Разбор задач 1-го этапа Всесибирской открытой олимпиады школьников по информатике 2015

Задача 1. Годовая оценка

Данную задачу следует решать в целых числах, это позволит избежать проблем с точностью, возникающих при использовании вещественных чисел. Для начала посчитаем количество двоек, троек, четверок и пятерок, которые получил Рома, параллельно считая общую сумму оценок в переменной sum. Предположим, что $border[i]$ – граница для получения оценки $i, 2 \le i \le 5$, причем $border[2] = 0$ (т.е. граница для получения двойки – 0 баллов). Пройдемся по списку баллов, которые Рома получил за контрольные работы. Пусть k – количество баллов за текущую контрольную работу. Тогда, если $k \ge border[i]$ для некоторого i, то Рома получил как минимум оценку i. Нам нужно лишь найти наибольшее такое i. Мы посчитали sum – сумму всех оценок и знаем количество оценок $i, 2 \le i \le 5$, осталось дать ответ на задачу – какую итоговую оценку получит наш ученик и какой у него будет средний балл. Средний балл вычисляется как sum/n, где n – общее количество оценок Ромы. Для получения пятерки средний балл Ромы должен быть не меньше 4.7, откуда следует, что $sum/n \ge 4.7$. Домножив на $10n$ обе части неравенства, получим $10 \cdot sum \ge 47 \cdot n$. Если выполняется данное условие и отсутствуют оценки 3 и 2, то Роман получит итоговую оценку 5. Условия для получения оценок 4 и 3 составляются аналогично.

Задача 2. Жадные гномики

Сначала рассмотрим наивное решение, которое просто моделирует описанный в задаче процесс, а именно: запишем в некоторую переменную значение N, и теперь на каждом шаге будем просто умножать эту переменную на 2 и вычитать 1. Какие тогда у нас возникают проблемы?

Во-первых, так как все типы данных, которыми мы пользуемся в программе, имеют конечную длину, то значение переменной быстро переполнится. Даже если бы такой проблемы не было, нам бы всё равно потребовалось много операций, поэтому такое решение не успевает исполняться за отведённое время. По этим причинам такое решение набирает 20 баллов.

Для начала избавимся от переполнения. Заметим, что нам неважно, какое мы в итоге получаем число, нам нужно, чтобы оно делилось на N. Здесь удобно воспользоваться следующими соотношениями:

$(a + b) \bmod m = (a \bmod m + b \bmod m) \bmod m$

$(a \cdot b) \bmod m = ((a \bmod m) \cdot (b \bmod m)) \bmod m,$

где $a \bmod m$ обозначает операцию взятия остатка от деления a на m. Теперь распишем подробнее операции, которые мы выполняем: $N \Rightarrow 2N - 1 \Rightarrow 2(2N - 1) - 1 \Rightarrow \ldots \Rightarrow X, X \bmod N = 0$. Получается, что на каждом шаге можно считать не реальное значение выражения, а лишь его остаток от деления на N. В таком случае, как только на каком-то шаге получается элемент, остаток от деления на N которого равен 0, то можно сказать, что это и будет искомый шаг, что прямо следует из вышеупомянутых свойств. При таком решении появляется возможность определять, в каком случае не удастся получить число, делящееся на N. А именно: если на некотором этапе получается такой остаток, который уже встречался, и при этом, остаток 0 ещё не получен, то и далее он не встретится. Действительно – остаток, который получается на следующем шаге,

определяется однозначно, поскольку остаток повторился, то далее следующие остатки будут повторятся по циклу, поэтому остаток 0 уже не встретится. Асимптотика такого решения – $O(N)$ и оно набирает 50 баллов.

В основе полного решения лежит применение теоремы Эйлера. Рассмотрим элементы последовательности, которые получаются после раскрытия скобок: $N \Rightarrow 2N-1 \Rightarrow 4N-3 \Rightarrow 8N-7 \Rightarrow \ldots$, в общем случае полученная последовательность имеет вид: $2^kN - 2^k - 1$. Это можно заметить, выписав первые несколько элементов последовательности, а доказать формально, например, воспользовавшись методом математической индукции. Следовательно, нужно решить уравнение: $2^kN - 2^k - 1 \equiv 0 (\mathrm{mod}\ N)$.

