Question 1:

Let X and Y be two decision problems. Suppose we know that X reduces to Y in polynomial time. Which of the following statements are true? Explain

Y is at least as hard as X. If you can solve Y, you can solve X --> [X <=p Y]

A. If Y is NP-complete then so is X.
B. If X is NP-complete then so is Y.
C. If Y is NP-complete and X is in NP then X is NP-complete.
D. If X is NP-complete and Y is in NP then Y is NP-complete. True: Given that Y is harder than X, if X is NP Complete, then Y is by definition NP-Complete as well. Y can also be part of NP (in addition to being part of NP-Complete). As well, x can be reduced to Y.
E. If X is in P, then Y is in P.
F. If Y is in P, then X is in P. True: if Y is in P, then X must be as well as X is easier than Y and P is the easiest set of problems (when considering P, NP, NP-Complete).
G. X and Y can't both be in NP

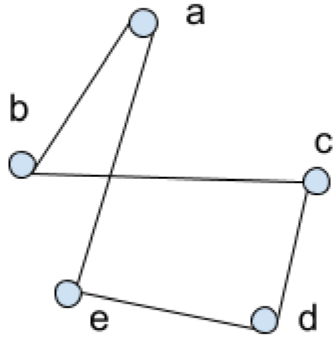
Question 2:

A Hamiltonian path in a graph is a simple path that visits every vertex exactly once. Show that HAM-PATH = { (G, u, v): there is a Hamiltonian path from u to v in G } is NP-complete. You may use the fact that HAM-CYCLE is NP-complete.

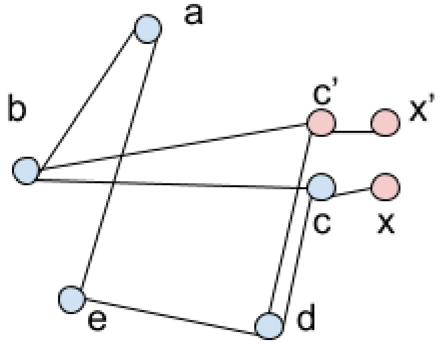
HAM-PATH will be henceforth referred to as Q

1. Showing that Q ∈ NP: A polynomial time algorithm for checking Q looks like:
    a. For each vertex in path:
        i. Remove that vertex from the master set of vertices
    b. If any are left in the master set:
        i. There is an error, and solution is invalid
    c. Else
        i. Success!
2. Showing that R reduces to Q for some R ∈ NP-Complete:
    a. R will be the NP-Complete function HAM-CYCLE described above.
    b. Algorithm for reducing arbitrary R into a Q problem:
        i. Choose vertex in G=(v,e) in R:
            1. Clone that vertex v into v’
            2. To v’ connect new vertex x’
            3. To vertex v, connect new vertex x
            4. Now, this is a hamiltonian path problem between vertices x and x’.

c. To prove that R reduces to Q, consider two graphs, G_c and G_p:



G



G'

Here, we suppose that G has a hamiltonian cycle (c, b, a, e, d, c). If we add vertex c' (clone of vertex c), and vertices x' and x as shown, then we can be certain that there exists a hamiltonian path that connects x and x', as c and c' are identical vertices. Therefore, any hamiltonian cycle (such as the one shown above) will be able to 'start' or 'end' at either c or c', which, when connected to x and x' implies the existence of a hamiltonian path.

i. If we consider that we have G', G' must be a valid solution to the Hamiltonian Path as G was established to be a valid Hamiltonian cycle, and we have cloned one of the cycle nodes, and 'broken out' the endpoints, ensuring that the graph without x and x' is still a hamiltonian cycle, thus when we start at x and end at x', this becomes a Hamiltonian path.
ii. If we consider that we have G, this is a valid hamiltonian cycle by definition, and cloning a vertex c to c' and breaking out the endpoints will create G'. Therefore, removing x, x' will leave us with a valid Hamiltonian cycle, and removing cloned node c' will have no effect on the nature of this as a hamiltonian cycle.

Question 3:

K-COLOR. Given a graph G = (V,E), a k-coloring is a function c: V -> {1, 2, … , k} such that c(u) ≠ c(v) for every edge (u,v) ∈ E. In other words the number 1, 2, .., k represent the k colors and adjacent vertices must have different colors. The decision problems K-COLOR asks if a graph can be colored with at most K colors.

A. The 2-COLOR decision problem is in P. Describe an efficient algorithm to determine if a graph has a 2-coloring. What is the running time of your algorithm?
   a. Select vertex v ∈ V
      c = 0
      Assign v color c

Enqueue v
While the queue is not empty:
Pop a from queue
c = c xor 1
For each unvisited neighbor to a, v, do:
Visit v
Color v with c
Add v to the queue

b. For each vertex v in the graph:
For each adjacent vertex a to v:
If v and a share the same color, then the graph cannot be 2-colored.
c. This algorithm was inspired by a post on CheggStudy. Also, this algorithm has a running time of O(V+E) as the main part of this algorithm is essentially implementing BFS so that takes O(V+E). The second part also takes O(V+E) so in total, this makes 2*O(V+E) == O(V+E).

B. The 3-COLOR decision problem is NP-complete by using a reduction from SAT. Use the fact that 3-COLOR is NP-complete to prove that 4-COLOR is NP-complete.
    a. 4-COLOR will be henceforth referred to as Q
    b. Showing that Q $\in$ NP:
        i. For each vertex v in the graph:
        For each adjacent vertex a to v:
        If v and a share the same color, then the graph cannot be k-colored.
    c. Showing that R reduces to Q for some R $\in$ NP-Complete
        i. R will in this case be the 3-COLOR, NP-Complete problem.
        ii. Take a valid 3-COLOR graph G. Add a new vertex x and connect it to all points in the graph G. Color x a new color not already in G to form G'. G' is 4-COLOR, and G is 3-COLOR.
        iii. Prove R(G) iff Q(G'):
            1. If we have graph G as a 3-COLOR graph, and we add x to form G', we obviously have a 4-COLOR graph as all nodes except x fulfil the 3-COLOR requirements, and x counts as the 4th color, making this a 4-COLOR graph
            2. If we have graph G' as a valid 4-COLOR graph, and we remove vertex x, we will end up with a 3-COLOR graph, as no other nodes have the color of x.