1 Обязательная часть

1. ***Фишки на матрице.***

   Тут есть два варианта решения. В зависимости от значения k стоит выбрать лучший по времени.

   Первый вариант.

   Этот способ не зависит от значения k и тратит полином памяти. Время работы: $\mathcal{O}(C_{w+h-2}^{h-1} \cdot (w+h))$. Откуда что берется: выбираем, на каком из $w+h-2$ ходов делаем шаг вверх, это C_{w+h-2} шагов алгоритма. На каждом шаге за w+h-2 этот путь "восстанавливаем проходим по нему и смотрим цену, наберет ли к концу все k типов.

   Второй вариант.

   Этот способ зависит от k и тратит гораздо больше памяти, чем первый, но в некоторых ситуациях может сработать значительно быстрее засчет асимптотики. Итак, время работы нашего античемпиона: $\mathcal{O}(wh \cdot 2^k)$. Алгоритм работает как обычный алгоритм, про коровку, которая хочет кушать, но сидит на диете, поэтому надо съесть как можно меньше травы. Но только массив у нас теперь трехмерный: третье поле mask. $d[i][j][mask]$ - минимальная стоимость пути до клетки $[i,j]$, на котором мы взяли хотя бы по одной фишке каждого типа, отмеченного в mask. А масок у нас всего 2^k - по количеству подмножеств k типов.

2. ***Покраска в 4 цвета.***

   Как покрасить в два цвета? Покрасить в два цвета можно тогда и только тогда, когда нет циклов в графе. Как это проверить? Бфс. Запускаемся от всех белых вершин, если в какой-то момент нас посылают в серую - отказываемся туда идти, обвиням граф в цикличности, говорим, что красить его запрещено. Иначе первую вершину бфса называем 1, ее смежных друзей - 2, их смежных друзей опять 1 и так чередуем.

   Как теперь расширить это на 4 цвета. Заметим, что покраска в графа в 4 цвета это все равно что отдельно покрасить два его непересекающихся "подграфа"(которые в объединении дают весь граф) в два цвета - первый в 1 и 2, второй в 3 и 4. Итак, что мы делаем: сначала предподсчет, все подмножества нашего графа проверяем на ацикличность бфсом. Займет $\mathcal{O}^*(2^n)$. Теперь опять бежим по подмножествам. Если найдем хоть одно такое подмножество H графа G, что H и $G \backslash H$ ациклична (это мы уже узнали во время предподсчета), то есть их оба можно покрасить в два цвета, то их объединение G можно покрасить в 4 цвета, то есть все хорошо.

3. ***Рюкзак в несколько заходов.***

   Решаем задачу в четыре этапа. Нулевой: предподсчитаем за $\mathcal{O}(2^n)$ вес и стоимость каждого подмножества. Первый: динамикой за $\mathcal{O}(2^n)$ бежим по подмножествам вещей и смотрим, можно ли их унести за один подход и сколько заработаем. Можно унести, если их вес меньше W. Если нельзя унести - ставим в ячейку $-\infty$. Можно - ставим их стоимость, она предподсчитана. Получили $d1[H]$

   Второй: бежим по подмножествам H, считаем, можно ли их унести и с какой стоимостью максимальной. Как это делаем. Бежим по их подмножествам T (то есть тут сложность уже $\mathcal{O}(3^n)$) и считаем, что рассматриваемое сейчас мы унесли последним. Тогда ответ будет $d2[H] = max(d1[T] + d1[H \backslash T])$. Третий этап: То же самое для трех подходов:
   $d3[H] = max(d1[T] + d2[H \backslash T])$. Ну и заключительный этап, при помощи которого получим ответ: $d4[H] = max(d1[T] + d3[H \backslash T])$. Трижды работали $\mathcal{O}(3^n)$, два $\mathcal{O}(2^n)$. В итоге: $\mathcal{O}(3^n)$.

4. ***Циклы в неориентированном графе.***

   По сути надо проверить, существует ли гамильтонов путь в подмножествах A. Как это делаем. Для каждой пары (подмножество вершин, вершина) буем считать, существует ли гамильтонов путь для этого подмножества вершин, заканчивающихся в выделенной вершине. То есть у нас есть двумерный массив $d[mask][j]$ размером $\mathcal{O}(2^n \cdot n)$. И на каждом шаге за линию считаем значение в ячейке массива.

   $$d[mask][j] = d[(1 << j) \oplus mask][u] \text{ для всех u, смежных с j и лежащих в mask, т.е. } (1 << k)\ \&\&\ mask! = 0).$$

5. ***Разбиение на циклы.***

   - $\mathcal{O}(3^n)$

     За $\mathcal{O}(2^n \cdot n)$ Делаем предподсчет - бежим по всем подграфам и есть ли цикл, проходящий по всем вершинам, это делается бфсом. Затем перебираем все подграфы графа. На каждом шаге перебираем подподграфы и суммируем ответы на них. Подробнее: Пусть есть граф G. Перебираем его подграфы H. В каждом H перебираем T. Для T и $H \backslash T$ мы уже знаем, сколько способов разбить их на циклы. Тогда способов разбить H на циклы: $d[H] = \sum_T d[T] \cdot d[H \backslash T]$. И надо разделить это добро пополам, так как по два раза посчитали каждый способ ($T = T'T = H \backslash T'$). А когда T - пустое, то считаем d[T] равным 1, если при предподсчете узнали, что T - один большой цикл. Иначе - 0. Способов разбить G на циклы: $d[G]$.

     Решение свелось к перебору всех подмножеств подмножеств. Мы уже знаем, что это делается за $\mathcal{O}(3^n)$.

6. ***Число совершенных паросочетаний.***

   Динамичка. $d[i][H]$ - количество способов покрыть подмножество вершин H той части, что с m вершинами, используя первые i вершин другой части.

   $$d[i][H] = \sum_k d[i-1][H \oplus (1 << k)] + f[i-1][H],$$
   где k из H, смежная с i из второй части.

Работает за $\mathcal{O}(2^m \cdot nm)$ - двумерный массив, бежим по вершинам, линия каждый шаг.

7. ***Расписание.***

За $\mathcal{O}^*(2^{|Q|})$ делаем предподсчет: перебираем подмножества уроков Q и считаем, можно ли все уроки из него провести одновременно. Для этого: все группы различны, преподы различны, для каждого урока можно найти аудиторию. Последнее переформулировывается так: представим в виде двудольного графа наши уроки (одна часть - препод+группа, вторая - класс), должно найтись такое паросочитание, что покрывает все вершины первой части. Это мы умеем делать за полином.

Сама динамика $d[H]$ - меньшее время, за которое пройдут все уроки из H. $d[H] = min_{T \subseteq H}(d[H \oplus T]) + 1$, если в предподсчете выяснили, что все уроки из T можно провести одновременно. Мы перебираем подмножества подмножеств, поэтому работает за $\mathcal{O}(3^n)$.