Задача A. Фотографии на память

Данную задачу можно решать несложным динамическим программированием, либо жадностью. Для начала заметим, что в любом случае выгодно отсортировать всех существ по росту, так как в рамках одной фотографии, чем ближе существа по росту друг к другу, тем лучше.

Далее, если хотим делать динамическое программирование, до делаем массив $dp[n]$ — минимальное число фотографий, которое нужно сделать, чтобы сфотографировать первых n существ. Тогда:

- $dp[0] = 0$. Нет существ, нет фотографий.
- $dp[i] = max(dp[i-1], dp[i-2], dp[i-3]) + 1$. При этом, можно брать $dp[i-2]$ или $dp[i-3]$ только тогда, когда выполнены необходимые условия.

Также можно довериться интуиции или доказать, что всегда выгодно брать на каждую следующую фотографию, как можно больше существ. Тогда, если n существ, уже сфотографированы, смотрим следующих трех, если можно их разместить на одной фотографии, то размещаем и переходим к $n+3$, если нет, пытаемся взять двух, если и тут нет, берем одного.

Независимо от выбора решения, асимптотическая сложность будет $O(n)$.

Задача B. Волшебные тройки

Требуется посчитать количество троек чисел a, b и c ($1 \leqslant a < b < c \leqslant n$), таких что $a \cdot b$, $a \cdot c$ и $b \cdot c$ — квадраты натуральных чисел.

Рассмотрим разложение числа x на простые множители в виде $x = p_1^{q_1} \cdot p_2^{q_2} \cdot \dots \cdot p_k^{q_k}$. Заметим, что $a \cdot b$ является квадратом натурального числа, если в разложении числа $a \cdot b$ все q_i — четны. Значит, в разложениях a и b множества простых чисел, входящих в нечетной степени, должны совпадать.

Таким образом, найдем для каждого числа x от 1 до n множество простых, которые входят в x в нечетной степени. После этого, разобьем числа на группы с совпадающими множествами. Ответом является сумма по всем группам, количество способов выбрать три числа в группе. Количество способов выбрать три числа из m штук равно $\frac{m \cdot (m-1) \cdot (m-2)}{6}$.

Задача C. Магическая ПСП

Дано множество чисел $A = \{a_1, a_2, \dots, a_n\}$. Требуется построить правильную скобочную последовательность, в которой мультимножество расстояний между парными скобками будет равно A.

Несложно заметить, что расстояние между парными скобками всегда четно, поэтому если в A есть нечетное число, решения точно не существует.

Рассмотрим самое большое число из A. Пара скобок, соответствующая этому расстоянию, не может быть вложена ни в какую другую пару скобок. Поэтому, можно поставить такую пару скобок в самом начале скобочной поледовательности.

Таким образом, задачу можно решить методом динамического программирования по подмножествам. Состоянием является мультимножество чисел, являющееся подмножеством A. Для каждого мультимножества нас интересует, существует ли ПСП, соответствующая такому мультимножеству. И если искомая ПСП существует, сохраним любую подходящую. Пустому мультимножеству соответствует пустая ПСП. Для любого другого множества, выберем самое большое число из множества. Пусть оно равно x. Поставим первую пару скобок на расстоянии x. Тогда оставшиеся числа из множества нужно разбить на две части: одно будет содержать $\frac{x}{2}$ чисел, и будет соответствовать ПСП внутри первой пары скобок, и вторая часть будет содержать все оставшиеся числа и будет соответствовать ПСП правее первой пары скобок. Значит, нужно перебрать способы выбрать $\frac{x}{2}$ чисел из текущего множества.

Для оценки асимптотики, просуммируем по всем множествам $C_{k-1}^{\lfloor (k-1)/2 \rfloor}$, где k — количество чисел в множестве. Получается порядка $4 \cdot 10^8$. На практике, такое решение уже укладывается в ограничение по времени. Можно еще ускорить решение, если заметить, что различных состояний может быть меньше, чем 2^n, потому что в A могут встречаться повторяющиеся числа.

Задача D. Сокровищница

Требуется построить несколько вложенных друг в друга латинских квадратов с заданными размерами и совпадающими верхними-левыми углами.

Для начала, научимся определять, когда решения не существует. Докажем, что размеры вложенных друг в друга латинских квадратов должны отличаться хотя бы в 2 раза. Допустим, существуют два латинских квадрата с размерами $a \times a$ и $b \times b$, вложенные друг в друга, и при этом $a < b$ и $a \cdot 2 > b$. Тогда рассмотрим a чисел, которые встречаются в меньшем квадрате. Они должны встречаться в каждой строке большого. При этом, в строках и столбцах с номерами от 1 до a они уже встречаются в меньшем квадрате. Значит, в строках с номерами $a+1 \dots b$ должно быть по a таких чисел. При этом, все эти числа должны стоять в столбцах с номерами $a+1 \dots b$. Поэтому, с одной стороны нужно поставить хотя бы $(b-a) \cdot a$ чисел, а с другой стороны у нас есть для этого только $(b-a) \cdot (b-a)$ доступных клеток. Из предположения, что $a \cdot 2 > b$ следует, что доступных клеток меньше количества чисел, которые необходимо поставить. Пришли к противоречию. Значит, размеры двух вложенных латинских квадратов должны отличаться хотя бы в 2 раза.

Оказывается, что во всех остальных случаях решение можно построить. Научимся решать задачу для двух квадратов размерами $a \times a$ и $b \times b$. Тогда мы будем уметь решать задачу для любой последовательности $a_1, a_2, \dots, a_n$ просто решая задачу сначала для a_1 и a_2, потом для a_2 и a_3 и так далее. Для начала, расставим числа $1 \dots a$ на пересечении строк и столбцов с номерами $a+1 \dots b$. Это можно сделать просто расставив 1 на главной диагонали, 2 на один правее, 3 еще на один правее и так далее (после столбца номер b идет столбец номер $a+1$). Осталось расставить числа $a+1 \dots b$. Причем, каждое число должно встречаться ровно в одной строке и ровно в одном столбце. Несложно заметить, что позиции одного числа соответствуют ребрам из полного паросочетания в двудольном графе, левой долей которого являются строки, правой долей — столбцы, а ребрами — свободные клетки. Докажем, что полное паросочетание всегда будет существовать. В общем случае, у нас есть матрица $x \times x$ и в каждом столбце и каждой строке свободны ровно y клеток. Применим лемму Холла. Рассмотрим некоторое множество строк размера k. Из этого множества выходит ровно $k \times y$ ребер. По принципу Дирихле, ребра ведут минимум в k столбцов. Значит, лемма Холла выполнена и полное паросочетание существует. Причем, оно будет существовать и после того, как мы найдем одно полное паросочетание и займем очередным числом соответствующие клетки. Для того, чтобы находить паросочетания, можно воспользоваться алгоритмом Куна или Хопкрофта-Карпа. В зависимости от выбранного алгоритма и от оптимальности кода, решение может проходить или не проходить последнюю подзадачу.