Задача A. Постройка дорожной сети

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

Странствующий торговец прибыл в королевство Lazyland. Он планирует начать свое путешествие из столицы, посетить каждый из городов королевства, а затем вернуться в столицу. Торговец знает координаты каждого города, однако дороги еще не были построены. Очевидно, не имея дорог, не возможно посетить все города королевства!

Известно, что в королевстве есть n городов. Их координаты равны: (x_1, y_1), $(x_2, y_2), \ldots, (x_n, y_n)$. Столица находится в точке (x_1, y_1). Король выразил свое желание помочь торговцу. Он прикажет построить для него дороги. Однако, жители страны слишком ленивые.

Как и все жители страны, инженеры тоже очень ленивы, поэтому если потребуется построить дорогу от точки (x_i, y_i) до точки (x_j, y_j), они будут использовать только горизонтальные или вертикальные отрезки. Поэтому нетрудно заметить, что длина такой дороги получится равной $|x_i - x_j| + |y_i - y_j|$.

Так как рабочие тоже невероятно ленивы, они согласятся построить не более, чем $n - 1$ дорогу для торговца — они слышали, что $n - 1$ дороги достаточно, чтобы путешествовать между всеми городами. Концами каждой дороги по понятным причинам будут какие-либо два различных города.

Торговцу дали право выбрать, какие именно дороги должны быть построены. Помогите ему вычислить длину минимального маршрута, который начинается в столице, обходит все города и затем возвращается в столицу при условии, что можно построить любые $n - 1$ дороги.

Формат входных данных

В первой строке записано число n ($1 \leqslant n \leqslant 10^4$) — количество городов в королевстве.

В каждой из следующих n строк записаны два числа x_i и y_i ($-1000 \leqslant x_i, y_i \leqslant 1000$) — координаты городов. Координаты столицы записаны в первой из этих строк.

Гарантируется, что никакие два города не находятся в одной точке.

Формат выходных данных

Выведите длину кратчайшего маршрута торговца, удовлетворяющего описанным в задаче требованиям.

Примеры

стандартный ввод	стандартный вывод
3 1 1 2 2 3 3	8
4 2 1 -1 2 -2 -1 1 -2	24
6 1 2 2 3 2 2 3 4 4 3 3 1	16

Задача B. Минимальное остовное дерево по каждому ребру

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

Дан связный неориентированный взвешенный граф без петель и кратных ребер, состоящий из n вершин и m ребер.

Для каждого ребра графа (u, v) найдите вес такого остовного дерева, что это ребро (u, v) входит в него, и при этом вес этого остовного дерева минимален.

Весом остовного дерева называется сумма весов ребер, входящих в остовное дерево.

Формат входных данных

В первой строке записаны два целых числа n и m ($1 \leqslant n \leqslant 2 \cdot 10^5, n - 1 \leqslant m \leqslant 2 \cdot 10^5$) — количество вершин и ребер в графе.

В каждой из следующих m строк записаны три целых числа $u_i, v_i, w_i (1 \leqslant u_i, v_i \leqslant n, u_i \neq v_i, 1 \leqslant w_i \leqslant 10^9)$ — вершины графа, соединенные i-м ребром, и вес этого ребра.

Формат выходных данных

Выведите m строк: i-я строка должна содержать минимальный вес такого остовного дерева, содержащего i-е ребро.

Ребра нумеруются от 1 до m в порядке их появления во входных данных.

Пример

стандартный ввод	стандартный вывод
5 7 1 2 3 1 3 1 1 4 5 2 3 2 2 5 3 3 4 2 4 5 4	9 8 11 8 8 8 9

Задача C. Кукушки

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	2 секунды
Ограничение по памяти:	512 мегабайт

Британские учёные решили заняться орнитологией и понаблюдать за жизнью необычных кукушек. Для этого они вырастили дерево и построили на нём n гнёзд, в каждом из которых живёт кукушка. Наблюдение за деревом состоит в том, что в некоторые моменты времени учёные оценивают, можно ли подложить определённое яйцо в гнездо к некоторой кукушке или нет.

Каждое яйцо может вынашиваться только в двух определённых гнёздах. Каждое яйцо задаётся неупорядоченной парой различных чисел (x, y) . Яйцо (x, y) может вынашиваться в любом из гнёзд x и y и не может вынашиваться в других гнёздах. Обратите внимание, яйцо (x, y) не отличается от яйца (y, x).

