Содержание

В некоторых задачах большой ввод и вывод. Имеет смысл пользоваться супер быстрым вводом-выводом: `http://acm.math.spbu.ru/~sk1/algo/input-output/`

Задачи

Задача A. Ядра [0.3 sec, 256 mb]

Капитан Вася всегда держит на своем корабле запас пушечных ядер для борьбы с пиратами. Так как он привык во всем поддерживать порядок, он хранит ядра в виде пирамид. Каждый из слоев одной пирамиды является равносторонним заполненным ядрами треугольником, сторона которого содержит ровно k ядер. Сторона основания пирамиды состоит из n ядер, в следующем слое сторона состоит из $n-1$ ядра, и т.д., пока на вершину не будет положено одно ядро (которое является равносторонним треугольником со стороной 1).

Например, пирамида размера 3 состоит из трех уровней, выглядящих так (сверху вниз):

```
  X

  X
 X X

  X
 X X
X X X
```

Ясно, что каждый из треугольников может содержать только 1, 3, 6, 10 и т.д. ядер. Таким образом, пирамида может содержать только 1, 4, 10, 20, и т.д. ядер.

Вася отправляется в плавание и берет с собой ровно m ядер. Какое минимальное число пирамид требуется ему сложить из них на своем корабле?

Формат входных данных

В первой строке входного файла записано количество тестов $1 \leqslant T \leqslant 20$. В последующих T строках задается количество ядер в i-м тесте $1 \leqslant m_i \leqslant 300\,000$.

Формат выходных данных

Для каждого из T тестов входного файла выведите в отдельной строке минимальное количество пирамид.

Пример

balls.in	balls.out
5 1 5 9 15 91	1 2 3 3 2

Задача B. Расстояние от корня [0.3 sec, 256 mb]

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 ($1 \leqslant n \leqslant 100$). В следующих $n-1$ строках заданы вершины, являющиеся предками вершин 2, 3, ..., 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

rootdist.in	rootdist.out
3 1 1	1 2 2 3
3 1 2	2 1 3

Задача C. Любители Кошек [0.15 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

catlover.in	catlover.out
3 3 1 2 2 3 3 1	1

Задача D. Avia. Авиаперелеты [1.5 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

avia.in	avia.out
4 0 10 12 16 11 0 8 9 10 13 0 22 13 10 17 0	10

Задача E. Кодовый замок [2 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

codelock.in	codelock.out
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 F. Дорожные работы [2.5 sec, 256 mb]

В республике Икс издавна действует двухпартийная система. Каждый год граждане, имеющие избирательные права, голосуют, какой партии они больше доверяют — партии Мошенников или партии Грабителей, и в течение этого года вся реальная власть сосредоточена в руках избранной партии.

В последние M лет между партиями разразилась нешуточная война по перестройке дорожной сети республики «под себя». Партия Мошенников стремится построить как можно больше государственных дорог, чтобы прикарманить побольше бюджетных денег на их «обслуживание», а партия Грабителей стремится сделать платными как можно большее число дорог. Движение на всех дорогах республики Икс двустороннее.

Известно, что в течение одного года правления партии Мошенников удавалось построить ровно одну новую дорогу (которая поначалу является бесплатной), а партии Грабителей — ввести плату за проезд по одной из бесплатных на текущий момент дорог (при этом деньги на содержание этой дороги выделяются уже не из бюджета, а из средств, вырученных за проезд).

Президент республики, в настоящее время не имеющий реального политического влияния, решил привлечь внимание общественности к проблеме дорог. Он назвал дорожную сеть *удобной* (для простых граждан), если из любого города можно доехать до любого, используя только бесплатные дороги, но при этом количество бесплатных дорог (а, соответственно, и бюджетные средства на их содержание, полученные сбором налогов с граждан республики) — минимально возможное.

Вам поручено написать программу, которая определяет, была ли дорожная сеть удобной по завершении i-го года «дорожной войны».

