Trust me, I'm the Doctor.

Доктор Кто

Алгоритмы и структуры данных–2

2024–2025 учебный год

SET 6. Домашняя работа

Графы–1. Анализ связности, минимальный остов и кратчайшие пути

февраль

					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28		
пн	вт	ср	чт	пт	сб	вс

создано командой преданных своему делу энтузиастов

Немного инструкций

Задания в рамках домашней работы SET 6 подразделяются на два блока:

1. *Блок A* «Аналитические задания» — задачи, связанные с анализом и применением различных графовых алгоритмов построения минимальных остовных деревьев и кратчайших путей.

   > Решения заданий Блока A оформляются в письменном виде в любом удобном формате (LaTeX, текстовый документ, скан, изображение и др.) и загружаются для оценки в соответствующую форму раздела `SET 6. Домашняя работа` на странице курса в `LMS` по ссылке: `https://edu.hse.ru/course/view.php?id=224738`.

2. *Блок P* «Задания на разработку» — задачи, связанные с реализацией графовых алгоритмов.

   > Решения заданий Блока P загружаются в систему `CODEFORCES` и проходят автоматизированное тестирование. Для загрузки нужно перейти на `https://dsahse.contest.codeforces.com` и выбрать соответствующее соревнование. Доступ к соревнованию предоставлен по тем же учетным данным, что и к системе Яндекс.Контест.

Домашняя работа SET 6 содержит 2 обязательные задачи в Блоке A и 5 обязательных задачи в Блоке P. Баллы, которые можно получить за решение задач, распределены следующим образом:

Блок A		**Блок P**						
A1	A2	P1	P2	P3	P4	P5	P6b	P7b
12	15	6	7	7	6	6	7	9

Задачи, помеченные “b” не являются обязательными — баллы за их решение относятся к *бонусным*.

Важные даты

1. Домашняя работа SET 4 открыта с **14:30 10 февраля 2025 г**.
2. Прием решений завершается в **02:00 24 февраля 2025 г**.
3. Защита решения бонусной задачи A3b принимается до **7 марта 2025 г**.

Содержание

Успехов!

Задача A1. Как построить минимальный остов?

Ниже приведены три алгоритма ALG_1, ALG_2 и ALG_3, которые из заданного связного неориентированного графа $G = (V, E)$ выбирают некоторое множество его ребер T.

```
ALG_1(G):
  отсортировать ребра грава G
  в порядке невозрастания весов;
  T = E;
  foreach (e ∈ E в порядке невозрастания весов)
    if (ребра T - {e} образуют связный граф)
      T = T - {e};
  return T;
```

```
ALG_2(G):
  T = ∅;
  foreach (e ∈ E, выбранное случайным образом)
    if (ребра T ∪ {e} образуют граф без циклов)
      T = T ∪ {e};
  return T;
```

```
ALG_3(G):
  T = ∅;
  foreach (e ∈ E, выбранное случайным образом)
    T = T ∪ {e};
    if (в T имеется цикл из ребер c ⊆ T)
      e_max = ребро с максимальным весом
              в цикле c;
      T = T - {e_max};
  return T;
```

Система оценки

1. 7 баллов Для каждого из трех представленных алгоритмов обоснуйте его наиболее эффективную *по временной сложности* реализацию, в особенности, с точки зрения используемых структур данных и операций над ними. Обоснуйте оценки сложности. Представьте исходный код на языке `C++` для каждой из соответствующих реализаций, в которых используемые структуры данных достаточно отразить на уровне интерфейса — приводить полный код используемых структур данных *не нужно*.

2. 5 баллов Для каждого из трех представленных алгоритмов определите, формируется ли в множестве ребер T *минимальное остовное дерево* исходного графа G. Обоснуйте свой ответ и приведите (контр)примеры.

Задача A2. Кратчайший блиц!

