Задача A. Велосипедные дорожки

Имя входного файла:	`bike.in`
Имя выходного файла:	`bike.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Андрюша живет в пригороде. Неподалеку от Андрюшиного дома проложены две велосипедные дорожки, каждая из которых имеет вид круга с радиусом r. У дорожек нет общих точек.

Андрюшин дом расположен около одной из дорожек, а его школа расположена около другой дорожки. Каждый день Андрюша ездит в школу и обратно на велосипеде. Он заметил, что когда он едет по дорожке, его скорость составляет u, а когда едет просто по полю, то v, причем $u < v$. Теперь Андрюша хочет узнать, за какое минимальное время он сможет добраться из школы домой.

Введем систему координат таким образом, что центр дорожки, расположенной около Андрюшиного дома, находится в точке $(0, 0)$, а центр дорожки где расположена его школа, находится в точке $(0, d)$. Радиус каждой дорожки равен r. Андрюшин дом расположен в точке (x_1, y_1), а его школа — в точке (x_2, y_2). Скорость Андрюши по дороге равна u, а по полю — v.

Формат входного файла

Входной файл содержит восемь вещественных чисел: d, r, x_1, y_1, x_2, y_2, u и v ($1 \le r \le 100$, $2r < d \le 100$, $1 \le v < u \le 10$, $\sqrt{x_1^2 + y_1^2} = r$, $\sqrt{x_2^2 + (y_2 - d)^2} = r$). Все неравенства выполнены с точностью до 10^{-9}.

Формат выходного файла

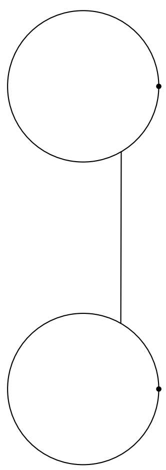
Выведите одно вещественное число — минимальное время, которое Андрюше требуется, чтобы добраться из дома до школы. Выводите ответ с точностью не меньше 10^{-6}.

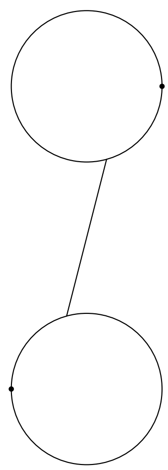
Пример

bike.in	bike.out

`20 5`
`5 0 5 20`
`2 1`
	`16.5757337181`

`20 5`
`-5 0 5 20`
`2 1`
	`17.2040517249`





Задача B. Диверсия

Имя входного файла:	`diversion.in`
Имя выходного файла:	`diversion.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

В королевстве Далеком n городов, соединенных m двусторонними дорогами. Некоторые дороги вымощены камнем, а другие представляют собой лишь обычные проселочные дороги. Столица королевства расположена в городе номер 1. Дороги устроены таким образом, что можно из любого города добраться до любого другого, перемещаясь исключительно по вымощенным камнем дорогам, причем количество каменных дорог — минимальное возможное. Проселочные же дороги были построены таким образом, что если каменная дорога оказывается заблокирована или уничтожена, то все равно можно добраться из любого города до любого другого по дорогам.

Обозначим количество каменных дорог, по которым требуется проехать, чтобы попасть из города u в город v как $s(u, v)$. Дороги подчиняются следующему правилу: если два города u и v соединены дорогой (неважно, каменной или проселочной), то либо $s(1, u) + s(u, v) = s(1, v)$, либо $s(1, v) + s(v, u) = s(1, u)$.

Король соседнего королевства планирует напасть на Далекое. В качестве начала операции предполагается уничтожить некоторые дороги. Расчеты показали, что финансирования, выделенного министерству атаки, достаточно, чтобы уничтожить ровно одну каменную и одну проселочную дорогу. Король хотел бы уничтожить такую пару дорог, чтобы после этого хотя бы для каких-нибудь двух городов стало невозможно добраться из одного города в другой.

Теперь он просит министра атаки посчитать количество возможных диверсионных планов. Однако министр атаки обучен только атаковать, считать для него слишком сложно. Помогите ему!

Формат входного файла

Первая строка входного файла содержит n и m — количество городов и дорог, соответственно ($3 \le n \le 20\,000$, $m \le 100\,000$). Следующие m строк описывают дороги, каждая строка содержит три целых числа — номера городов, соединенных соответствующей дорогой, и 1, если соответствующая дорога вымощена камнем, или 0, если соответствующая дорога — проселочная. Никакие два города не соединены более чем одной дорогой, никакая дорога не соединяет город сам с собой.

