Содержание

Обратите внимание, входные данные лежат в **стандартном потоке ввода** (он же stdin), вывести ответ нужно в **стандартный поток вывода** (он же stdout).

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 `GNU C++` компиляторы с суффиксом `inc`, они позволяют пользоваться дополнительной библиотекой. Под ними можно сдать вот это.

Must have

Задача 6A. Диаметр графа [0.1 sec, 256 mb]

Дан связный взвешенный неориентированный граф.

Рассмотрим пару вершин, расстояние между которыми максимально среди всех пар вершин. Расстояние между ними называется *диаметром графа*. *Эксцентриситетом вершины* v называется максимальное расстояние от вершины v до других вершин графа. *Радиусом графа* называется наименьший из эксцентриситетов вершин. Найдите диаметр и радиус графа.

Формат входных данных

В первой строке входного файла единственное число: N ($1 \leqslant N \leqslant 100$) — количество вершин графа. В следующих N строках по N чисел — матрица смежности графа, где -1 означает отсутствие ребра между вершинами, а любое неотрицательное число — присутствие ребра данного веса. На главной диагонали матрицы всегда нули; веса рёбер не превышают 1000.

Формат выходных данных

В выходной файл выведите два числа — диаметр и радиус графа.

Пример

stdin	stdout
4 0 -1 1 2 -1 0 -1 5 1 -1 0 4 2 5 4 0	8 5

Замечание

Решения больше 20 строк кода считаются неприличными.

Задача 6B. Path. Кратчайший путь [0.5 sec, 256 mb]

Дан взвешенный ориентированный граф и вершина s в нем. Требуется для каждой вершины u найти длину кратчайшего пути из s в u.

Формат входных данных

Первая строка входного файла содержит n, m и s — количество вершин, ребер и номер выделенной вершины соответственно ($2 \leqslant n \leqslant 2\,000$, $1 \leqslant m \leqslant 6\,000$).

Следующие m строк содержат описание ребер. Каждое ребро задается стартовой вершиной, конечной вершиной и весом ребра. Вес каждого ребра — целое число, не превосходящее 10^{15} по модулю. В графе могут быть кратные ребра и петли.

Формат выходных данных

Выведите n строк — для каждой вершины u выведите длину кратчайшего пути из s в u, «*» если не существует путь из s в u и «-» если не существует кратчайший путь из s в u.

Пример

stdin	stdout
6 7 1 1 2 10 2 3 5 1 3 100 3 5 7 5 4 10 4 3 -18 6 1 -1	0 10 - - - *

Замечание

Форд-Беллман обыкновенный.

Пожалуйста, будьте аккуратны с бесконечностями. И минус бесконечностями.

Обязательные задачи

Задача 6C. Цикл отрицательного веса [0.1 sec, 256 mb]

Дан ориентированный граф. Определите, есть ли в нем цикл отрицательного веса, и если да, то выведите его.

Формат входных данных

Во входном файле в первой строке число N ($1 \leqslant N \leqslant 100$) — количество вершин графа. В следующих N строках находится по N чисел — матрица смежности графа. Все веса ребер не превышают по модулю 10 000. Если ребра нет, то соответствующее число равно 100 000.

Формат выходных данных

В первой строке выходного файла выведите «YES», если цикл существует или «NO» в противном случае. При его наличии выведите во второй строке количество вершин в искомом цикле и в третьей строке — вершины входящие в этот цикл в порядке обхода.

Пример

stdin	stdout
2 0 -1 -1 0	YES 2 1 2

Замечание

Петля – тоже цикл.
Можно сдать Флойдом, можно Форд-Беллманом.

Задача 6D. Currency Exchange [0.1 sec, 256 mb]

Several currency exchange points are working in our city. Let us suppose that each point specializes in two particular currencies and performs exchange operations only with these currencies. There can be several points specializing in the same pair of currencies. Each point has its own exchange rates, exchange rate of A to B is the quantity of B you get for $1A$. Also each exchange point has some commission, the sum you have to pay for your exchange operation. Commission is always collected in source currency.

For example, if you want to exchange 100 US Dollars into Russian Rubles at the exchange point, where the exchange rate is 29.75, and the commission is 0.39 you will get $(100 - 0.39) \cdot 29.75 = 2963.3975$ RUR.

You surely know that there are N different currencies you can deal with in our city. Let us assign unique integer number from 1 to N to each currency. Then each exchange point can be described with 6 numbers: integer A and B – numbers of currencies it exchanges, and real RAB, CAB, RBA and CBA – exchange rates and commissions when exchanging A to B and B to A respectively.

Nick has some money in currency S and wonders if he can somehow, after some exchange operations, increase his capital. Of course, he wants to have his money in currency S in the end. Help him to answer this difficult question. Nick must always have non-negative sum of money while making his operations.

Формат входных данных

The first line contains four numbers: N – the number of currencies, M – the number of exchange points, S – the number of currency Nick has and V – the quantity of currency units he has. The following M lines contain 6 numbers each – the description of the corresponding exchange point – in specified above order. Numbers are separated by one or more spaces. $1 \leqslant S \leqslant N \leqslant 100$, $1 \leqslant M \leqslant 100$, V is real number, $0 \leqslant V \leqslant 10^3$.

