Problem #1

Suppose we know that X reduces to Y in polynomial time.
X -> Y
X <=p Y

a) If Y is NP-complete then so is X.
We can’t infer X is NP-complete. X can be NP-complete, NP, or P.

b) If X is NP-complete then so is Y.
We can’t infer Y is NP-complete. Y can be NP-complete or NP-hard.

c) If Y is NP-complete and X is in NP then X is NP-complete.
X doesn’t necessarily have to be NP-complete. X can be also P or NP.

d) If X is NP-complete and Y is in NP then Y is NP-complete.
Yes, in this case Y is NP-complete.

e) X and Y can't both be NP-complete.
If X and Y are in NP-complete, both are NP-complete.

f) If X is in P, then Y is in P.
Y can be P, NP, NP-complete, or NP-hard.

g) If Y is in P, then X is in P.
Yes. X must be in P because X reduces to Y.

Therefore, answer is d) and g).

Problem #2

a) SUBSET-SUM ≤p COMPOSITE.

This is invalid.
A NP-complete problem doesn't necessarily reduce to a problem in NP.

b) If there is an O(n^3) algorithm for SUBSET-SUM, then there is a polynomial time algorithm for COMPOSITE.

This is valid.
SUBSET-SUM is a polynomial algorithm problem. Since SUBSET-SUM is NP-complete, SUBSET-SUM is in NP, and all problems in NP reduce to SUBSET-SUM. So, all problems in NP bound to polynomial algorithm. COMPLETE is in NP. Therefore, there is a polynomial time algorithm for COMPOSITE.

c) If there is a polynomial algorithm for COMPOSITE, then P = NP.

This is invalid.
COMPOSITE is in NP. COMPOSITE is one of instances in problems in NP. Even though COMPOSITE can be solved in polynomial time, this doesn't conclude that all other problems in NP can be solved in polynomial time. Therefore, we can't say P = NP.

d) If P ≠ NP, then no problem in NP can be solved in polynomial time.

This is invalid.
The equivalent statement is '*if some problems in NP can be solved in polynomial time, then P = NP*'. This is not true. This can't conclude that P = NP. This statement is logically equivalent to c). We can use the same reason as the one to c).

Problem #3

a) 3-SAT ≤p TSP.

True.

3-SAT is NP-complete and is a problem in NP. TSP is NP-complete, which implies that all problems in NP reduce to TSP. Since 3-SAT is in NP, 3-SAT reduces to TSP.

b) If P ≠ NP, then 3-SAT ≤p 2-SAT.

False.

2-SAT is a polynomial algorithm problem, which means 2-SAT is in P. 3-SAT is in NP. Since P is in NP and P is not equal to NP as the assumption, a problem in NP can't reduce to a problem in P which is not equal to NP. This is structurally impossible. Therefore, 3-SAT cannot reduce to 2-SAT.

c) If P ≠ NP, then no NP-complete problem can be solved in polynomial time.

True.

Equivalent statement: If some NP-complete problems can be solved in polynomial time, P = NP. This statement implies that all problems in NP are reducible to the NP-complete problem. Since NP-complete is solvable in polynomial time, problems in NP are also solvable in polynomial time, which is P = NP.

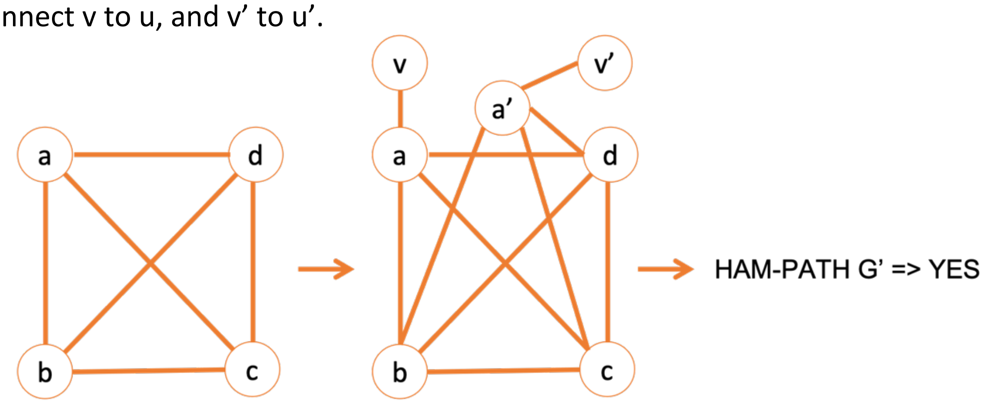
Problem #4

1) Show HAM-PATH is in NP.
If we are given a certificate solution, we can go through every vertex in the graph by checking starting vertex u is adjacent to ending vertex v. We also check the validity of the edge between two vertices, and mark a visited vertex. And we can also mark visited vertex. By using an adjacency list, the time efficiency would be at most O(V+E) where V is the number of vertices and E is the number of edges. This will be completed in polynomial time. Therefore HAM-PATH is in NP.

2) Show that HAM-CYCLE <=p HAM-PATH.

a) We need to show a polynomial algorithm to transform HAM-CYCLE into an instance of HAM-PATH.

