Задача A. Самое дешевое ребро

Имя входного файла:	`stdin`
Имя выходного файла:	`stdout`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Дано подвешенное дерево с корнем в первой вершине. Все ребра имеют веса (стоимости). Вам нужно ответить на M запросов вида "найти у двух вершин минимум среди стоимостей ребер пути между ними".

Формат входных данных

В первой строке файла записано одно числ — n (количество вершин).

В следующих $n-1$ строках записаны два числа — x и y. Число x на строке i означает, что x — предок вершины i, y означает стоимость ребра.

$x < i, |y| \leqslant 10^6$.

Далее m запросов вида (x, y) — найти минимум на пути из x в y $(x \neq y)$.

Ограничения: $2 \leqslant n \leqslant 5 \cdot 10^4, 0 \leqslant m \leqslant 5 \cdot 10^4$.

Формат выходных данных

m ответов на запросы.

Пример

stdin	stdout
5 1 2 1 3 2 5 3 2 2 2 3 4 5	2 2

Задача B. День Объединения

Имя входного файла:	`stdin`
Имя выходного файла:	`stdout`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

В Байтландии есть целых n городов, но нет ни одной дороги. Король решил исправить эту ситуацию и соединить некоторые города дорогами так, чтобы по этим дорогам можно было бы добраться от любого города до любого другого. Когда строительство будет завершено, Король планирует отпраздновать День Объединения. К сожалению, казна Байтландии почти пуста, поэтому Король требует сэкономить деньги, минимизировав суммарную длину всех построенных дорог.

Формат входных данных

Первая строка входного файла содержит натуральное число n ($1 \leqslant n \leqslant 5\,000$) — количество городов в Байтландии. Каждая из следующих n строк содержит два целых числа x_i, y_i — координаты i-го города ($-10\,000 \leqslant x_i, y_i \leqslant 10\,000$). Никакие два города не расположены в одной точке.

Формат выходных данных

Первая строка выходного файла должна содержать минимальную суммарную длину дорог. Выведите число с точностью не менее 10^{-3}.

Примеры

stdin	stdout
6 1 1 7 1 2 2 6 2 1 3 7 3	9.65685

Задача C. Остовное дерево 2

Имя входного файла:	`stdin`
Имя выходного файла:	`stdout`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Требуется найти в связном графе остовное дерево минимального веса.

Формат входных данных

Первая строка входного файла содержит два натуральных числа n и m — количество вершин и ребер графа соответственно. Следующие m строк содержат описание ребер по одному на строке. Ребро номер i описывается тремя натуральными числами b_i, e_i и w_i — номера концов ребра и его вес соответственно ($1 \leqslant b_i, e_i \leqslant n$, $0 \leqslant w_i \leqslant 100\,000$). $n \leqslant 20\,000, m \leqslant 100\,000$.

Граф является связным.

Формат выходных данных

Первая строка выходного файла должна содержать одно натуральное число — вес минимального остовного дерева.

Примеры

stdin	stdout
4 4 1 2 1 2 3 2 3 4 5 4 1 4	7

Задача D. Range Minimum Query

Имя входного файла:	`stdin`
Имя выходного файла:	`stdout`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Компания Giggle открывает свой новый офис в Судиславле, и вы приглашены на собеседование. Ваша задача — решить поставленную задачу.

Вам нужно создать структуру данных, которая представляет из себя массив целых чисел. Изначально массив пуст. Вам нужно поддерживать две операции:

- запрос: «`? i j`» — возвращает минимальный элемент между i-ым и j-м, включительно;
- изменение: «`+ i x`» — добавить элемент x после i-го элемента списка. Если $i = 0$, то элемент добавляется в начало массива.

Конечно, эта структура должна быть достаточно хорошей.

Формат входных данных

Первая строка входного файла содержит единственное целое число n — число операций над массивом ($1 \leqslant n \leqslant 200\,000$). Следующие n строк описывают сами операции. Все операции добавления являются корректными. Все числа, хранящиеся в массиве, по модулю не превосходят 10^9.

Формат выходных данных

Для каждой операции в отдельной строке выведите её результат.