Формат выходного файла

Выведите одно целое число — количество способов организовать диверсию.

Пример

diversion.in	diversion.out
6 7 1 2 1 2 3 1 1 4 0 3 4 1 4 5 1 3 6 0 5 6 1	4

Задача C. Голова на плечах

Имя входного файла:	`headand.in`
Имя выходного файла:	`headand.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Изготовитель всемирно известного шампуня, компания «Голова на плечах» всерьез заботится о качестве своей продукции. В частности, она постоянно улучшает различные показатели своего шампуня, с целью чего постоянно проводит исследования.

Но, согласно законодательству, на человеке проводить исследования нельзя, а проводить исследования на животных компания считает ниже своего достоинства.

Поэтому учёные, работающие в компании, разработали математическую модель человека и проводят исследования на ней.

Человек, согласно этой модели, состоит из головы и плеч. Голова представляет собой окружность с центром в точке $(0;0)$ и радиусом R, а плечи — бесконечную прямую $y = -K$, где $R < K$.

Объектом изучения исследователей являются волосы. Каждый волос в данной модели представлен отрезком, начинающимся на голове (строго на окружности) и заканчивающимся на плечах (строго на прямой). При этом ни один волос не имеет с окружностью головы более одной общей точки.

В данный момент учёные озабочены проблемой *секущихся* волос. Пара волос называется *секущейся*, если соответствующие этим волосам отрезки имеют общую точку.

Дана математическая модель человека. Найдите количество секущихся пар волос.

Формат входного файла

В первой строке даны два целых числа R, K ($1 \le R < K \le 1000$).

Во второй строке дано целое число N ($0 \le N \le 100000$) — количество волос в модели человека.

В следующих N строках находится по 4 вещественных числа X_h, Y_h, X_s, Y_s — координаты начала и конца очередного волоса. Первая пара чисел соответствует концу, лежащему на окружности головы, вторая пара соответствует концу, лежащему на плечах.

Гарантируется, что никакой волос не имеет с окружностью головы более одной общей точки. Также гарантируется, что среди начальных и конечных точек нет одинаковых.

Формат выходного файла

В выходной файл выведите число секущихся пар волос.

Примеры

headand.in	headand.out
1 2 3 0 -1 -2 -2 1 0 2 -2 -1 0 -1 -2	1
1 10 3 -1 0 -1 -10 0 -1 0 -10 1 0 1 -10	0

Задача D. Упаковка чисел

Имя входного файла:	`packing.in`
Имя выходного файла:	`packing.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

При передаче данных по сети важно их эффективно кодировать, чтобы лучше использовать пропускную способность канала. Мы опишем процедуру декодирования чисел при одном эффективном способе их кодирования. Этот способ использует переменную длину кодов, для обеспечения единственности декодирования ни один код числа не является префиксом кода другого числа. Вы же должны реализовать процедуру оптимального кодирования.

Метод применим для кодирования целых чисел от -2^{63} до $2^{63}-1$ (тип `long` в Java, `int64` в Дельфи, `__int64` в C++). Упакованное число занимает от 1 до 9 байт, в зависимости от своего значения.

Декодирование происходит следующим образом. Сначала рассматривается первый байт числа. Просматривая его биты от старшего к младшему, найдем первый ноль. Пусть q — количество просмотренных при этом единиц (если первый бит равен `0xff`, то $q = 8$). Число q означает, сколько следующих байтов относятся к декодируемому числу. Если $q = 8$, то эти байты — стандартная восьмибайтная запись числа, причем старшие байты идут сначала.

В противном случае выполним следующее. Если $q + 2$-й бит первого байта равен нулю (биты нумеруются с единицы, сначала идет старший бит, если $q = 7$, то рассматривается старший бит второго байта) то старшие единицы первого байта заменяются на нули, и число дополняется нулевыми байтами до размера 8 байтов.

Иначе (если соответствующий бит равен единице) $q + 1$-й бит (который всегда равен нулю) заменяется на единицу, число же дополняется слева байтами `0xff` до восьми байтов.

В обоих случаях получается стандартная восьмибайтная запись числа, причем старшие байты идут сначала.

