

Задача А. Волновой обход графа

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

Пусть *расстояние* от вершины u до вершины v — это минимальное количество рёбер в пути между u и v ; так, расстояние между u и u — 0, а расстояние между любыми двумя различными соседними вершинами — 1.

Волновым обходом графа из вершины v назовём такую последовательность вершин u_1, u_2, \dots, u_r , что:

- $u_1 = v$,
- Каждая вершина графа, достижимая из v , встречается в ней хотя бы один раз, и
- Каждая следующая вершина последовательности удалена от вершины v не меньше, чем предыдущая.

Задан связный неориентированный граф и его вершина v . Выведите любой волновой обход этого графа.

Формат входных данных

В первой строке заданы числа N , M и v через пробел — количество вершин и рёбер в графе и начальная вершина обхода ($1 \leq N \leq 100$, $0 \leq M \leq 10\,000$, $1 \leq v \leq N$). Следующие M строк содержат по два числа u_i и v_i через пробел ($1 \leq u_i, v_i \leq N$); каждая такая строка означает, что в графе существует ребро между вершинами u_i и v_i .

Формат выходных данных

В первой строке выведите число r — количество вершин в найденном волновом обходе ($1 \leq r \leq 10\,000$; гарантируется, что обход, удовлетворяющий этим ограничениям, существует). Во второй строке выведите сами числа u_1, u_2, \dots, u_r через пробел.

Примеры

<i>стандартный ввод</i>	<i>стандартный вывод</i>
3 2 1 1 2 2 3	3 1 2 3
4 4 1 1 2 2 3 3 4 4 1	4 1 2 4 3

Задача В. (p, q) -лошадь

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

(p, q) -лошадь — это обобщение обычного шахматного коня. (p, q) -лошадь своим ходом перемещается на p клеток в одном направлении и на q — в другом (перпендикулярном). Например, $(3, 4)$ -лошадь может переместиться с клетки $(5, 6)$ на клетки $(1, 3)$, $(2, 2)$, $(2, 10)$, $(1, 9)$, $(8, 10)$, $(9, 9)$, $(8, 2)$ и $(9, 3)$. Очевидно, что обычный шахматный конь — это $(2, 1)$ -лошадь.

Ваша задача — определить минимальное число ходов, которое требуется (p, q) -лошади, чтобы добраться от одной клетки шахматной доски $M \times N$ до другой. За пределы доски выходить запрещается.

Формат входных данных

Единственная строка входных данных содержит восемь целых чисел $M, N, p, q, x_1, y_1, x_2, y_2$ ($1 \leq x_1, x_2 \leq M \leq 100$, $1 \leq y_1, y_2 \leq N \leq 100$, $0 \leq p \leq 100$, $0 \leq q \leq 100$).

Формат выходных данных

В первой строке выведите целое число k — минимальное число ходов, которое требуется (p, q) -лошади, чтобы добраться из клетки (x_1, y_1) в клетку (x_2, y_2) . Далее должны следовать $k + 1$ строк, в которых должны быть записаны последовательные положения (p, q) -лошади на этом пути.

Если (p, q) -лошадь не может добраться из (x_1, y_1) в (x_2, y_2) , выведите единственное число -1 .

Примеры

<i>стандартный ввод</i>	<i>стандартный вывод</i>
3 3 1 1 1 1 3 3	2 1 1 2 2 3 3
2 2 1 1 1 1 1 2	-1

Задача С. Пошаговый обход графа

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

Пошаговым обходом графа из вершины v назовём такую последовательность вершин u_1, u_2, \dots, u_r , что:

- $u_1 = u_r = v$,
- Каждая вершина графа, достижимая из v , встречается в ней хотя бы один раз, и
- Между любыми двумя соседними вершинами последовательности в графе существует ребро.

Задан связный неориентированный граф и его вершина v . Выведите любой пошаговый обход этого графа.