Примеры

stdin	stdout
8 + 0 5 + 1 3 + 1 4 ? 1 2 + 0 2 ? 2 4 + 4 1 ? 3 5	4 3 1

Задача E. Вперёд!

Имя входного файла:	`stdin`
Имя выходного файла:	`stdout`
Ограничение по времени:	6 секунд
Ограничение по памяти:	256 мегабайт

Капрал Дукар любит раздавать приказы своей роте. Самый любимый его приказ — “Вперёд!”. Капрал строит солдат в ряд и отдаёт некоторое количество приказов, каждый из них звучит так: “Рядовые с l_i по l_j — вперёд!”

Перед тем, как Дукар отдал первый приказ, солдаты были пронумерованы от 1 до n, слева направо. Услышав приказ “Рядовые с l_i по l_j — вперёд!”, солдаты, стоящие на местах с l_i по l_j включительно, продвигаются в начало ряда, в том же порядке, в котором были.

Например, если в какой-то момент солдаты стоят в порядке $1, 3, 6, 2, 5, 4$, то после приказа “Рядовые с 2 по 3 — вперёд!”, порядок будет таким: $3, 6, 1, 2, 5, 4$. А если потом Капрал вышлет вперёд солдат с 3 по 4, то порядок будет уже таким: $1, 2, 3, 6, 5, 4$.

Вам дана последовательность из приказов Капрала. Найдите порядок, в котором будут стоять солдаты после исполнения всех приказов.

Формат входных данных

В первой строке входного файла указаны числа n и m ($2 \leqslant n \leqslant 100\,000$, $1 \leqslant m \leqslant 100\,000$) — число солдат и число приказов. Следующие m строк содержат приказы в виде двух целых чисел: l_i и r_i ($1 \leqslant l_i \leqslant r_i \leqslant n$).

Формат выходных данных

Выведите в выходной файл n целых чисел — порядок, в котором будут стоять солдаты после исполнения всех приказов.

Пример

stdin	stdout
6 3 2 4 3 5 2 2	1 4 5 2 3 6

Задача F. Переворачивания

Имя входного файла:	`stdin`
Имя выходного файла:	`stdout`
Ограничение по времени:	5 секунд
Ограничение по памяти:	256 мебибайт

Учитель физкультуры школы с углубленным изучением предметов уже давно научился считать суммарный рост всех учеников, находящихся в ряду на позициях от l до r. Но дети играют с ним злую шутку. В некоторый момент дети на позициях с l по r меняются местами. Учитель заметил, что у детей не очень богатая фантазия, поэтому они всегда «переворачивают» этот отрезок, т. е. l меняется с r, $l+1$ меняется с $r-1$ и так далее. Но учитель решил не ругать детей за их хулиганство, а все равно посчитать суммарный рост на всех запланированных отрезках.

Формат входных данных

В первой строке записано два числа n и m ($1 \leqslant n, m \leqslant 200\,000$) — количество детей в ряду и количество событий, произошедших за все время. Во второй строке задано n натуральных чисел — рост каждого школьника в порядке следования в ряду. Рост детей не превосходит $2 \cdot 10^5$. Далее в m строках задано описание событий: три числа q, l, r в каждой строке ($0 \leqslant q \leqslant 1$, $1 \leqslant l \leqslant r \leqslant n$). Число q показывает тип события: 0 показывает необходимость посчитать и вывести суммарный рост школьников на отрезке $[l, r]$; 1 показывает то, что дети на отрезке $[l, r]$ «перевернули» свой отрезок. Все числа во входном файле целые.

Формат выходных данных

Для каждого события типа 0 выведите единственное число на отдельной строке — ответ на этот запрос.

Пример

stdin	stdout
5 6 1 2 3 4 5 0 1 5 0 2 4 1 2 4 0 1 3 0 4 5 0 3 5	15 9 8 7 10

Задача G. Приказы

Имя входного файла:	`stdin`
Имя выходного файла:	`stdout`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мебибайт

Вася работает в НИИГСД (НИИ Государственных Структур Данных). Он изучает приказы правительства далёкого государства.

В том государстве все города расположены вдоль одной дороги. Они пронумерованы в порядке обхода. Изначально качество жизни в каждом из них равно нулю.

