Содержание

Вы не умеете читать/выводить данные, открывать файлы? Воспользуйтесь примерами.

В некоторых задачах большой ввод и вывод. Пользуйтесь быстрым вводом-выводом.

В некоторых задачах нужен STL, который активно использует динамическую память (set-ы, map-ы) переопределение стандартного аллокатора ускорит вашу программу.

Обратите внимание на компилятор `GNU C++11 5.1.0 (TDM-GCC-64) inc`, который позволяет пользоваться дополнительной библиотекой. Под ним можно сдать вот это.

Must have

Задача 2A. Любители Кошек [0.2 sec, 256 mb]

В университетском клубе любителей кошек зарегистрировано n членов. Естественно, что некоторые из членов клуба знакомы друг с другом. Нужно сосчитать, сколькими способами можно выбрать из них троих, которые могли бы свободно общаться (то есть, любые два из которых знакомы между собой).

Формат входных данных

В первой строке входного файла заданы числа n и m ($1 \leqslant n \leqslant 1\,000$, $1 \leqslant m \leqslant 30\,000$), где m обозначает общее число знакомств. В последующих m строках идут пары чисел a_i b_i, обозначающие, что a_i знаком с b_i. Информация об одном знакомстве может быть записана несколько раз, причем даже в разном порядке (как (x, y), так и (y, x)).

Формат выходных данных

В выходной файл необходимо вывести количество способов выбрать троих попарно знакомых друг с другом людей из клуба.

Пример

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

Замечание

Предполагается решение за $\mathcal{O}(nm)$. Если каждый треугольник сразу считать ровно один раз, получится быстрее.

Задача 2B. Longpath. Длиннейший путь [0.1 sec, 256 mb]

Дан ориентированный граф без циклов. Требуется найти в нем длиннейший путь.

Формат входных данных

Первая строка входного файла содержит два натуральных числа n и m — количество вершин и дуг графа соответственно. Следующие m строк содержат описания дуг по одной на строке. Ребро номер i описывается двумя натуральными числами b_i и e_i — началом и концом дуги соответственно ($1 \leqslant b_i, e_i \leqslant n$).

Входной граф не содержит циклов и петель.

$n \leqslant 10\,000$, $m \leqslant 100\,000$.

Формат выходных данных

Первая строка выходного файла должна содержать одно натуральное число — количество дуг в длиннейшем пути.

Пример

стандартный ввод	стандартный вывод
5 5 1 2 2 3 3 4 3 5 1 5	3

Замечание

В орграфе без циклов умеем почти что угодно считать dfs-ом.

Задача 2C. Avia. Авиаперелеты [1.0 sec, 256 mb]

Главного конструктора Петю попросили разработать новую модель самолета для компании «Air Бубундия». Оказалось, что самая сложная часть заключается в подборе оптимального размера топливного бака.

Главный картограф «Air Бубундия» Вася составил подробную карту Бубундии. На этой карте он отметил расход топлива для перелета между каждой парой городов.

Петя хочет сделать размер бака минимально возможным, для которого самолет сможет долететь от любого города в любой другой (возможно, с дозаправками в пути).

Формат входных данных

Первая строка входного файла содержит натуральное число n ($1 \leqslant n \leqslant 1000$) — число городов в Бубундии. Далее идут n строк по n чисел каждая. j-ое число в i-ой строке равно расходу топлива при перелете из i-ого города в j-ый. Все числа не меньше нуля и меньше 10^9. Гарантируется, что для любого i в i-ой строчке i-ое число равно нулю.

Формат выходных данных

Первая строка выходного файла должна содержать одно число — оптимальный размер бака.

Пример

стандартный ввод	стандартный вывод
4 0 10 12 16 11 0 8 9 10 13 0 22 13 10 17 0	10

Замечание

Простая задача

Обязательные задачи

Задача 2D. Condense 2. Конденсация графа [0.3 sec, 256 mb]

Требуется найти количество ребер в конденсации ориентированного графа. Примечание: конденсация графа не содержит кратных ребер.

Формат входных данных

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

Формат выходных данных

Первая строка выходного файла должна содержать одно число — количество ребер в конденсации графа.

Пример

стандартный ввод	стандартный вывод
4 4 2 1 3 2 2 3 4 3	2

Замечание

Стандартный алгоритм.

Задача 2E. Bridges. Мосты [0.3 sec, 256 mb]

Дан неориентированный граф. Требуется найти все мосты в нем.

Формат входных данных

Первая строка входного файла содержит два натуральных числа n и m — количество вершин и ребер графа соответственно ($n \leqslant 20\,000$, $m \leqslant 200\,000$).

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

Формат выходных данных

Первая строка выходного файла должна содержать одно натуральное число b — количество мостов в заданном графе. На следующей строке выведите b целых чисел — номера ребер, которые являются мостами, в возрастающем порядке. Ребра нумеруются с единицы в том порядке, в котором они заданы во входном файле.

