Задание 1

Дано: гиперкуб.

Алгоритм:

1. Для начала заматим, что граф G, вершинами которого являются вершины гиперкуба, а ребрами - ребра гиперкуба, двудольный.
2. В полученном графе G удалим все вершины, покрашенные в черный цвет (граф (G_1), очевидно, остался двудольным).
3. Найдем в графе (G_1) максимальное независимое множество. Перекрасим вершины не входящие в это множество.

Время: O(VE)

Корректность: допустим, что можно оставить непрекрашенными больше вершин. Заметим, что какие-то две из них гарантированно соединены ребром, иначе возникнет противоречие с максимальностью независимого множества. Противоречие.

Исправления: зафиксируем систему координат в $\mathbb{R}^d$. Тогда вершина кодируется строкой из d символов (нули и единицы), например (0,0,1,0,1,1,1,1,1,1), для $d = 10$. Следовательно, всего вершин 2^d, ребро между вершинами есть, если они отличаются на 1 в одном разряде. Отсюда мы сразу можем разбить вершины на две доли: в первой вершины с четным количеством 1, а во второй с нечетным. У каждой вершины d инцидентных ей ребер, следовательно, всего ребер $d2^d$.
Время: $O(d4^d)$

Задание 2

Дано: граф G.

Алгоритм:

1. Найдем какое-нибудь минимальное вершинное покрытие графа G. Пусть его размер sz.

2. Удалим из графа G лексиграфически минимальную вершину и все инцидентные ей ребра. Получим граф G_1.

3. Найдем какое-нибудь минимальное вершинное покрытие в графе G_1. Его размер sz_1

$sz_1 == sz$ Возвращаемся к пункту 2 и берем вторую лексиграфически минимальную вершину.

$sz_1 == sz - 1$ Возвращаемся к пункту 2, но теперь $G := G_1$, $sz := sz_1$.

Время: $O(V^2E)$ - для каждой вершины не более одного поиска вершинного покрытия.

Корректность: Перебор вершин идет в лексиграфическом порядке -> Очевидно.

Задание 3

Дано: граф G.

Алгоритм:

1. Разделим каждую вершину на две, получим двудольный граф G_1. В первой доле вершины, отвечающие за исходящие, во второй за входящие ребра.

2. Найдем максимальное паросочетание. Если оно не совершенно, то разбить вершины на циклы нельзя, иначе просто восстановливаем циклы, переходя по ребрам из паросочетания.

Время: O(VE)

Корректность: если совершенное паросочетание нашлось, то восстановить разделение на циклы довольно легко - просто берем вершину, ходим по ребрам из G_1, причем из второй доли в первую ведут только ребра из паросочетания, если не нашли, то допустим, что разбиение на циклы есть, но такое разбиение дает нам и совершенное паросочетание (просто ребра этих циклов). Противоречие.

Так как мы только что установили биекцию между наборами циклами и паросочетаниями, то задача поиска минимального набора циклов сводится к задаче поиска минимального совершенного паросочетания, а это мы уже умеем делать.

Задание 4

Дано: граф G.

Алгоритм:

1. Разделим каждую вершину на две, получим двудольный граф G_1. В первой доле вершины, отвечающие за исходящие, во второй за входящие ребра.
2. Найдем максимальное паросочетание. Множество ребер из паросочетания и будет ответом.

Время: O(VE)

Корректность: степень каждой вершины $\leq 2 \Rightarrow$ граф, составленный из ребер из мультимножества представляет объединение циклов, путей и изолированных вершин (только если и в исходном графе эта вершина была изолирована). Для максимизации размера мультимножество необходимо максимизировать количество циклов. Пусть k - количество вершин не принадлежащих циклам, а m - размер паросочетания, n - количество вершин в графе. Тогда $k = n - m$. Т.к. вершина без пары - это начало/конец пути. Опять биекция между количеством циклов и размером паросочетания.

Исправление: покажем, что любое мультимножество переделывается в паросочетание. Посмотрим на граф, состоящий только из ребер из мультимножества. Это объединение циклов, путей и изолированных вершин.

