Задача A. Выражения

Имя входного файла:	`expr.in`
Имя выходного файла:	`expr.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мебибайт

Алиса и Боб играют в интересную игру.

Они выбрали два числа a и b. И теперь хотят написать два корректных арифметических выражения A и B, состоящих из целых неотрицательных чисел и символов `(`, `)`, `*`, `+`. Причём таких, что значение выражения A равно a, а значение выражения B равно b. Кроме того, Алиса и Боб хотят выбрать такие выражения A и B, что их *различие* будет минимальным. Помогите им!

Различием выражений назовём минимальное количество операций вставки (вставка любого символа в любую позицию в выражении), удалений (удаление любого символа из выражения) и замен (замена любого символа в выражении на любой другой), которое нужно сделать, чтобы из одного выражения получить другое.

Формат входных данных

Единственная строка содержит два числа, записанных через пробел — a, b ($1 \leqslant a, b \leqslant 300$).

Формат выходных данных

В первой строке выведите выражение A, во второй строке выведите выражение B.

Выражения должны иметь длину не больше 500 и быть корректными арифметическими выражениями из целых неотрицательных чисел и символов `(`, `)`, `*`, `+`. В процессе вычисления выражений не должны появляться значения большие 10000.

Если удовлетворяющих ограничениям ответов с минимальным *различием* несколько, выведите любой.

Пример

expr.in	expr.out
11 30	5+6 5*6
248 28	248 28

Задача B. Сон о сыре

Имя входного файла:	`floating-cheese.in`
Имя выходного файла:	`floating-cheese.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мебибайт

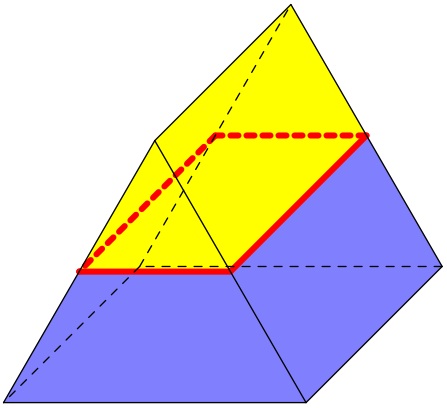
Вам приснилось, что в воду опустили кусок сыра. Кто и зачем это сделал, не важно.

Важно то, что сыр имеет плотность ρ кг/м^3, а сам кусок представляет собой прямую правильную треугольную призму (высота призмы перпендикулярна основаниям, которые имеют форму правильного треугольника) с длиной ребра основания d метров и высотой в d метров (см. рисунок).

Сыр либо тонет (и тогда полностью скрывается под водой), либо плавает, причём одна из его квадратных граней остаётся параллельной плоскости воды, а сама призма находятся выше этой грани.

Плотность воды во сне по-прежнему 1000 кг/м^3, плотность воздуха внезапно оказалась равна нулю, а плотность сыра не может быть равна плотности воды.

Рассмотрим слой сыра, получающийся при сечении сыра плоскостью воды. Сыр интересуется (всякое во сне бывает...): а каков периметр этого слоя?



Формат входных данных

В одном входном файле может описываться сразу несколько ситуаций.

Первая строка входного файла содержит число t — количество описанных ситуаций ($1 \leqslant t \leqslant 10^4$).

Каждая ситуация задаётся на отдельной строке двумя целыми числами: длиной ребер куска сыра d ($1 \leqslant d \leqslant 10^5$) (в метрах) и плотностью сыра ρ ($1 \leqslant \rho \leqslant 1001, \rho \neq 1000$) (в килограммах на кубический метр).

Формат выходных данных

Для каждой ситуации выведите единственное число — искомый периметр в метрах с абсолютной или относительной погрешностью не хуже 10^{-9}.

Примеры

floating-cheese.in	floating-cheese.out
2 1 500 2 1001	3.414213562 0

Задача C. Перестановки

Имя входного файла:	`permutation.in`
Имя выходного файла:	`permutation.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мебибайт

Вам дана таблица a размера $2 \times n$, заполненная целыми числами. В каждой строке все числа различны. Кроме того, $1 \leqslant a_{r,i} \leqslant n$ для любых $1 \leqslant r \leqslant 2$, $1 \leqslant i \leqslant n$. Инверсией в таблице назовём тройку чисел r, i, j, такую, что $i < j$ и $a_{r,i} > a_{r,j}$.

