Содержание

Общая информация:

Вход в контест: `http://contest.yandex.ru/contest/817/`
Дедлайн на задачи 19-го ноября в 23:59.
К каждой главе есть более простые задачи (base), посложнее (advanced), и сложные (hard).
В скобках к каждой главе написано сколько любых задач из этой главы нужно сдать.
Сайт курса: `http://compscicenter.ru/courses/algorithms-1/2014-autumn/`
Семинары ведет Сергей Владимирович Копелиович,
контакты: burunduk30@gmail.com, vk.com/burunduk1

В каждом условии 2 таймлимита: для `C/C++` и для `Java, Python`.

08.Base [2/2]

1 Задача 8A. Сумма расстояний [0.5 sec (1 sec)]

Дан связный неориентированный граф. Требуется найти сумму расстояний между всеми парами вершин.

Формат входных данных

Первая строка входного файла содержит два натуральных числа n и m — количество вершин и ребер графа соответственно ($1 \leqslant n \leqslant 1000$, $0 \leqslant m \leqslant 10\,000$).

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

Гарантируется, что граф связен.

Формат выходных данных

Первая строка выходного файла должна содержать одно натуральное число — сумму попарных расстояний между вершинами.

Пример

sumdist.in	sumdist.out
5 5 1 2 2 3 3 4 5 3 1 5	16

2 Задача 8B. Диаметр графа [0.5 sec (1 sec)]

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

diameter.in	diameter.out
4 0 -1 1 2 -1 0 -1 5 1 -1 0 4 2 5 4 0	8 5

08.Advanced [3/5]

3 Задача 8C. Стоимость проезда [0.5 sec (1 sec)]

Страна состоит из n городов и m дорог. Города пронумерованы числами от 1 до n. Город с номером s является столицей. Все дороги односторонние, проход по каждой дороге стоит ровно 1 золотой. Требуется найти минимальные стоимости проезда от каждого города до столицы.

Формат входных данных

В первой строке файла записаны три целых числа — n, s и m (количество городов, номер столичного города и количество дорог).

В следующих m строках записаны пары чисел. Пара чисел (a, b) означает, что есть дорога из города a в город b.

Ограничения: $1 \leqslant n \leqslant 10^5, 0 \leqslant m \leqslant 10^5$.

Формат выходных данных

Выведите n чисел — минимальные стоимости проезда от городов до столицы. Если от какого-то города не существует ни одного пути до столицы, выведите -1.

Пример

bfsrev.in	bfsrev.out
3 2 2 1 2 2 3	1 0 -1

4 Задача 8D. Кратчайший путь коня [0.5 sec (1 sec)]

На шахматной доске размером 8×8 заданы две клетки. Соедините эти клетки кратчайшим путем коня.

Формат входных данных

Программа получает на вход координаты двух клеток, каждая в отдельной строке. Координаты клеток задаются в виде буквы (от “a” до “h”) и цифры (от 1 до 8) без пробелов.

Формат выходных данных

Программа должна вывести путь коня, начинающийся и заканчивающийся в данных клетках и содержащий наименьшее число клеток.

Пример

knight1.in	knight1.out
a1 h8	a1 b3 a5 b7 d8 f7 h8

5 Задача 8E. Get the Duck to the Sink [0.5 sec (1 sec)]

Как-то профессор Налейпиво заметил, что один из студентов на его лекции уделяет слишком много внимания мобильному телефону. Подкравшись сзади (а несмотря на большие размеры, профессор Налейпиво умеет незаметно подкрадываться), профессор обнаружил смягчающее вину студента обстоятельство — тот не отправлял SMS-ки, а увлечённо играл в следующую игру.

Есть поле размером $N \times M$ ячеек и несколько (возможно ноль) стен между ячейками. Одна из ячеек является *стоком*, в то время как одна из оставшихся занята *уткой*. Ваша задача привести *утку* в *сток*. Единственный способ передвижения *утки*, доступный вам — это *скольжение*, что подразумевает, что вы можете толкнуть *утку* в одном из четырёх направлений, и она будет *скользить* в нем, пока не упрется в стену. Вы не можете толкнуть *утку* снова, пока она не остановится. Уровень считается пройденным, если *утка* останавливается в *стоке*. Если *утка* просто *проскользнула* через *сток*, не остановившись, уровень не считается пройденным.