Теперь опишем процесс подкладывания яйца в имеющиеся гнезда: пусть учёные хотят подложить яйцо (x, y) в гнездо x . Если в гнезде x нет яйца, то яйцо (x, y) просто остаётся в этом гнезде, и процесс на данном шаге завершается. Если же в гнезде x лежит какое-то яйцо (x, p), то кукушка кладёт яйцо (x, y) в данное гнездо, а яйцо (x, p) пытается подложить в гнездо p аналогичным образом, и процесс продолжается.

Вам предлагается отвечать на вопросы учёных. Всего есть три типа вопросов:

(Теоретический) Закончится ли процесс, если подложить яйцо (x, y) в гнездо x ? Так как вопрос чисто теоретический, оно не добавляется на самом деле, и состояние гнёзд не меняется.

(Практический) Закончится ли процесс, если подложить яйцо (x, y) в гнездо x ? Если процесс закончится, то яйцо добавляется в реальности согласно описанному процессу.

(Теоретический) Сколько существует упорядоченных пар различных чисел (x, y), таких что яйцо (x, y) можно подложить в гнездо x с учётом имеющихся в гнёздах яиц? При этом для каждого яйца ответ определяется независимо от других добавляемых яиц.

Формат входных данных

В первой строке вводятся три целых числа n,m, q, $(2 \leqslant n \leqslant 200000, 0 \leqslant m \leqslant n, 1 \leqslant q \leqslant 600000)$, где n — количество гнёзд на дереве, m — количество яиц, которые учёные уже положили, q — количество вопросов, которые задают учёные.

В каждой из m последующих строк следуют по два числа x_i, y_i, означающих, что в гнезде x_i лежит яйцо (x_i, y_i) . Гарантируется, что все x_i различны и что $x_i \neq y_i$ для всех i .

В следующих q строках описаны вопросы учёных. Вопросы даны в том порядке, в котором на них требуется отвечать. Первое число t_j в строке описывает тип вопроса.

Если $t_j = 1$ или $t_j = 2$, то далее идут два различных числа x_j и y_j, описывающих яйцо, которое фигурирует в соответствующем вопросе.

Если $t_j = 1$, то яйцо не требуется добавлять в текущую расстановку.

Если $t_j = 2$, то яйцо требуется добавить, если процесс добавления потребует конечного числа перекладываний.

Если $t_j = 3$, то требуется определить количество упорядоченных пар (x, y), таких что яйцо (x, y) можно добавить в гнездо x с тем, чтобы процесс когда-нибудь завершился. В реальности никакие яйца в расстановку не добавляются.

Формат выходных данных

Для каждого вопроса первого и второго типа выведите единственное слово «Yes» или «No» в зависимости от того, закончится ли процесс перекладывания.

Для каждого запроса третьего типа выведите количество искомых упорядоченных пар.

Пример

стандартный ввод	стандартный вывод
5 3 8 1 2 5 1 2 4 1 1 2 3 2 1 2 3 2 4 2 2 5 3 3 1 4 5	Yes 20 Yes 8 No Yes 0 No

Задача D. Соединение и разъединение

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	3 секунды
Ограничение по памяти:	512 мегабайт

Вы когда-нибудь слышали про обход в глубину? Например, используя этот алгоритм, вы можете проверить является ли граф связным за время $O(E)$. Вы можете даже посчитать количество компонент связности за то же время.

А вы когда нибудь слышали про систему непересекающихся множеств? Используя эту структуру, вы можете быстро обрабатывать запросы "Добавить ребро в граф" и "Посчитать количество компонент связности в графе".

А вы когда-нибудь слышали о *динамической* задаче связности? В этой задаче вам необходимо обрабатывать три типа запросов:

1. Добавить ребро в граф.
2. Удалить ребро из графа.
3. Посчитать количество компонент связности в графе.

Можно считать, что граф является неориентированным. Изначально граф является пустым.

