Содержание

Разминка и динамика 2

Задача A1. Различные подпоследовательности [0.1 sec, 256 mb] 2

Задача A2. Идеальный путь [0.4 sec, 256 mb] 3

Задача A3. Рюкзак 36-го размера [0.4 sec, 256 mb] 4

Задача A4. Капитан Оптимальность [2 sec (0.3 sec), 256 mb] 5

Задача A5. Сервера [1.5 sec, 256 mb] 6

Задача A6. Две лесопилки [0.1 sec, 256 mb] 7

Задача A7. Мультипликативная сумма [0.4 sec, 256 mb] 8

Задача A8. Неприводимые многочлены [0.1 sec, 256 mb] 9

Задача A9. Японский кроссворд [0.1 sec, 256 mb] 10

Структуры данных – 1 11

Задача B1. Приказы [1, 256] 11

Задача B2. Фыр-Фыр-Фенвик [2, 256] 12

Задача B3. Отрезки, пересекающие прямоугольник [3, 256] 13

Задача B4. Добавление и удаление точек [2, 256] 14

Задача B5. Галактический апокалипсис [1, 256] 15

Задача B6. Persistent Array On [0.7, 256] 16

Динамика по профилю 17

Задача C1. Длинные домино [0.1, 256] 17

Задача C2. Шах-мат-блок [0.2, 256] 18

Задача C3. Гамильтоновые гриды [4, 756] 19

Задача C4. Путешествие лучника [0.3, 256] 20

Структуры данных – 2 21

Задача D1. Дерево [1, 256] 21

Задача D2. Строки в дереве [1, 256] 22

Задача D3. Inspector is Coming [1, 256] 23

Задача D4. MeX на пути дерева [1, 256] 24

Хещи 25

Задача E1. Массивы-палиндромы [0.1, 256] 25

Задача E2. Подпалиндромы [1, 256] 26

Пихаем квадраты 27

Задача F1. Переворачивания [2, 4] 27

Обратите внимание, входные данные лежат в **стандартном потоке ввода** (он же stdin), вывести ответ нужно в **стандартный поток вывода** (он же stdout).

Обратите внимание на `GNU C++` компиляторы с суффиксом `inc`.
Под ним можно пользоваться дополнительной библиотекой (`optimization.h`).
То есть, использовать быстрый ввод-вывод: **пример про числа и строки**.
И быструю аллокацию памяти (ускоряет vector-set-map-весь-STL): пример.

Для тех, кто хочет разобраться, как всё это работает.
Короткая версия быстрого ввода-вывода (тык) и короткая версия аллокатора (тык).

Разминка и динамика

Задача A1. Различные подпоследовательности [0.1 sec, 256 mb]

Дана последовательность целых чисел длины n. Нужно найти количество различных непустых подпоследовательностей.

Формат входных данных

На первой строке n. На второй строке n целых чисел. $1 \leqslant n \leqslant 3 \cdot 10^5$, $1 \leqslant a_i \leqslant 3 \cdot 10^5$.

Формат выходных данных

Количество различных подпоследовательностей по модулю $10^9 + 7$.

Пример

stdin	stdout
3 1 1 2	5
35 1 2 3 4 5 6 7 8 9 10 1 2 3 4 5 6 7 8 9 10 1 2 3 4 5 6 7 8 9 10 1 2 3 4 5	941167856

Задача A2. Идеальный путь [0.4 sec, 256 mb]

Есть лабиринт развлечений из n комнат и m цветных переходов между комнатами. Участник изначально попадает в комнату 1, цель — добраться до комнаты n. Несколько участников одновременно стартуют из комнаты 1. Каждый участник пройдет некоторым путем из комнаты 1 в комнату n, записывая цвета переходов, по которым прошел. Выиграет участник с кратчайшей последовательностью, а если таких несколько, с лексикографически меньшей среди кратчайших. Эндрю тоже участвует и очень хочеть выиграть. Помогите ему, найдите кратчайшую, а из таких лексикографически минимальную последовательность переходов из 1 в n.

Замечание

Последовательность $(a_1, a_2, \ldots, a_k)$ лексикографически меньше последовательности $(b_1, b_2, \ldots, b_k)$ если существует i такое, что $a_i < b_i$ и $a_j = b_j$ для всех $j < i$.

Формат входных данных

Первая строка содержит целые числа n и m — количество комнат и переходов соответственно ($2 \leqslant n \leqslant 100\,000$, $1 \leqslant m \leqslant 200\,000$). Следующие m строк содержат описания переходов. Каждый переход задается тремя целыми числами: a_i, b_i и c_i — номера комнат, которые он соединяет, и цвет перехода ($1 \leqslant a_i, b_i \leqslant n$, $1 \leqslant c_i \leqslant 10^9$). Каждый переход может быть использован в обоих направлениях. Две комнаты могут быть соединены несколькими переходами, также может быть переход из комнаты в саму себя. Гарантируется, что из комнаты 1 можно как-нибудь попасть в комнату n.