Ниже приведены четыре вопроса, связанные с практическим анализом различных алгоритмов поиска кратчайших путей в графах. Ответы на эти вопросы должны быть обоснованы и дополнены подтверждающими или опровергающими примерами.

Система оценки

1. 3 балла Предположим, что «длина» пути рассчитывается не как общая сумма весов ребер, а как их произведение. Модифицируйте алгоритм Дейкстры для поиска кратчайших путей по указанному правилу. Для каких графов модифицированный алгоритм `DijkstraMULT(G, start)` будет обеспечивать корректный поиск таких кратчайших путей? Почему?

2. 5 баллов Разработайте алгоритм `RestoreGraph(dist[][])`, который по заданной матрице кратчайших путей `dist` между всеми парами вершин графа $G = (V, E)$ восстанавливает его исходное представление. Например, на выходе этого алгоритма может быть получен список смежности графа G. Вы можете выбрать любое представление для восстанавливаемого графа, за исключением списка ребер. Есть ли случаи, в которых однозначное восстановление графа по матрице `dist` невозможно? Почему?

3. 3 балла В ядре реализации алгоритма Флойда-Уоршелла (для поиска кратчайших путей между всеми парами вершин в графе), представленного ниже, допущена ошибка. Приведите пример графа, для которого кратчайшие пути будут определяться неверно, а также соответствующую частичную трассировку (пошаговое исполнение) алгоритма.

```
for i = 1 to n
    for j = 1 to n
        for k = 1 to n
            dist[i][j] = min(dist[i][j], dist[i][k] + dist[k][j])
```

4. 4 балла Возможно ли определить такой ориентированный взвешенный граф $G = (V, E)$, в котором некоторая дуга (v_i, v_j) лежит как на кратчайшем пути из вершины $a \in V$ в вершину $b \in V$, так и на кратчайшем пути из вершины b в вершину a? Охарактеризуйте структуру такого графа и определите, возникнут ли ограничения применимости известных алгоритмов поиска кратчайших путей в G.

Задача P1. Сортировка графа конденсации

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1 секунда
Ограничение по памяти:	128 мегабайт

Всем известно, что Тардис внутри больше, чем снаружи. Немудрено, что Клара потерялась, пока искала путь до кухни (она хотела приготовить суфле).

Чтобы помочь Кларе ориентироваться в космическом корабле, Доктор решил сделать для девушки карту. Он гений и любит, когда всё красиво и сложно, поэтому решил представить комнаты корабля и коридоры между ними в виде ориентированного графа (иногда Тардис вредничает и пускает по коридорам только в одну сторону). Чтобы сделать красивую карту, необходимо построить конденсацию этого графа. Вы, как близкий друг Доктора, вызвались помочь с этим.

Более конкретно, Вам дан ориентированный граф с n вершинами и m ребрами. Найдите компоненты сильной связности этого графа и топологически отсортируйте его граф конденсации, в котором каждая компонента связности исходного графа «стягивается» в одну вершину.

Формат входных данных

Граф задан во входном файле следующим образом: первая строка содержит числа n ($1 \leqslant n \leqslant 2 \cdot 10^4$) и m $\left(1 \leqslant m \leqslant \min\left(\frac{n \cdot (n-1)}{2}, 2 \cdot 10^5\right)\right)$.

Каждая из следующих m строк содержит описание ребра — два целых числа в диапазоне от 1 до n — номера начала и конца ребра.

Формат выходных данных

На первой строке выведите число k — количество компонент сильной связности в заданном графе. На следующей строке выведите n чисел — для каждой вершины выведите номер компоненты сильной связности, которой принадлежит эта вершина. Компоненты сильной связности должны

быть занумерованы таким образом, чтобы для любого ребра номер компоненты сильной связности его начала не превышал номера компоненты сильной связности его конца.

