Задание на восьмую неделю.

№1

Пусть $mindist(x, y)$ алгоритм выдающий наикратчайшее расстояние от a до b в графе. Он работает за $O(V+E)$, т. к. с помощью BFS можно найти наикратчайшее расстояние от вершины запуска x до всех остальных, включая y.
Тогда с помощью этого алгоритма за $O(V+E)$ найдем $mindist(a, b)$, $mindist(a, c)$, $mindist(b, c)$, $mindist(b, d)$, $mindist(a, d)$, где c, d вершины соединенные данным ребром. Это ребро принадлежит хотя бы одному наикратчайшему пути, тогда и только тогда, когда:

$$mindist(a, b) - 1 =$$

$$= min(mindist(a, c) + mindist(d, b), mindist(b, c) + mindist(d, a))$$

Таким образом, за $O(V+E)$ можно проверить принадлежность ребра хотя бы одному наикратчайшему пути.

№2

В реализации алгоритма через очереди, каждый раз из нее берется первый элемент, и в конец добавляются достижимые из него вершины. Модифицируем его: если из первой в очереди вершины идет ребро веса нуль в некоторую вершину, то тогда будем добовлять эту вершину не в конец очереди, а в начало. Тогда все вершины будут достижимы за такое же расстояние как смежные с ними через ребро нуль вершины. Все остальное осталось без изменений, а значит алгоритм работает корректно.

№4a)

Применим стандартный алгоритм Ф-У, сделав $V-1$ итераций. После сделаем V-ую итерацию, если в матрице изменится некоторый элемент A_{ii}, то значит соответствующая ему i-ая вершина входит в отрицательный цикл. Тогда если между парой вершин существует путь, проходящий через нее, то расстояние между ними $-\infty$. Тогда всем элементам k, m, для которых $A_{ki} < +\infty$, и $A_{im} < +\infty$, сделаем $A_{km} = -\infty$. После этого снова сделаем итерацию, если изменится еще один элемент вида A_{tt}, повторим процедуру еще раз и т. д. Т. к. всего кол-во таких элементов V, то доп итераций придется сделать не больше V. Соответственно, сложность алгоритма не меняется. При этом, все расстояния определяются с учетом отрицательных циклов.

№4)

$$d_0 = \begin{bmatrix} 0 & 1 & 3 & -1 \\ \infty & 0 & \infty & -2 \\ \infty & -2 & 0 & -4 \\ \infty & \infty & \infty & 0 \end{bmatrix} \Rightarrow d_1 = \begin{bmatrix} 0 & 1 & 3 & -1 \\ \infty & 0 & \infty & -2 \\ \infty & -2 & 0 & -4 \\ \infty & \infty & \infty & 0 \end{bmatrix} \Rightarrow$$

$$d_2 = \begin{bmatrix} 0 & 1 & 3 & -1 \\ \infty & 0 & \infty & -2 \\ \infty & -2 & 0 & -4 \\ \infty & \infty & \infty & 0 \end{bmatrix} \Rightarrow d_3 = \begin{bmatrix} 0 & 1 & 3 & -1 \\ \infty & 0 & \infty & -2 \\ \infty & -2 & 0 & -4 \\ \infty & \infty & \infty & 0 \end{bmatrix} \Rightarrow$$

$$d_4 = \begin{bmatrix} 0 & 1 & 3 & -1 \\ \infty & 0 & \infty & -2 \\ \infty & -2 & 0 & -4 \\ \infty & \infty & \infty & 0 \end{bmatrix}$$

№5

b) Очевидно, что на i-ом шаге расстояние до каждой вершины от начальной вершины D будет являться минимальным среди путей длины не больше i. Тогда, если найти кратчайшие пути из D во все остальные, и посчитать длину максимального из этих путей, равную S, то после S шагов метки перестанут обновляться. Сделав это для данного графа, получим что $S = 4$, что соответствует кратчайшему пути из D в F. $\Rightarrow$ Через четыре итерации.

c) Для этого, необходимо, что бы для вершины A существовала вершина B: $d_m in(A, B) = V - 1$. Если из A выходит более двух ребер, предположим в C и D, то тогда длина наикратчайшего пути между A и любой другой вершиной заведомо $\leq V-1$ (иначе в пути есть циклы, и выкинув их можно уменьшить длину). Значит, A соединена ровно с одно вершиной A_1. Тогда задача сводится, к тому что $d_m in(A_1, B) = V-2$ в подграфе из $V-1$ вершин (все, кроме A). Тогда аналогично предыдущему, A_1 сединена ровно с одной вершиной: A_2, и т. д. до B. Значит, искомое множество графов это все и только все простые пути из V вершин (петли могут присутствовать).

№6

Предположим, что в графе есть цикл,включающий в себя вершин u, v. Тогда из вершины u можно добраться до вершины v и обратно: $uu_1 \dots u_n v u_n \dots u$. Тогда, из сущестования частичного порядкка между всеми смежными вершинами и его транзитивностью получим: uRv и vRu. А это противоречит свойству антисимметрии. Таким образом, в графе нет циклов. Но тогда в к каждой компоненте связности по одной вершине. $\Rightarrow$ Кол-во вершин в конденсации равно кол-ву вершин в исходном графе.

№7

a) Т. к. v достижима из u, но не наоборот, то граф можно считать ацикличным. Тогда рассмотрим работу алгоритма DFS при прохождение вершины v. Если до этого мы уже были в u, то значит в дереве DFS v потомок u, тогда из свойства DFS время выхода из v будет раньше u. Если, мы еще не были в вершине u, то т. к. u недостижима из v, алгоритм сначала выйдет из v, побывав во всех достижимых из нее вершин, а только после сможет зайти в u. Тогда, очевидно, время выхода из u больше v в любом случае, ч. т. д.

b) Предположим, что в графе есть ребро (u, v) и $t_u < t_v$. v недостижима из u, т. к. иначе в графе есть цикл. Тогда, из предыдущего пункта $t_v > t_u$, получаем противоречие.

№8

a) Произвольные вершины a, b в одной компоненте связности $\Leftrightarrow$ есть пути $(a, b), (b, a)$. Таким образом, если a, b были в одной компоненте, то после инвертации они так же будут в одной. При этом, вершины из разных компонент не могут оказаться в одной. $\Rightarrow$ Коппоненты сильной связности не меняются.
b) Временные затраты: инвертация ребер (за $O(E)$); DFS (за $O(|V| + |E|)$). Будем, параллельно с DFS записывать порядок вершин по времени выхода из вершины (из пункта a) оно всегда растет). Тогда, после этого, останется только применить еще раз DFS. $\Rightarrow$ Всего времени $O(E + V)$. В доп памяти необходимо хранить массив времен выхода для каждой вершины, он занимает $O(V)$.
c) Да, верно. Т. к. из определения конденсации, следует что полученый граф будет ациклическим.