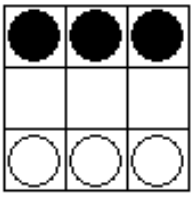
Содержание

В скобках указано число баллов за задачу.

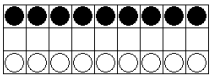
Интересные задачи

Задача 17A (3). Игра в пешки [0.1 sec, 256 mb]

В свободное время Дед Мороз Петрович и Дед Мороз Егорыч играют в следующую игру. На доске 3×3 пешки расставляются следующим образом:



Пешки ходят и бьют по обычным шахматным правилам, к которым добавляется ещё одно: бить обязательно. Проигрывает тот, кто не может сделать ход. Первыми ходят белые. В течение 100 лет Петрович играл белыми и всегда выигрывал. Однажды Егорычу это надоело, и он принёс доску 3×5. Но и теперь он, играя чёрными, до сих пор проигрывает. «Как так?» — подумал Егорыч и решил купить доску $3 \times N$:



Вот тут Петровичу надо подумать, какими играть, чтобы выиграть. Вам нужно помочь ему это сделать.

Формат входных данных

Целое число N ($1 \leqslant N \leqslant 10^9$).

Формат выходных данных

«White», если Петровичу надо играть белыми, и «Black», если надо играть чёрными. Егорыч и Петрович каждый раз ходят по наилучшей для себя стратегии.

Пример

stdin	stdout
3	White
4	Black
5	White

Подсказка по решению

Решите задачу для малых $N \leqslant 1000$. Вспомните, как на практике мы научились решение для малых N применять для больших N?

Задача 17B (3). Шары и урны [1.5 sec, 256 mb]

Рассмотрим n различных шаров и n различных урн, стоящих в ряд. Изначально каждая урна содержит ровно один из шаров.

Для перемещения шаров есть специальное устройство. Пользоваться им очень просто. Сначала необходимо выбрать непрерывный отрезок урн. После этого устройство поднимает шары изо всех урн этого отрезка. Наконец, необходимо выбрать другой непрерывный отрезок урн такой же длины, после чего устройство перемещает поднятые шары в этот отрезок. Каждая урна может вместить любое количество шаров.

По данной последовательности перемещений шаров при помощи устройства выясните для каждого шара, в какой урне он будет находиться после всех перемещений.

Формат входных данных

Первая строка ввода содержит два целых числа n и m — количество урн и количество перемещений ($1 \leqslant n \leqslant 100\,000$, $1 \leqslant m \leqslant 50\,000$). Каждая из следующих m строк содержит по три числа count_i, from_i и to_i, которые означают, что устройство одновременно перемещает все шары из урны from_i в урну to_i, все шары из урны from_i+1 в урну to_i+1, ..., все шары из урны $\mathrm{from}_i+\mathrm{count}_i-1$ в урну $\mathrm{to}_i+\mathrm{count}_i-1$ ($1 \leqslant \mathrm{count}_i, \mathrm{from}_i, \mathrm{to}_i \leqslant n$, $\max(\mathrm{from}_i, \mathrm{to}_i)+\mathrm{count}_i \leqslant n+1$).

Формат выходных данных

Выведите ровно n чисел от 1 до n: конечные позиции всех шаров. Первое число должно задавать конечную позицию шара, который изначально был в первой урне, второе число — позицию шара из второй урны и так далее.

Примеры

stdin	stdout
2 3 1 1 2 1 2 1 1 2 1	1 1
10 3 1 9 2 3 7 3 8 3 1	1 2 1 2 3 4 1 2 2 8

Подсказка по решению

Это задача про персистентное декартово дерево. Задача offline, поэтому представим себе процесс в обратном порядке (сделаем операции с конца). Чтобы получить OK, вам нужен будет garbage collector. Моё решение использует 0.39 секунд и 12 мегабайт.

Задача 17C (3). Intercity Express [3.0 sec, 256 mb]

Андрей разрабатывает систему для продажи железнодорожных билетов. Он собирается протестировать ее на Междугородней Экспресс линии, которая соединяет два больших города и имеет $n-2$ промежуточных станций, то есть в итоге есть n станций, пронумерованных от 1 до n.