1. Цикл -> паросочетание: ориентируем ребра (все ребра одинаково) -> получили набор ребер, у любых двух ребер нет общей вершины.

2. Путь -> паросочетание: ориентируем ребра (все ребра одинаково) -> получили набор ребер, у любых двух ребер нет общей вершины.

Задание 5

Дано: набор прямых.

Алгоритм:

1. Пусть прямые - вершины графа G. Между вершинами i, j проведем ребро, если
   $(line_i \parallel line_j) | ((0, line_i(0)) \in line_j)$

2. Найдем максимальное независимое множество.

Время: $O(n^3)$

Корректность: очевидно

Исправление: прямая задается парой (k, b)
Алгоритм:

1. Пусть вершины первой доли - это набор k, вершины второй доли - набор b. Соединим две вершины (k_i, b_j) ребром, если существует прямая $y = k_i x + b_j$.

2. Найдем максимальное паросочетание.

Время: $O(n^2)$

Корректность: прямые параллельны, если их k равны, пересекаются в 0, если их b равны. В результате все k и b должны быть разные.

Задание 6

Дано: граф G.

Алгоритм:

1. Строим эйлеров обход графа G.
2. Выкидываем из этого цикла все ребра с четными номерами вхождения (нумерация начинается с любой вершины).
3. Из оставшихся ребер получился 2^{k-1}-регулярный граф. Возвращаемся к пункту 1.
4. $k = 1$ - осталось паросочетание.

Время: O(E). На первой итерации $O(E)$, на второй - $O(\frac{E}{2})$ и т.д.

Корректность: очевидно.

Задание д2

Дано: граф G.

Алгоритм:

1. Сортируем вершины первой доли по весу.
2. Запускаем Куна, который перебирает вершины в отсортированном порядке.

Время: O(VE)

Корректность: пусть есть такая вершина v, которую возьмет алгоритм, но которой нет в оптимальном ответа.
Следовательно, ни одно ребро из инцидентных ей не лежит в ответе, но, так как алгоритм хотел ее взять, на расстоянии 2 от нее есть вершина с меньшим весом, возьмем первое ребро этого пути и уберем второе. Мы

увеличили вес паросочетания.

Исправления: пусть есть такая вершина v, которую возьмет алгоритм, но ее нет в оптимальном ответе.
Возьмем оптимальный ответ, запустим dfs из v (из второй доли в первую только по ребрам из паросочетания), т.к. наш алгоритм ее взял, то когда-нибудь мы дойдем до вершины (u), значение в которой меньше, чем в v (или найдем чередующийся путь, но это нам подходит), возьмем этот путь и поменяем всем ребрам на этом пути принадлежность к паросочетанию, причем все вершины на этом пути (кроме u), все еще принадлежат паросочетанию -> паросочетание корректно. В итоге, мы смогли увеличить вес оптимального паросочетания. Противоречие.

Задание д3

Дано: граф G.
Алгоритм:

1. Сортируем вершины первой доли по весу.
2. Сортируем ребра каждой вершины (сначала те, которые ведут в вершины с большим весом).
3. Запускаем Куна.

Задание д4

Дано: граф G, его матрица смежности A (смежность вершин первой и второй доли).
Алгоритм:

1. Посчитаем определитель матрицы A mod 2. Если он 0, то кол-во совершенных паросочетаний четно, иначе нечетно.

Время: $O(V^3)$

Корректность: совершенное паросочетание - перестановка вершин первой доли, причем $a_{\sigma(i)i} = 1$, для любого i. Тогда количество совершенных паросочетаний:

$$\sum_\sigma \prod_{i=0}^{n-1} a_{\sigma(i)i}$$

Заметим, что это то же самое что и дискриминант с точность до $(-1)^{sign(\sigma)}$ в каждом слагаемом, но заметим, что $-1\%2 = 1\%2$, следовательно, можно заменить $(-1)^{sign(\sigma)}$ на 1. Получили, что по модулю 2 детерминант и количество совершенных паросочетаний равны.