Далее последовательно издаются указы вида «уровень жизни в городах с i по j должен стать не меньше x».

Также есть некоторые официальные заявления. Они имеют следующую форму: «средний уровень жизни в городах с i по j равен x». Вася нуждается в помощи с проверкой этих утверждений: для каждого из них известны i и j, требуется подсчитать верное значение x.

Можете считать, что каждый приказ исполняется, а также в каждый момент времени каждый город имеет минимальный неотрицательный уровень жизни, удовлетворяющий всем приказам.

Формат входных данных

Ввод состоит из одного или более тестов. Каждый тест начинается строкой с двумя целыми числами n и k — числом городов и событий, соответственно. Следующие k строк содержат по одному описанию события:

1. `^` i j x означает приказ: после этого, все города с номерами от i до j включительно должны иметь уровень жизни не менее x ($1 \leqslant x \leqslant 10^9$, $1 \leqslant i \leqslant j \leqslant n$).
2. `?` i j означает официальное заявление: следует подсчитать средний уровень жизни в городах с i по j включительно ($1 \leqslant i \leqslant j \leqslant n$).

В конце ввода будет помещён тест с $n = k = 0$, который не требуется обрабатывать.

Сумма n по всему вводу не превысит 100 000. Сумма k по всему вводу не превысит 100 000.

Формат выходных данных

Для каждого официального заявления выведите на отдельной строке искомый средний уровень жизни в виде несократимой дроби с наименьшим возможным натуральным знаменателем. Если знаменатель равен 1, выведите вместо дроби целое число. Следуйте формату вывода, как это показано в примере.

Пример

stdin	stdout
10 10 ? 1 10 ^ 1 10 1 ? 1 10 ^ 2 3 10 ^ 3 4 5 ? 2 2 ? 3 3 ? 4 4 ? 1 5 ? 1 10 0 0	0 1 10 10 5 27/5 16/5

Задача H. Динамический Лес

Имя входного файла:	`stdin`
Имя выходного файла:	`stdout`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Вам нужно научиться обрабатывать 3 типа запросов:

1. Добавить ребро в граф (`link`).
2. Удалить ребро из графа (`cut`).
3. По двум вершинам a и b, определить, лежат ли они в одной компоненте связности (`get`).

Изначально граф пустой (содержит N вершин, не содержит ребер). Гарантируется, что в любой момент времени граф является лесом. При добавлении ребра гарантируется, что его сейчас в графе нет. При удалении ребра гарантируется, что оно уже добавлено.

Формат входных данных

Числа N и M ($1 \leqslant N \leqslant 10^5 + 1$, $1 \leqslant M \leqslant 10^5$) — количество вершин в дереве и, соответственно, запросов. Далее M строк, в каждой строке команда (`link` или `cut`, или `get`) и 2 числа от 1 до N — номера вершин в запросе.

Формат выходных данных

В выходной файл для каждого запроса `get` выведите 0, если не лежат, или 1, если лежат.

Пример

stdin	stdout
3 7 get 1 2 link 1 2 get 1 2 cut 1 2 get 1 2 link 1 2 get 1 2	0101
5 10 link 1 2 link 2 3 link 4 3 cut 3 4 get 1 2 get 1 3 get 1 4 get 2 3 get 2 4 get 3 4	110100

Задача I. Внутреняя точка 2

Имя входного файла:	`stdin`
Имя выходного файла:	`stdout`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Дан совсем невыпуклый простой N-угольник и K точек. Напомним, N-угольник называется простым, если не имеет ни самопересечений, ни самокасаний. Для каждой точки нужно определить, где она находится — внутри, на границе, или снаружи.

Формат входных данных

В первой строке дано целое число T — количество тестов.
Далее идут T тестов. Тесты разделены переводом строки.
N ($3 \leqslant N \leqslant 10^5$). Далее N точек — вершины многоугольника.
K ($0 \leqslant K \leqslant 10^5$). Далее K точек — запросы.
Все координаты — целые числа по модулю не превосходящие 10^9.
Суммарное количество N и K не превосходит $2 \cdot 10^6$.