Формат выходных данных

На первой строке выведите k — длину кратчайшей последовательности переходов из комнаты 1 в комнату n. На следующей строке k чисел — цвета переходов в порядке их прохождения.

Пример

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

Задача A3. Рюкзак 36-го размера [0.4 sec, 256 mb]

Даны n предметов с положительными целыми весами w_i. И рюкзак веса W. Стоимость предмета равна его весу. Ваша задача – унести предметы максимальной суммарной стоимости.

Формат входных данных

На первой строке n ($2 \leqslant n \leqslant 36$) и W ($1 \leqslant W \leqslant 10^9$).
Следующая строка содержит веса предматов — n целых от 1 до 10^9.

Формат выходных данных

Выведите одно число — максимальную суммарную стоимость предметов в рюкзаке.

Примеры

stdin	stdout
3 10 6 5 3	9

Подсказка по решению

Восстанавливать ответ не нужно.
Это meet-in-the-middle.
Вам пригодится сортировка и «два указателя» для массивов длины $2^{n/2}$.
На C++ не забудьте использовать 64-битный тип.

Задача A4. Капитан Оптимальность [2 sec (0.3 sec), 256 mb]

Имеется калькулятор, который умеет выполнять с текущим числом на экране следующие операции:

- Умножить текущее число на A.
- Умножить текущее число на B.
- Прибавить к текущему числу единицу.
- Прибавить к текущему числу число C.

Здесь A, B и C — некоторые целочисленные константы. Определите, какое наименьшее количество операций требуется, чтобы получить из числа 1 число n.

Формат входных данных

Во вводе задано несколько строк. Каждая строка содержит очередной тестовый случай, задаваемый четырьмя целыми числами n, A, B и C ($1 \leqslant n \leqslant 10^{19}$, $2 \leqslant A, B \leqslant 10$, $1 \leqslant C \leqslant 10$). Входные данные содержат не более 100 тестов.

Формат выходных данных

Для каждого тестового случая выведите $k+1$, где k — количество сделанных операций. За ним выведите все $k+1$ промежуточных чисел, включая 1 и n. Если последовательностей из k операций несколько, можно вывести любую из них.

Пример

stdin	stdout
1 7 8 9 5 2 3 9 576 8 9 7 100000 2 3 1	1 1 4 1 2 4 5 4 1 8 64 576 19 1 2 6 18 19 38 76 77 231 693 694 2082 2083 6249 6250 12500 25000 50000 100000

Замечание

У этой задаче есть две версии в тестирующей системе.
Отличаются они лишь ограничением по времени.

Задача A5. Сервера [1.5 sec, 256 mb]

Спонсор этой задачи — всесибирская олимпиада 2009. Новосибирск – город мечты!

Компьютерная сеть в некотором доме строилась по принципу присоединения нового компьютера к последнему из уже подключенных. Никакие два компьютера, будучи подключенными в сеть, между собой дополнительно никак не связывались. Таким образом, в сеть были объединены последовательно N компьютеров. Соседи обменивались информацией между собой, но в какой-то момент поняли, что им нужны прокси-серверы. Компьютерное сообщество дома решило установить прокси-серверы ровно на K компьютеров. Осталось только решить, какие именно компьютеры выбрать для этой цели. Главным критерием является ежемесячная стоимость обслуживания серверами всех компьютеров.

Для каждого компьютера установлен тариф его обслуживания, выраженный в рублях за метр провода. Стоимость обслуживания одного компьютера каким-то сервером равна тарифу компьютера, умноженному на суммарную длину провода от этого компьютера до сервера, которым он обслуживается.

Ваша задача написать программу, которая выберет такие K компьютеров, чтобы установить на них прокси-серверы, что общие затраты на обслуживание всех компьютеров были бы минимальными

Формат входных данных

В первой строке входного файла записано два целых числа N и K — количество компьютеров в сети и количество прокси-серверов, которые нужно установить ($1 \le K \le N \le 2000$).

Все компьютеры в сети пронумерованы числами от 1 до N по порядку подключения.

Во второй строке записано одно целое число T_1 — тариф обслуживания первого компьютера.

В следующих $N-1$ строках записано через пробел по два целых неотрицательных числа L_i, T_i — информация об остальных компьютерах в сети по порядку номеров. L_i — длина провода, соединяющего $i-$ компьютер с соседним с меньшим номером, T_i — тариф обслуживания данного компьютера ($2 \le i \le N$). Все L_i и T_i от 0 до 10^6.