Вы можете некоторым образом переставить столбцы таблицы, то есть, применить к ним некоторую перестановку b. Перестановкой назовем массив из n различных целых чисел, $1 \leqslant b_i \leqslant n$ для $1 \leqslant i \leqslant n$. Тогда после перестановки столбцов по b получится некоторая таблица a', в которой $a'_{r,i} = a_{r,b_i}$. Вам требуется найти такую перестановку b, что количество инверсий в таблице a' минимально.

Формат входных данных

В первой строке записано целое число n ($1 \leqslant n \leqslant 10^5$) — количество столбцов таблицы. В следующих двух строках через пробел записано по n целых чисел. i-е число r-й строки соответствует значению $a_{r,i}$.

Формат выходных данных

Выведите n целых чисел $b_1, b_2, \ldots, b_n$ ($1 \leqslant b_i \leqslant n$ для $1 \leqslant i \leqslant n$) — перестановку, минимизирующую число инверсий в итоговой таблице. Если таких перестановок несколько — выведите любую.

Пример

permutation.in	permutation.out
2 2 1 2 1	2 1

Задача D. Последовательности

Имя входного файла:	`sequence.in`
Имя выходного файла:	`sequence.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мебибайт

Вам дана последовательность a, состоящая из n элементов. Найдите количество последовательностей x_n, таких, что $0 \leqslant x_i \leqslant a_i$ для всех $1 \leqslant i \leqslant n$ и $x_i \leqslant x_{i+1}$ для всех $1 \leqslant i \leqslant n-1$. Так как это количество может быть очень большим, выведите его остаток от деления на $10^9 + 7$.

Формат входных данных

В первой строке записано целое число n ($1 \leqslant n \leqslant 300$) — количество элементов в a. В следующей строке через пробел записаны n чисел $a_1, a_2, \ldots, a_n$ ($1 \leqslant a_i \leqslant 10^8$).

Формат выходных данных

Выведите единственное число — количество искомых последовательностей по модулю $10^9 + 7$.

Пример

sequence.in	sequence.out
3 1 1 2	7

Задача E. Квадраты

Имя входного файла:	`square.in`
Имя выходного файла:	`square.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мебибайт

Сегодня Артур и Мерлин узнали, что такое *факториал*. Назовём числом *n-факториал* произведение всех чисел от 1 до n включительно.

Артур и Мерлин играют в «Квадраты». Суть этой игры в следующем: вначале Артур и Мерлин фиксируют число n. Потом Артур выбирает натуральное число k ($k \geqslant 1$). Далее Мерлин выписывает какие-то делители числа *n-факториал* (не обязательно различные). Артур пытается стереть не более k чисел так, чтобы произведение всех оставшихся чисел было полным квадратом (если чисел после стирания не осталось, будем считать, что произведение равно 1). Если он сможет это сделать, то выиграет, иначе выиграет Мерлин.

Артур хочет узнать, при каком минимальном k он сможет выиграть при любой игре Мерлина.

Формат входных данных

В первой строке дано число t ($1 \leqslant t \leqslant 300$) — количество тестов в файле. Далее следует t строк, описывающих тест. Каждый тест состоит из одного числа n ($1 \leqslant n \leqslant 300$).

Формат выходных данных

Выведите t строк: в i-й строке ответ на i-й тест (искомое число k).

Пример

square.in	square.out
3 2 3 4	1 2 2

Задача F. Всем чмоки в этом чатике!

Имя входного файла:	`chat.in`
Имя выходного файла:	`chat.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мебибайт

Сегодня Мэри, как программисту социальной сети «Телеграфчик», предстоит реализовать сложную систему управления чатами.

Задача Мэри усложняется тем, что в социальную сеть «Телеграфчик» внедрена продвинутая система шифрования «ZergRus», простая, как всё гениальное. Суть её в том, что в системе хранится одна переменная zerg, которая принимает значения от 0 (включительно) до $p = 10^6 + 3$ (исключая p) и меняется в зависимости от событий в системе.

В социальной сети всего n пользователей ($1 \leqslant n \leqslant 10^5$). В начале дня каждый пользователь оказывается в своём собственном чате, в котором больше никого нет. Переменная zerg в начале дня устанавливается равной 0.