Формат входных данных

В первой строке находятся два целых числа N и K — количество вершин и количество запросов, сответственно ($1 \leqslant N \leqslant 300\,000$, $1 \leqslant K \leqslant 300\,000$). Следующие K строк содержат запросы, по одному в строке. Каждый запрос имеет один из трех типов:

1. `+` u v: Добавить ребро между вершинами u и v. Гарантируется, что такого ребра нет.
2. `-` u v: Удалить ребро между u и v. Гарантируется, что такое ребро есть.
3. `?`: Посчитать количество компонент связности в графе.

Вершины пронумерованы целыми числами от 1 до N. Во всех запросах $u \neq v$.

Формат выходных данных

Для каждого запроса типа '`?`', Выведите количество компонент связности в момент запроса.

Пример

стандартный ввод	стандартный вывод
5 11 ? + 1 2 + 2 3 + 3 4 + 4 5 + 5 1 ? - 2 3 ? - 4 5 ?	5 1 1 2

Задача E. Всем чмоки в этом чатике!

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

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

стандартный ввод	стандартный вывод
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

Задача F. Зависть

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

Для заданного связного неориентированного взвешенного графа G минимальным остовным деревом называется подграф G, содержащий все вершины G, являющийся деревом, а также имеющий минимально возможную сумму весов ребер.

Вам дан граф G. Если вы запустите алгоритм по поиску минимального остовного дерева, вы найдете лишь одно минимальное остовное дерево, а другие ребра будут завидовать. Вам дано несколько запросов, каждый запрос содержит подмножество ребер графа G, определите, существует ли минимальное остовное дерево этого графа, содержащее все эти ребра, или нет.

Формат входных данных

Первая строка содержит два целых числа n, m ($2 \leqslant n, m \leqslant 5 \cdot 10^5, n - 1 \leqslant m$) — количество вершин и ребер в графе, соответственно.

i-я из следующих m строк содержит три целых числа u_i, v_i, w_i ($u_i \neq v_i, 1 \leqslant w_i \leqslant 5 \cdot 10^5$) — концы и вес i-го ребра. Возможно, что существует более одного ребра между некоторыми парами вершин. Гарантируется, что данный граф связен.

Следующая строка содержит одно целое число q ($1 \leqslant q \leqslant 5 \cdot 10^5$) — количество запросов.

Далее следуют q строк, i-я из них содержит описание i-го запроса. Она начинается с целого числа k_i ($1 \leqslant k_i \leqslant n - 1$) — размера подмножества ребер, а затем следуют k_i различных целых чисел от 1 до m — индексы ребер в подмножестве. Гарантируется, что сумма значений k_i для всех $1 \leqslant i \leqslant q$ не превосходит $5 \cdot 10^5$.

Формат выходных данных

Для каждого запроса выведите «YES» (без кавычек), если существует минимальное остовное дерево со всеми данными ребрами, и «NO» (конечно же, без кавычек) иначе.

Пример

стандартный ввод	стандартный вывод
5 7 1 2 2 1 3 2 2 3 1 2 4 1 3 4 1 3 5 2 4 5 2 4 2 3 4 3 3 4 5 2 1 7 2 1 2	YES NO YES NO

Замечание

Рассмотрим пример из теста.

Минимальное остовное дерево из ребер $(1, 3, 4, 6)$ содержит все ребра из первого запроса, поэтому ответ на первый запрос «YES».

Ребра из второго запроса образуют цикл длины 3, поэтому не существует остовного дерева, включающего в себя эти три ребра. Поэтому ответ «NO».

Задача G. Своя игра

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	5 секунд
Ограничение по памяти:	768 мегабайт

У маленького Даниэля есть мешок конфета и N карточек с числами. На i-й карточке написано число P_i. Пока Даниэль ел конфеты, он придумал игру. В этой игре Даниэль мог произвольное число раз повторять следующую процедуру: связать ниткой две карточки с числами a и b и съесть $\min(P_a\%P_b, P_b\%P_a)$ конфет, где $x\%y$ означает остаток от деления x на y нацело.

В своей игре Даниэль хочет связать некоторые карточки так, чтобы в конце, если бы он поднял одну карточку, то поднялись бы и все остальные. Каждая карточка может быть связана с произвольным числом других карточек. Даниэль не хочет заработать себе диабет, поэтому он хочет достичь своей цели, съев при этом минимально возможное число конфет. Помогите Даниэлю морально подготовиться и вычислите, сколько конфет ему придётся съесть.