В Междугороднем Экспресс поезде есть s мест, пронумерованных с 1 до s. В тестирующем режиме система имеет доступ к базе данных, содержащей проданные билеты в направлении от станции 1 до станции n и должна отвечать на вопросы, можно ли продать билет от станции a до станции b, и если да, нужно найти минимальный номер места, которое свободно на протяжении всего пути между a и b.

Изначально система имеет только доступ на чтение, то есть даже если есть свободное место, она должна сообщить об этом, но не должна изменять данные.

Помогите Андрею протестировать его систему написанием программы, которые будет находить ответы на вопросы.

Формат входных данных

Первая строка содержит число n — количество станций, s — количество мест и m — количество уже проданных билетов ($2 \leqslant n \leqslant 10^9$, $1 \leqslant s \leqslant 100\,000$, $0 \leqslant m \leqslant 100\,000$).

В следующих m строках описаны билеты, описание каждого билета состоит из трех чисел: c_i, a_i и b_i — номер места, которое занимает владелец билета, номер станции, с которой продан билет и номер станции, до которой продан билет ($1 \leqslant c_i \leqslant s$, $1 \leqslant a_i < b_i \leqslant n$).

Следующие строки содержат число q — количество запросов ($1 \leqslant q \leqslant 100\,000$). Специальное значение p должно поддерживаться в течение считывания запросов. Изначально $p = 0$.

Следующие $2q$ строк описывают запросы. Каждый запрос описывается двумя числами: x_i и y_i ($x_i \leqslant y_i$).

Чтобы получить города a и b между которыми нужно проверить наличие места, используется следующая формула:

$a = x_i + p$, $b = y_i + p$. Ответ на запрос — число 0, если нет места на каждом отрезке между a и b, или минимальный номер свободного места.

После ответа на запрос, надо приравнять число p полученному ответу на запрос.

Формат выходных данных

Для каждого запроса выведите ответ на него.

Примеры

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

Обратите внимание, что запросы выглядят так:
$(1, 2)$, $(2, 3)$, $(3, 4)$, $(4, 5)$, $(1, 3)$, $(2, 4)$, $(3, 5)$, $(1, 4)$, $(2, 5)$, $(1, 5)$.

Подсказка по решению

Как решать? Представьте себе сканирующую прямую, движущуюся слева направо.
Представьте, что пришли в правый конец r_i отрезка-запроса.
Храните массив – для каждого места «последний момент времени, когда оно стало свободно».
Тогда нужно найти «минимальное место, освободившееся не позднее l_i».
Это запрос к одномерному дереву отрезков. Да, нужно сохранить все версии.

Задача 17D (3). Опекуны карнотавров [0.6 sec, 256 mb]

Карнотавры очень внимательно относятся к заботе о своем потомстве. У каждого динозавра обязательно есть старший динозавр, который его опекает. В случае, если опекуна съедают (к сожалению, в юрский период такое не было редкостью), забота о его подопечных ложится на плечи того, кто опекал съеденного динамозавра. Карнотавры — смертоносные хищники, поэтому их обычаи строго запрещают им драться между собой. Если у них возникает какой-то конфликт, то, чтобы решить его, они обращаются к кому-то из старших, которому доверяют, а доверяют они только тем, кто является их опекуном или опекуном их опекуна и так далее (назовем таких динозавров суперопекунами). Поэтому для того, чтобы решить спор двух карнотавров, нужно найти такого динозавра, который является суперопекуном для них обоих. Разумеется, беспокоить старших по пустякам не стоит, поэтому спорщики стараются найти самого младшего из динозавров, который удовлетворяет этому условию. Если у динозавра возник конфликт с его суперопекуном, то этот суперопекун сам решит проблему. Если у динозавра нелады с самим собой, он должен разобраться с этим самостоятельно, не беспокоя старших. Помогите динозаврам разрешить их споры.

Формат входных данных

Во входном файле записано число M, обозначающее количество запросов ($1 \le M \le 200\,000$). Далее на отдельных строках следуют M запросов, обозначающих следующие события:

- `+ v` — родился новый динозавр и опекунство над ним взял динозавр с номером v. Родившемуся динозавру нужно присвоить наименьший натуральный номер, который до этого еще никогда не встречался.
- `- v` — динозавра номер v съели.
- `? u v` — у динозавров с номерами u и v возник конфликт и вам надо найти им третейского судью.