В течение дня происходят события типов:

1. Участник с номером $(i + zerg) \bmod n$ посылает сообщение всем участникам, сидящим с ним в чате (в том числе и себе самому), при этом переменная zerg заменяется на $(30 \cdot zerg + 239) \bmod p$.

2. Происходит слияние чатов, в которых сидят участники с номерами $(i + zerg) \bmod n$ и $(j + zerg) \bmod n$. Если участники и так сидели в одном чате, то ничего не происходит. Если в разных, то чаты объединяются, а переменной zerg присваивается значение $(13 \cdot zerg + 11) \bmod p$.

3. Участник с номером $(i + zerg) \bmod n$ хочет узнать, сколько сообщений он не прочитал, и прочитать их. Если участник прочитал q новых сообщений, то переменной zerg присваивается значение $(100\,500 \cdot zerg + q) \bmod p$.

Вы поможете Мэри реализовать систему, обрабатывающую эти события?

Формат входных данных

В первой строке входного файла записаны натуральные числа n ($1 \leqslant n \leqslant 10^5$) — число пользователей социальной сети. и m ($1 \leqslant m \leqslant 3 \cdot 10^5$) — число событий, произошедших за день. В следующих m строках содержится описание событий. Первое целое число в строке означает тип события t ($1 \leqslant t \leqslant 3$). Если $t = 1$, далее следует число i ($0 \leqslant i < n$), по которому можно вычислить, какой участник послал сообщение. Если $t = 2$, далее следуют числа i и j ($0 \leqslant i, j < n$), по которым можно вычислить номера участников, чаты с которыми должны объединиться. Если $t = 3$, далее следует число i ($0 \leqslant i < n$), по которому можно вычислить номер участника, желающего узнать, сколько у него сообщений, и прочитать их.

Формат выходных данных

Для каждого события типа 3 нужно вывести число непрочитанных сообщений у участника.

Пример

chat.in	chat.out	Пояснение
4 10 1 0 1 2 1 1 1 2 3 1 2 1 2 1 3 3 3 2 3 2 3 2	1 1 2	4 10 1 0 1 1 1 2 1 3 3 0 2 0 1 1 1 3 0 2 2 1 3 1

Задача G. Суммы в прямоугольниках

Имя входного файла:	`subsums.in`
Имя выходного файла:	`subsums.out`
Ограничение по времени:	3 секунды
Ограничение по памяти:	256 мебибайт

У Джона Доу есть массив a, состоящий из n целых чисел. Таблица b размера $n \times n$ получается по следующему правилу: $b_{0,i} = a_i$ для $0 \leqslant i < n$. $b_{ij} = b_{i-1,j-1}$ для $0 < i < n$; $0 < j < n$. $b_{i,0} = b_{i-1,n-1}$.

Джона интересуют суммы элементов в прямоугольниках этой таблицы, то есть суммы вида $\sum_{i=x_1}^{x_2} \sum_{j=y_1}^{y_2} b_{ij}$ — сумма всех таких b_{ij}, что $x_1 \leqslant i \leqslant x_2$ и $y_1 \leqslant j \leqslant y_2$ для заданных четырёх чисел x_1, y_1, x_2, y_2.

Формат входных данных

В первой строке через пробел записаны два целых числа n и q ($1 \leqslant n, q \leqslant 5 \cdot 10^5$) — размер массива и количество запросов. Во второй строке находится n целых чисел, разделённых пробелами, — массив a ($|a_i| \leqslant 10^6$).

В следующих q строках записано через пробел по четыре целых числа: x_1, y_1, x_2 и y_2 ($0 \leqslant x_1 \leqslant x_2 < n$, $0 \leqslant y_1 \leqslant y_2 < n$). Ответом на запрос будет являться число $\sum_{i=x_1}^{x_2} \sum_{j=y_1}^{y_2} b_{ij}$ — сумма всех таких b_{ij}, что $x_1 \leqslant i \leqslant x_2$ и $y_1 \leqslant j \leqslant y_2$.

Формат выходных данных

Для каждого запроса в отдельной строке выведите ответ на него. Ответы на запросы выводите в том же порядке, в каком заданы запросы.

Пример

subsums.in	subsums.out
6 5 0 1 2 3 4 5 0 0 5 5 0 0 5 0 1 1 2 2 2 2 3 4 2 2 4 3	90 15 6 15 9

Задача H. Адские круги

Имя входного файла:	`circles.in`
Имя выходного файла:	`circles.out`
Ограничение по времени:	3 секунды
Ограничение по памяти:	256 мебибайт

I'm on the highway to hell...

