Разбор задачи «Книги на лето»

Для начала посчитаем количество символов в каждой книге. Очевидно, что для книги номер i это значение равно $book_i = PPP \cdot SSS$.

Теперь возьмем некоторую перестановку последовательности $book_i$, в которой $book_{i1} > book_{i2}$. Заметим, что возможна ситуация, когда времени на чтение книги $book_{i1}$ не хватает, а на чтение $book_{i2}$ хватает, тогда необходимо их переставить. Следуя далее таким рассуждениям, приходим к тому, что самый оптимальный вариант перестановки является последовательность по возрастанию. Поэтому отсортируем массив $book$ по возрастанию и будем набирать книги жадно, пока нам хватает времени.

Заметим, что максимальное количество символов во всех книгах, может быть равно 10^{11}, а максимальное количество символов, которое может успеть прочитать Вася — 10^{36}. Для того, чтобы решать задачу в целых числах достаточно проверить, что заданные на вход K и T — скорость чтения (символы в минуту) и свободное время в минутах, которое есть у Васи, заведомо больше суммарного количества символов, требуемых для прочтения книг.

Разбор задачи «Домик в Триангляндии»

Для полного решения этой задачи необходимо было рассматривать не точки по отдельности, а векторы, которые эти точки образуют. Две точки $P_1(x_1, y_1)$ и $P_2(x_2, y_2)$ образуют два вектора: $v_1(x_2 - x_1, y_2 - y_1)$ и $v_2(x_1 - x_2, y_1 - y_2)$, которые выходят из точек P_1 и P_2 соответственно.

Для каждой точки будем рассматривать векторы, которые из нее выходят. Если из точки выходят два перпендикулярных вектора, то они образуют один прямоугольный треугольник.

Заметим, что для каждой точки исходящие векторы можно сгруппировать по направлению, то есть достаточно хранить количество векторов, исходящих из данной точки в определенном направлении. Обозначим количество векторов, исходящих из точки по направлению l, как $\mathrm{c}(l)$. Тогда если два направления l_1 и l_2 перпендикулярны, то они образуют суммарно $c(l_1) \cdot c(l_2)$ прямоугольных треугольников.

Для того, чтобы сгруппировать векторы по направлению, необходимо привести сонаправленные векторы разной длины к общему виду. Так как все точки имеют целочисленные координаты, то мы можем вектор вида $v(x, y)$ приводить к виду $v'\left(\frac{x}{\text{НОД}(x,y)}, \frac{y}{\text{НОД}(x,y)}\right)$. Тогда любые два сонаправленных вектора перейдут в один вектор.

Заметим, что два приведенных таким образом вектора $v_1'(x_1, y_1)$ и $v_2'(x_2, y_2)$ перпендикулярны тогда и только тогда, когда $\begin{cases} x_1 = -y_2 \\ y_1 = x_2 \end{cases}$ или $\begin{cases} x_1 = y_2 \\ y_1 = -x_2 \end{cases}$. То есть направление, заданное координатами (x, y), образует $c(x, y) \cdot \Big(c(-y, x) + c(y, -x)\Big)$ прямоугольных треугольников. После суммирования по всем направлениям, нужно разделить ответ на 2, так как каждую пару направлений мы посчитаем дважды.

Координаты точек лежат в пределах от -1000 до 1000, то есть координаты векторов лежат в пределах от -2000 до 2000. Сдвинем систему координат так, чтобы все точки имели неотрицательные координаты. Тогда координаты векторов будут лежать в пределах от 0 до 4000. Теперь для каждой точки мы можем завести массив размера 4001×4001 и находить количество векторов данного направления за O(1).

Количество направлений, по которым нам нужно пройти, в худшем случае равно n^2. То есть итоговая асимптотика решения будет $O(n^2)$.

Разбор задачи «Иннокентий и одна очень известная страна»

Будем решать задачу методом динамического программирования: сначала выберем произвольный город u, в котором расположим филиал. После чего посчитаем значение $dp[v]$ — путь наибольшей длины, начинающийся в вершине v, не проходящий через города на пути из v в u. Пусть w_i — города, соседние с v, тогда

$$dp[v] = \begin{cases} \max(dp[w_i]) + 1 & \text{, если количество } w_i \text{ не равно } 0 \\ 1 & \text{, если количество } w_i \text{ равно } 0 \end{cases}$$

Тогда ответом для u будет сумма квадратов значений динамики в соседних городах.

Теперь посмотрим, как изменится ответ при переносе филиала из u в некоторый новый соседний город v: мы пока не знаем максимальный путь для v, начинающийся в u. Но несложно заметить, что он равен максимальному пути для u, не начинающемуся в v, плюс 1. Таким образом можно обойти города из того города, для которого мы посчитали динамику, и восстановить ответ для остальных.

Разбор задачи «Футболки участникам»

Посчитаем сначала количество позитивных исходов P, когда N делится нацело на $2 * a + 3 * b$.
Если N — четное, то b тоже должно быть четным , и, наоборот, если N нечетное, то и b нечетное.
В нечетном случае $P = \lfloor \frac{\lfloor \frac{N}{3} \rfloor + 1}{2} \rfloor$.
В четном случае $P = \lceil \frac{\lfloor \frac{N}{3} + 1 \rfloor}{2} \rceil$.
Количество всех исходов $Q = (a + 1) * (b + 1)$.
Ответ — это $\frac{P}{Q}$.

Разбор задачи «Сокровища на карте»

Для решения этой задачи необходимо построить граф, каждая вершина которого является клеткой поля. Однако, заметим, что по всем клеткам мы можем ходить какое угодно количество раз, и только клетки с бонусами изменяют наше состояние — количество набранных баллов. Соответственно, построим граф, состоящий из $Q+1$ вершины — всех клеток с бонусами и стартовой клетки. Чтобы сделать это, можно запустить из всех $Q + 1$ вершин обход в ширину, причем, если мы можем дойти по пустым клеткам из вершины v в вершину u, то в искомый граф добавляем такое ребро.

Далее заметим, что жадные стратегии не работают, т.к. возможна ситуация, когда в графе необходимо пройти несколько вершин с отрицательным весом, прежде чем достичь некоторую вершину с большим по модулю положительным весом. Будем решать задачу с помощью перебора по битовым маскам. Определим маску $mask$, в которой будем единицами задавать те вершины, которые можно посетить игроку, а нулями — те, в которые идти запрещено. Тогда запустим другой обход в ширину из стартовой вершины, с учетом ограничений, которые задаются маской. Среди всех вариантов обхода выберем тот, который дает нам максимально достижимый результат.