Формат входных данных

В первой строке заданы числа N , M и v через пробел — количество вершин и рёбер в графе и начальная вершина обхода ($1 \leq N \leq 100$, $0 \leq M \leq 10\,000$, $1 \leq v \leq N$). Следующие M строк содержат по два числа u_i и v_i через пробел ($1 \leq u_i, v_i \leq N$); каждая такая строка означает, что в графе существует ребро между вершинами u_i и v_i .

Формат выходных данных

В первой строке выведите число r — количество вершин в найденном пошаговом обходе ($1 \leq r \leq 10\,000$; гарантируется, что обход, удовлетворяющий этим ограничениям, существует). Во второй строке выведите сами числа u_1, u_2, \dots, u_r через пробел.

Примеры

<i>стандартный ввод</i>	<i>стандартный вывод</i>
3 2 1 1 2 2 3	5 1 2 3 2 1
4 4 1 1 2 2 3 3 4 4 1	5 1 2 3 4 1

Задача D. Связность

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

В этой задаче требуется проверить, что граф является *связным*, то есть что из любой вершины можно по рёбрам этого графа попасть в любую другую.

Формат входных данных

В первой строке заданы числа N и M через пробел — количество вершин и рёбер в графе, соответственно ($1 \leq N \leq 100$, $0 \leq M \leq 10\,000$). Следующие M строк содержат по два числа u_i и v_i через пробел ($1 \leq u_i, v_i \leq N$); каждая такая строка означает, что в графе существует ребро между вершинами u_i и v_i .

Формат выходных данных

Выведите «YES», если граф является связным, и «NO» в противном случае.

Примеры

<i>стандартный ввод</i>	<i>стандартный вывод</i>
3 2 1 2 3 2	YES
3 1 1 3	NO

Задача Е. Зависимости между функциями

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

Даны названия функций и зависимости между ними. Нужно выписать названия функций по одному разу в таком порядке, что:

- Если функция A зависит от функции B , функция B должна быть выписана раньше функции A .
- Если предыдущий критерий позволяет в какой-то момент выписать более чем одну функцию, то выписана должна быть та из них, название которой лексикографически минимально.

Известно, что между функциями нет циклических зависимостей.

Формат входных данных

В первой строке входных данных задано целое число n — количество функций ($1 \leq n \leq 50$). Следующие n строк содержат описания функций. Каждая из этих строк имеет вид $\text{name}_i d_{i,1} d_{i,2} \dots d_{i,k_i}$; здесь name_i — имя функции, а $d_{i,l}$ — номера функций (считая с нуля), от которых зависит данная. Имя функции непусто и состоит из не более чем 50 заглавных букв латинского алфавита; имена всех функций различны. Соседние элементы строки отделены друг от друга одним пробелом. Длина каждой строки не превосходит 200 символов.

Формат выходных данных

Выведите n строк. В каждой строке выведите название одной из функций. Функции должны быть перечислены в требуемом порядке.

Примеры

<i>стандартный ввод</i>	<i>стандартный вывод</i>
3 B 1 C A 1	C A B
3 B 1 C A 0	C B A
10 K A B 1 1 C 2 D 3 E 4 F 5 G 6 H 7 I 8	A B C D E F G H I K

Задача F. Кодовый замок

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

Петя опоздал на тренировку по программированию! Поскольку тренировка проходит в воскресенье, главный вход в учебный корпус, где она проходит, оказался закрыт, а вахтёр ушёл куда-то по своим делам. К счастью, есть другой способ проникнуть в здание — открыть снаружи боковую дверь, на которой установлен кодовый замок.

На пульте замка есть d кнопок с цифрами $0, 1, \dots, d-1$. Известно, что код, открывающий замок, состоит из k цифр. Замок открывается, если последние k нажатий кнопок образуют код.

Поскольку Петя не имеет понятия, какой код открывает замок, ему придётся перебрать все возможные коды из k цифр. Но, чтобы как можно скорее попасть на тренировку, нужно минимизировать количество нажатий на кнопки. Помогите Пете придумать такую последовательность нажатий на кнопки, при которой все возможные коды были бы проверены, а количество нажатий при этом оказалось бы минимально возможным.