Given a graph G, we produce a new graph G' such that G has a Hamiltonian cycle path if and only if G' has a Hamiltonian path. This is done by selecting an arbitrary vertex u in G, and adding a copy u' of u with all edges of it. And add vertices v and v' to the G' and connect v to u, and v' to u'.



G (HAM-CYCLE) G' (HAM-PATH)

HAM-CYCLE: a, b, c, d, a HAM-PATH: v, a, b, c, d, a', v'

b) Prove HAM-CYCLE can be solved by using HAM-PATH. We need to show that the graph G has a HAM-CYCLE *if and only if* the graph G' has a HAM-PATH path.

i) If G has a HAM-CYCLE path, G' has a HAM-PATH path. We get a HAM-PATH path by starting at v and follow the cycle path from G. Instead of ending up at d, the path goes to d, a' and v'. The endpoints are v and v' that shows it's a Hamiltonian path. The path is <v, a, b, c, d, a', v'>.

ii) If G' has a HAM-PATH path, G has a HAM-CYCLE path. By disregarding points v, v', a', we can obviously get a cycle path <a, b, c, d, a>.

Since 1) and 2) are true, HAM-PATH is NP-complete.

Problem #5

1) Show LONG-PATH is in NP.

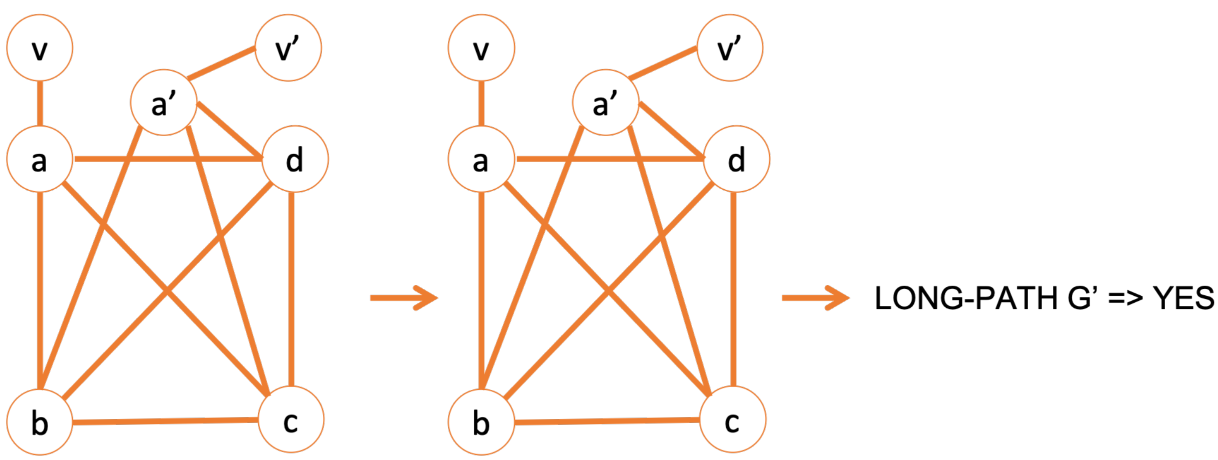
If we are given a certificate solution of vertices, <u, …, v>, we can use the same logic as we find a route from HAM-PATH, and we count the cumulative weight as we go through to see if it's at least k. Assuming that every edge's weight is the same, to find the longest path we need to check all vertices and mark visited vertex not to be repeated. If the total weight of edges is greater than or equal to k, there exists a solution, otherwise there's none. This can be done in polynomial time. So, LONG-PATH is in NP.

2) Show R <=p LONG-PATH for some NP-complete R.

a) We will use HAM-PATH to show it is reducible to LONG-PATH because it has a similar structure to LONG-PATH. HAM-PATH is in NP-complete as shown in previous problem.

b) Show HAM-PATH <=p LONG-PATH.

We need to show a polynomial algorithm to transform HAM-PATH into an instance of LONG-PATH. We can create a graph G' which is identical to graph G, and set an integer m = the number of vertices in path solution - 1. In this case, G' has 7 vertices, and its path has minimum length of 7-1=6.



G (HAM-PATH)
HAM-PATH: v, a, b, c, d, a', v'

G' (LONG-PATH)
LONG-PATH: v, a, b, c, d, a', v'
m = 7 - 1 = 6

c) We need to show that graph G has a HAM-PATH if and only if the graph G' has a LONG-PATH.

> **i)** If G has a HAM-PATH, then G' has a LONG-PATH. This is true because as long as the graph has visited all vertex not repeatedly, we have the total weight of m-1 where m is the number of vertices in the path.
> **ii)** If G' has a LONG-PATH, then G has a HAM-PATH. If there exists longest path, it's obvious that there exists HAM-PATH path in which every vertex has been visited and the path has different end points.

Since 1) and 2) are true, LONG-PATH is NP-Complete.

<citations>
http://www.csc.kth.se/utbildning/kth/kurser/DD2354/algokomp10/Ovningar/Exercise6_Sol.pdf
https://www.csie.ntu.edu.tw/~lyuu/complexity/2016/20161129s.pdf