Формат выходных данных

Для каждого запроса одна строка — `INSIDE`, `BORDER` или `OUTSIDE`.
Тесты следует разделять переводом строки.

Примеры

stdin	stdout
1 4 0 0 2 0 2 2 0 2 4 1 1 0 0 0 1 0 3	INSIDE BORDER BORDER OUTSIDE

Задача J. Междугородний Экспресс

Имя входного файла:	`stdin`
Имя выходного файла:	`stdout`
Ограничение по времени:	5 секунд
Ограничение по памяти:	256 мегабайт

Андрей разрабатывает программу для электронной продажи билетов. Сейчас он хочет протестировать ее на «Междугороднем Экспрессе», маршрут которого соединяет два крупных города. По пути между городами экспресс останавливается в $n-2$ населенных пунктах и, таким образом всего проходит n остановок, пронумерованных от 1 до n.

В «Междугороднем Экспрессе» s посадочных мест, пронумерованных от 1 до s. У программы Андрея есть доступ к базе данных, в которой содержится информация об уже проданных билетах по напровлению из города 1 в город n. Нужно, чтобы программа умела отвечать, возможно ли продать билет со станции a до станции b и, если возможно, выдать номер минимального места, свободного на всем промежутке от a до b.

Пока система должна работать в тестовом режиме, поэтому никаких билетов она не продает и, выведя номер свободного места, она не должа его резервировать.

Помогите Андрею дописать эту программу.

Формат входных данных

В первой строке файла через пробел написаны числа n, s и m — число уже проданных билетов ($2 \leqslant n \leqslant 10^9$, $1 \leqslant s \leqslant 100\,000$, $0 \leqslant m \leqslant 100\,000$).

Далее m строк содержат по три числа, описывающих каждый из билетов: c_i — номер занятого места, a_i — станция отправления и b_i — станция прибытия ($1 \leqslant c_i \leqslant s$, $1 \leqslant a_i < b_i \leqslant n$).

В следующей строке записано число q — количество запросов к программе ($1 \leqslant q \leqslant 100\,000$).

Для обработки запросов должно поддерживатся специальное число p, изначально равное 0.

Дальше идет $2q$ чисел, описывающие запросы. Каждый запрос описывается двумя числами: x_i и y_i ($x_i < y_i$). Чтобы получить a и b — номера станций запроса нужно воспользоваться следующими формулами: $a = x_i + p$, $b = y_i + p$. Ответ на запрос — 0, если свободных мест на данном отрезке станций нет, иначе — номер минимального свободного места.

Ответив на запрос, нужно присвоить p значение ответа.

Формат выходных данных

Для каждого запроса выведите ответ на него.

Пример

stdin	stdout
5 3 5 1 2 5 2 1 2 2 4 5 3 2 3 3 3 4 10 1 2 1 2 1 2 2 3 -2 0 2 4 1 3 1 4 2 5 1 5	1 2 2 3 0 2 0 0 0 0

Замечание: запросы должны получиться вот такие: $(1,2)$, $(2,3)$, $(3,4)$, $(4,5)$, $(1,3)$, $(2,4)$, $(3,5)$, $(1,4)$, $(2,5)$, $(1,5)$.

Задача K. Миллиардеры

Имя входного файла:	`stdin`
Имя выходного файла:	`stdout`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Возможно, вы знаете, что из всех городов мира больше всего миллиардеров живёт в Москве. Но, поскольку работа миллиардера подразумевает частые перемещения по всему свету, в определённые дни какой-то другой город может занимать первую строчку в таком рейтинге. Ваши приятели из ФСБ, ФБР, MI5 и Шин Бет скинули вам списки перемещений всех миллиардеров за последнее время. Ваш работодатель просит посчитать, сколько дней в течение этого периода каждый из городов мира был первым по общей сумме денег миллиардеров, находящихся в нём.

