Содержание

Обязательные задачи 3

Задача 15A. Самое дешевое ребро [1 sec, 256 mb] 3

Задача 15B. Центроиды дерева [1 sec, 256 mb] 4

Задача 15C. БДБД [1 sec, 256 mb] 5

Дополнительные задачи 6

Задача 15D. Строки в дереве [1 sec, 256 mb] 6

Задача 15E. Сбалансированные строки [1 sec, 256 mb] 7

Пример работы с файлами.

Если вы не умеете читать/выводить данные, или открывать файлы, воспользуйтесь примерами. http://acm.math.spbu.ru/~sk1/algo/sum/

Пример работы с файлами.

В некоторых задачах большой ввод и вывод. Про ввод-вывод в C++:
http://acm.math.spbu.ru/~sk1/algo/input-output/cpp_common.html
Имеет смысл пользоваться супер быстрым вводом-выводом. Две версии:
http://acm.math.spbu.ru/~sk1/algo/input-output/io_export.cpp.html
http://acm.math.spbu.ru/~sk1/algo/input-output/fread_write_export.cpp.html

Выделение памяти.

В некоторых задачах нужен STL, который активно использует динамическую память (set-ы, map-ы) переопределение стандартного аллокатора ускорит вашу программу:
http://acm.math.spbu.ru/~sk1/algo/memory.cpp.html

Обязательные задачи

Задача 15A. Самое дешевое ребро [1 sec, 256 mb]

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

minonpath.in	minonpath.out
5 1 2 1 3 2 5 3 2 2 2 3 4 5	2 2

Задача 15B. Центроиды дерева [1 sec, 256 mb]

Дано дерево из n вершин. У каждой вершины есть цвет. Нужно обработать q запросов (v_i, c_i): найти расстояние от v_i до ближайшей к v_i вершины цвета c_i. Расстоянием между вершинами называется минимальное количество рёбер в пути между ними.

Формат входных данных

На первой строке число n ($1 \leqslant n \leqslant 10^5$), следующая строка содержит числа $p_1, p_2, \ldots, p_{n-1}$. $0 \leqslant p_i < i$. p_i – отец вершины i в дереве. Далее строка с числами $a_0, a_1, \ldots, a_{n-1}$. $0 \leqslant a_i < n$. a_i – цвет вершины i. Далее строка с числом q ($1 \leqslant q \leqslant 10^5$). Следующие q строк содержат запросы $v_i q_i$ ($0 \leqslant v_i < n$, $0 \leqslant c_i < n$).

Формат выходных данных

Для каждого запроса выведите одно число – расстояние до ближайшей вершины нужного цвета, или -1, если в дереве нет вершин такого цвета.

Примеры

centroid.in	centroid.out
5 0 1 1 3 1 2 3 2 1 9 0 1 0 2 0 3 1 0 2 1 2 2 3 3 3 1 4 2	0 1 2 -1 2 1 2 1 1

Задача 15C. БДБД [1 sec, 256 mb]

Большая Древесная База Данных создана для того, чтобы в ней можно было надежно сохранить и раскрасить любое дерево. В новой версии БДБД запланирован новый функционал, для реализации которого потребуется вновь переосмыслить теорию графов.

В БДБД хранится взвешенное дерево. В языке запросов Системы Управления Большой Древесной Базы Данных (СУБДБД) предусмотрены два вида запросов:

1. «1 v d c» — покрасить все вершины, находящиеся на расстоянии не более d от вершины v, в цвет c. Все вершины изначально окрашены в цвет с номером 0.
2. «2 v» — вывести цвет вершины v.

Необходимо запрограммировать СУБДБД и ответить на все запросы пользователя.

Формат входных данных

В первой строке число N ($1 \leqslant N \leqslant 10^5$) — количество вершин дерева.

Следующие $N-1$ строк содержат по три числа в строке a_i, b_i, w_i ($1 \leqslant a_i, b_i \leqslant N$, $a_i \neq b_i$, $1 \leqslant w_i \leqslant 10^4$) — i-ое ребро имеет вес w_i и соединяет вершины a_i и b_i.

В следующей строке число Q ($1 \leqslant Q \leqslant 10^5$) — число запросов. В каждой из Q следующих строк запросы одного из двух видов:

1. Числа 1, v, d, c ($1 \leqslant v \leqslant N$, $0 \leqslant d \leqslant 10^9$, $0 \leqslant c \leqslant 10^9$).
2. Числа 2, v ($1 \leqslant v \leqslant N$).

Все числа во входных данных целые.

Формат выходных данных

Для каждого запроса второго типа выведите цвет запрошенной вершины.

Примеры

lwdb.in	lwdb.out
5 1 2 30 1 3 50 3 4 70 3 5 60 8 1 3 72 6 2 5 1 4 60 5 2 3 2 2 1 2 144 7 2 4 2 5	6 6 0 5 7

Дополнительные задачи

Задача 15D. Строки в дереве [1 sec, 256 mb]

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 $3 \cdot 10^5$.

Интерфейс

Вам необходимо реализовать функцию
`Solve (str, len, tree, n, v1, v2)`.

- `str` — строка, которую нужно найти
- `len` — длина строки
- `tree` — массив рёбер (`edge`)
- `n` — количество вершин в дереве
- `v1`, `v2` — переменные, в которые следует записать концы найденного вами пути

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

Примеры

treestr.in	treestr.out
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

Замечание

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 15E. Сбалансированные строки [1 sec, 256 mb]

Вам дано дерево из n вершин. Вершины пронумерованы целыми от 1 до n. Каждой вершине соответствует метка с символом '(' или ')'. Обозначим за $l[u \to v]$ строку, полученнную конкатенацией всех меток на простом пути из u в v. Заметьте, для любой пары u и v есть ровно один простой путь.

Сбалансированные строки определяются следующим образом.

- Пустая строка сбалансирована.
- Для любой сбалансированной s, строка "(" s ")" сбалансированна.
- Для любых сбалансированных строк s и t, их конкатенация сбалансированна.
- Любая другая строка **не** сбалансированна.

Посчитайте количество таких пар вершин дерева (u, v), что строка $l[u \to v]$ сбалансированна. Пары (u, v) и (v, u) считаются одинаковыми, смотрите первый пример.

Формат входных данных

На первой строке число вершин дерева n ($2 \leqslant n \leqslant 10^5$). Следующая строка содержит n сиволов '(' и ')'. i-й из них – метка i-й вершины дерева. Следующие $n - 1$ строк содержат рёбра дерева – пары чисел a_i и b_i ($1 \leqslant a_i, b_i \leqslant n$).

Формат выходных данных

Выведите одно число – ответ на задачу.

Примеры

balanced-strings.in	balanced-strings.out
2 () 1 2	1
4 (()) 1 2 2 3 3 4	2
5 ()()) 1 2 2 3 2 4 1 5	4