Формат входных данных

В первой строке записаны через пробел два целых числа d и k — количество кнопок на пульте и размер кода, соответственно ($2 \leq d \leq 10$, $1 \leq k \leq 20$).

Формат выходных данных

В первой строке выведите искомую последовательность. Если последовательностей минимальной длины, перебирающих все возможные коды, несколько, можно выводить любую из них. Гарантируется, что d и k таковы, что минимальная длина последовательности не превосходит одного мегабайта.

Пример

<i>стандартный ввод</i>	<i>стандартный вывод</i>
2 3	0001011100

Пояснение к примеру

Последовательность в примере перебирает все коды длины 3 в следующем порядке: 000, 001, 010, 101, 011, 111, 110, 100.

Задача G. Мосты

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

Дан неориентированный граф. Требуется найти все мосты в нём.

Формат входных данных

Первая строка содержит два натуральных числа n и m — количество вершин и рёбер графа соответственно ($n \leq 20\,000$, $m \leq 200\,000$).

Следующие m строк содержат описание рёбер по одному на строке. Ребро номер i описывается двумя натуральными числами b_i и e_i — номерами концов ребра ($1 \leq b_i, e_i \leq n$).

Формат выходных данных

Первая строка должна содержать одно натуральное число b — количество мостов в заданном графе. В следующей строке выведите b целых чисел — номера рёбер, которые являются мостами, в возрастающем порядке. Рёбра нумеруются с единицы в том порядке, в котором они заданы во входных данных.

Пример

<i>стандартный ввод</i>	<i>стандартный вывод</i>
6 7	1
1 2	3
2 3	
3 4	
1 3	
4 5	
4 6	
5 6	

Задача Н. Точки сочленения

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

Вам задан неориентированный связный граф с N вершинами и M рёбрами ($1 \leq N \leq 20\,000$, $1 \leq M \leq 200\,000$). В графе отсутствуют петли и кратные рёбра.

Найдите все точки сочленения в заданном графе.

Формат входных данных

Граф задан следующим образом: первая строка содержит числа N и M . Каждая из следующих M строк содержит описание ребра: два целых числа из диапазона от 1 до N — номера концов ребра.

Формат выходных данных

В первой строке выведите число C — количество точек сочленения в заданном графе. В следующей строке выведите C целых чисел — номера вершин, которые являются точками сочленения, в возрастающем порядке.

Пример

<i>стандартный ввод</i>	<i>стандартный вывод</i>
5 6	1
1 2	3
2 3	
1 3	
3 4	
3 5	
4 5	

Задача I. Конденсация графа

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

Конденсация ориентированного графа (V, E) — это ориентированный граф (V', E') , который строится следующим образом. Сначала построим множество вершин нового графа и зададим соответствие $f : V \rightarrow V'$ так, чтобы каждой из вершин нового графа соответствовала хотя бы одна вершина старого графа, а кроме того, если для пары вершин u и v старого графа из u есть путь в v , а из v есть путь в u , в новом графе u и v соответствует одна и та же новая вершина, то есть $f(u) = f(v)$. После этого построим множество рёбер нового графа так: по каждому ребру старого графа $(u, v) \in E$ построим ребро нового графа $(f(u), f(v)) \in E'$. Наконец, удалим из получившегося графа все петли и кратные рёбра.

По заданному ориентированному графу найдите количество рёбер в его конденсации.

Формат входных данных

Первая строка содержит два натуральных числа n и m — количество вершин и рёбер графа соответственно ($1 \leq n \leq 10\,000$, $1 \leq m \leq 100\,000$). Следующие m строк содержат описание рёбер, по одному на строке. Ребро номер i описывается двумя натуральными числами b_i и e_i — началом и концом ребра соответственно ($1 \leq b_i, e_i \leq n$). В графе могут присутствовать кратные рёбра и петли.

Формат выходных данных

Первая строка должна содержать одно число — количество рёбер в конденсации графа.