Формат выходных данных

В первую строку выходного файла необходимо вывести одно целое число — минимальную стоимость обслуживания всех компьютеров всеми серверами. Во второй строке должны быть записаны через пробел K номеров компьютеров, на которые необходимо установить серверы. При существовании нескольких вариантов размещения разрешается вывести любой.

Пример

stdin	stdout
3 1 10 2 2 3 3	19 1
3 2 10 2 2 3 3	4 1 3

Задача A6. Две лесопилки [0.1 sec, 256 mb]

От вершины до подножья холма растет N старых деревьев. Районная администрация решила в санитарных целях срубить эти деревья, а чтобы снизить стоимость мероприятия перевезти все древесину на лесопилки. Деревья могут быть перевезены только в одном направлении – вниз. У подножья холма находится лесопилка, а также две дополнительные лесопилки могут быть построены на холме вдоль дороги. Вам предстоит определить, где наиболее выгодно построить эти лесопилки, чтобы минимизировать стоимость транспортировки древесины. Перевозка 1 килограмма древесины на 1 метр стоит 1 копейку.

Формат входных данных

Первая строка входного файла содержит натуральное число N – количество деревьев ($1 \leqslant N \leqslant 20\,000$). Деревья занумерованы от 1 до N начиная с вершины холма. Следующие N линий содержат по два целых числа w_i и d_i ($1 \leqslant w_i, d_i \leqslant 10\,000$) – вес дерева номер i и расстояние между деревьями i и i+1. Последнее из этих чисел (d_n) задает расстояние от нижнего дерева до лесопилки.

Формат выходных данных

Выведите единственное число – минимальную стоимость сплава деревьев вниз по реке.

Пример

stdin	stdout
9 1 2 2 1 3 3 1 1 3 2 1 6 2 1 1 2 1 1	26

Пояснение к примеру

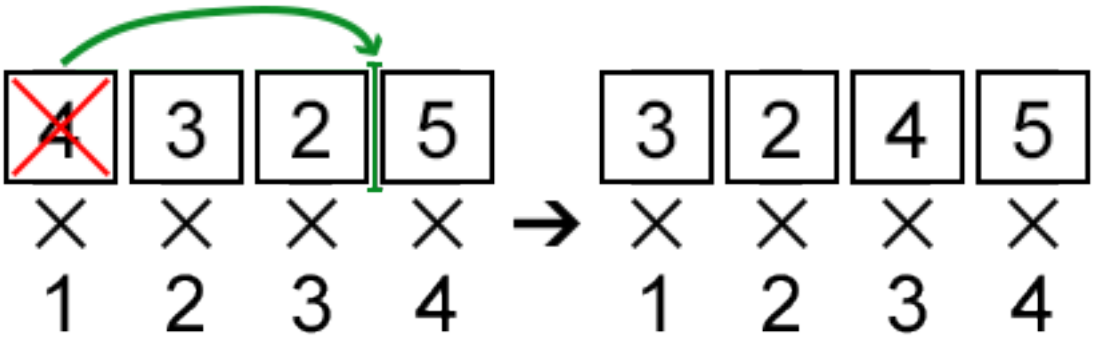
В примере выгодно поставить лесопилки у деревьев с номерами 3 и 6.

Задача A7. Мультипликативная сумма [0.4 sec, 256 mb]

Спонсоры сегодняшней задачи — Codeforces. Codeforces — мечты сбываются!

Хотя Блейк и является начальником Криса, это никак не мешает их дружбе. Обычно по вечерам они собираются вместе в баре и обсуждают там увлекательные задачи, в которых требуется максимизировать какую-нибудь величину. В этот раз им попалась особенная задача...

Пусть дан массив a длины n. Назовём *характеристикой* массива число $c = \sum_{i=1}^{n} a_i \cdot i$ — сумму произведений i-го числа в массиве a на число i. Разрешается **ровно один раз** сделать следующую операцию: выбрать некоторый элемент массива и переставить его в любое место в массиве. В частности, разрешается переставить элемент в начало, в конец или поставить его обратно на ту же позицию. Необходимо получить массив с максимально возможным значением характеристики.



Формат входных данных

В первой строке входных данных записано одно число n ($2 \leqslant n \leqslant 200\,000$) — размер массива a.

Во второй строке содержится n целых чисел a_i ($|a_i| \leqslant 1\,000\,000$) — элементы массива a.

Формат выходных данных

Выведите единственное целое число — максимально возможное значение характеристики массива a, если можно переставить ровно один элемент.