Формат входных данных

В первой строке ввода заданы два числа — N ($1 \leqslant N \leqslant 1000$), число городов в республике Икс, и M ($1 \leqslant M \leqslant 100\,000$), продолжительность порядком затянувшейся «дорожной войны». Далее следуют M строк, первый символ каждой из которых — это `F`, если в данный год у власти была партия Мошенников, и `R` — если партия Грабителей, а далее в строке следуют два числа — номера городов u_i и v_i — пара городов, дорога между которыми стала объектом пристального внимания соответствующей партии (была построена новая дорога, если у власти была партия Мошенников, и одна из существующих дорог была сделана платной, если у власти была партия Грабителей). Вполне возможна ситуация, когда между двумя городами окажется более одной дороги, или будет построена дорога из города в себя — мало ли, что там удумает партия Мошенников.

Гарантируется, что входные данные корректны, то есть, все числа u_i и v_i лежат в пределах от 1 до N, и если известно, что в какой-то год дорога между двумя городами была сделана платной, то это значит, что перед началом года была хотя бы одна бесплатная дорога между этими городами.

Формат выходных данных

Для каждого года выведите в отдельной строке `YES`, если дорожная сеть по завершении соответствующего года была удобной, и `NO` в противном случае.

Пример

roadwork.in	roadwork.out
4 8 F 1 2 F 1 3 R 1 3 F 2 3 F 3 4 F 1 3 R 1 3 F 1 1	NO NO NO NO YES NO YES NO

Задача G. Новая модель телефона [0.5 sec, 256 mb]

Компания Gnusmas разработала новую модель мобильного телефона. Основное достоинство этой модели — ударопрочность: её корпус сделан из особого сплава, и телефон должен выдерживать падение с большой высоты.

Компания Gnusmas арендовала n-этажное здание и наняла экспертов, чтобы те при помощи серии экспериментов выяснили, с какой высоты бросать телефон можно, а с какой — нельзя. Один эксперимент заключается в том, чтобы бросить телефон с какого-то этажа и посмотреть, сломается он от этого или нет. Известно, что любой телефон этой модели ломается, если его сбросить с x-го этажа или выше, где x — некоторое целое число от 1 до n, включительно, и не ломается, если сбросить его с более низкого этажа. Задача экспертов заключается в том, чтобы узнать число x и передать его рекламному отделу компании.

Задача осложняется тем, что экспертам предоставлено всего k образцов новой модели телефона. Каждый телефон можно бросать сколько угодно раз, пока он не сломается; после этого использовать его для экспериментов больше не удастся.

Подумав, эксперты решили действовать так, чтобы минимизировать максимально возможное количество экспериментов, которое может потребоваться произвести. Чему равно это количество?

Формат входных данных

В первой строке входного файла записаны через пробел два целых числа n и k — количество этажей в здании и количество предоставленных телефонов ($1 \leqslant n \leqslant 100\,000$, $0 \leqslant k < n$).

Формат выходных данных

В выходной файл выведите единственное число — минимальное количество экспериментов, которое потребуется совершить, чтобы узнать число x и использовать не более k телефонов. Если решить задачу невозможно, выведите вместо этого -1.

Примеры

newphone.in	newphone.out
4 2	2
4 1	3

В первом примере сначала следует бросить телефон со второго этажа. Если он сломается, то второй бросок следует сделать с первого этажа. В случае поломки станет известно, что $x = 1$. Иначе мы узнаем, что $x = 2$.

Если же при броске со второго этажа телефон не сломался, бросим телефон с третьего этажа. При поломке будет ясно, что $x = 3$. Иначе из условия $1 \leqslant x \leqslant 4$ следует, что $x = 4$.

Всего будет сделано два эксперимента. В них будет использовано не более чем два телефона.

Во втором примере следует сначала бросить единственный данный нам телефон с первого этажа, если он не сломается, то со второго, а если опять не сломается, то с третьего. При первой же поломке мы узнаем точное значение x. Если после трёх бросков телефон так и не сломался, то $x = 4$.