Изначально есть один прадинозавр номер 1; гарантируется, что он никогда не будет съеден.

Формат выходных данных

Для каждого запроса типа «?» в выходной файл нужно вывести на отдельной строке одно число — номер самого молодого динозавра, который может выступить в роли третейского судьи.

Примеры

stdin	stdout
11 + 1 + 1 + 2 ? 2 3 ? 1 3 ? 2 4 + 4 + 4 - 4 ? 5 6 ? 5 5	1 1 2 2 5

Подсказка по решению

Решите сперва задачу в случае, когда дерево не меняется и никто не умирает.
Что делать с добавлениями? На теорпрактике обсуждалось.
Что делать с удалением? Подняться ещё выше вверх... СНМ подскажет, до куда.

Задача 17E (3). Дерево [1.5 sec, 256 mb]

Рассмотрим корневое дерево. Изначально дерево состоит из одного лишь корня. На сыновьях каждой вершины определён порядок слева направо. Допустимые операции с деревом таковы:

- Добавить в дерево лист.
- Удалить из дерева лист.
- Найти количество вершин на пути между двумя листьями.
- Найти количество вершин «под путём» между двумя листьями.

Множество вершин, лежащих «под путём» между листьями u и v, определяется следующим образом. Рассмотрим путь $u = w_0 - w_1 - \ldots - w_k = v$ между ними. Выделим в нём ту вершину w_c, в которой оба ребра пути идут в её сыновей. Пусть для определённости левое из этих рёбер приближает нас к вершине u, а правое — к вершине v. Тогда лежащими «под путём» объявляются следующие вершины:

- Все сыновья w_c, лежащие между w_{c-1} и w_{c+1}, а также все вершины их поддеревьев;
- Для $i = 1, 2, \ldots, c-1$ — все сыновья w_i, лежащие правее сына w_{i-1}, а также все вершины их поддеревьев;
- Для $i = c+1, c+2, \ldots, k-1$ — все сыновья w_i, лежащие левее сына w_{i+1}, а также все вершины их поддеревьев.

Напишите программу, которая по последовательности запросов перестраивает дерево согласно запросам на добавление и удаление вершин, а также вычисляет ответы на запросы о количестве вершин на пути и «под путём».

Формат входных данных

В первой строке ввода содержится одно целое число n — количество запросов ($0 \leqslant n \leqslant 300\,000$). Каждая из следующих n строк описывает один запрос. Возможные виды запросов таковы:

`l` x	добавить новый лист как самого левого сына вершины x
`r` x	добавить новый лист как самого правого сына вершины x
`a` x y	добавить новый лист как сына вершины x, находящегося непосредственно справа от вершины y; все сыновья x, находившиеся до этого справа от y, после добавления оказываются правее новой вершины; гарантируется, что y является сыном x
`d` x	удалить вершину x; гарантируется, что в этот момент вершина x не удалена и является листом
`p` x y	найти количество вершин на пути между x и y, включая сами эти вершины; гарантируется, что x и y являются листьями
`q` x y	найти количество вершин «под путём» между x и y, включая сами эти вершины; гарантируется, что x и y являются листьями

Добавляемые вершины нумеруются с единицы в том порядке, в котором они добавляются в запросах. Корень дерева имеет номер 0 и листом ни в какой момент времени не считается.

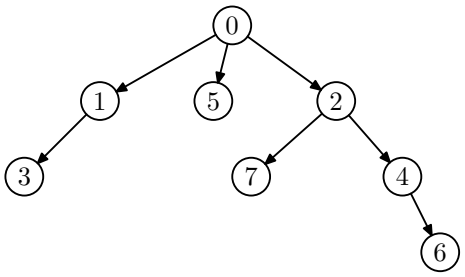
Формат выходных данных

Выполняя запросы по порядку, на каждый запрос вида или выведите ответ на него на отдельной строке.

Пример

stdin	stdout
10 l 0 r 0 l 1 r 2 a 0 1 r 4 p 6 5 l 2 q 3 6 d 7	5 2

Иллюстрация



На рисунке показано дерево до выполнения последнего запроса — удаления вершины 7.
Путь между вершинами 6 и 5 содержит пять вершин: 6 – 4 – 2 – 0 – 5.
«Под путём» между вершинами 3 и 6 находится две вершины — 5 и 7.