Вам заданы несколько целых чисел, закодируйте каждое из них таким образом, чтобы длина закодированного числа была минимальной возможной и оно декодировалось в исходное число.

Формат входного файла

Первая строка входного файла содержит n — количество тестовых примеров ($1 \le n \le 10\,000$). Следующие n строк содержит по одному числу в интервале между -2^{63} и $2^{63}-1$, включительно.

Формат выходного файла

Выведите n строк, каждая строка должна содержать закодированную версию соответствующего числа, записанную как последовательность шестнадцатеричных цифр, старшие байты должны идти сначала. Используйте строчные буквы латинского алфавита.

Пример

packing.in	packing.out
9 0 1 2 3 100 500 1000000 -1 -100	00 01 02 03 8064 81f4 cf4240 7f bf9c

Задача E. Восстановление перестановки

Имя входного файла:	`permutation.in`
Имя выходного файла:	`permutation.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Знаменитая гипотеза Улама утверждает, что если рассмотреть граф G и построить мультимножество G_{min} графов, получаемых из G удалением по очереди каждой из его вершин вместе со всеми инцидентными ей ребрами, то граф G можно однозначно восстановить по G_{min}. Гипотеза Улама до сих пор не доказана, не известно и контрпримера.

В этой задаче мы рассмотрим аналогичную задачу для перестановок. Рассмотрим перестановку $a = \langle a_1, a_2, \ldots a_n \rangle$ чисел от 1 до n. Обозначим как a/i перестановку $n-1$ чисел, полученных из a удалением числа i и уменьшением на единицу всех чисел, бóльших i.

Например, если $a = \langle 1, 3, 5, 2, 6, 4 \rangle$, то $a/2 = \langle 1, 2, 4, 5, 3 \rangle$.

Вам заданы в некотором порядке перестановки $a/1, a/2, \ldots, a/n$. Требуется восстановить исходную перестановку a.

Формат входного файла

Первая строка входного файла содержит n — размер исходной перестановки ($5 \le n \le 300$). Следующие n строк содержат по $n-1$ числу каждая и задают a/i для всех i в некотором порядке.

Формат выходного файла

Выведите n целых чисел — перестановку a. Гарантируется, что такая перестановка существует.

Пример

permutation.in	permutation.out
6 1 3 5 2 4 1 3 4 2 5 1 2 4 5 3 2 4 1 5 3 1 4 2 5 3 1 3 2 5 4	1 3 5 2 6 4

В приведенном примере перестановки заданы в следующем порядке: $a/6$, $a/4$, $a/2$, $a/1$, $a/3$ и $a/5$.

Задача F. Прямоугольный многоугольник

Имя входного файла:	`polygon.in`
Имя выходного файла:	`polygon.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Прямоугольный многоугольник — это многоугольник, стороны которого параллельны осям координат. Многоугольник должен иметь связную границу без самопересечений и самокасаний. Никакие две последовательные стороны не должны быть параллельны.

У Пети есть несколько палочек различной длины. Он хотел бы составить из них прямоугольный многоугольник. Он планирует использовать палочки в качестве горизонтальных сторон многоугольника, а вертикальные он собирается нарисовать.

Теперь Петю интересует вопрос — а какое максимальное количество палочек он сможет использовать.

Формат входного файла

Первая строка входного файла содержит n — количество палочек ($1 \le n \le 100$). Вторая строка содержит n целых чисел — длины палочек. Длина каждой палочки не превышает 200.

Формат выходного файла

На первой строке выходного файла выведите число l — максимальное количество палочек, которое Петя сможет использовать. Следующие $2l$ строк должны содержать координаты вершин многоугольника в порядке обхода. Первые две вершины должны быть концами горизонтального ребра. Если есть несколько решений, выведите любое. Координаты вершин не превышают 10^9.

Если составить многоугольник невозможно, выведите $l = 0$.

Пример

polygon.in	polygon.out
4 1 2 3 5	3 0 0 1 0 1 1 3 1 3 2 0 2
4 1 2 4 8	0
4 1 1 1 1	4 0 0 1 0 1 1 2 1 2 -2 1 -2 1 -1 0 -1

Задача G. Забавная последовательность

Имя входного файла:	`sequence.in`
Имя выходного файла:	`sequence.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Определим последовательность a_i следующим образом: $a_1 = 1$, $a_n = a_{n-1} + 3$, если число n уже встречалось в последовательности a (то есть $\exists j : a_j = n$), и $a_n = a_{n-1} + 2$, иначе.