AC/DC

Когда-то давно у Дьявола в Аду был $k+1$ круг. На каждом из них томились грешники, причём некоторые в очень удобных котлах. Не так давно Невероятно Важный Совет Демонов решил, что передвигаться между кругами очень неудобно, поэтому была построена магистраль в виде прямой, проходящая через все круги (которые, на самом деле, являются окружностями, но демоны не сильно разбираются в геометрии).

На открытие магистрали собрались все грешники (в Аду не часто происходит что-то новое). Они сразу же разбрелись по ней и стали изучать, как устроена магистраль. Один из дозорных ангелов сразу же заметил, что в аду как-то пусто, и немедленно доложил об этом Богу. Он не растерялся и тут же отдал приказ уничтожить пустые круги Ада, который тотчас же был исполнен.

Дьяволу это крайне не понравилось, он решил встать в какое-то место на магистрали и создать новые круги Ада, с блекджеком и демонессами. Для оптимизации работ все грешники должны сразу оказаться на каких-то кругах, чтобы их потом не пришлось гонять туда-сюда. Конечно, создание каждого круга требует много сил, поэтому Дьявол хочет создать минимальное число кругов (мы всё ещё помним, что они являются окружностями), причём центры кругов будут ровно в той точке, куда встанет Дьявол (он хочет всегда находиться в центре внимания). Вам, как главному программисту Ада, поручено найти место, куда нужно встать Дьяволу.

Формат входных данных

Первая строка входного файла содержит n — число грешников ($1 \leqslant n \leqslant 200\,000$).

На следующей строке заданы n целых чисел — координаты грешников на магистрали, по модулю не превосходящие 60 000.

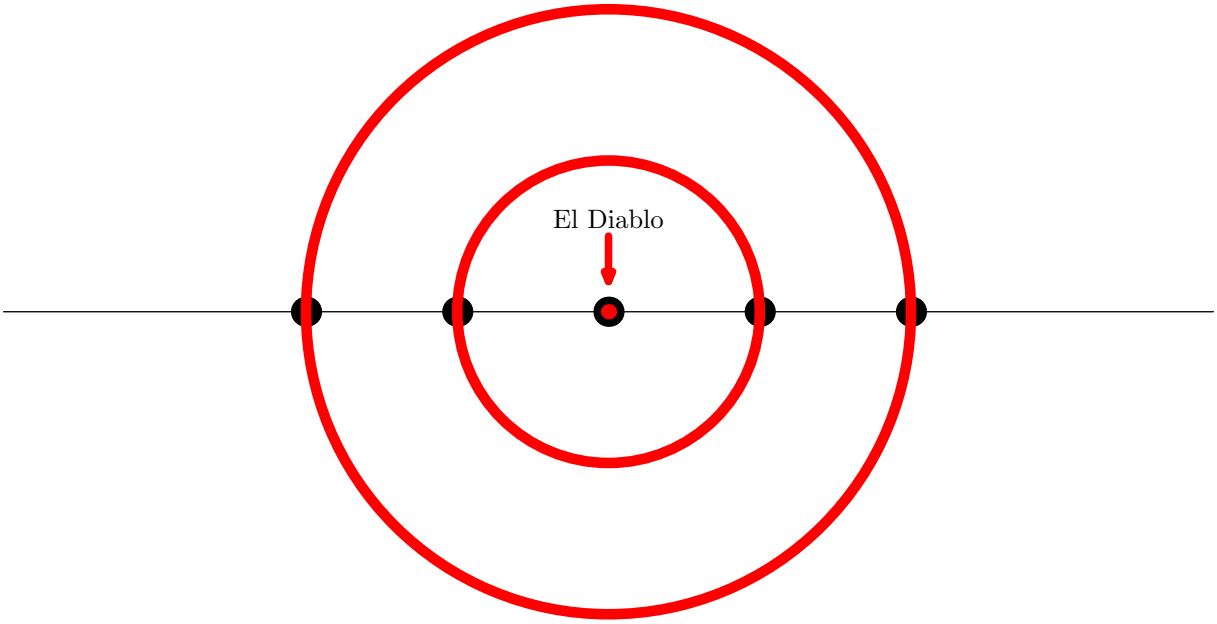
Формат выходных данных

В первой строке выходного файла выведите вещественную координату — куда надо встать Дьяволу. Во второй строке выведите минимальное количество кругов, необходимое для возвращения всех грешников на круги своя. В третьей строке выведите разделённые пробелами вещественные радиусы кругов. Заметим, что круги могут иметь нулевой радиус.