Формат входных данных

В первой строке записано число n — количество миллиардеров ($1 \leqslant n \leqslant 10\,000$). Каждая из следующих n строк содержит данные на определённого человека: его имя, название города, где он находился в первый день данного периода, и размер состояния. В следующей строке записаны два числа: m — количество дней, о которых есть данные ($1 \leqslant m \leqslant 50\,000$), k — количество зарегистрированных перемещений миллиардеров ($0 \leqslant k \leqslant 50\,000$). Следующие k строк содержат список перемещений в формате: номер дня (от 1 до $m-1$), имя человека, название города назначения. Вы можете считать, что миллиардеры путешествуют не чаще одного раза в день, и что они отбывают поздно вечером и прибывают в город назначения рано утром следующего дня. Список упорядочен по возрастанию номера дня. Все имена и названия городов состоят не более чем из 20 латинских букв, регистр букв имеет значение. Состояния миллиардеров лежат в пределах от 1 до 100 миллиардов.

Формат выходных данных

В каждой строке должно содержаться название города и, через пробел, количество дней, в течение которых этот город лидировал по общему состоянию миллиардеров, находящихся в нём. Если таких дней не было, пропустите этот город. Города должны быть отсортированы по алфавиту (используйте обычный порядок символов: ABC...Zabc...z).

Примеры

stdin	stdout
5 Abramovich London 15000000000 Deripaska Moscow 10000000000 Potanin Moscow 5000000000 Berezovsky London 2500000000 Khodorkovsky Chita 1000000000 25 9 1 Abramovich Anadyr 5 Potanin Courchevel 10 Abramovich Moscow 11 Abramovich London 11 Deripaska StPetersburg 15 Potanin Norilsk 20 Berezovsky Tbilisi 21 Potanin StPetersburg 22 Berezovsky London	Anadyr 5 London 14 Moscow 1

Дополнительные задачи

Задача L. Таможенные пошлины

Имя входного файла:	`stdin`
Имя выходного файла:	`stdout`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мебибайт

Недавно королева страны AlgoLand придумала новый способ отмывания денег для своего королевского двора. Она решила, что всякий житель, желающий совершить путешествие из одного города страны в другой, должен расплатиться за это желание своими деньгами.

В стране AlgoLand есть N городов, пронумерованных от 1 до N. Некоторые города соединены дорогами, движение по которым разрешено в двух направлениях. Начиная движение по какой-нибудь дороге, путешественник обязательно должен доехать до ее конца.

Предположим теперь, что житель страны хочет совершить путешествие из города A в город B. Новый указ королевы гласит, что при проезде по любой дороге страны во время этого путешествия, полицейские могут взять с этого жителя таможенную пошлину в пользу королевского двора (а могут и не взять). Если при этом у жителя недостаточно денег для уплаты пошлины, то он автоматически попадает в тюрьму. Указ также устанавливает величину пошлины для каждой дороги страны. Так как королева заботится о жителях своей страны, то она запретила полицейским брать с жителя пошлину более чем три раза во время одного путешествия.

Отметим, что если существует несколько способов попасть из города A в город B, то житель может выбрать для путешествия любой из них по собственному желанию.

Напишите программу, которая определяет, какую минимальную сумму денег должен взять с собой житель, чтобы гарантированно не попасть в тюрьму во время путешествия.

Формат входных данных

Первая строка входного файла содержит числа N и M ($2 \leqslant N \leqslant 10\,000$, $1 \leqslant M \leqslant 100\,000$), разделенные пробелом — количества городов и дорог. Следующие M строк описывают дороги. Каждая из этих строк описывает одну дорогу и содержит три числа X, Y, Z ($1 \leqslant X, Y \leqslant N$; $X \neq Y$; $1 \leqslant Z \leqslant 1\,000\,000\,000$), разделенных пробелами, означающие, что дорога соединяет города X и Y и пошлина за проезд по ней равна Z денежных единиц. Все числа Z целые. Последняя строка содержит числа A и B ($1 \leqslant A, B \leqslant N$; $A \neq B$) — номера начального и конечного городов путешествия. Гарантируется, что существует хотя бы один способ проезда из A в B.

Формат выходных данных

Единственная строка выходного файла должна содержать одно число, равное минимальной сумме денег, которую должен взять с собой житель, чтобы иметь возможность совершить путешествие из города A в город B и при этом гарантированно не попасть в тюрьму независимо от действий полицейских.