Система оценки

Подзадача	Баллы	Дополнительные ограничения	Необходимые подзадачи	Информация о проверке
0	–	тест из условия	–	полная
1	0.5	$n \leqslant 10, m \leqslant 20$	0	первая ошибка
2	1	$n \leqslant 100, m \leqslant 2000$	0–1	первая ошибка
3	1	$n \leqslant 1000, m \leqslant 2 \cdot 10^4$	0–2	первая ошибка
4	1	$n \leqslant 4000, m \leqslant 2 \cdot 10^5$	0–3	первая ошибка
5	1	$n \leqslant 10^4, m \leqslant 2 \cdot 10^5$	0–4	первая ошибка
6	1.5	$n \leqslant 2 \cdot 10^4, m \leqslant 2 \cdot 10^5$	0–5	первая ошибка

Пример

стандартный ввод	стандартный вывод
10 19 1 4 7 8 5 10 8 9 9 6 2 6 6 2 3 8 9 2 7 2 9 7 4 5 3 6 7 3 6 7 10 8 10 1 2 9 2 7	2 1 2 2 1 1 2 2 2 2 1

Задача P2. Энергоснабжение планет

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1 секунда
Ограничение по памяти:	128 мегабайт

Доктор снова помогает жителям других планет. В этот раз вызовом стало соединение всех городов планеты Трензалор с бесконечным источником энергии. Для этого достаточно расположить этот источник в любом из городов и соединить некоторые города линиями энергоснабжения так, чтобы каждый город получал энергию напрямую из источника или из другого города, в который уже провели энергию.

Доктору известна стоимость построения линии энергоснабжения между некоторыми парами городов. Он решил выбрать одну из двух наиболее экономичных схем энергоснабжения (стоимость схемы равняется сумме стоимостей соединений пар городов). Напишите программу, которая вычисляет стоимость двух наиболее экономных схем энергоснабжения городов.

Формат входных данных

В первой строке входного файла находятся два натуральных числа, разделенных пробелом: N ($3 \leqslant N \leqslant 100$), количество городов на Трензалоре, и M ($3 \leqslant M \leqslant 2000$) — количество возможных соединений между ними. В каждой из последующих M строк находятся по три числа: A_i, B_i, C_i, разделенных пробелами, где C_i — стоимость прокладки линии энергоснабжения ($1 \leqslant C_i \leqslant 300$) от города A_i до города B_i ($i = 1, 2, ..., N$).

Формат выходных данных

В единственной строке выходного файла должны содержаться два натуральных числа S_1 и S_2, разделенных пробелом – две наименьшие стоимости схем ($S_1 \leqslant S_2$). Отметим, что $S_1 = S_2$ тогда и только тогда, когда существует несколько схем надежного энергоснабжения наименьшей стоимости.

Система оценки

Подзадача	Баллы	Дополнительные ограничения	Необходимые подзадачи	Информация о проверке
0	–	тест из условия	–	полная
1	1	$N \leqslant 30, M \leqslant 50$	0	первая ошибка
2	2	$N \leqslant 70, M \leqslant 300$	0–1	первая ошибка
3	2	$N \leqslant 90, M \leqslant 1000$	0–2	первая ошибка
4	2	$N \leqslant 100, M \leqslant 2000$	0–3	первая ошибка

Пример

стандартный ввод	стандартный вывод
5 8 1 3 75 3 4 51 2 4 19 3 2 95 2 5 42 5 4 31 1 2 9 3 5 66	110 121

Замечание

Гарантируется, что для входных данных существует две различные схемы надёжного энергоснабжения.

Задача P3. Разрезание времени

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1 секунда
Ограничение по памяти:	64 мегабайта

Что случилось?! Тардис взорвалась, и мир буквально разорван в клочья! С каждой секундой связь между разными временами становится всё слабее, и между некоторыми уже нельзя перемещаться!

Чтобы снова всех спасти, Доктору нужно всегда знать, в какие времена он всё ещё может попасть. Помогите ему с этой задачей!

