

Задача А. Треугольники

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

Во время проектирования Церебро (компьютера, позволяющего определить местоположение любого мутанта на Земле) Ксавьер столкнулся со следующей проблемой. Задача поиска всех мутантов одновременно чрезвычайно сложна. Каждый мутант обладает своей силой, и каждый мутант наводит некоторые помехи, мешающие поиску. Если представить Землю плоской (при поиске в ограниченной области это разумное допущение), то и каждого мутанта, и Церебро можно представить точкой на плоскости. Каждая тройка мутантов, такая, что Церебро находится строго внутри треугольника, образованного этими мутантами, добавляет одну единицу помех. Если Церебро находится вне или на границе такого треугольника, помехи не возникают. Таким образом, суммарное число помех — сумма помех по всем тройкам мутантов. Помогите Чарльзу найти это число.

Формат входных данных

В первой строке входного файла записано одно целое число n ($3 \leq n \leq 10^5$) — число мутантов.

Во второй строке входного файла записана пара целых чисел x, y ($-10^9 \leq x, y \leq 10^9$) — координаты Церебро.

В следующих n строках записаны пары чисел x_i, y_i ($-10^9 \leq x_i, y_i \leq 10^9$) — координаты мутантов.

Известно, что координаты всех точек в условии (и мутантов, и Церебро) различны.

Формат выходных данных

В единственной строке выходного файла выведите искомое число помех.

Пример

<code>triangles.in</code>	<code>triangles.out</code>
6 0 0 1 1 -1 1 1 -1 -1 -1 2 0 0 2	3
3 0 0 0 1 -1 -1 1 -1	1

Задача В. Геном

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

Боливар Траск изучает геном мутантов в своей лаборатории, чтобы сделать своих Стражей ещё более совершенными.

На геном мутанта влияют гены X , Y и Z . Они взаимодействуют между собой следующим образом: каждый раз под действием катализатора все гены делятся пополам, после этого половина сформированных генов X и половина Y взаимодействуют, чтобы получить ген Z . Вторая половина X и половина Z формируют ген Y . И оставшиеся половины Y и Z производят ген X . Таким образом, если геном состоял из x_0 , y_0 , z_0 соответствующих генов, то после одного применения катализатора он будет состоять из x_1 , y_1 , z_1 генов, где $x_1 = y_0 + z_0$, $y_1 = x_0 + z_0$ и $z_1 = x_0 + y_0$.

Траск исследует отношение генов X и Y . Для исследований ему нужно знать, на сколько гена X будет больше, чем гена Y после k применений катализатора. Ваша задача — написать программу, которая рассчитает $x_k - y_k$, где x_k и y_k — количество генов X и Y соответственно после k применений катализатора.

Формат входных данных

Во единственной строке входного файла дано четыре числа x , y , z и k ($0 \leq x, y, z, k \leq 10^{18}$) — первоначальное количество генов X , Y и Z соответственно, а также количество применений катализатора.

Формат выходных данных

В единственной строке выходного файла выведите, на сколько гена X больше, чем гена Y после k применений катализатора.

Пример

genom.in	genom.out
1 1 1 4	0
4 3 2 1	-1

Замечание

Во втором примере после одного применения катализатора будет 5 генов X и 6 генов Y . Соответственно, гена X на -1 больше, чем гена Y .

Задача С. Стеки

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

В давние времена, когда Магнето и Чарльз Ксавьер работали в одной команде, они любили играть в одну игру. Правила победы, очередность ходов и всё остальное их не интересовало — кто убедительнее докажет, что сейчас его ход, тот и ходит. Определение победителя тоже выливалось в философский спор, не имеющий отношения к игре, и переходящий к вопросу о роли мутантов в обществе.

Нас же интересует сам процесс игры. У игроков есть n изначально пустых стеков. Каждый игрок может сделать один из трёх ходов:

- **A l r x** — положить на вершину каждого из стеков с l по r число x ;
- **G x** — спросить у соперника число, лежащее на вершине x -го стека;
- **R i** — отменить i -й по порядку запрос добавления числа на стеки: удалить соответствующее число из каждого стека.