Формат входных данных

В первой строке находится кол-во карточек N, $(1 \leqslant N \leqslant 10^5)$

Далее следуют N строк, в i-й из них находится одно число P_i, $(1 \leqslant P_i \leqslant 10^7)$

Формат выходных данных

Выведите одно число — минимальное кол-во конфет, которое придётся съесть Даниэлю.

Примеры

стандартный ввод	стандартный вывод
4 2 6 3 11	1
4 1 2 3 4	0
3 4 9 15	4

Замечание

В первом примере Даниэль может связать первую и вторую карточку за 0 конфет, вторую и третью за 0 конфет, первую и четвёртую за 1 конфету.

Задача H. Разностный MST

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	2 секунды
Ограничение по памяти:	1024 мегабайта

Дан массив $x_1, x_2, \ldots, x_n$.

Давайте создадим неориентированный граф на n вершинах, в котором изначально нет ребер.

После этого для каждой пары (u, v), такой что $u < v$, добавим в граф ребро между вершинами u и v веса $x_v - x_u$.

Найдите вес минимального остовного дерева в получившемся графе.

Формат входных данных

Первая строка входных данных содержит одно целое положительное число t ($1 \leqslant t \leqslant 300\,000$) — количество тестовых примеров.

Первая строка каждого тестового примера содержит одно целое положительное число n ($1 \leqslant n \leqslant 300\,000$) — количество элементов массива.

Вторая строка каждого тестового примера содержит n целых чисел $x_1, x_2, \ldots, x_n$ ($-300\,000 \leqslant x_i \leqslant 300\,000$) — элементы массива.

Гарантируется, что сумма n по всем тестовым примерам не превышает $300\,000$.

Формат выходных данных

Для каждого тестового примера выведите одно целое число — вес минимального остовного дерева в данном графе.

Пример

стандартный ввод	стандартный вывод
2 5 1 2 3 4 5 3 10 45 10	4 -35

Задача I. Еще одна задача про остов

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

Костя с Ваней при подготовке очередной задачи на персистентную двумерную динамическую структуру данных нашли массив a_i размера n, состоящий из целых чисел, каждое из которых находится в промежутке $[0, m-1]$. Костя, как любитель цветных графов, сразу создал полный неориентированный граф из n вершин, вершине v был задан цвет a_v. Ваня, как любитель взвешенных графов, сразу задал каждому ребру (v, u) вес $f(a_v, a_u)$.

$$f(a,b) = \begin{cases} a+b & a+b < m \\ a+b-m, & \text{иначе} \end{cases}$$

Теперь ребят интересует вопрос: каким же будет вес минимального остовного дерева для нашего графа? Помогите им найти ответ.

Формат входных данных

Программа получает на вход два числа n, m ($1 \leqslant n \leqslant 2 \cdot 10^5$, $1 \leqslant m \leqslant 10^9$).
В следующей строке вводится n целых чисел — массив a_i ($0 \leqslant a_i \leqslant m-1$).

Формат выходных данных

Выведите одно число — суммарный вес остовного дерева.

Пример

стандартный ввод	стандартный вывод
3 2 0 1 1	1

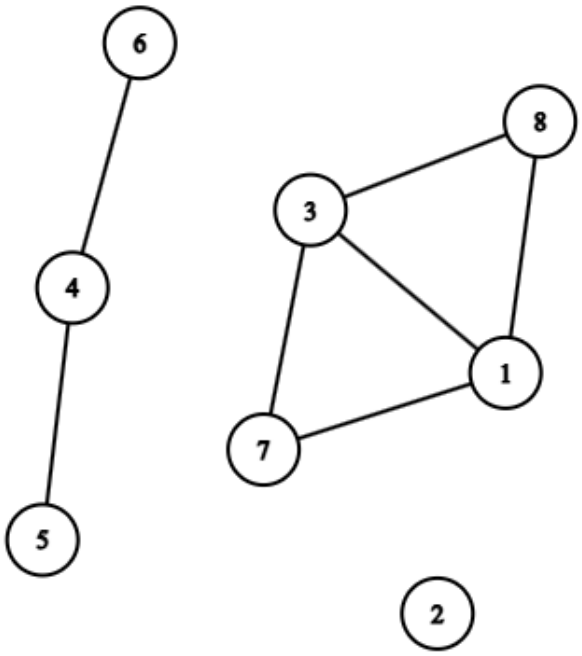
Задача J. Включение графа

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	4 секунды
Ограничение по памяти:	512 мегабайт

Компонентой связности неориентированного графа назовем такое множество вершин этого графа S, что:

- для каждой пары вершин (u, v) из множества S существует путь между вершинами u и v;
- не существует ни одной вершины вне S, из которой есть путь в вершину из S.