Представим каждый момент времени как вершину в неориентированном графе. Над этим графом в заданном порядке производят операции следующих двух типов:

- `cut` — разрезать связь между двумя временами (т.е., удалить ребро из графа);
- `ask` — проверить, можно ли из одного времени попасть в другое напрямую или за несколько путешествий (т.е., лежат ли две вершины графа в одной компоненте связности).

Известно, что после выполнения всех операций типа `cut` перемещения во времени стали невозможны (т.е., рёбер в графе не осталось).

Найдите результат выполнения каждой из операций типа `ask`.

Формат входных данных

Первая строка входного файла содержит три целых числа, разделённые пробелами — количество вершин графа n ($1 \leqslant n \leqslant 50000$), количество рёбер m ($0 \leqslant m \leqslant 100000$) и количество операций k ($m \leqslant k \leqslant 150000$).

Следующие m строк задают рёбра графа; i-ая из этих строк содержит два числа u_i и v_i ($1 \leqslant u_i, v_i \leqslant n$), разделённые пробелами — номера концов i-го ребра. Вершины нумеруются с единицы, а граф не содержит петель и кратных рёбер.

Далее следуют k строк, описывающих операции:

- Операция `cut` задаётся строкой «`cut` u v» ($1 \leqslant u,\ v \leqslant n$), которая означает, что из графа удаляют ребро между вершинами u и v.
- Операция типа `ask` задаётся строкой «`ask` u v» ($1 \leqslant u, v \leqslant n$), которая означает, что необходимо узнать, лежат ли в данный момент вершины u и v в одной компоненте связности.

Гарантируется, что каждое ребро графа встретится в операциях типа `cut` ровно один раз.

Формат выходных данных

Для каждой операции `ask` во входном файле выведите на отдельной строке слово «YES», если две указанные вершины лежат в одной компоненте связности, и «NO» в противном случае. Порядок ответов должен соответствовать порядку операций ask во входном файле.

Система оценки

Подзадача	Баллы	Дополнительные ограничения	Необходимые подзадачи	Информация о проверке
0	—	тесты из условия	—	полная
1	2	$n \leqslant 1000$	0	первая ошибка
2	5	—	1	первая ошибка