Пример

стандартный ввод	стандартный вывод
6 7 1 2 2 3 3 4 1 3 4 5 4 6 5 6	1 3

Задача 2F. Мосты и компоненты [0.3 sec, 256 mb]

Дан неориентированный граф (не обязательно связный). Граф может содержать петли и кратные ребра.

Выведите все компоненты реберной двусвязности графа (максимальные подмножества вершин, такие что подграф на них не теряет связность при удалении любого ребра).

Формат входных данных

Первая строка содержит числа n и m ($1 \leqslant n \leqslant 100\,000$, $0 \leqslant m \leqslant 100\,000$) — количество вершин и ребер в графе.

Следующие m строк задают ребра графа.

Формат выходных данных

В первой строке выведите количество компонент, в следующих за ней строках выведите сами компоненты, по одной на строку.

Вершины в каждой компоненте должны идти в возрастающем порядке, компоненты нужно вывести в лексикографическом порядке.

Примеры

стандартный ввод	стандартный вывод
3 2 1 2 2 3	3 1 2 3
3 3 1 2 2 3 3 1	1 1 2 3
2 2 1 2 1 2	1 1 2
7 8 1 5 5 6 1 6 5 4 4 3 4 2 3 2 7 2	3 1 5 6 2 3 4 7

Замечание

Стандартный алгоритм.

Задача 2G. Points. Точки сочленения [0.3 sec, 256 mb]

Дан неориентированный граф без петель а кратных рёбер. Требуется найти все точки сочленения в нем.

Формат входных данных

Первая строка входного файла содержит два натуральных числа n и m — количество вершин и ребер графа соответственно ($n \leqslant 20\,000$, $m \leqslant 200\,000$).

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

Формат выходных данных

Первая строка выходного файла должна содержать одно натуральное число b — количество точек сочленения в заданном графе. На следующей строке выведите b целых чисел — номера вершин, которые являются точками сочленения, в возрастающем порядке.

Пример

стандартный ввод	стандартный вывод
9 12 1 2 2 3 4 5 2 6 2 7 8 9 1 3 1 4 1 5 6 7 3 8 3 9	3 1 2 3

Замечание

Стандартный алгоритм.

Задача 2H. Компоненты вершинной двусвязности [0.5 sec, 256 mb]

Компонентой вершинной двусвязности графа $\langle V, E\rangle$ называется максимальный по включению подграф (состоящий из вершин и ребер), такой что любые два ребра из него лежат на вершинно простом цикле.

Дан неориентированный граф без петель. Требуется выделить компоненты вершинной двусвязности в нем.

Формат входных данных

Первая строка входного файла содержит два натуральных числа n и m — количества вершин и ребер графа соответственно ($n \leqslant 20\,000$, $m \leqslant 200\,000$).

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

Формат выходных данных

В первой строке выходного файла выведите целое число k — количество компонент вершинной двусвязности графа. Во второй строке выведите m натуральных чисел $a_1, a_2, \ldots, a_m$, не превосходящих k, где a_i — номер компоненты вершинной двусвязности, которой принадлежит i-е ребро. Ребра нумеруются с единицы в том порядке, в котором они заданы во входном файле.

Примеры

стандартный ввод	стандартный вывод
5 6 1 2 2 3 3 1 1 4 4 5 5 1	2 1 1 1 2 2 2

Замечание

Стандартный алгоритм.

Задача 2I. Из истории банка Гринготтс [0.1 sec, 256 mb]

Чтобы понять название задачи, можно прочитать красивую легенду.
http://acm.timus.ru/problem.aspx?space=1&num=1441

Задача же заключается в том, чтобы рёбра неориентированного графа разбить на минимальное число путей.

Формат входных данных

Дан граф. На первой строке число вершин n ($1 \leqslant n \leqslant 20\,000$) и число рёбер m ($1 \leqslant m \leqslant 20\,000$). Следующие m строк содержат описание рёбер графа. Каждая строка по два числа a_i b_i ($1 \leqslant a_i, b_i \leqslant n$). Между каждыми двумя вершинами не более одного ребра. Граф связен.

Формат выходных данных

На первой строке минимальное число путей. На каждой следующей описание очередного пути – номера вершин в порядке прохождения.

Примеры

стандартный ввод	стандартный вывод
7 7 1 2 4 1 6 7 5 7 7 4 2 3 4 2	3 5 7 4 2 1 4 2 3 6 7

Замечание

Задача с практики.

Дополнительные задачи

Задача 2J. King's Assassination [1.0 sec, 256 mb]

Дан граф из n вершин и m ребер. Граф ориентированный. Нужно определить число вершин, содержащихся на всех путях из s в t (сами s и t учитывать не нужно).