Примеры

stdin	stdout
4 4 3 2 5	39
5 1 1 2 7 1	49
3 1 1 2	9

Замечание

В первом примере можно удалить первый элемент из массива и вставить его перед третьим (перед числом 5), таким образом получив максимальный ответ: $3{\cdot}1{+}2{\cdot}2{+}4{\cdot}3{+}5{\cdot}4 = 39$.

Во втором примере максимальный ответ можно получить, удалив пятый элемент из массива и вставив его перед третьим: $1 \cdot 1 + 1 \cdot 2 + 1 \cdot 3 + 2 \cdot 4 + 7 \cdot 5 = 49$.

Задача A8. Неприводимые многочлены [0.1 sec, 256 mb]

Найти количество неприводимых унитарных $f \in Z_p[x] : \deg f = n$ $(p \in P)$.

Напомним, $\mathbb{Z}_p$ — это поле вычетов по модулю p. $\mathbb{Z}_p[x]$ — множество всех многочленов над полем $\mathbb{Z}_p$, то есть многочленов, коэффициенты которых являются остатками по модулю p.

Ненулевой многочлен называется *унитарным*, если его старший коэффициент равен 1.

Многочлен $f \in Z_p[x]$ называется *неприводимым*, если не существует таких многочленов $g, h \in Z_p[x] : \deg g > 0, \deg h > 0$ и $f(x) = g(x)h(x)$.

Формат входных данных

В единственной строке входного файла два числа: простое p и натуральное n. $1 \leqslant p, n \leqslant 10^9$; $1 \leqslant p^n < 10^{18}$.

Формат выходных данных

Выведите единственное число — количество неприводимых унитарных многочленов степени n над полем остатков по модулю p.

Примеры

stdin	stdout
2 2	1
3 4	18

Замечание

В первом примере единственный такой многочлен — $x^2 + x + 1$.

Задача A9. Японский кроссворд [0.1 sec, 256 mb]

Решение японских кроссвордов (они часто называются “Закраска числами”) – очень популярное развлечение среди новых русских. В этом виде кроссвордов клетки должны закрашиваться чёрным или оставаться пустыми в соответствии с числами, заданными на сторонах прямоугольной сетки. Если всё сделано правильно, чёрные клетки складываются в картинку. Числа на сторонах сетки – длины последовательных отрезков закрашенных клеток в строке или столбце. Например, ключ “4 8 3” означает, что есть отрезки из четырёх, восьми и трёх заполненных клеток, в таком порядке, с хотя бы одной пустой клеткой между последовательными группами. Конечно, программисты не настолько умны, как новые русские, поэтому мы не просим вас решить весь кроссворд. Вашей задачей будет написать программу, которая находит как можно больше информации о каждой клетке одной строки кроссворда по некоторой уже имеющейся информации об этой строке.

Формат входных данных

Первая строка содержит длину строки L ($1 \leqslant L \leqslant 400$) и число групп последовательных клеток, которые должны быть закрашены K ($0 \leqslant K \leqslant L$). Вторая строка содержит K целых чисел – длины этих групп. Третья строка содержит L символов, описывающих текущую информацию о клетках этой строки (эта информация может быть получена анализом данных столбцов и других строк):

- `'.'` означает клетку, которая определённо пуста,
- `'X'` (латинская заглавная) означает клетку, которая определённо должна быть закрашена,
- `'?'` означает, что об этой клетке нет информации.

Формат выходных данных

Выведите строку, содержащую L символов, описывающих наиболее полную информацию о клетках строки в том же формате, что и во вводе, или слово “`Impossible`”, если исходная информация противоречива, и никакая строка не может соответствовать входным данным.

Примеры

stdin	stdout
10 2 4 2 ?????.?X??	?XXX?.?X?.
9 0 ??????.X?	Impossible

Структуры данных – 1

Задача B1. Приказы [1, 256]

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 B2. Фыр-Фыр-Фенвик [2, 256]

Даны n точек с весами на плоскости. Каждая задаётся тремя числами x_i, y_i, w_i (координаты и вес). Вам нужно обработать m запросов двух типов:

- `get rx ry` – посчитать сумму весов точек, у которых $x_i \leqslant rx$ и $y_i \leqslant ry$.
- `change i z` – задать i-й точке новый вес равный z.

Формат входных данных

На первой строке число n ($1 \leqslant n \leqslant 100\,000$). На следующих n строках тройки целых чисел x_i, y_i, w_i ($0 \leqslant x_i, y_i, w_i < 10^9$). Следующая строка содержит количество запросов m ($1 \leqslant m \leqslant 300\,000$). На следующих m строках описания запросов в формате `get rx, ry` и `change i z`. Здесь $1 \leqslant i \leqslant n$, а остальные числа целые от 0 до 10^9-1.