Например, у графа на картинке ниже три компоненты связности: $\{1, 3, 7, 8\}$, $\{2\}$, $\{4, 5, 6\}$.



Скажем, что граф A включает граф B, если каждая компонента связности графа B является подмножеством какой-то компоненты связности графа A.

Вам даны два графа, A и B, оба из которых состоят из n вершин, пронумерованных от 1 до n. Изначально в графах нет ребер. Вы должны обрабатывать запросы двух типов:

- добавить ребро в какой-то из графов;
- удалить ребро из какого-то из графов.

После каждого запроса вы должны вычислить минимальное количество ребер, которое нужно добавить в граф A, чтобы A включал B. Обратите внимание — вы только считаете это количество ребер, но не добавляете их.

Формат входных данных

В первой строке задано два целых числа n и q ($2 \leqslant n \leqslant 4 \cdot 10^5; 1 \leqslant q \leqslant 4 \cdot 10^5$) — количество вершин и запросов соответственно.

Далее следуют q строк, в i-й из которых задан i-й запрос. Описание запроса начинается с символа c_i (либо A, либо B) — граф, к которому надо совершать запрос. Далее следуют два целых числа x_i и y_i ($1 \leqslant x_i, y_i \leqslant n; x_i \neq yi$). Если в соответствующем графе существует ребро (x_i, y_i), его надо оттуда удалить, иначе его надо добавить в этот граф.

Формат выходных данных

На каждый запрос выведите одно целое число — минимальное количество ребер, которое нужно добавить в граф A, чтобы он включал граф B.

Пример

стандартный ввод	стандартный вывод
6 9 A 2 3 B 1 3 A 2 1 A 3 2 B 5 6 A 6 5 A 3 4 A 4 2 A 4 3	0 1 0 1 2 1 1 0 1

Задача K. По модулю 3

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	4 секунды
Ограничение по памяти:	512 мегабайт

Наверняка вы видели задачи, которые требуют вывести ответ по модулю $10^9 + 7$, $10^9 + 9$ или 998244353. Но видели ли вы когда-нибудь задачу, где нужно брать ответ по модулю 3?

Вам дан функциональный граф, состоящий из n вершин, пронумерованных от 1 до n. Это ориентированный граф, в котором у каждой вершины есть ровно одна исходящая дуга. Граф задан массивом $g_1, g_2, ..., g_n$, где g_i означает, что есть дуга, которая идет от i к g_i. Для некоторых вершин исходящие дуги могут являться петлями.

Изначально все вершины графа окрашены в цвет 1. Вы можете выполнить следующую операцию: выбрать вершину и цвет от 1 до k, а затем окрасить эту вершину и все вершины, которые от нее достижимы. Эту операцию можно выполнять сколько угодно раз (можно и ноль).

Вам нужно обработать q запросов. Запрос описывается тремя целыми числами x, y и k. Для каждого запроса вам нужно:

- присвоить $g_x := y$;
- затем вычислить количество различных раскрасок графа для данного значения k (две раскраски различны, если существует хотя бы одна вершина, окрашенная в разные цвета в этих двух раскрасках); поскольку ответ может быть очень большим, выведите его по модулю 3.

Обратите внимание, что в каждом запросе начальная раскраска графа сбрасывается (все вершины изначально имеют цвет 1 в каждом запросе).

Формат входных данных

Первая строка содержит два целых числа n и q ($1 \leqslant n, q \leqslant 2 \cdot 10^5$).

Вторая строка содержит n целых чисел $g_1, g_2, ..., g_n$ ($1 \leqslant g_i \leqslant n$).

Далее следуют q строк, i-я из них содержит три целых числа x_i, y_i и k_i ($1 \leqslant x_i, y_i \leqslant n; 1 \leqslant k_i \leqslant 10^9$).

Формат выходных данных

Для каждого запроса выведите одно целое число — количество различных раскрасок графа для данного значения k, взятое по модулю 3.

Примеры

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