Заметим, что 2^kN всегда без остатка делится на N, а значит оно в уравнение вклада не вносит. Перенося слагаемое –1 вправо, получаем модульное уравнение следующего вида: $2^k \equiv 1 \pmod N$, где требуется найти минимальное возможное положительное k, если оно существует. Ранее было показано, что в конечном счёте получается некоторый цикл. Заметим, что если ответ существует, то первый же элемент лежит в этом цикле. Это следует из того, что при $k = 0$, $2^k \equiv 1 \pmod N$, и нам нужно найти первое ненулевое k, для которого тоже будет выполняться это соотношение. Значит цикл должен прийти в конечном счёте в ту же точку. Теперь, основываясь на этом, воспользуемся теоремой Эйлера, которая гласит: $a^{\varphi(N)} \equiv 1 (\mathrm{mod}\ N)$, если N и a являются взаимно простыми (в обратном случае ответа не существует), где $\varphi(N)$ – это количество натуральных чисел, меньших N и взаимно простых с ним.

Для нахождения $\varphi(N)$ можно воспользоваться стандартным алгоритмом со сложностью $O(\sqrt{N})$. Поскольку $\varphi(N)$ может быть не первым числом, при возведении двойки в степень которого мы получим 1, то это число ещё не является ответом. Тем не менее, несложно заметить, что ответ будет делителем $\varphi(N)$. Чтобы показать это обозначим через t – длину вышеописанного цикла, который и будет ответом на задачу. В 1 мы могли попасть только одним способом – пройти один или несколько раз по этому циклу, откуда следует, что: $t \cdot v = \varphi(N)$, а значит ответ будет делителем $\varphi(N)$. Таким образом, мы можем перебрать все делители $\varphi(N)$, их будет порядка $O(\sqrt[3]{N})$, и для каждого из них проверить, подходит ли он. Это можно сделать при помощи быстрого возведения в степень за $O(\log N)$. Асимптотика итогового решения – $O(\sqrt{N} + \sqrt[3]{N} \log N)$.

Задача 3. Реми и кулинарный поединок

Из условия задачи можно вывести такую формальную её постановку: сколько существует способов расставить перегородки между элементами массива таким образом, чтобы между каждых двух последовательных перегородок находилась строго монотонная последовательность. Между двух соседних элементов разрешается ставить не более одной перегородки, в конце массива обязательно должна стоять перегородка. В начале массива (перед первым элементом) перегородки быть не должно.

Для решения задачи воспользуемся методом динамического программирования. Обозначим за $dp[i]$ количество способов расставить перегородки между первыми i элементами массива. Очевидно, что $dp[1] = 1, dp[2] = 2$. Рассмотрим теперь k-й элемент последовательности, и будем считать, что уже посчитаны значения динамики в состояниях $dp[k-1], dp[k-2]$.... Переберём позиции j $(0 \le j \le k-1)$ в которых мы могли поставить последнюю перегородку. Если мы пытаемся поставить перегородку после j-го элемента, то необходимо убедиться, что отрезок элементов с j-го по k-й является монотонным. Это нетрудно сделать за линейное время. Т.к. перебирать позицию j можно справа налево, проверка на монотонность может делаться «на ходу» (проверяя, что текущий j-й

элемент не нарушает монотонности). Во всех позициях, которые допускают выставление перегородки по описанному выше критерию, мы можем прибавить значение уже посчитанной динамики: каждый раз нужно обновить значение $dp[k]$, прибавив $dp[j]$. В итоге, динамика имеет N состояний, каждое из которых вычисляется за $O(N)$ в худшем случае. (На самом деле, k-е состояние вычисляется за количество действий, пропорциональное наибольшей длине монотонной последовательности, заканчивающейся в k-м элементе). Итоговая сложность: $O(N^2)$

Рассмотрим теперь более оптимальное и простое линейное решение, использующее несколько другие рассуждения. Будем поддерживать $dp[i]$, имеющее тот же смысл, что и в предыдущем решении. Рассмотрим теперь k-й элемент последовательности, а также два предыдущих элемента. Обозначим их a, b, c соответственно слева направо. Возможны два случая:

1. Три элемента a, b, c образуют монотонную последовательность ($a \le b \le c$). В этом случае мы можем как ставить, так и не ставить перегородку между bc. Перегородки, выставленные до элемента b, полностью учитываются значением динамики $dp[k-1]$, поэтому $dp[k] = 2 \cdot dp[k-1]$.

2. Три элемента a, b, c не образуют монотонную последовательность. Тогда мы обязаны поставить перегородку либо между ab, либо между bc. Следовательно, $dp[k] = dp[k-1] + dp[k-2]$.

Таким образом, мы получили решение за $O(N)$.