Формат выходных данных

Для каждого запроса типа "`get`" выведите одно целое число на отдельной строке — ответ на запрос.

Пример

stdin	stdout
3 1 1 10 2 3 100 3 2 1000 4 get 2 3 change 2 200 get 3 3 get 3 2	110 1210 1010

Задача B3. Отрезки, пересекающие прямоугольник [3, 256]

Будем считать, что отрезок и прямоугольник пересекаются, если они имеют хотя бы одну общую точку.

Вам дан набор из n горизонтальных и вертикальных отрезков на плоскости. Выполните q запросов. Каждый запрос представляет из себя прямоугольник, со сторонами параллельными осям координат, а ответом на запрос является множество отрезков, пересекающих этот прямоугольник.

Формат входных данных

В первой строке дано целое число n ($1 \leqslant n \leqslant 50\,000$) — количество отрезков.

В следующих n строках дано по четыре целых числа x_1, y_1, x_2, y_2 — координаты двух концов отрезка. Гарантируется, что все отрезки параллельны осям координат. Отрезки могут быть вырождены в точку.

В следующей строке дано целое число q ($1 \leqslant q \leqslant 50\,000$) — количество запросов.

В следующих q строках даны по четыре целых числа x_1, y_1, x_2, y_2 ($x_1 \leqslant x_2$, $y_1 \leqslant y_2$) — координаты левого нижнего и правого верхнего угла прямоугольника из запроса.

Все координаты целые, по модулю не превосходят 10^6.

Формат выходных данных

Пусть I_j – множество индексов отрезков, пересекающих j-й прямоугольник. Выведите медиану множества. Если $|I_j| \bmod 2 = 0$, выведите меньшее из двух чисел. Медиана пустого множества равна -1. Отрезки нумеруются от 1 до n в порядке, в котором они идут во входных данных. Гарантируется, что $\sum_j |I_j| \leqslant 10^7$.

Примеры

stdin	stdout
3 0 0 0 5 -1 4 5 4 6 6 6 10 4 0 0 6 6 2 2 4 5 -1 -1 1 6 -2 -2 -1 -1	2 2 1 -1

Замечание

В примере ответами на запросы являются следующие множества:
$\{1, 2, 3\}$, $\{2\}$, $\{1, 2\}$, $\{\}$.
Соответственно, ответы на запросы: $2, 2, 1, -1$.

Задача B4. Добавление и удаление точек [2, 256]

Не все $N^2 \log N$ одинаковы полезны...

Какая-то лекция

У вас в каждый момент есть мультимножество A точек на плоскости.
Нужно научиться обрабатывать запросы трех типов:

- Добавить точку в мультимножество A
- Удалить точку из мультимножества A
- Вычислить $\sum\limits_{p \in A} \max\limits_{q \in A} distance(p, q)$.

Формат входных данных

Число запросов N ($1 \leqslant N \leqslant 3000$). Далее N строк, описывающие запросы, точный формат смотрите в примере. Координаты точек — целые число от 0 до 3000. Точки могут совпадать. Запрос *удалить точку* должен удалять ровно одну точку (гарантируется, что такая точка в мультимножестве на момент запроса есть).

Формат выходных данных

После каждой операции с множеством выводите текущую сумму максимальных расстояний.
Абсолютная погрешность не должна превышать 10^{-6}.

Пример

stdin	stdout
6 + 0 0 + 5 5 + 5 0 - 5 5 - 5 0 - 0 0	0.00000000000000000000 14.14213562373095100000 19.14213562373095100000 10.00000000000000000000 0.00000000000000000000 0.00000000000000000000

Задача B5. Галактический апокалипсис [1, 256]

Давным-давно в одной далекой-далекой галактике, было N планет. Также было $N-1$ межпланетных магистралей, соединявших между собой все планеты (не обязательно напрямую). Иными словами, сеть планет и магистралей образовывала дерево. Кроме того, каждая магистраль имеет свой показатель интересности, заданный неотрицательным целым числом. Пара планет (A, B) называется скучной, если выполняются следующие условия:

- A и B – различные планеты.
- В действующей сети межпланетных магистралей существует путь между A и B.
- Побитовый XOR показателей интересности всех магистралей в этом пути равен 0.

Ныне в галактике правит злой император, и он планирует использовать Силу, чтобы уничтожить все межпланетные магистрали в определенном порядке. Для того, чтобы спасти вселенную от гибели, вам необходимо определить количество пар скучных планет и после каждого разрушения вновь подсчитывать эту величину.