Профессор остановил лекцию и попросил студента создать несколько своих уровней. Он расчертил на доске уровень, нанёс стены, поместил *сток*, и теперь студенту надо разместить где-то *утку*. Профессор выбрал несколько мест, куда бы он хотел поместить её, и предложил студенту описать алгоритм прохождения уровня. Студент быстро понял, что это ловушка — среди них есть такие, начав игру из которых, довести *утку* до *стока* невозможно. А сумеете ли это сделать Вы?

Ваша задача — имея размеры поля, позицию стен и *стока*, а также выбранные позиции для *утки*, найти среди выбранных позиций такие, начав игру из которых пройти уровень возможно.

Формат входных данных

Первая строка содержит два числа N и M — размеры поля.

Далее следует $2N + 1$ строк, каждая по $2M + 1$ символов, где $2k$-ый символ $2i$-ой стоки либо пробел, если ячейка пуста, либо `S` если ячейка содержит сток либо `D` если ячейка входит в список выбранных для размещения утки.

$(2k + 1)$-ый символ $2i$-ой строки либо пробел, если $k > 0$ и $k < M$ и нет стены между ячейками (k,i) и $(k + 1,i)$; либо `|` в противном случае.

$2k$-ый символ $(2i + 1)$-ой строки либо пробел, если $i > 0$ и $i < N$ и нет стены между ячейками (k,i) и $(k,i + 1)$; либо `-` в противном случае.

$(2k + 1)$-ый символ $(2i + 1)$-ой строки всегда `+`.

$1 \leqslant N \leqslant 1000$

$1 \leqslant M \leqslant 1000$

Формат выходных данных

Выведите поле из входного файла, заменив буквы `D` на пробел в ячейках, начиная с которых, нельзя пройти уровень.

Примеры

getduck.in	getduck.out
5 5 +-+-+-+-+-+ | | + +-+ + + + | S D D| + + + + + + | D | | + + + + + + | | | | + + + +-+ + | | +-+-+-+-+-+	+-+-+-+-+-+ | | + +-+ + + + | S D | + + + + + + | D | | + + + + + + | | | | + + + +-+ + | | +-+-+-+-+-+

6 Задача 8F. Зависимости между функциями [0.5 sec (1 sec)]

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 входного файла задано целое число n — количество функций ($1 \leqslant n \leqslant 50$). Следующие n строк содержат описания функций. Каждая из этих строк имеет вид $\texttt{name}_i\ d_{i,1}\ d_{i,2}\ \ldots\ d_{i,k_i}$; здесь $\texttt{name}_i$ — 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 в выходной файл n строк. В каждой строке выведите название одной из функций. Функции должны быть перечислены в требуемом порядке.

Примеры

functions.in	functions.out
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

7 Задача 8G. Расстояние между вершинами [0.5 sec (1 sec)]

Дан неориентированный взвешенный граф. Найти вес минимального пути между двумя вершинами.

Формат входных данных

Первая строка входного файла содержит натуральные числа N, M, вторая строка содержит натуральные числа S и F ($N \leqslant 5\,000$, $M \leqslant 100\,000$, $1 \leqslant S, F \leqslant N$, $S \neq F$) — количество вершин и ребер графа а также номера вершин, длину пути между которыми требуется найти.

Следующие M строк по три натуральных числа b_i, e_i и w_i — номера концов i-ого ребра и его вес соответственно ($1 \leqslant b_i, e_i \leqslant n$, $0 \leqslant w_i \leqslant 100\,000$).

Формат выходных данных

Первая строка должна содержать одно натуральное число — вес минимального пути между вершинами S и F. Во второй строке через пробел выведите вершины на кратчайшем пути из S в F в порядке обхода.

Если путь из S в F не существует, выведите -1.