Подсказка по решению

Это задача про Эйлеров обход и LCA. Длина пути выражается через высоты вершин и LCA. Число путей под путём через позиции этих вершин в Эйлеровом обходе и длину пути.

Запросы даны в оффлайн, значит, все запросы «добавление новых вершин» можно сделать заранее.

Не пробовали ещё дерево Фенвика? Самое время.

Задача 17F (3). Count Online [3.2 sec, 256 mb]

Вам дано множество точек на плоскости.

Нужно уметь отвечать на два типа запросов:

- `?` x_1 y_1 x_2 y_2 — сказать, сколько точек лежит в прямоугольнике $[x_1..x_2] \times [y_1..y_2]$. Точки на границе и в углах тоже считаются. $x_1 \leqslant x_2$, $y_1 \leqslant y_2$.
- `+` x y — добавить в множество точку `(x + res % 100, y + res % 101)`. Где `res` — ответ на последний запрос вида `?`, а `%` — операция взятия по модулю.

Формат входных данных

Число точек N ($1 \leqslant N \leqslant 50\,000$). Далее N точек. Число запросов Q ($1 \leqslant Q \leqslant 100\,000$). Далее Q запросов. Все координаты от 0 до 10^9.

Формат выходных данных

Для каждого запроса `GET` одно целое число — количество точек внутри прямоугольника.

Пример

stdin	stdout
5 0 0 1 0 0 1 1 1 1 1 9 ? 0 1 1 2 + 1 2 + 2 2 ? 1 0 2 2 ? 0 0 0 0 + 3 3 ? 3 3 3 3 ? 4 3 4 3 ? 4 4 5 5	3 3 1 0 0 3

Замечание

На самом деле добавлялись точки `(4, 5)`, `(5, 5)`, `(4, 4)`.

Подсказка по решению

Представьте, что точки не добавляются? Тогда мы умеем эту задачу решать двумя способами – за $\mathcal{O}(\log^2)$ деревом отрезков сортированных массивов и за $\mathcal{O}(\log n)$ персистентным деревом отрезков.

Чтобы поддержать добавление, самое простое – корневая по запросам (отложенные операции). Не забудьте пошевелить константу, минимизируя время.

Задача 17G (2). Island. Островные государства [0.2 sec, 256 mb]

Суровые феодальные времена переживала некогда великая островная страна Байтландия. За главенство над всем островом борются два самых сильных барона. Таким образом, каждый город страны контролируется одним из правителей. Как водится издревле, некоторые из городов соединены двусторонними дорогами. Бароны очень не любят друг друга и стараются делать как можно больше пакостей. В частности, теперь для того чтобы пройти по дороге, соединяющей города различных правителей, надо заплатить пошлину — один байтландский рубль.

Программист Вася живет в городе номер 1. С наступлением лета он собирается съездить в город N на Всебайтландское сборище программистов. Разумеется, он хочет затратить при этом как можно меньше денег и помочь ему здесь, как обычно, предлагается Вам.

Формат входных данных

В первой строке входного файла записано два числа N и M ($1 \leqslant N, M \leqslant 100\,000$) — количество городов и количество дорог соответсвенно.

В следующий строке содержится информация о городах — N чисел 1 или 2 — какому из баронов принадлежит соответствующий город.

В последних M строках записаны пары $1 \leqslant a, b \leqslant N$, $a \neq b$. Каждая пара означает наличие дороги из города a в город b. По дорогам Байтландии можно двигаться в любом направлении.

Формат выходных данных

Если искомого пути не существует, выведите единственное слово `impossible`. В противном случае в первой строке напишите минимальную стоимость и количество посещенных городов, а во вторую выведите эти города в порядке посещения. Если минимальных путей несколько, выведите любой.

Пример

stdin	stdout
7 8 1 1 1 1 2 2 1 1 2 2 5 2 3 5 4 4 3 4 7 1 6 6 7	0 5 1 2 3 4 7
5 5 1 2 1 1 2 1 2 2 3 3 5 1 4 4 5	1 3 1 4 5

Подсказка по решению

Вы умеете решать эту задачу за $\mathcal{O}(N + M)$ поиском в ширину.