Формат входных данных

Первая строка содержит одно целое число N ($1 \leqslant N \leqslant 100\,000$). Каждая из следующих $N-1$ строк содержит три целых числа A_i, B_i, Z_i ($1 \leqslant A_i, B_i \leqslant 100\,000$, $0 \leqslant Zi \leqslant 10^9$), которые означают, что планеты с номерами A_i и B_i соединены магистралью с показателем интересности Z_i . Последняя строка содержит $N-1$ число: перестановку натуральных чисел от 1 до $N-1$, отражающую порядок уничтожения магистралей (если i-е число в строке равно j, то император уничтожит дорогу между планетами A_j и B_j на i-м шаге).

Формат выходных данных

Выведите N строк, в k-й строке выведите одно число – количество пар скучных планет после уничтожения $k-1$ дорог.

Пример

stdin	stdout
2 1 2 0 1	1 0
3 1 2 4 2 3 4 1 2	1 0 0
4 1 2 0 2 3 0 2 4 0 3 1 2	6 3 1 0

Задача B6. Persistent Array On [0.7, 256]

Дан массив (вернее, первая, начальная его версия).

Нужно уметь в *online* отвечать на два запроса:

- `create i j x` — создать из i-й версии новую, в которой j-й элемент равен x, а остальные элементы такие же, как в i-й версии.
- `get i j` — сказать, чему равен j-й элемент в i-й версии.

Это интерактивная задача. Запросы нужно обрабатывать в online, результат каждого запроса `flush`-ить до чтения следующего.

Формат входных данных

Количество чисел в массиве N ($1 \leqslant N \leqslant 10^5$) и N элементов массива. Далее количество запросов M ($1 \leqslant M \leqslant 10^5$) и M запросов. Формат описания запросов можно посмотреть в примере. Если уже существует K версий, новая версия получает номер $K + 1$. И исходные, и новые элементы массива — целые числа от 0 до 10^9. Элементы в массиве нумеруются числами от 1 до N.

Формат выходных данных

На каждый запрос типа `get` вывести соответствующий элемент нужного массива.

Пример

stdin	stdout
6 1 2 3 4 5 6 11 create 1 6 10 create 2 5 8 create 1 5 30 get 1 6 get 1 5 get 2 6 get 2 5 get 3 6 get 3 5 get 4 6 get 4 5	6 5 10 5 10 8 6 30

Подсказка по решению

Сделайте ровное красивое дерево. Глубины $\log_2 n$.

Замечание

Буфферизированные способы ввода, например, `readInt`, не подойдут, так как пытаются читать сразу «до конца файла». Пример корректного кода:

```
string query;
for (int i = 0; i < queryCnt; i++) {
    getline(cin, query);
    process(query);
    cout << flush; // cout << endl; тоже делает flush
}
```

Динамика по профилю

Задача C1. Длинные домино [0.1, 256]

Найдите количество способов замостить прямоугольник размера $m \times n$ длинными домино — прямоугольниками размера 3×1.

Каждое домино должно полностью находиться внутри прямоугольника, домино не должны накладываться.

Формат входных данных

Входной файл содержит m и n ($1 \leqslant m \leqslant 9$, $1 \leqslant n \leqslant 30$).

Формат выходных данных

Выведите количество способов замостить прямоугольник $m \times n$ длинными домино.

Пример

stdin	stdout
3 3	2
3 10	28

Задача C2. Шах-мат-блок [0.2, 256]

Вам нужно заблокировать всех **B**arsuk-ов на шахматной доске. Для этого нужно расставить **W**olkov-ов. Барсук заблокирован, если он не может дойти до другого барсука, переходя в соседние по стороне клетки, и не проходя через волков.

Формат входных данных

Доска 8×8, «`.`» обозначает пустую клетку, «`B`» барсука.

Формат выходных данных

Выведите число способов расставить волков, чтобы все барсуки были заблокированы.

Примеры

stdin	stdout
.B.B.B.B B.B.B.B. .B.B.B.B B.B.B.B. .B.B.B.B B.B.B.B. .B.B.B.B B.B.B.B.	1
B......B .... B......B	834135523322500070
.... BB......	0

Задача C3. Гамильтоновые гриды [4, 756]

Найдите число гамильтоновых цикла у грида $n \times m$.
Клетки считаются соседними, если имеют общее ребро.

Формат входных данных

Целые числа n и m ($2 \leqslant n \leqslant 15$, $2 \leqslant m \leqslant 100$).
Тесты в данной задаче растут плавно: $n, m \leqslant 10$, $n, m \leqslant 12$, $n, m \leqslant 14$, $n \leqslant 14$, $n \leqslant 15$.

Формат выходных данных

Выведите количество гамильтоновых циклов, вычисленное по модулю 10^9.