Задача H. Выбор вершин взвешенного дерева [0.5 sec, 256 mb]

Дан граф, являющийся деревом. В вершинах графа написаны целые числа. Множество вершин графа называется *допустимым*, если никакие две вершины этого множества не соединены ребром.

Рассмотрим все допустимые множества вершин графа. Для каждого такого множества вычислим сумму чисел, написанных в его вершинах. Какова максимальная из этих сумм?

Формат входных данных

Граф в этой задаче задан в виде *корневого дерева*. В графе выделена вершина — *корень дерева*. Для каждой вершины i, не являющейся корнем, задан номер вершины-предка p_i в корневом дереве. Дерево, заданное таким образом, состоит из рёбер $i - p_i$ для всех вершин i, кроме корня.

В первой строке входного файла записано целое число n — количество вершин в графе ($1 \leqslant n \leqslant 100$). В следующих n строках задан граф. В i-й из этих строк записаны через пробел два целых числа p_i и q_i; здесь p_i — номер вершины-предка i-ой вершины, а q_i — число, записанное в этой вершине. Для корня дерева $p_i = 0$; для всех остальных вершин $1 \leqslant p_i \leqslant n$. Числа q_i не превосходят по модулю 10 000.

Гарантируется, что заданный во входном файле граф является деревом.

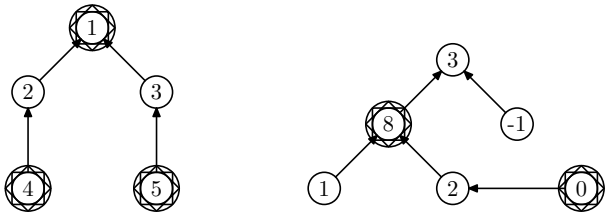
Формат выходных данных

В первой строке выходного файла выведите одно число — максимальную сумму чисел в допустимом множестве.

Примеры

selectw.in	selectw.out
5 0 1 1 2 1 3 2 4 3 5	10
6 5 8 6 0 5 -1 1 1 0 3 1 2	8

На рисунке показаны графы, заданные в примерах. В каждом графе выделено допустимое множество с максимальной суммой чисел в вершинах.



Задача I. Здоровье Графа [1 sec, 256 mb]

Этично ли удалять рёбра у связанного графа?

Граф Безциклов решил проверить своё здоровье. Он хочет проверить, что все его рёбра достаточно крепко держатся в нём. Для этого он хочет посчитать *устойчивость* некоторых из них. *Устойчивостью* ребра называется количество простых путей, проходящих через это ребро.

Формат входных данных

Все числа в файле целые.
$0 \leqslant N \leqslant 10^5$, $0 \leqslant M \leqslant 10^5$ — количество вершин и рёбер.
Затем M пар чисел $1 \leqslant v_i, u_i \leqslant N$ — i-ое ребро соединяет вершины v_i и u_i.
$0 \leqslant Q \leqslant 10^5$ — количество запросов.
Затем Q чисел $1 \leqslant e_i \leqslant M$.
Граф неориентирован. Гарантируется, что Граф ацикличен.

Формат выходных данных

Для i-ого запроса вывести устойчивость e_i-ого ребра.

Примеры

health.in	health.out
2 1 1 2 1 1	1

Задача J. Простые пути в дереве [1.5 sec, 256 mb]

Дан неориентированный связный граф из n вершин и $n-1$ ребра. Требуется для каждого ребра посчитать суммарную длину простых путей, проходящих через это ребро. Длиной пути здесь называется количество ребер в пути.

Формат входных данных

На первой строке целое число n ($2 \leqslant n \leqslant 300\,000$). Следующие $n-1$ строка содержат пары чисел от 1 до n — ребра графа.