For each point exchange rates and commissions are real, given with at most two digits after the decimal point, $10^{-2} \leqslant rate \leqslant 10^2$, $0 \leqslant commission \leqslant 10^2$.

Let us call some sequence of the exchange operations simple if no exchange point is used more than once in this sequence. You may assume that ratio of the numeric values of the sums at the end and at the beginning of any simple sequence of the exchange operations will be less than 10^4.

Формат выходных данных

If Nick can increase his wealth, output `YES`, in other case output `NO`.

Примеры

stdin	stdout
3 2 1 10.0 1 2 1.0 1.0 1.0 1.0 2 3 1.1 1.0 1.1 1.0	NO
3 2 1 20.0 1 2 1.0 1.0 1.0 1.0 2 3 1.1 1.0 1.1 1.0	YES

Замечание

Отличается от разобранного на практике наличием комиссии.

Задача 6E. Цикл минимального среднего веса [0.7 sec, 256 mb]

Дан ориентированный граф, содержащий n вершин и m ребер. Каждое ребро имеет некоторый вес. Простым циклом называется последовательность рёбер $(a_1, b_1), (a_2, b_2), \ldots, (a_k, b_k)$ такая, что все a_i различны, $b_i = a_{i+1}$, $b_k = a_1$. Весом цикла называется сумма весов составляющих его ребер. Средним весом цикла называется отношение веса цикла к числу составляющих его ребер.

Требуется найти в графе **простой** цикл минимального среднего веса.

Формат входных данных

Первая строка входного файла содержит числа n и m ($3 \leqslant n \leqslant 1000$, $3 \leqslant m \leqslant 2000$). Следующие m строк описывают ребра графа. Каждое ребро описывается тремя целыми числами u_i, v_i и w_i: номер начальной вершины, номер конечной вершины и вес ребра ($1 \leqslant u_i, w_i \leqslant n$, $-1000 \leqslant w_i \leqslant 1000$). Гарантируется, что в графе есть цикл. В графе может быть несколько ребер между одной и той же парой вершин, а также могут быть петли.

Формат выходных данных

На первой строке выведите число z — минимальный возможный средний вес цикла в заданном графе. На второй строке выведите число k — число ребер в таком цикле. На третьей строке выведите k чисел — номера ребер в порядке обхода вдоль цикла. Ребра пронумерованы от 1 до m в порядке, в котором они заданы во входном файле. Если возможных циклов несколько, можно вывести любой. Требуется вывести простой цикл. Число z должно быть выведено с абсолютной или относительной погрешностью 10^{-9}.

Пример

stdin	stdout
5 8 1 2 10 2 3 1 3 4 2 4 5 3 5 1 8 5 5 7 5 2 4 3 5 4	2.5 4 2 3 4 7

Замечание

Решение с бинпоиском зайдёт.

А можно потренироваться писать Карпа.

Дополнительные задачи

Задача 6F. Потенциал [2.5 sec, 256 mb]

Дан взвешенный ориентированный граф. Пусть у каждой вершины есть потенциал Φ_i. Тогда к весу каждого ребра прибавляется потенциал начала и вычитается потенциал конца.

Требуется найти такие целые Φ_i, чтобы веса у всех рёбер были одинаковыми.

Формат входных данных

В первой строке задано целое число t — количество тестовых случаев.

В первой строке каждого тестового случая заданы целые числа n и m ($1 \leqslant n \leqslant 300\,000$, $0 \leqslant m \leqslant 300\,000$) — количество вершин и рёбер в графе. В следующих m строках задано по три целых числа x_i, y_i и w_i ($1 \leqslant x_i, y_i \leqslant n$, $-10^9 \leqslant w_i \leqslant 10^9$) — начало, конец и вес ребра. Гарантируется, что граф не содержит кратных рёбер и петель.

Также гарантируется, что сумма всех n и m по всем тестовым случаям не превосходит $600\,000$.

Формат выходных данных

Для каждого тестового случая выведите «YES», если существует целочисленное решение, и «NO» в противном случае.

Если ответ положительный, то в следующей строке выведите n целых чисел — потенциалы вершин. Все выведенные числа должны быть не больше 10^{18} по абсолютной величине. Гарантируется, что если ответ существует, то существует и ответ, удовлетворяющий этому ограничению.

Если возможных ответов несколько, выведите любой из них.

Пример

stdin	stdout
2 5 4 1 2 -1 2 3 2 3 4 1 4 5 179 5 5 1 2 1 2 3 1 3 4 1 4 5 0 5 1 2	YES 0 -1 1 2 181 YES 0 0 0 0 -1

Задача 6G. Лифтостроитель Эдуард [0.2 sec, 256 mb]

Эдуард работает инженером в компании «Нетривиальные лифты». Его очередное задание — разработать новый лифт для небоскрёба из h этажей.