Задача 17H (4). Республика [1 sec, 256 mb]

Президент одной из новоиспеченных «демократий» на Ближнем Востоке после своего избрания взял курс на экономическое и социальное развитие страны. К сожалению, первая проблема, с которой столкнулся президент — плачевное состояние дорожной системы в стране. В силу того, что дороги однонаправленные, далеко не всегда можно от одного города добраться до другого, а это мешает развитию коммуникаций между городами и экономики вообще.

Вам поручено разработать план модернизации дорожной системы таким образом, чтобы от каждого города можно было добраться до каждого. План состоит в добавлении нескольких новых дорог. В силу того, что демократия в стране молодая, денег не хватает, поэтому количество добавленных дорог должно быть минимально возможным.

Формат входных данных

Первая строка входного файла содержит числа n и m ($1 \leqslant n \leqslant 10^5$), ($1 \leqslant m \leqslant 10^5$). Далее следуют m строк по два числа p и q, обозначающие, что из города p в город q ведет дорога.

Формат выходных данных

Выведите число k — количество дорог, которые требуется добавить. Далее выведите k строк по два числа — описания добавленных вами дорог.

Примеры

stdin	stdout
3 2 1 2 1 3	2 2 1 3 1

Подсказка по решению

Это сложная задача. Она была у нас в допах на теорпрактике. Начинается всё с конденсации ксс. Затем нужно выделить стоки и истоки и соединить их по циклу в правильном порядке.

Задача 17I (3). Конкатенация скобочных последовательностей [0.4 sec, 256 mb]

Даны n скобочных последовательностей из круглых скобок. Выбрать некоторое подмножество, выписать в произвольном порядке так, чтобы

- Результат являлся правильной скобочной последовательностью.
- Длина результата была максимально возможной.

Формат входных данных

На первой строке n ($1 \leqslant n \leqslant 1\,000$). Следующие n строк содержат непустые скобочные последовательности, суммарная длина не более $10\,000$.

Формат выходных данных

В первой строке выведите через пробел числа l и k, где l — максимальная длина правильной скобочной последовательности, а k — количество кусков, из которых она состоит. Во второй строке через пробел выведите k номеров использованных кусков в том порядке, в котором их следует выписывать. Куски занумерованы числами от 1 до n в том порядке, в котором они даны на входе. Если возможных ответов несколько, выведите любой.

Примеры

stdin	stdout
4 ((((( ))	4 3 1 3 4
3 () (())	6 3 2 3 1

Подсказка по решению

Мы разбирали что-то очень похожее в теме «жадности».

Задача 17J (4). Учимся красить [1 sec, 256 mb]

Дан случайный неориентированный граф G из n вершин и m ребер. Ваша задача — покрасить его вершины в минимальное количество цветов таким образом, чтобы смежные вершины были покрашены в разные цвета.

Формат входных данных

На первой строке число вершин n ($1 \leqslant n \leqslant 70$) и число ребер $m \geqslant 1$.
Следующие m строк содержат пары чисел от 1 до n — ребра графа.
В графе нет ни петель, ни кратных ребер.

Формат выходных данных

На первой строке выведите k — минимальное количество цветов. На следующей строке n целых чисел от 1 до k — цвета вершин. Если оптимальных раскрасок несколько, выведите любую.

Примеры

stdin	stdout
5 8 5 4 3 5 1 5 1 3 2 3 1 4 5 2 3 4	4 1 1 2 3 4

Подсказка по решению

Помните, у нас в курсе была неплохая жадность для решения этой задачи? Начнём с неё. Кстати, в неё можно добавить рандомчика, и запустить несколько раз.

Как теперь понять, можно ли уменьшить ответ ещё на 1? Для новой задачи «покрасить ровно в k цветов» тоже есть неплохая жадность – красить каждый раз вершину, которую сложнее всего покрасить. И сюда тоже можно random walk добавить.

Как это тестить локально? Рандомный граф вы сгенерить можете, да. Как проверить, что ответ корректный? Любой приличный random walk можно пускать в 10 раз дольше и assert-ить, что ответ не изменился.