Разбор задач обучающей интернет-олимпиады по информатике «Код успеха»

Возрастная номинация «7-11 классы»

Теоретическая часть

Задача 1. Задача коммивояжера

Ответом на задачу будет величина $|a–b|+|b–c|+|c–a|$. Модуль берется из-за того, что неизвестно, какой город находится левее. Также возможны и другие решения.

В зависимости от корректности работы алгоритма участник мог получить до 30 баллов

Задача 2. Арифметика

Посчитаем остаток от деления суммы цифр числа на $b–1$. Это и будет нашим ответом. Доказательство: рассмотрим число $a_1a_2…a_n$, записанное в системе счисления с основанием b. $a_1a_2…a_n=a_1b^{n–1}+a_2b^{n–2}+…+a_{n–1}b+a_n=$ $a_1(b^{n–1}–1)+a_1+a_2(b^{n–2}–1)+…+a_{n–1}(b–1)+a_{n–1}+a_n=a_1(b^{n-1}–1)+a_2(b^{n–2}–1)+…+$ $a_{n–1}(b–1)+(a_1+a_2+…+a_n)$. Нетрудно убедиться, что $b^{k–1}–1=(b–1)(b^{k–1}+b^{k–2}+…+b+1)$. Из этого равенства следует, что числа $a_1(b^{n–1}–1)$, $a_2(b^{n–2}–1)$,…, $a_{n–1}(b–1)$ делятся на $b–1$. Тогда остаток от деления числа $a_1a_2…a_n$, записанного в системе счисления с основанием b, равен остатку от деления суммы $a_1+a_2+…+a_n$ на $b–1$, что и требовалось доказать. Чтобы считать большое число (по условию, в нем может быть порядка миллиона цифр), достаточно хранить его как строку.

Неоптимальные решения получали не более 10 баллов.

Решения, не обосновывающие формулу, набирали не более 20 баллов.

Задача 3. Мозаика

Мозаика может быть построена в том случае, если $N^2*N^2=2*K$ (предлагаем доказать этот факт самостоятельно). Построением, удовлетворяющим заданным условиям, будет простая шахматная раскраска. Возможны и другие варианты раскраски.

Переборные решения получали не более 10 баллов.

Задача 4. Стрельба

Если поле имеет размер $1 \times M$, $N \times 1$ или 1×1, то ответ очевиден. Пусть $N > 1$ и $M > 1$. Тогда имеет смысл сделать один выстрел по горизонтали и один выстрел по вертикали. Таким образом, получим поле размера $(N–1) \times (M–1)$, в клетках которого остаются живые враги. Предположим, что выстрел прошел через клетку (A, B), тогда он проходит через клетки (2*A, 2*B), (3*A, 3*B) и так далее. Таким образом, ответом является количество пар чисел (A, B) таких, что A и B – взаимно простые и $1 \le A \le N–1$, $1 \le B \le M–1$. Воспользуемся динамическим программированием. Пусть T[N] – количество клеток, наибольший общий делитель координат которых равен N. Ответом будет являться величина 2 + T[1]. Рассмотрим число (N/K)*(M/K). Это будет количество клеток в прямоугольнике, таких, что их НОД кратен K, но так как в него будут входить числа с НОД, равным 2*K, 3*K и т. д., нам необходимо их вычесть. Тогда переходом будет T[K]=(N/K)*(M/K)–T[2*K]–T[3*K]–… Тогда сначала нам надо найти T[min(n–1,m–1)], а затем последовательно вплоть до единицы вычислять T[K].

Переборные решения получали не более 15 баллов.

Задача 5. Игра

Будем считать, что две закрашенные клетки находятся в одной компоненте связности в том случае, если они лежат либо на одной горизонтали, либо на одной вертикали, либо если существует множество закрашенных клеток такое, что можно переместиться от одной клетки к другой, двигаясь только по вертикали и горизонтали, поворачивая только в закрашенных клетках. Тогда ответом для компоненты связности будет количество различных координат клеток по оси абсцисс, умноженное на количество различных координат по оси ординат. Ответом для всей задачи будем сумма ответов для компонент. Посчитать ответ можно, например, с помощью обхода в глубину или системы непересекающихся множеств. С помощью системы непересекающихся множеств задача решается за O(KlogK).

Переборные решения получали не более 15 баллов.

Практическая часть

Задача 1. Очередь

Интуитивно понятно, что в кабинет надо посылать студента, если есть свободный кабинет и свободный студент. То есть, если мы в какой-то момент времени не пошлём студента в свободный кабинет, то ответ будет не оптимальный.

Докажем интуитивно понятный факт из второй подзадачи. Для начала найдём оптимальный ответ в случае, когда M = 1. Утверждается, что ответ для этого случая и есть ответ на задачу. Пусть мы выполнили волну для первого типа кабинета. Тогда время нахождения после этого каждого студента можно сделать (M - 1) * T. Понятно, что меньше нельзя. Так можно сделать, например, следующим образом. Пусть студент проходил i-ый тип в кабинете с номером j. Тогда либо до него этот кабинет кто-то проходил, и тогда после него он может пойти сразу в тот же кабинет, в который пошёл человек до него. Иначе он может просто пойти в любой свободный кабинет. (т.к. в таком случае всего прошло дальше не более чем K - 1 человек). То есть, ответ для M = 1 является ответом и для больших M. В этом случае интуитивно понятный факт тривиален – пусть есть студент, который должен пойти и не идёт. Если после него не появится никакого студента, который пойдёт в свободный кабинет, то очевидно, что процесс не оптимален. Иначе мы поменяем местами первого и второго студента. Несложно понять, что минимальное время не ухудшится. Если оно не ухудшилось, то где-то есть студент, который сидит дольше, чем эти два студента. Иначе ответ улучшится. Т.е. условие на номера студентов тоже выполняется.

Задача 2. Перестановки со спусками

Воспользуемся методом динамического программирования. DP[N][K] – количество перестановок длины N с K спусками. Начальные состояния: DP[I][0]=DP[I][I–1]=1, i=1,…,N. Рассмотрим переход. Пусть имеется перестановка чисел от 1 до N–1. Вставить в неё число N мы можем одним из следующих способов:

1. Вставить N в конец перестановки, тогда нового спуска не будет создано;
2. Вставить N между элементами, составляющими спуск, тогда нового спуска не будет создано;
3. В противном случае в перестановке становится одним спуском больше.

Таким образом, из перестановки длины N–1 с K спусками мы можем получить K+1 перестановку длины N с K спусками и N–K перестановку с K+1 спуском. Чтобы получить перестановку длины N с K спусками, получаем формулу перехода:

DP[N][K]=(K+1)*DP[N–1][K]+(N–K)*DP[N–1][K–1].

Ответ будет храниться в DP[N][v].

Задача 3. Счастье студента

Воспользуемся стандартной структурой “Дерево отрезков”, которая будет поддерживать запросы вида найти сумму на отрезке и прибавить к некоторому отрезку арифметическую прогрессию с первым членом a и разностью d. Очевидно, как с помощью такой структуры решать задачу. Структура поддерживается следующим образом. Помимо суммы на подотрезке в каждой вершине хранятся параметры прогрессии – первый член и разность. Как только мы хотим получить информацию о вершине, мы проталкиваем информацию о прогрессии детям и обновляем актуальную информацию в вершине. Таким образом, во время любого обращения к вершине дерева у нас в ней хранится актуальная информация. Итоговая асимптотика решения - O(M log N)