Формат выходных данных

$n-1$ строка. i-я строка должна содержать целое число — ответ для i-го ребра.

Система оценки

Подзадача 1 (50 баллов) $n \leqslant 3\,000$.
Подзадача 2 (50 баллов) $n \leqslant 300\,000$.

Пример

treedp.in	treedp.out
5 1 2 2 3 2 4 5 1	13 8 8 9

Бонус

Задача K. Редукция дерева [0.5 sec, 256 mb]

Задано неориентированное дерево, содержащее n вершин. Можно выбрать некоторое ребро и удалить его, при этом инцидентные ему вершины не удаляются. Таким образом можно удалить из дерева некоторый набор рёбер. В результате дерево распадается на некоторое количество меньших деревьев. Требуется, удалив наименьшее количество рёбер, получить в качестве хотя бы одной из компонент связности дерево, содержащее ровно p вершин.

Формат входных данных

Первая строка входного файла содержит пару натуральных чисел n и p ($1 \leqslant p \leqslant n \leqslant 1000$). Далее в $n-1$ строке содержатся описания рёбер дерева. Каждое описание состоит из пары натуральных чисел a_i, b_i ($1 \leqslant a_i, b_i \leqslant n$) — номеров соединяемых ребром вершин.

Формат выходных данных

В первую строку выведите наименьшее количество рёбер q в искомом наборе. Во вторую строку выведите номера удаляемых рёбер. Номера рёбер определяются порядком их задания по входном файле. Рёбра нумеруются с единицы. Если оптимальных решений несколько, разрешается выводить любое.

Система оценки

Подзадача 1 (50 балла) $1 \leqslant p \leqslant n \leqslant 200$.
Подзадача 2 (50 балла) $1 \leqslant p \leqslant n \leqslant 1000$.

Пример

tree.in	tree.out
11 6 1 2 1 3 1 4 2 6 2 7 1 5 2 8 4 9 4 10 4 11	2 3 6

Задача L. Сервера [1.5 sec, 256 mb]

Компьютерная сеть в некотором доме строилась по принципу присоединения нового компьютера к последнему из уже подключенных. Никакие два компьютера, будучи подключенными в сеть, между собой дополнительно никак не связывались. Таким образом, в сеть были объединены последовательно N компьютеров. Соседи обменивались информацией между собой, но в какой-то момент поняли, что им нужны прокси-серверы. Компьютерное сообщество дома решило установить прокси-серверы ровно на компьютеров. Осталось только решить, какие именно компьютеры выбрать для этой цели. Главным критерием является ежемесячная стоимость обслуживания серверами всех компьютеров.

Для каждого компьютера установлен тариф его обслуживания, выраженный в рублях за метр провода. Стоимость обслуживания одного компьютера каким-то сервером равна тарифу компьютера, умноженному на суммарную длину провода от этого компьютера до сервера, которым он обслуживается.

Ваша задача написать программу, которая выберет такие компьютеров, чтобы установить на них прокси-серверы, что общие затраты на обслуживание всех компьютеров были бы минимальными

Формат входных данных

В первой строке входного файла записано два целых числа N и K — количество компьютеров в сети и количество прокси-серверов, которые нужно установить ($1 \le K \le N \le 2000$).

Все компьютеры в сети пронумерованы числами от 1 до N по порядку подключения.

Во второй строке записано одно целое число T_1 — тариф обслуживания первого компьютера.

В следующих $N - 1$ строках записано через пробел по два целых неотрицательных числа L_i, T_i — информация об остальных компьютерах в сети по порядку номеров. L_i — длина провода, соединяющего $i-$ компьютер с соседним с меньшим номером, T_i — тариф обслуживания данного компьютера ($2 \le i \le N$). Все L_i и T_i не превышают 10^6.

Формат выходных данных

В первую строку выходного файла необходимо вывести одно целое число — минимальную стоимость обслуживания всех компьютеров всеми серверами. Во второй строке должны быть записаны через пробел K номеров компьютеров, на которые необходимо установить серверы. При существовании нескольких вариантов размещения разрешается вывести любой.