Примеры

stdin	stdout
2 3	1
5 6	154
8 8	4638576

Подсказка по решению

Динамика по скошенному профилю, состояний в слое не больше 3^n.
Основная оптимизация: построить граф состояний $\langle y, mask \rangle$.
Данную задачу можно сдать без прекалка.

Задача C4. Путешествие лучника [0.3, 256]

Назовём лучником шахматную фигуру, способную ходить на одно поле вперёд, назад, влево или вправо. Лучник стоит на поле $(1, 1)$ шахматной доски размера $n \times m$. Правое верхнее поле такой доски имеет номер (n, m). Цель лучника — обойти всю доску и вернуться в исходное поле, причём в процессе путешествия он должен побывать на каждом поле доски в точности один раз.

Сколькими способами лучник может обойти доску?

Формат входных данных

Целые числа n и m ($2 \leqslant n \leqslant 8$, $2 \leqslant m \leqslant 10^9$).

Формат выходных данных

Выведите количество способов обойти доску, вычисленное по модулю $10^9 + 7$.

Примеры

stdin	stdout
2 3	2
5 6	308
8 1000000000	841074556

Подсказка по решению

Матрицу можно ужать до размера 308×308.

Структуры данных – 2

Задача D1. Дерево [1, 256]

Задано подвешенное дерево, содержащее n ($1 \leqslant n \leqslant 1\,000\,000$) вершин. Каждая вершина покрашена в один из n цветов. Требуется для каждой вершины v вычислить количество различных цветов, встречающихся в поддереве с корнем v.

Формат входных данных

В первой строке входного файла задано число n. Последующие n строк описывают вершины, по одной в строке. Описание очередной вершины i имеет вид p_i c_i, где p_i — номер родителя вершины i, а c_i — цвет вершины i ($1 \leqslant c_i \leqslant n$). Для корня дерева $p_i = 0$.

Формат выходных данных

Выведите n чисел, обозначающих количества различных цветов в поддеревьях с корнями в вершинах $1, \ldots, n$.

Пример

stdin	stdout
5 2 1 3 2 0 3 3 3 2 1	1 2 3 1 1

Задача D2. Строки в дереве [1, 256]

Дано дерево. Дерево — это связный граф без циклов. На каждом ребре дерева написана строчная латинская буква. Между каждыми двумя вершинами существует ровно один простой путь, то есть путь по рёбрам дерева, проходящий через каждую вершину не более одного раза. Каждому пути соответствует строка, которая получается, если идти по этому пути и читать буквы на рёбрах в порядке следования. Путь можно проходить, начиная с любого его конца.

Также дана строка S. Соответствует ли она какому-либо простому пути в данном дереве?

Длина строки и размер дерева не превышают 300 000.

Вам необходимо реализовать функцию

`Solve (str, len, tree, n, v, u)`.

- `str` — строка, которую нужно найти
- `len` — длина строки
- `tree` — массив рёбер (`edge`)
- `n` — количество вершин в дереве
- `v`, `u` — переменные, в которые следует записать концы найденного вами пути

Функция должна возвращать 0 или 1 — присутствует ли строка в дереве.

Примеры

stdin	stdout
abc 4 1 2 c 4 3 a 2 4 b	YES 1 3
abc 1	NO
zy 6 1 2 x 1 3 y 1 4 z 3 5 a 4 6 b	YES 3 4

Пояснения к примерам

В первом примере строка `abc` соответствует пути 3–4–2–1.

Во втором примере в дереве нет ни одного ребра.

В третьем примере строка `zy` соответствует пути 3–1–4.

Задача D3. Inspector is Coming [1, 256]

Дано дерево из N вершин. Изначально все рёбра не помечены. Поступают Q запросов вида "пометить все рёбра на пути из u в v, вес которых от W_{min} до W_{max}". Выведите число помеченных рёбер в конце процесса.

Формат входных данных

Вам даны число N ($2 \leqslant N \leqslant 100\,000$) — число вершин в дереве. Следующие $N-1$ строка содержат описание рёбер дерева a_i, b_i и w_i ($1 \leqslant a_i, b_i \leqslant N$, $1 \leqslant w_i \leqslant 1\,000\,000$). Далее следует число запросов Q ($0 \leqslant Q \leqslant 100\,000$). Следующие Q строк описывают запросы, каждая содержит четыре целых числа u, v, W_{min}, and W_{max} ($1 \leqslant u, v \leqslant N$, $1 \leqslant W_{min} \leqslant W_{max} \leqslant 1\,000\,000$).

Формат выходных данных

Одно число – количество помеченных рёбер дерева.

Примеры

stdin	stdout
3 1 2 20 2 3 10 2 1 3 5 15 1 3 15 25	2
4 1 2 10 1 3 20 3 4 30 1 4 2 11 29	1

Замечание

Эту задачу нужно решать в offline.