Формат входных данных

Первая строка содержит n, m, s и t ($2 \leqslant n \leqslant 100\,000$, $1 \leqslant m \leqslant 300\,000$, $1 \leqslant s, t \leqslant n$, $s \neq t$).

Следующие m строк содержат пары чисел x_i и y_i — индексы вершин от 1 до n. Это означает что есть дорога из вершины с номером x_i в вершину с номером y_i.

Формат выходных данных

Число вершин k. Далее k чисел — номера вершин в возрастающем порядке.

Примеры

стандартный ввод	стандартный вывод
4 3 1 4 1 2 2 3 3 4	2 2 3
4 4 1 4 1 2 2 3 3 4 1 3	1 3
4 5 1 4 1 2 2 3 3 4 1 3 2 4	0

Замечание

Задача с практики. Удивительно, но решение для ациклического графа работает и при наличии циклов.

Задача 2K. Кодовый замок [1.0 sec, 256 mb]

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 $0, 1, \ldots, d-1$. 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 входного файла записаны через пробел два целых числа d и k — количество кнопок на пульте и размер кода, соответственно ($2 \leqslant d \leqslant 10$, $1 \leqslant k \leqslant 20$).

Формат выходных данных

В первой строке выходного файла 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 1 мебибайта.

Пример

стандартный ввод	стандартный вывод
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 2L. Раскраска в три цвета [0.1 sec, 256 mb]

Петя нарисовал на бумаге n кружков и соединил некоторые пары кружков линиями. После этого он раскрасил каждый кружок в один из трех цветов — красный, синий или зеленый.

Теперь Петя хочет изменить их раскраску. А именно — он хочет перекрасить каждый кружок в некоторый другой цвет так, чтобы никакие два кружка одного цвета не были соединены линией. При этом он хочет обязательно перекрасить каждый кружок, а перекрашивать кружок в тот же цвет, в который он был раскрашен исходно, не разрешается.

Помогите Пете решить, в какие цвета следует перекрасить кружки, чтобы выполнялось указанное условие.

Формат входных данных

Первая строка содержит два целых числа n и m — количество кружков и количество линий, которые нарисовал Петя, соответственно ($1 \leqslant n \leqslant 1\,000$, $0 \leqslant m \leqslant 20\,000$).

Следующая строка содержит n символов из множества {“R”, “G”, “B”} — i-й из этих символов означает цвет, в который раскрашен i-й кружок (“R” — красный, “G” — зеленый, “B” — синий).

Следующие m строк содержат по два целых числа — пары кружков, соединенных отрезками.

Формат выходных данных

Выведите в выходной файл одну строку, состоящую из n символов из множества {“R”, “G”, “B”} — цвета кружков после перекраски. Если решений несколько, выведите любое.

Если решения не существует, выведите в выходной файл слово “Impossible”.

Пример

стандартный ввод	стандартный вывод
4 5 RRRG 1 3 1 4 3 4 2 4 2 3	BBGR
4 5 RGRR 1 3 1 4 3 4 2 4 2 3	Impossible

Замечание

Задача с практики. Можно сдать за $O(nm)$, можно за $O(n + m)$.

Задача 2M. Chip Installation [0.5 sec, 256 mb]

Новый ЧИП скоро установят в новый летательный апарат, недавно выпущенной компанией `Airtram`. ЧИП имеет форму диска. Есть n проводов, которые нужно подсоединить к ЧИПу.

Каждый провод можно подсоединить в один из двух разъемов, допустимых для этого провода. Все $2n$ разъемов расположены на границе диска. По кругу. Каждый провод имеет свой цвет. Для повышения безопасности два провода одного цвета не могут быть подсоединены к соседним разъемам.

Дана конфигурация разъемов на ЧИПе, найдите способ подсоединить все провода, не нарушающий условия про цвета.

Формат входных данных

Первая строка содержит число n — количество проводов ($1 \leqslant n \leqslant 50\,000$). Вторая строка содержит n целых чисел от 1 до 10^9 — цвета проводов. Цвета проводов могут совпадать. Третья строка содержит $2n$ целых чисел от 1 до n описывающих разъемы. Число обозначает номер провода, который может быть подсоединен к данному разъему. Каждое число от 1 до n встречается ровно дважды. Разъемы перечислены в порядке "по кругу". 1-й разъем является соседним со 2-м и так далее, не забудьте, что n-й является соседним с 1-м.

Формат выходных данных

Если не существует способа подключить все провода, выведите одно слово “`NO`”.

Иначе выведите “`YES`” и n целых чисел. Для каждого провода выведите номер разъема, к которому нужно подключить этот провод. Разъемы нумеруются числами от 1 до $2n$ в том порядке, в котором они даны во входном файле.

Пример

стандартный ввод	стандартный вывод
2 1 1 1 1 2 2	YES 1 3
2 1 1 1 2 1 2	NO
2 1 2 1 2 1 2	YES 1 2

Замечание

Две задачи по цене одной: эта есть и в теордз.