Если же котёнку придется играть сколь угодно долго, при том, что его соперник сможет в любой момент выиграть, выведите «`LOSE INF`».

Примеры

labyrinth.in	labyrinth.out
4 4 1 2 1 3 2 4 3 4	LOSE 2 WIN 1 WIN 1 LOSE 0
6 6 1 2 2 3 3 4 4 1 4 5 5 6	DRAW DRAW DRAW DRAW WIN 1 LOSE 0
6 6 1 2 2 3 3 4 4 1 2 6 4 5	LOSE INF WIN INF LOSE INF WIN INF LOSE 0 LOSE 0

Задача I. Произведение графов

Имя входного файла:	`graphprod.in`
Имя выходного файла:	`graphprod.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	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 — определить, кто выиграет при правильной игре.

Формат входных данных

На первой строке будут даны числа M_1 и M_2 — количество вершин и рёбер в первом графе ($1 \leqslant M_1, M_2 \leqslant 100\,000$). 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.

Примеры

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