В один день Чарльз заметил, что Магнето стал отвечать слишком быстро. После непродолжительного наблюдения, он заметил, что хитрый Магнето написал программу, которая делает всё за него. Чарльзу это, конечно, не понравилось, но он не стал обвинять соперника в мошенничестве: это бы ещё больше увеличило напряжение в их отношениях. Вместо этого он решил сам автоматизировать процесс, чтобы не отставать от своего извечного соперника.

Напишите программу, которая сумеет играть в такую игру не хуже, чем Ксавьер и Магнето.

Формат входных данных

В первой строке даны числа n ($1 \leq n \leq 10^5$) — число стеков, и m ($1 \leq m \leq 10^5$) — число запросов.

В следующих m строках даны запросы в формате, описанном в условии. Гарантируется, что все номера стеков лежат в интервале от 1 до n , а числа, которые кладутся на стек, удовлетворяют ограничению ($1 \leq x \leq 10^9$). Для любого запроса на отмену добавления гарантируется, что соответствующий запрос добавления уже был исполнен, и ни один запрос не отменяется дважды.

Формат выходных данных

Для каждого запроса второго типа выведите одно число — число, находящееся на вершине соответствующего стека, или -1, если соответствующий стек пуст.

Пример

<code>stacks.in</code>	<code>stacks.out</code>
3 7	5
A 1 3 5	3
A 2 3 3	-1
G 1	3
G 2	
R 1	
G 1	
G 2	

Задача D. Последовательности

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

Настала решающая битва Людей Икс и Стражей. Профессор Икс понимает, что шансов выиграть в битве у них немного, но он старается использовать любую возможность. В армии Людей Икс есть $2 \cdot (n + 1)$ бойцов, и профессор знает, что для любого значения силы бойца x от 1 до $n + 1$ найдётся ровно два бойца с такой силой. Для успеха в бою Люди Икс должны выбрать $2 \cdot n$ бойцов и построиться в шеренгу так, что если один боец с силой x стоит в строю, то второй тоже находится в шеренге, и между ними стоит ровно $x - 1$ других бойцов. Два любых бойца с одинаковой силой должны остаться в резерве, чтобы прийти на помощь в подходящий момент.

Помогите Профессору Икс найти искомое построение или скажите, что его не существует.

Формат входных данных

В первой и единственной строке входного файла записано одно целое число n ($1 \leq n \leq 100$) — половина длины требуемого построения.

Формат выходных данных

Если искомого построения не существует, выведите одно число -1 .

Иначе, выведите $2 \cdot n$ чисел — искомое построение.

Пример

<code>sequence.in</code>	<code>sequence.out</code>
1	1 1
3	2 4 2 1 1 4

Задача Е. Экзамен

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

Чтобы попасть в школу для одарённых подростков Чарльза Ксавьера, нужно не только быть мутантом, но и уметь хорошо считать, чтобы уметь быстро оценить количество врагов и свои шансы на победу. Для этого все поступающие сдают экзамен по математике.

Джубили очень хочет попасть в школу и как раз сдает этот экзамен. Ей осталось решить последнее задание, но у неё возникли с ним большие трудности. Задание заключается в следующем: по данным целым неотрицательным числам l, r нужно найти количество таких целых чисел x на отрезке с l по r , что число $10 \cdot x$ — точный квадрат, а число $6 \cdot x$ — точный куб.

Помогите Джубили найти ответ, иначе её заветная мечта — попасть в школу Чарльза Ксавьера — не сбудется.

Число x называется точным квадратом, если существует такое целое число y , что $y^2 = x$. Аналогично определяется понятие «точный куб».

Формат входных данных

В первой и единственной строке входного файла даны два числа l, r ($0 \leq l \leq r \leq 10^{18}$) — отрезок, на котором надо найти ответ.

Формат выходных данных

В единственной строке выходного файла выведите одно число — ответ на задачу.

Пример

<code>exam.in</code>	<code>exam.out</code>
0 10	1
10000 100000	1

Задача F. Пароль

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

Это интерактивная задача.

Недавно Профессор Ксавьер с помощью своего Церебро смог узнать о готовящемся наступлении Магнето. Но этого мало, нужна полная информация о размере его войск, времени и месте первого удара. Взломав систему безопасности на компьютере Магнето, Профессор нашел на нём нужный файл. Но чтобы открыть этот файл, нужно знать пароль, который знает только сам Магнето.