Кстати, если не писали до этого LCA-offline, самое время попробовать.

Задача D4. MeX на пути дерева [1, 256]

Вам дано дерево из n вершин. Каждому ребру дерева сопоставлено неотрицательное целое число x_i. Ваша задача – ответить на q запросов. j-й из запросов – найти минимальное неотрицательное целое число y, отсутствующее на пути между вершинами a_j и b_j.

Формат входных данных

Первая строка содержит числа n и q ($2 \le n \le 10^5$, $1 \le q \le 10^5$), количество вершин в дереве и число запросов.

Следующие $n-1$ строка содержит описания рёбер дерева, тройки целых чисел u_i, v_i, x_i ($1 \le u_i, v_i \le n$, $u_i \ne v_i$, $0 \le x_i \le 10^9$).

Следующие q строк содержат пары целых чисел a_j, b_j ($1 \le a_j, b_j \le n$), обозначающие запросы.

Формат выходных данных

На каждый запрос a_j, b_j на отдельной строке выведите минимальное целое неотрицательное y, не лежащее на пути между a_j и b_j.

Пример

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

Хещи

Задача E1. Массивы-палиндромы [0.1, 256]

Кай работает в лаборатории изучения массивов, он экспериментирует с двумя массивами натуральных чисел: $A = [a_1, a_2, \ldots, a_n]$ длины n и $B = [b_1, b_2, \ldots, b_m]$ длины m.

Эксперимент, который проводит Кай, устроен следующим образом. У каждого из массивов отбрасывается произвольный, возможно пустой, префикс, а также произвольный, возможно пустой, суффикс, таким образом, чтобы оставшиеся части массивов имели равную длину. Обозначим получившиеся массивы как A' и B', а их длину как k. Затем Кай суммирует поэлементно получившиеся массивы, итоговый массив Кай обозначает как $C = [c_1, c_2, \ldots, c_k]$.

Пусть, например, $n = 5$, $A = [4, 3, 3, 2, 1]$, $m = 6$, $B = [4, 1, 5, 1, 3, 2]$, от массива A отбрасывается первый и последний элемент, от массива B три первых. После этого массивы имеют вид $A' = [3, 3, 2]$, $B' = [1, 3, 2]$, результат их поэлементного суммирования $C = [4, 6, 4]$.

Задача Кая заключается в том, чтобы получать такие C, которые являются *массивами-палиндромами*, то есть если числа на первой и последней позиции совпадают, числа на второй и предпоследней позиции совпадают, и так далее, для всех i числа на позициях i и $k - i + 1$ совпадают.

Помогите Каю понять, какой максимальный по длине массив-палиндром он может получить в результате эксперимента.

Формат входных данных

В первой строке ввода даны два целых числа n и m — количество элементов в первом и во втором массиве, соответственно ($1 \leqslant n, m \leqslant 100\,000$).

Во второй строке ввода даны n целых чисел a_i — массив A ($1 \leqslant a_i \leqslant 100$).

В третьей строке ввода даны m целых чисел b_j — массив B ($1 \leqslant b_j \leqslant 100$).

Формат выходных данных

Выведите единственное целое число — максимальное k, что Кай в результате эксперимента может получить массив-палиндром длины k.

Пример

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

Задача E2. Подпалиндромы [1, 256]

Дано слово и запросы двух типов:

- заменить i-ю букву в слове на букву c;
- проверить, является ли подстрока $s_j \dots s_k$ палиндромом.

Формат входных данных

В первой строке записано слово из n строчных латинских букв. Во второй строке записано целое число m — количество запросов ($5 \leqslant n, m \leqslant 10^5$). Следующие m строк содержат запросы. Каждый запрос имеет вид «`change` i a» или «`palindrome?` j k», где i, j, k — целые числа ($1 \leqslant i \leqslant n; 1 \leqslant j \leqslant k \leqslant n$), а символ c — строчная латинская буква.

Формат выходных данных

На все запросы второго типа выведите «`Yes`», если подслово $s_j \dots s_k$ является палиндромом, и «`No`» в противном случае.

Примеры

stdin	stdout
abcda 5 palindrome? 1 5 palindrome? 1 1 change 4 b palindrome? 1 5 palindrome? 2 4	No Yes Yes Yes

Пихаем квадраты

Задача F1. Переворачивания [2, 4]

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

Подсказка по решению

Это задача про «взять добротный квадрат» и попросить gpt пооптимизировать, используя mm256i и unroll-loops. Не забудьте включить AVX прагмами.

Если придумаете, как ещё улучшить, прекрасно. Авторский n^2 работает в 4.5 раз дольше авторского $n \log n$ на $n = 200\,000$.