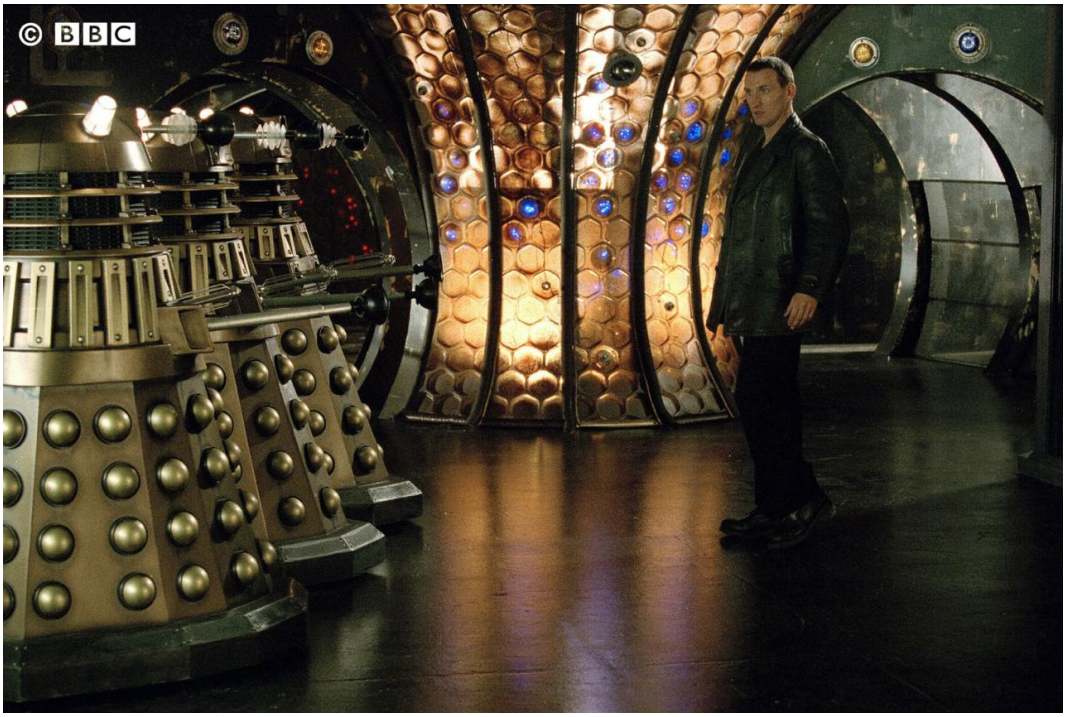
Пример

стандартный ввод	стандартный вывод
3 3 7 1 2 2 3 3 1 ask 3 3 cut 1 2 ask 1 2 cut 1 3 ask 2 1 cut 2 3 ask 3 1	YES YES NO NO

Задача P4. Алгоритм Беллмана-Форда

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	2.5 секунд
Ограничение по памяти:	64 мегабайта

Далеки прорываются на «Спутник 5»! Известно, что они пристыкуются к узлу с номером 0 и попытаются попасть в каждый из n узлов «Спутника». Время, за которое они проходят между узлами, может отличаться — местные жители под командованием Капитана Джека не отдадут космический корабль без боя! И даже может быть отрицательным, если Далеки попадут во временную ловушку Доктора.



Помогите Доктору спланировать оборону и реализуйте **алгоритм Беллмана-Форда** для поиска всех кратчайших путей из узла 0 в остальные в **ориентированном графе**.

Формат входных данных

На вход программы подается информация о слабо-связном ориентированном взвешенном графе в следующем формате.

В первой строке два целых числа n ($2 \leqslant n \leqslant 10^4$) и m $\left(n-1 \leqslant m \leqslant \min\left(n \cdot (n-1),\, 2 \cdot 10^4\right)\right)$ – количество вершин и дуг в графе соответственно. Все n вершин имеют номера, нумерация вершин от 0 до $n-1$.

В следующих m строках содержатся по 3 числа: u v w ($u \neq v$; $0 \leqslant u, v \leqslant n-1$; $-2 \cdot 10^5 \leqslant w \leqslant 10^6$), где u и v – номера вершин, которые связывает дуга (путь из u в v), а w – стоимость (вес) дуги.

Граф может содержать отрицательные циклы. Одну и ту же пару вершин соединяет не более двух дуг различной ориентации (для вершин u и v может быть и дуга (u, v), и дуга (v, u)). Гарантируется, что из вершины 0 существует путь до всех остальных вершин.

Формат выходных данных

В каждой строке требуется, требуется вывести длину пути от вершины под номером 0 до вершины с номером, соответствующим номеру строки (нумерация строк с 1). Если между вершиной 0 и вершиной x есть сколь угодно малый путь, то выведите `-inf`.

Система оценки

Подзадача	Баллы	Дополнительные ограничения	Необходимые подзадачи	Информация о проверке
0	−−	тесты из условия	–	полная
1	1	$n \leqslant 10, m \leqslant 20$	0	первая ошибка
2	1	$n \leqslant 100, m \leqslant 200$	1	первая ошибка
3	2	$n \leqslant 1000, m \leqslant 2000$	1–2	первая ошибка
4	2	$n \leqslant 10^4, m \leqslant 2 \cdot 10^4$	1–3	первая ошибка

Примеры

стандартный ввод	стандартный вывод
5 10 0 1 5 0 3 3 0 4 8 1 2 4 2 1 7 2 3 9 3 1 2 4 0 6 4 2 10 4 3 1	5 9 3 8
4 4 0 1 3 1 2 0 2 3 1 3 2 -2	3 -inf -inf

Задача P5. Алгоритм Флойда-Уоршелла

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

Однажды, гуляя по планете Шан Шен, Доктор с Донной разговаривали о графах и поиске кратчайших путей:

- А этих Уоршелла и Флойда ты тоже знал?
- Эх, Флойд-Флойд, играли как-то с ним в нарды...