Задача 4. Взволнованный водитель

Опишем решение за $O(NM)$. Такое решение заключается в том, чтобы для каждой из точек, в которой Афанасий волнуется, пройти циклом по всем камерам и проверить, видит ли камера его или нет. Один из способов сделать такую проверку – это заметить, что точка находится в другой полуплоскости тогда и только тогда, когда угол между вектором нормали и вектором между точкой, из которой исходит нормаль и точкой, в которой требуется произвести проверку, более 90 градусов. Для этого нам не надо находить сам угол, нам достаточно проверить тот факт, что скалярное произведение заданных векторов меньше 0. То же самое можно было бы понять и из геометрического смысла скалярного произведения. Если при таком решении полностью вычислять угол, то потребуется вещественная арифметика, которая не является точной, поэтому помимо того, что такое решение – долгое, оно может также выдавать неправильные ответы на некоторые тесты, и набирает около 40 баллов.

В решении с использованием скалярного произведения мы уже можем не прибегать к вещественной арифметике, а значит все ответы будут точными. Тем не менее, оно не успевает по времени, и получает порядка 50 баллов.

Теперь ускорим наше решение, для этого посмотрим на отдельную камеру. Во-первых, у нас может быть случай, когда линия, которая разделяет две полуплоскости камеры параллельна линии, вдоль которой двигается Афанасий. В таком случае, у нас либо эта камера будет наблюдать все точки, которые предстоит посетить кроту, либо – ни одну. Поэтому для такой камеры достаточно проверить только одну из точек, в которой мы побываем. Пусть теперь эти две линии не параллельны. В таком случае у нас эти две линии обязательно где-нибудь пересекаются, причём, может быть один из двух вариантов: либо все точки до точки пересечения осматривались камерой (включительно), а все после – не осматриваются, либо наоборот. Так как точки, в которых

требуется узнать количество наблюдающих камер упорядочены, то определив точку пересечения, мы можем определить до какой точки (либо начиная с какой) эту камеру нужно посчитать. Это можно сделать, например, при помощи бинарного поиска и массива, в каждом элементе которого будем хранить число – на сколько изменилось количество камер после того, как Афанасий проедет данную по счёту точку. В таком случае, если количество камер уменьшилось, то в элементе массива будет храниться отрицательное число. При учёте каждой камеры нам требуется изменить не более одного числа в массиве. Можно было бы вместо использования бинарного поиска добавить все точки пересечения в некоторый массив вместе с точками для проверки и воспользоваться сортировкой. Далее, проходя слева направо по массиву, поддерживать количество наблюдающих сейчас камер. В обоих случаях нужно обратить отдельное внимание на то, что если точка, которую мы проверяем, находится на той же линии камеры, то считается, что камера наблюдает за точкой. Такие решения работает за время $O(N \log M)$.

Стоит отметить, что здесь также хочется прибегнуть к вещественной арифметике, когда мы пересекаем прямые, однако в таком случае всё же возникают проблемы с точностью, поэтому такое решение набирает около 80 баллов.

Чтобы избежать использование вещественных чисел, мы можем хранить и искать все точки пересечения в виде p/q, где p и q – целые числа. В таком случае нам требуется написать свои арифметические операции над таким дробями. Такое решение набирает полный балл.

Задача 5. Петя и IT

Опишем решение за $O(N^2)$. Построим два графа G и H: в графе G проведём только все существующие изначально ребра, а в графе H наоборот: только те ребра, которых нет в графе G. Обозначим за $rev[j]$ стоимость «мирового переворота» из вершины j. Запустим поиск в ширину в графе G из стартовой вершины (с номером 1), а в графе H из конечной вершины (с номером N). Обозначим за $dist_1[i]$ – расстояние от начала до i-й вершины графа G, за $dist_2[i]$ – расстояние от конечной вершины до i-й в графе H. Тогда ответ – это минимум из $dist_1[N]$ (если добираться, не прибегая к «мировому перевороту») и минимума по всем j величины $dist_1[j]+rev[j]+dist_2[j]$ (дойти до вершины j в графе G, совершить мировой переворот и пойти до конца в графе H). Решение работает за $O(N^2)$, ведь суммарно в графы G и H будет добавлено $N \cdot (N-1)/2$ ребер (для каждой пары различных городов – по ребру).

Модифицируем описанное решение. Обойдём граф G, используя метод поиска в ширину. Пусть величины $dist_1[v]$ посчитаны. Заметим, что для произвольной вершины v возможны две ситуации:

1. Из вершины v есть ребро в конечную вершину. Тогда кратчайший путь до конечной вершины, проходящий через текущую вершину v, равен $dist_1[v] + 1$.

2. Из вершины v ребра в конечную вершину нет. В таком случае ясно, что после совершения мирового переворота ребро из v в конечную вершину появится. Поэтому кандидатом на кратчайший путь является $dist_1[v] + rev[v] + 1$.

Таким образом, если после обхода в ширину рассмотреть все описанные возможности, мы получим ответ. Решение работает за $O(N + M)$, т.к. использует только обход в ширину графа G.