,

Задача A. Арифметическая или геометрическая?

Если числа x_1, x_2 и x_3 являются тремя подряд идущими членами арифметической прогрессии, то это эквивалентно тому, что $x_1 - x_2 = x_2 - x_3$, или же $x_1 + x_3 = 2 \cdot x_2$. Аналогично, если x_1, x_2 и x_3 являются тремя подряд идущими членами геометрической прогрессии, то это эквивалентно $x_1 \cdot x_3 = x_2^2$.

Упорядочиваем a, b и c по возрастанию и проверяем оба равенства. Заметим, что при проверке второго равенства может переполниться даже `unsigned long long`, так что требуется или использовать тип `__int128`, или решать задачу на языках с поддержкой длинной арифметики (например, Питоне), или вместо проверки произведения проверять, что дробь x_1/x_2 равна дроби x_2/x_3: подсчитать наибольший общий делитель, сократить на него, после чего сравнить числители и знаменатели.

Заметим, что одновременно быть тремя подряд идущими членами как арифметической, так и геометрической прогрессии с указанными в условиями свойствами разности и знаменателя никакие три целых числа быть не могут (хотя этот факт для решения и не имеет никакого значения — просто переход на соответствующую ветвь выполнен не будет). Задачу на доказательство этого факта оставляем читателю как несложное упражнение по математике.

Задача B. Большая загрузка

Будем хранить имена файлов в качестве ключа ассоциативного массива (std::map в C++), соответствующим значением будет целое число — количество раз, которое файл встретился. Как только приходит не уникальное имя файла — увеличиваем значение по данному ключу на 1, генерируем имя файла и кладём его в массив строк (или сразу в контейнер, подобный std::set). Уникальное имя кладём туда же в неизменном виде. После того, как все имена файлов обработаны, сортируем массив и выводим список файлов (в случае с std::set просто проходим итератором от начала до конца и выводим содержимое в соответствующем порядке).

Задача C. Варианты сообщения

Построим граф, в котором вершинами являются пользователи, а две вершины соединены ребром, если пользователи являются друзьями. Запустим обход в ширину из вершины, соответствующей автору сообщения, и вычислим минимальные расстояния для каждого пользователя. Если оно равно бесконечности, то пользователь не видел сообщения и не мог его перепечатать. В противном случае пользователь видел сообщение после d итераций. На каждой итерации чётность длины сообщения меняется, то есть на нечётных итерациях сообщения совпадать не могут (так как будут иметь разную длину). На чётных итерациях сообщения могут совпадать (вставили ту же букву, которая была удалена на предыдущей итерации, или удалили ту же букву, которая была вставлена). То есть ответом к задаче будет количество вершин, расстояние до которых от вершины, соответствующей автору сообщения, чётно.

Задача D. Генерация непростых чисел

Заведём массив $dp[i][j]$, где i — длина числа, составленного из набора цифр 0,2,4,5,6,8, а j — остаток от деления числа на 3, хранящий количество чисел соответствующей длины с соответствующим остатком, и массив $dp1[i][j]$ аналогичной структуры для цифр 0,2,4,6,8. Тогда ответ будет равен $dp[N][1] + dp[N][2] - (dp1[N][1] + dp1[N][2]))$, то есть количеству N-значных чисел с остатками, отличными от нуля, составленными из всех разрешённых цифр, минус количество N-значных чисел с остатками, отличными от нуля, составленными из этого же множества без пятёрки (единственной нечётной цифры, не взаимно простой с 10).

Для пересчёта динамики заметим, что остаток 0 имеют 0 и 6, остаток 1 — 4, остаток 2 — 2, 5, 8, после чего $dp[i][0] = dp[i-1][0] * 2 + 3 * dp[i-1][1] + dp[i-1][2]$ (так как чтобы получить остаток 0, к остаткам 0 предыдущей длины можно приписать 0 и 6, к остаткам 1 — 2, 5 и 8, к остаткам 2 — только 4). Аналогично считаются и остальные два перехода; при вычислении $dp1$ 3 заменяется на 2, так как пятёрка в том наборе отсутствует.

При инициализации типичной ошибкой будет учитывать однозначный 0, который находится вне диапазона (то есть старт в виде $dp[1][i]$ идёт с 1 3 1 (1 2 1 для $dp1$), а не с 2 3 1).

Разумеется, все вычисления проходят по модулю.

,

Задача E. Дневник олимпиадного школьника

Задача разделяется на две части: техническая по сути задача по подсчёту оценок (при этом избежать лишнего кода помогут правильно организованные классы или структуры), и задача по обработке запросов с возможными ошибками.

В первой части некоторую техническую сложность может вызвать разбор строки с буквами задач; наиболее просто это делать, считывая строку целиком и затем просто считывая два символа (так как имя задачи — ровно два символа) и пропуская третий (пробел или конец строки). Также эффективен форматный ввод-вывод.

Наиболее простым вариантом решения второй части является создание ассоциативного массива с ключами в виде строк, которые получаются не более, чем двумя обменами от имеющихся имён, и со значениями в виде оценок. Если у полученной строки уже есть значение — это значит, что строка может быть получена из двух разных строк и в качестве значения помещается специальное число, которое будет выведено как знак вопроса. При генерации строк следует предотвратить ситуацию «ложного знака вопроса», когда строка совпадает с уже полученной из текущей строки (из-за того, что поменяли две одинаковые буквы); как вариант, можно завести std::set для каждого имени, а после завершения обработки имени перекладывать содержимое в ассоциативный массив).