К счастью, Профессор нашёл на компьютере программу, которая используется для восстановления пароля. Этой программе можно подавать на вход строку произвольной длины, тогда на выходе программа выдаст сообщение о том, входит ли введённая строка в строку-пароль как подстрока. Ксавьеру также удалось узнать длину пароля и то, что пароль хранится в бинарном виде — он состоит только из нулей и единиц.

Теперь он хочет сделать несколько запросов к программе и полностью узнать пароль. Но у него не очень много времени — довольно скоро Магнето поймет, что его компьютер взломан, и поменяет все пароли. Чарльз точно знает, что успеет сделать 1024 запроса к программе. Помогите ему — скажите, какие запросы надо делать.

Протокол взаимодействия

В самом начале программа жюри сообщает вашей программе натуральное число n ($1 \leq n \leq 1000$) — длину строки-пароля.

Дальше во время взаимодействия вашей программы с программой жюри несколько раз повторяются следующие действия:

- ваша программа сообщает программе жюри непустую строку, состоящую из символов 0 и 1, которую вы хотите проверить на принадлежность паролю
- программа жюри сообщает вашей программе:
 - «Success», если ваша строка и есть пароль
 - «0», если в пароле нет такой подстроки
 - «1», если в пароле есть такая подстрока
- в случае, если вы нашли пароль, вам необходимо завершить работу своей программы
- в противном случае, описанные действия начинают повторяться сначала

Гарантируется, что пароль состоит только символов 0 и 1, а также имеет длину ровно n .

Пример

stdin	stdout
3	000
0	001
0	010
0	011
0	100
0	101
Success	

Замечание

Для корректной работы программы после *каждой* операции вывода данных вам необходимо делать следующие операции:

- В языке Pascal: `flush(output);`
- В C/C++: `fflush(stdout);`
- В Java: `System.out.flush();`
- В Python: `sys.stdout.flush();`

Кроме этого, не забывайте после каждой выведенной строки ставить перевод строки.

Задача G. Росомаха и стеллаж

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

Зверю на день рождения подарили стеллаж с книгами. Стеллаж имеет форму дерева, где вершинам соответствуют полки с книгами, а рёбрам — соединения полок между собой. В каждой вершине находится несколько (возможно, ноль) книг.

Это не простое дерево, оно обладает следующим свойством: у каждой вершины не более двух детей, и количество книг в каждой вершине равно суммарному количеству книг в её детях. Также известно, что в листьях дерева не более одной книги.

Зверь был очень рад этому подарку, поэтому сразу повесил его на стену. Но Росомаха, проходя мимо, случайно зацепил дерево своими адамантиевыми когтями, и оно упало на пол. Росомаха собрал все книги, которые нашёл, но некоторых явно не хватало, поэтому он решил взять несколько из своей коллекции и добавить их незаметно на полки, чтобы получившееся дерево обладало изначальным свойством. Но после того, как он это сделал, стало только хуже. Теперь надо всё исправить и всё-таки вернуть дереву его изначальное свойство. Зверь скоро вернётся, поэтому Росомахе надо действовать как можно быстрее.

Немного подумав, он понял, что не всегда выгодно только добавлять книги на полки, иногда выгодно их убирать оттуда. За одну секунду Росомаха может либо положить на какую-то полку книгу, либо забрать её с какой-то из полок. Ваша же задача заключается в следующем: помогите Росомахе найти наименьшее время, за которое он сможет получить такими операциями дерево, обладающее тем же свойством, что и дерево, подаренное изначально Зверю на день рождения.

Формат входных данных

В первой строке входного файла дано число n — количество вершин в дереве ($1 \leq n \leq 5000$). Во второй строке входного файла даны n целых чисел a_i ($0 \leq a_i \leq 5000$) — количество книг на i -й полке. В i -й из следующих $n - 1$ строк даны два числа a, b ($1 \leq a, b \leq n$) — ребро дерева.

Корень дерева находится в вершине с номером 1.

Можно считать, что в коллекции Росомахи бесконечное число книг.

Формат выходных данных

В единственной строке выходного файла выведите минимальное количество секунд, которое понадобится Росомахе, чтобы скрыть свою оплошность.

Пример

tree.in	tree.out
2 1 0 1 2	1
5 5 1 3 0 1 1 2 1 3 3 4 3 5	4

Задача Н. Забег

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

Команда Людей Икс решила выяснить, кто из них самый быстрый бегун. Для этого они организовали трассу, состоящую из контрольных точек. Трасса представлена в виде неориентированного графа с n вершинами, обозначающими контрольные точки, и m рёбрами. Для каждого ребра известна его длина.