Пример

stdin	stdout
5 6 1 2 10 1 3 4 3 2 3 1 4 1 4 5 2 5 2 3 1 2	6

Задача M. Динамический Лес 2

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

Вам нужно научиться обрабатывать 3 типа запросов:

1. Добавить ребро в граф (`link`).
2. Удалить ребро из графа (`cut`).
3. По двум вершинам a и b вернуть длину пути между ними (или -1, если они лежат в разных компонентах связности) (`get`).

Изначально граф пустой (содержит N вершин, не содержит ребер). Гарантируется, что в любой момент времени граф является лесом. При добавлении ребра гарантируется, что его сейчас в графе нет. При удалении ребра гарантируется, что оно уже добавлено.

Формат входных данных

Числа N и M ($1 \leqslant N \leqslant 10^5 + 1$, $1 \leqslant M \leqslant 10^5$) — количество вершин в дереве и, соответственно, запросов. Далее M строк, в каждой строке команда (`link` или `cut`, или `get`) и 2 числа от 1 до N — номера вершин в запросе.

Формат выходных данных

В выходной файл для каждого запроса `get` выведите одно число — расстояние между вершинами, или -1, если они лежат в разных компонентах связности.

Пример

стандартный ввод	стандартный вывод
3 7 get 1 2 link 1 2 get 1 2 cut 1 2 get 1 2 link 1 2 get 1 2	-1 1 -1 1
5 10 link 1 2 link 2 3 link 4 3 cut 3 4 get 1 2 get 1 3 get 1 4 get 2 3 get 2 4 get 3 4	1 2 -1 1 -1 -1

Задача N. Игровой автомат

Имя входного файла:	`stdin`
Имя выходного файла:	`stdout`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мебибайт

Создатели игровых автоматов выпустили на рынок математическую игру. Игра состоит в следующем: в каждом автомате создателями задана уникальная комбинация из $M + 1$ числа в P-ичной системе счисления. Каждое число состоит из N разрядов. Первые M чисел во время игры не меняют своего значения, обозначим их a_1, a_2, …, a_M. Во время игры автомат несколько раз случайным образом выбирает из первых M чисел одно число и поразрядно прибавляет его к $M + 1$-му числу (будем называть $M + 1$-е число счетчиком) по модулю P, тем самым, изменяя его (т. е. счетчик поразрядно накапливает сумму всех сложений по модулю P). Игрок получает выигрыш, если в результате игры счетчик обнулится.

Поразрядное сложение по модулю P выполняется следующим образом: если в каком-либо разряде числа получено значение, большее $P - 1$, то оно уменьшается на P, например, при $P = 5$ и $N = 3$ результат сложения чисел 123 и 144 равен 212.

Вам прислали на инспекцию несколько таких автоматов, удостоверьтесь в том, что выигрыш принципиально возможен.

Формат входных данных

Первая строка файла содержит натуральное число — количество автоматов, присланных на инспекцию. В следующих строках описываются сами игровые автоматы. Каждый автомат описывается отдельно в следующем формате. Первая строка содержит числа P, N, M ($1 \leqslant N, M \leqslant 100$, $2 \leqslant P \leqslant 255$). Следующие $M + 1$ строк содержат по N чисел в P–ичной системе счисления, каждое из которых — значение одного разряда P–ичного числа из уникальной комбинации чисел описываемого автомата. Значения разрядов P–ичного числа задаются как числа в десятичной записи через пробел. Суммарный размер входного файла не превосходит 40 000 байт.

Формат выходных данных

Ответ по каждому автомату должен содержаться в отдельной строке. Ответ — это число 0, если игроку вообще не удастся выиграть. Если же выигрыш возможен, то ответ — это число 1 и далее — M чисел через пробел в этой же строке файла: k_1, k_2, …, k_M, где значение k_i ($k_i \leqslant P$) указывает, сколько раз нужно прибавить к счетчику число a_i, чтобы в результате всех сложений счетчик обнулился. Если решение не единственно, то выведите любое из них.

Пример

stdin	stdout
3 4 2 2 2 2 2 2 3 3 3 2 4 1 0 2 0 0 0 0 1 2 1 14 2 2 12 12 10 10 3 3	0 1 1 0 0 2 0