Нетрудно видеть, что первые 8 членов этой последовательности таковы: $1, 3, 6, 8, 10, 13, 15, 18$.

Ваша задача — вычислить a_n.

Формат входного файла

Первая строка входного файла содержит целое число n ($1 \le n \le 10^5$).

Формат выходного файла

В выходной файл выведите ответ на задачу.

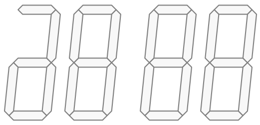
Примеры

sequence.in	sequence.out
1	1
8	18

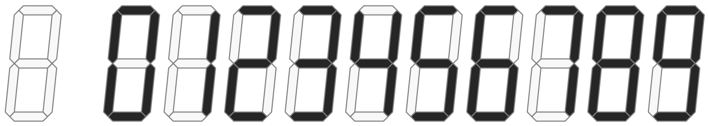
Задача H. Наручные часы

Имя входного файла:	`watch.in`
Имя выходного файла:	`watch.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

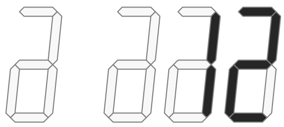
Вы приобрели новые электронные наручные часы с жидкокристаллическим дисплеем. Дисплей отображает часы и минуты с помощью четырех элементов, каждый из которых отображает одну цифру.



Три из них состоят из семи полосок, каждая из которых может быть либо белой (неотличимой от фона), либо черной. Вид такого элемента и отображаемые им цифры показаны на рисунке:



Четвертый элемент предназначен для отображения старшей цифры ча́са. Если она равна нулю, то элемент полностью неактивен (все полоски белые), иначе показывается соответствующая цифра. Вот как выглядит элемент и отображаемые им цифры:



Вам хочется проверить, все ли в порядке с новым приобретением, а именно, нет ли таких полосок в каком-либо из элементов, которые либо всегда белые, либо всегда черные.

Вы хотите начать проверку в некоторое начальное время. Напишите программу, которая определяет, сколько времени такая проверка должна продолжаться, чтобы полностью убедиться в исправности Ваших часов.

Формат входного файла

В первой строке входного файла находится время начала проверки в формате `HH:MM`, то есть сначала записан час, затем идет двоеточие, а после него минута. И часы, и минуты записаны с лидирующими нулями, если таковые имеются. $00 \le HH \le 23$, $00 \le MM \le 59$.

Формат выходного файла

Выведите число минут, необходимых для проверки Ваших часов, если она началась в заданное время.

Пример

watch.in	watch.out
00:00	1200
02:39	1041

Задача I. Свадьба

Имя входного файла:	`wedding.in`
Имя выходного файла:	`wedding.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Одна предприимчивая и очень симпатичная дамочка с прелестнейшем именем Горгона решила заработать себе денег на роскошную жизнь. N молодых людей так влюблены в нее, что предложили руку и сердце. К несчастью для них, Горгона видит в них только мешок с деньгами. Она планирует выйти замуж и почти сразу же развестись с некоторыми из молодых людей ради денежной выгоды. Все, что ей нужно, это подзаработать как можно больше денег (и уж, конечно, остаться незамужней). По законам этой прекрасной страны при разводе каждый из супругов получает половину всего имущества.

Вы планируете опубликовать статью, в которой опишете всю подлость и меркантильность этой особы. Для того чтобы статья получилась особенно красочной, нужно указать максимальную сумму денег, которую сможет получить Горгона.

Формат входного файла

В первой строке записано целое число N — количество молодых людей, без памяти влюбленных в Горгону ($1 \leq N \leq 40$). Далее следует N чисел — сумма денег на счету каждого молодого человека. В последней строке записано целое число A — сумма денег на счету Горгоны. Суммы денег на счету — целые неотрицательные числа, не превосходящие 10^9.

Формат выходного файла

В выходной файл выведите единственное число — максимальную сумму денег, которой сможет обладать Горгона после своей махинации. Ответ выводите с точностью до шести знаков.

Примеры

wedding.in	wedding.out
2 5 10 5	7.500000
3 1 3 2 0	2.125000