По правилам забега каждый участник должен пробежать k контрольных точек так, чтобы последовательность вершин, соответствующих им, представляла простой путь. Однако Профессор Икс выяснил, что в системе, регистрирующей участника на контрольной точке, есть ошибка. Для каждого участника система запоминает только последнюю посещённую контрольную точку. Профессор Икс решил схитрить — он решил составить не кратчайший простой путь, а такой маршрут минимальной длины из k контрольных точек, что каждые две последовательные точки различны, и между ними есть ребро.

Формат входных данных

В первой строке входного файла дано два числа n и m ($1 \leq n \leq 100000, 1 \leq m \leq 200000$) — количество вершин и количество ребер в графе. В следующей строке дано целое число k ($2 \leq k \leq 100000$) — длина маршрута. В i -й из следующих m строк дана тройка чисел a_i, b_i, w_i ($1 \leq a_i, b_i \leq n, 1 \leq w_i \leq 10000$) — описание i -го ребра. Гарантируется, что в графе нет петель.

Формат выходных данных

Выведите минимальную длину маршрута.

Пример

run.in	run.out
3 3 3 1 2 1 2 3 1 1 3 2	2

Задача I. Нечетное или четное?

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

Несмотря на то, что Джубили является полноправным членом команды Людей Икс, она всё же школьница и обязана делать домашнее задание.

Джина Грей задала ей на дом непростое задание. Нужно определить чётность огромного арифметического выражения с двумя неизвестными x и y , если известна чётность этих переменных.

К сожалению, обязанности супергероя не оставили Джубили ни сил, ни времени. Она просит вас написать программу, которая определит чётность данного ей выражения.

Формат входных данных

В самой первой строке входного файла дана строка s ($1 \leq |s| \leq 10^6$) — выражение, содержащее в себе цифры, знаки $+$, $-$, $*$, а также переменные x и y . Во второй и третьей строках входного файла содержится описание чётности переменных x и y в следующем формате: `Odd`, если значение переменной нечетно и `Even` в противном случае. Гарантируется, что во входной строке нет двух подряд стоящих знаков арифметических операций, что строка начинается с цифры или переменной, все числа без ведущих нулей, а также что слева и справа от переменной стоят арифметические знаки, начало или конец строки.

Формат выходных данных

В единственной строке выходного файла выведите `Odd`, если выражение нечётно, и `Even`, если нет.

Пример

<code>oddoreven.in</code>	<code>oddoreven.out</code>
<code>1+23+x+456*y-7</code> <code>Odd</code> <code>Even</code>	<code>Even</code>
<code>x*x*x*y-x</code> <code>Odd</code> <code>Even</code>	<code>Odd</code>

Задача J. Игра в домино

Имя входного файла: dominoes.in
Имя выходного файла: dominoes.out
Ограничение по времени: 2 секунды
Ограничение по памяти: 256 мегабайт

Собравшись в очередной раз, чтобы поиграть в домино, Люди Икс осознали, что игра им порядком надоела. Они решили придумать что-нибудь новое. Как обычно, с гениальной идеей выступил Гамбит. Он предложил следующую незамысловатую игру: по данному набору доминошек надо уметь определять длину самой длинной цепочки.

Каждая доминошка представляет собой пару чисел a, b — количество точек на двух половинах доминошки. Цепочкой называется последовательность доминошек, которую можно выложить в линию так, что для любых двух соседних доминошек с номерами $i, i+1$ в этой линии верно следующее: $b_i = a_{i+1}$. Доминошки нельзя поворачивать и переворачивать.

Люди Икс пока не научились оптимально играть в эту игру, поэтому обратились к вам за помощью.

Формат входных данных

В первой строке входного файла дано число n ($1 \leq n \leq 100000$) — количество доминошек. В следующих n строках даны пары чисел a_i, b_i ($0 \leq b_i \leq a_i \leq 10^9$) — описание доминошек.

Формат выходных данных

В единственной строке выходного файла выведите одно число — максимальную длину цепочки из доминошек.

Пример

dominoes.in	dominoes.out
7 2 6 5 6 2 5 2 2 6 8 2 2 0 2	6