В данной задаче предлагается реализовать **алгоритм Флойда-Уоршелла** для поиска кратчайших путей между всеми парами вершин в **ориентированном графе**.

Формат входных данных

На вход программы подается информация о слабо–связном ориентированном взвешенном графе в следующем формате:

В первой строке два целых числа n ($2 \leqslant n \leqslant 500$) и m $\left(n-1 \leqslant m \leqslant \min\left(n \cdot (n-1),\, 2 \cdot 10^5\right)\right)$ – количество вершин и дуг в графе соответственно. Все n вершин имеют номера, нумерация вершин от 0 до $n-1$.

В следующих m строках содержатся по 3 числа: u v w ($u \neq v$; $0 \leqslant u, v \leqslant n-1$; $1 \leqslant w \leqslant 10^9$), где u и v – номера вершин, которые связывает дуга (путь из u в v), а w – стоимость (вес) дуги.

Одну и ту же пару вершин соединяет не более двух дуг различной ориентации (для вершин u и v может быть и дуга (u, v), и дуга (v, u)).

Формат выходных данных

На выход выдаются все найденные кратчайшие пути в следующем формате.

Каждая строка содержит три компоненты: u v w, где

- u — номер вершины, из которой ведёт путь,
- v — номер вершины, в которую ведёт путь и
- w — вес найденного пути.

При выводе путей номера вершин выводятся в лексикографическом порядке, т. е., первым выводится путь из вершины 0 в вершину 1, вторым – из вершины 0 в вершину 2 и т.д. Таким образом, в ответе должно быть $n \cdot (n-1)$ строк.

Если из u не существует пути в некоторую вершину v (из-за ориентации дуг), соответствующая строка имеет формат: «u v -1». Смотрите пример.

Система оценки

Подзадача	Баллы	Дополнительные ограничения	Необходимые подзадачи	Информация о проверке
0	–	тесты из условия	–	полная
1	1	$n \leqslant 20, m \leqslant 100$	0	первая ошибка
2	1	$n \leqslant 100, m \leqslant 10^4$	1	первая ошибка
3	2	$n \leqslant 300, m \leqslant 10^5$	1–2	первая ошибка
4	2	$n \leqslant 500, m \leqslant 2 \cdot 10^5$	1–3	первая ошибка

Примеры

стандартный ввод	стандартный вывод
4 6 0 1 4 0 2 6 0 3 3 1 2 9 3 1 1 2 3 7	0 1 4 0 2 6 0 3 3 1 0 -1 1 2 9 1 3 16 2 0 -1 2 1 8 2 3 7 3 0 -1 3 1 1 3 2 10
5 11 1 0 5 0 2 8 3 0 7 0 4 8 1 2 6 1 3 7 4 1 2 3 2 4 2 4 9 4 3 4 3 4 2	0 1 10 0 2 8 0 3 12 0 4 8 1 0 5 1 2 6 1 3 7 1 4 9 2 0 16 2 1 11 2 3 13 2 4 9 3 0 7 3 1 4 3 2 4 3 4 2 4 0 7 4 1 2 4 2 8 4 3 4

Задача P6b. Сложные путешествия во времени

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1 секунда
Ограничение по памяти:	64 мегабайта

Доктор снова попал в передрягу и оказался без Тардис. Ему срочно нужно отправиться в будущее(или прошлое), чтобы спасти Понд и Рори. К счастью, на планете, где он оказался, уже много веков существуют машины времени общественного пользования. Они работают следующим образом.