Если правильных ответов несколько вы можете вывести любой из них. Для всех точек минимальное расстояние до найденных окружностей не должно превышать 10^{-6}.

Примеры

circles.in	circles.out
4 1 1 2 2	1.5 1 0.5
5 1 2 3 4 5	3.0 3 0.0 1.0 2.0


El Diablo

Задача I. Таблица

Имя входного файла:	`table.in`
Имя выходного файла:	`table.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мебибайт

В таблице $n \times n$ расставьте числа от 1 до $2n-1$ (числа могут повторяться) так, чтобы для любого i в кресте, образованном i-й строкой и i-м столбцом, встречались все числа от 1 до $2n-1$.

Кстати, $n = 2^k$.

Формат входных данных

В единственной строке дано число k ($1 \leqslant k \leqslant 9$).

Формат выходных данных

Если такое расположение чисел возможно, выведите в первой строке «`Yes`» (без кавычек), а в последующих строках — саму таблицу. В противном случае выведите «`No`» (без кавычек).

Пример

table.in	table.out
1	Yes 2 1 3 2

Задача J. Слону нужен ваш совет!

Имя входного файла:	`torobishop.in`
Имя выходного файла:	`torobishop.out`
Ограничение по времени:	2 секунда
Ограничение по памяти:	256 мебибайт

Как известно, не все спортивные программисты любят шахматы. Ещё меньше поклонников среди участников олимпиад у шахмат на тороидальной доске. Тем не менее...

Дана тороидальная шахматная доска размера $n \times m$. От обыкновенной доски того же размера она отличается тем, что верхняя горизонталь соединена с нижней, а левая вертикаль с правой.

Горизонтали и вертикали пронумерованы последовательными натуральными числами: от 1 до n снизу вверх и от 1 до m слева направо, соответственно. Каждая клетка имеет координату, образованную упорядоченной парой (номер горизонтали, номер вертикали).

В клетке с координатой (x_1, y_1) стоит Ваш закадычный друг — слон. Напомним, слоны в шахматах перемещаются по диагонали на любое количество клеток за ход. Обратите внимание, что диагонали на тороидальной доске не такие, как на обычной. Например, из клетки (n, m) можно попасть в клетку $(1, 1)$ за один ход (на одну клетку по диагонали вправо вверх).

Слон мечтает попасть в клетку (x_2, y_2) как можно быстрее (за наименьшее количество ходов), а не то можно пропустить самое интересное! Слон нетерпелив, а потому он просит Вас сообщить ему, какое минимальное количество ходов ему требуется сделать, чтобы оказаться в заветной клетке. А если попасть в клетку (x_2, y_2) из (x_1, y_1) невозможно, то слон должен быть осведомлён об этом заранее (то есть прямо сейчас!), чтобы не двигаться зря.

Не оставляйте слона в беде! Они ведь, знаете, страшно мстят тем, кто не хочет им помогать...

Формат входных данных

В одном входном файле может описываться сразу несколько ситуаций.

Первая строка входного файла содержит число t — количество описанных ситуаций ($1 \leqslant t \leqslant 100$).

Каждая ситуация задаётся на трёх строках следующим образом: в первой строке через пробел записаны два целых числа n ($1 \leqslant n \leqslant 10^5$) и m ($1 \leqslant m \leqslant 10^5$). В следующей строке записаны координаты слона (x_1, y_1), а в третьей — координаты заветной клетки (x_2, y_2) ($1 \leqslant x_i \leqslant n, 1 \leqslant y_i \leqslant m$).

Формат выходных данных

Для каждой ситуации выведите единственное число — минимальное количество ходов, которое требуется слону на то, чтобы попасть из начального положения в требуемое.

Если попасть в (x_2, y_2) невозможно, выведите -1.

Примеры

torobishop.in	torobishop.out
3 2 4 1 1 2 4 8 8 1 1 2 1 1 6 1 2 1 2	1 -1 0