Пример

distance.in	distance.out
4 4 1 3 1 2 1 3 4 5 3 2 2 4 1 4	3 1 2 3

08.Hard [0/1]

8 Задача 8H. Грязь [0.5 sec (1 sec)]

— Здравствуйте! Могу я поговорить с Петровым? Алё, милый, привет... ты знаешь, у нас дома небольшая авария произошла... Но твой компьютер не пострадал, не волнуйся. Но теперь там немного грязно. Ну, то есть очень грязно. Но ты не волнуйся, я приготовила тебе твои болотные сапоги, у входа стоят. А грязь я уберу, как будет свободное время. Когда? Ну, наверное, когда в отпуск пойду. А, ну когда вернёмся из Турции. А, ну значит в следующий отпуск, но обязательно уберу. А пока я у мамы поживу. И ты, кстати, тоже можешь. Ну, как хочешь, я же не заставляю... Только пока я не убрала, ты там грязь не разводи, сильно сапогами по грязи не шлёпай и когда по чистому ходишь, сапоги снимай и тапочки обувай, я их тоже возле входа поставила, ты их бери с собой, когда идёшь по грязи и переобувай. А когда по чистому идёшь, бери сапоги, там грязь в разных местах. Программисты, как известно, не самые трудолюбивые люди, поэтому убирать грязь не станут. Но переобувать болотные сапоги каждый раз, когда переходишь от грязного пола к чистому и наоборот — удовольствие ниже среднего, уж лучше пройти лишние несколько метров. Чтобы прожить время до следующего отпуска с комфортом, надо срочно выработать способ добираться из одной точки квартиры с минимальным количеством переобуваний по пути, ну а уж среди них, конечно, выбрать самый короткий.

Формат входных данных

В первой строке даны два целых числа M и N — размеры квартиры (в у.е.). $1 \leqslant N, M \leqslant 500$. Два целых числа во второй строке — координаты компьютера (в у.е.), а два целых числа в третьей строке — координаты холодильника (тоже в у.е.). Далее идут M строк по N символов в каждой — план квартиры. На плане 1 означает чистое место, 2 — грязное, 0 — стена или непроходимая грязь. Переходить можно только на клетки, имеющие общую вершину с данной, при переходе с чистой на грязную и наоборот надо переобуваться. Холодильник и компьютер находятся не в клетках, помеченных нулём. Левая верхняя клетка плана имеет координаты $(1, 1)$.

Формат выходных данных

Длина кратчайшего пути (количество преодоленных квадратиков, включая начальный и конечный) с минимальным количеством переобуваний, и, через пробел, количество переобуваний (переобувание проходит при переходе с грязного на чистое и наоборот). Если пройти к холодильнику невозможно, вывести числа 0 0.

Пример

dirt.in	dirt.out
3 7 1 1 3 7 1200121 1212020 1112021	8 4

9 Задача 8I. Кратчайший путь [0.5 sec (1 sec)]

Надеюсь, все вы умеете искать в ориентированном графе кратчайший путь. В этой задаче вам предлагается свое умение продемонстрировать.

Вам дан ориентированный взвешенный граф. Веса ребер — целые числа от 1000 до 2000. Нужно несколько раз (не более 1000) ответить на следующий запрос: длина кратчайшего пути из некоторой вершины s в некоторую вершину t.

Формат входных данных

На первой строке числа N и M ($1 \leqslant N \leqslant 10^5$, $0 \leqslant M \leqslant 2 \cdot 10^5$) — количество вершин и ребер нашего графа, соответственно. Вершины нумеруются целыми числами от 1 до N. Далее M строк содержат информацию о ребрах графа. Каждое ребро задается тремя числами — номер начала, номер конца и вес. Все веса — целые числа от 1000 до 2000. В графе могут быть и петли, и кратные ребра. Следующая строка содержит число K ($1 \leqslant K \leqslant 10^3$) — количество запросов. В следующих K строках задаются запросы. Каждый запрос описывается двумя числами — из какой вершины, и в какую должен вести путь.

Формат выходных данных

Для каждого запроса выведите на отдельной строке целое число — длину кратчайшего пути. Если кратчайшего пути не существует следует вывести -1.

Пример

shortest.in	shortest.out
5 5 1 2 2000 1 3 1000 1 4 1200 2 3 1500 3 4 1500 4 1 5 2 4 3 3 1 2	-1 3000 0 2000

Замечание

Путем в графе называется такая последовательность ребер, что конец i-го совпадает с началом $i+1$-го. Длиной пути называется суммарный вес ребер. Путь является кратчайшим, если его длина минимальна.