Если сейчас год X, то из него можно совершить путешествие в прошлое или будущее на любое положительное число лет Y, десятичная запись которого является подстрокой десятичной записи числа X. Стоимость такого путешествия равна сумме цифр числа Y.

Необходимо за минимальную стоимость попасть из года a в год b, при этом все промежуточные года, которые посетит Доктор, должны быть положительными (нельзя переместиться в год раньше создания первой общественной машины времени) и не должны превышать n (в год $(n+1)$ все такие машины времени запретили).

Формат входных данных

Входной файл содержит три целых числа: n, a, b ($1 \leqslant a, b \leqslant n \leqslant 5000$).

Формат выходных данных

Если из года a нельзя попасть в год b, выведите в выходной файл одно число -1.

Если же такая последовательность путешествий существует, в первой строке выходного файла выведите минимальную сумму денег, которую Доктор потратит на путешествия. Во второй строке выходного файла выведите число k — минимальное количество путешествий, которое Доктор при этом совершит. В последующих k строках выведите сами путешествия по одному в строке. Каждая строка должна иметь вид «+число» или «–число», в зависимости от того, на сколько лет он в очередной раз прыгал в будущее или прошлое.

Система оценки

Подзадача	Баллы	Дополнительные ограничения	Необходимые подзадачи	Информация о проверке
0	−−	тесты из условия	–	полная
1	2	$n \leqslant 1000$	0	первая ошибка
2	2	$n \leqslant 5000$	1	первая ошибка
3	3	$n \leqslant 100000$	0–2	первая ошибка

Примеры

стандартный ввод	стандартный вывод
20 12 18	5 3 -2 +10 -2
100 5 43	29 8 +5 +1 +1 +1 +13 +26 -5 -4

Задача P7b. Вторжение Киберлюдей

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Киберлюди снова вторгаются на Землю. Доктор, изучив их броню, выяснил удивительный факт: если подать на некоторые транзисторы их центрального процессора электрический ток, а на некоторых перекрыть его, костюм может закоротить!

Чтобы правильно рассчитать, на какие транзисторы какой ток подавать, Доктор просит Вас — верного друга и соратника —решить задачу 2-SAT. Нужно подобрать значения n булевых переменных так, чтобы все m утверждений вида $(x_{i_1} = e_1) \vee (x_{i_2} = e_2)$ обратились в истину. В данной задаче вам гарантируется, что решение существует.

Формат входных данных

Входной файл состоит из одного или нескольких тестов.

Каждый тест описывается следующим образом. На первой строке число переменных n и число утверждений m. Каждая из следующих m строк содержит числа i_1, e_1, i_2, e_2, задает утверждение $(x_{i_1} = e_1) \vee (x_{i_2} = e_2)$ $(0 \leqslant i_j < n,\ 0 \leqslant e_j \leqslant 1)$. Ограничения: сумма всех n не больше 100 000, сумма всех m не больше 300 000.

Формат выходных данных

Для каждого теста выведите строку из n нулей и единиц — значения переменных. Если у данной задачи `2-SAT` есть несколько решений, выведите любое.

Система оценки

Подзадача	Баллы	Дополнительные ограничения	Необходимые подзадачи	Информация о проверке
0	–	тесты из условия	–	полная
1	1	$n \leqslant 10,\ m \leqslant 20$ $\sum n_k \leqslant 100,\ \sum m_k \leqslant 300$	0	первая ошибка
2	1	$n \leqslant 10, m \leqslant 150000$	0–1	первая ошибка
3	2	$n \leqslant 50000, m \leqslant 20$	0–1	первая ошибка
4	2	$n \leqslant 100, m \leqslant 1000$	0–3	первая ошибка
5	3	ограничения из условия	0–4	первая ошибка

Пример

стандартный ввод	стандартный вывод
1 0 2 2 0 0 1 0 0 1 1 1 3 4 0 1 1 0 0 0 2 1 1 1 2 0 0 0 0 1	0 01 000