Пример

server.in	server.out
3 1 10 2 2 3 3	19 1
3 2 10 2 2 3 3	4 1 3

Задача M. Жить или не жить? [0.5 sec, 256 mb]

Гамильтонов путь — это путь в графе, проходящий по каждой вершине ровно один раз. Задача о его нахождении является NP-полной, а значит, современная наука не обладает возможностью эффективно решить её.

В этой задаче не нужно реализовывать алгоритм, который работает на всех возможных графах. Чтобы получить `Accepted`, достаточно верно найти *Гамильтонов путь* на имеющихся у жюри 99 тестах.

Формат входных данных

На первой строке входного файла записаны целые числа n и k — количества вершин и рёбер, соответственно. В следующих k строках записаны номера вершин, соединённых рёбрами. Петли и кратные рёбра отсутствуют, граф неориентирован. Число вершины в графе не больше 40. Думаете это мало? а теперь попробуйте сдать задачу :-)

Гарантируется, что в графе есть гамильтонов путь.

Формат выходных данных

Выведите перестановку из n чисел — номера вершин в порядке гамильтонова пути.

Пример

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

Задача N. Casino. Казино [0.5 sec, 16 mb]

Вновь открытое казино предложило оригинальную игру.

В начале игры крупье выставляет в ряд несколько фишек разных цветов. Кроме того, он объявляет, какие последовательности фишек игрок может забирать себе в процессе игры. Далее игрок забирает себе одну из заранее объявленных последовательностей фишек, расположенных подряд. После этого крупье сдвигает оставшиеся фишки, убирая разрыв. Затем игрок снова забирает себе одну из объявленных последовательностей и так далее. Игра продолжается до тех пор, пока игрок может забирать фишки.

Рассмотрим пример. Пусть на столе выставлен ряд фишек '`rrrgggbbb`', и крупье объявил последовательности '`rg`' и '`gb`'. Игрок, например, может забрать фишки '`rg`', лежащие на третьем и четвёртом местах слева. После этого крупье сдвинет фишки, и на столе получится ряд '`rrggbbb`'. Ещё дважды забрав фишки '`rg`', игрок добьётся того, что на столе останутся фишки '`bbb`' и игра закончится, так как игроку больше нечего забрать со стола. Игрок мог бы действовать и по-другому — на втором и третьем ходах забрать не последовательности '`rg`', а последовательности '`gb`'. Тогда на столе остались бы фишки '`rrb`'. Аналогично, игрок мог бы добиться того, чтобы в конце остались ряды '`rrr`' или '`rbb`'.

После окончания игры полученные фишки игрок меняет на деньги. Цена фишки зависит от её цвета.

Требуется написать программу, определяющую максимальную сумму, которую сможет получить игрок.

Формат входных данных

В первой строке входного файла записано число K ($1 \le K \le 26$) — количество цветов фишек. Каждая из следующих K строк начинается со строчной латинской буквы, обозначающей цвет. Далее в той же строке через пробел следует целое число X_i ($1 \le X_i \le 150$, $i = 1..K$) — цена фишки соответствующего цвета.

В $(K+2)$-ой строке описан ряд фишек, лежащих на столе в начале игры. Ряд задается L строчными латинскими буквами ($1 \le L \le 150$), которые обозначают цвета фишек ряда.

В следующей строке содержится число N ($1 \le N \le 150$) — количество последовательностей, которые были объявлены крупье. В следующих N строках записаны эти последовательности. Гарантируется, что сумма длин этих N строк не превосходит 150 символов, и все они непустые.

Формат выходных данных

В выходной файл выведите единственное целое число — максимальную сумму денег, которую может получить игрок.

Пример

casino.in	casino.out
6 a 1 b 4 d 2 x 3 f 1 e 3 fxeeabadd 2 aba ed	16