Пример

<i>стандартный ввод</i>	<i>стандартный вывод</i>
4 4 2 1 3 2 2 3 4 3	2

Задача J. Дерево

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

Дан неориентированный граф. Проверьте, является ли он деревом.

Формат входных данных

В первой строке входных данных заданы через пробел два целых числа n и m — количество вершин и рёбер в графе, соответственно ($1 \leq n \leq 100$). В следующих m строках заданы рёбра; i -я из этих строк содержит два целых числа u_i и v_i через пробел — номера концов i -го ребра ($1 \leq u_i, v_i \leq n$). Граф не содержит петель и кратных рёбер.

Формат выходных данных

В первой строке выведите «YES», если граф является деревом, и «NO» в противном случае.

Примеры

<i>стандартный ввод</i>	<i>стандартный вывод</i>
3 2 1 2 1 3	YES
3 3 1 2 2 3 3 1	NO

Задача К. Расстояние от корня

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

В заданном корневом дереве найдите вершины, максимально удалённые от корня. Расстоянием между вершинами считается количество рёбер в пути.

Формат входных данных

В первой строке задано n — количество вершин в дереве ($1 \leq n \leq 100$). В следующих $n - 1$ строках заданы вершины, являющиеся предками вершин $2, 3, \dots, n$. Вершина 1 является корнем дерева.

Формат выходных данных

В первой строке выведите максимальное расстояние от корня до остальных вершин дерева. Во второй строке выведите, сколько вершин дерева находятся от корня на таком расстоянии. В третьей строке выведите номера этих вершин через пробел в порядке возрастания.

Примеры

<i>стандартный ввод</i>	<i>стандартный вывод</i>
3	1
1	2
1	2 3
3	2
1	1
2	3

Задача L. Лего

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

Рассмотрим трёхмерную конструкцию из блоков, аналогичных блокам известного конструктора «Лего». Конструкция состоит из нескольких прямоугольных слоёв одинакового размера, расположенных один над другим. Каждый слой состоит из блоков. Каждый блок — это плоская фигура из кубиков, связанная по стороне. Соседние слои скреплены между собой таким образом, что нижний кубик в каждом столбике скреплен с верхним. Различные же блоки в одном слое никак не скреплены между собой.

Два блока A и B считаются связанными, если существует такая цепочка блоков $A = C_1, C_2, C_3, \dots, C_{n-1}, C_n = B$, что каждые два соседних блока C_i и C_{i+1} в ней скреплены. Очевидно, всю конструкцию можно поделить на компоненты связности. Найдите их количество.

Формат входных данных

В первой строке ввода записаны через пробел три числа x , y и z — длина, ширина и высота конструкции, соответственно ($1 \leq x, y, z \leq 100$). Далее размещены z блоков по y строк, описывающие конструкцию. В каждой из этих строк ровно x символов. Перед каждым блоком расположена дополнительно одна пустая строка. Каждый кубик описывается одной заглавной буквой английского алфавита. Два кубика в одном слое принадлежат одному блоку, если они соседние по стороне и обозначены одной и той же буквой.

Формат выходных данных

Выведите одно число — количество компонент связности в заданной конструкции.

Примеры

<i>стандартный ввод</i>	<i>стандартный вывод</i>
2 2 2 AA AA AB CD	1
3 3 2 AAB BCB BAA CAA CEB DDB	2
4 3 1 AABV ACCB AABV	3

Пояснения к примерам

В первом примере четыре отдельных кубика второго слоя крепятся к одному блоку, занимающему весь первый слой. В конструкции одна компонента связности.

Во втором примере блоки из двух кубиков образуют связный «колодец». Два блока в центре связаны друг с другом, но не связаны с «колодцем». Количество компонент связности в этой конструкции равно двум.

В третьем примере в конструкции всего один слой, состоящий из трёх блоков. Отметим, что блоки не обязательно имеют прямоугольную форму. В этой конструкции три компоненты связности.