У Эдуарда есть идея-фикс: он считает, что четырёх кнопок должно хватать каждому. Его последнее конструктивное предложение предполагает следующие кнопки:

- Подняться на a этажей вверх
- Подняться на b этажей вверх
- Подняться на c этажей вверх
- Вернуться на первый этаж

Исходно лифт находится на первом этаже. Пассажир использует три первые кнопки, чтобы попасть на нужный этаж. Если пассажир пытается переместиться на этаж, которого не существует, то есть нажать одну из первых трёх кнопок на этаже со слишком большим номером, лифт не перемещается.

Чтобы доказать, что план достоин реализации, Эдуард хочет подсчитать количество этажей, до которых возможно доехать с его помощью.

Формат входных данных

В первой строке записано целое число h — количество этажей небоскрёба ($1 \leqslant h \leqslant 10^{18}$).

Во второй строке записаны целые числа a, b и c — параметры лифта ($1 \leqslant a, b, c \leqslant 100\,000$).

Формат выходных данных

Выведите одно целое число — количество этажей, доступных с первого с помощью лифта.

Пример

stdin	stdout
15 4 7 9	9

Замечание

Самое важное – увидеть правильный граф.

Задача 6H. Грамматика [3 sec, 256 mb]

Формальной грамматикой называется способ описания формального языка, представляющий собой четвёрку $\Gamma = \langle \Sigma, N, S \in N, P \subset N^+ \times (\Sigma \cup N)^* \rangle$, где Σ — *алфавит*, элементы которого называют *терминалами*, N — множество, элементы которого называют *нетерминалами*, S — начальный нетерминал грамматики, P — набор *правил вывода* вида $\alpha \to \beta$.

Здесь N^+ — это все строки из одного или нескольких элементов множества N (непустые строки из нетерминалов), а $(\Sigma \cup N)^*$ — это все строки из нуля, одного или нескольких элементов множества $(\Sigma \cup N)$ (строки из терминалов и нетерминалов, включая пустую).

Грамматика называется *контекстно-свободной*, если в левой части каждого правила вывода всегда стоит только один нетерминал, то есть верно более сильное условие для правил: $P \subset N \times (\Sigma \cup N)^*$.

Для примера рассмотрим такую грамматику (второй тестовый случай примера) над алфавитом $\Sigma = \{\texttt{a}, \texttt{b}\}$, множеством нетерминалов $N = \{S, A\}$ и двумя правилами вывода:

1. $S \to \texttt{b}A$
2. $A \to \texttt{aa}$

Она, очевидно, является контекстно-свободной.

Чтобы получить язык, описываемый грамматикой, нужно взять строку, составленную из одного начального нетерминала S, и применить к ней один или несколько раз какие-либо правила вывода. Применение правила вывода состоит в нахождении в текущей строке в каком-либо месте строки из левой части этого правила и замены её на правую часть этого же правила вывода. *Языком* грамматики Γ будет называться множество всех строк, состоящих **только** из терминальных символов и выводимых по такому принципу.

Например, в языке описанной выше грамматики есть строка `baa`, выводимая следующим образом: $S \to \texttt{b}A \to \texttt{baa}$. Более того, других строк в её языке нет.

Но бывают и грамматики, в языке которых бесконечное количество строк. А бывают и такие, язык которых пуст.

Вам дана контекстно-свободная грамматика, алфавит в которой состоит из двух терминалов «`a`» и «`b`». Ваша задача — проверить, есть ли в её языке строка, в которой символ «`a`» встречается строго большее число раз, чем символ «`b`».

Нетерминалы в этой задаче будут нумероваться числами от 1 до n. Начальный нетерминал всегда имеет номер 1.

Формат входных данных

Во вводе заданы один или несколько тестовых случаев.

В первой строке каждого тестового случая записаны два числа n и m — число нетерминалов и количество правил вывода ($1 \leqslant n \leqslant 100$, $1 \leqslant m \leqslant 50\,000$).

Далее следуют m строк, каждая из которых описывает одно правило вывода в следующем виде. Сначала заданы A_i и k_i — номер нетерминала в левой части ($1 \leqslant A_i \leqslant n$) и число объектов в правой части правила вывода ($0 \leqslant k_i \leqslant 100$), далее следует k_i объектов, каждый из которых — либо нетерминал $B_{i,j}$ ($1 \leqslant B_{i,j} \leqslant n$), либо один из терминалов «`a`» или «`b`». Объекты разделяются одиночными пробелами.

Общая сумма значений n по всем тестовым случаям не превосходит 1000, а общая сумма значений m не превосходит 50 000. Размер ввода не превышает 5 мегабайт.

Ввод завершается строкой из двух нулей.

Формат выходных данных

Для каждого из тестовых случаев необходимо вывести на отдельной строке «YES», если в языке данной грамматики есть строка, в которой количество букв «a» строго больше, чем количество букв «b», и «NO» в противном случае.

Пример

stdin	stdout
2 2 1 2 a 2 2 1 b 2 2 1 2 b 2 2 2 a a 2 2 1 2 b 2 2 3 a a 1 0 0	NO YES NO