1. Consider the following problem called BoxDepth: Given a set of n axis parallel rectangles (this means that the length and breadth of the rectangle are parallel to the x and y-axes), how big is the largest subset of these rectangles that contain a common point.
   (a) Write down the decision version of the above problem.
   (b) Give a polynomial time reduction from BoxDepth to the problem of computing the largest clique in a graph. In fact, you can try a reduction between the respective decision problems.
   (c) Describe a polynomial-time algorithm to solve BoxDepth. Any polynomial-time algorithm would suffice. Don't try to optimize the running time.
   (d) Does this give a polynomial time algorithm for the problem of computing maximum size clique of a given graph? Why? or Why not?

2. A Hamiltonian path in an undirected graph $G(V, E)$ is a simple path in G that passes through each vertex in G exactly once. Similarly, a Hamiltonian cycle is a cycle that starts from a vertex $v \in V$, passes through every other vertex exactly once and returns to v. The HamPath and HamCycle problem is to decide if a given input graph G has a Hamitonian path and Hamiltonian cycle respectively. Note that these are decision problems.
   (a) Show that HamPath $\leq$ HamCycle. (*Hint: add a new vertex and connect to all vertices in the graph*)
   (b) Show that HamCycle $\leq$ HamPath.(*Hint: Similar to previous part, try adding two new vertices and one duplicate for every vertex. You need to decide how to connect these new vertices in such a way that a hamiltonian path in the new graph will help you recover a hamiltonian cycle in the original graph.*)

3. Given a graph $G(V, E)$, a k-coloring of the graph is an assignment of colors $c : V \to \{1, 2, \dots k\}$ such that if $(u, v) \in E$, then $c(u) \neq c(v)$. The 3-Colorability problem is to test if a given graph has a 3-coloring.
   (a) Argue that 3-Colorability is in NP. In fact, this problem is known to be at least as hard as the Clique problem that we saw in the class (But this exercise is not about that).
   (b) Suppose that you are given a subroutine, that takes as input a graph G, and tells you whether or not the graph has a 3-coloring or not. Using this subroutine, and assuming that a call to the subroutine takes constant time, design a polynomial time algorithm to compute a 3-coloring $c : \{1, 2, 3\}$, if one exists. *Hint: The idea is to force one color each for a vertex and see if it can be extended to a 3-oloring of all the vertices satisfying the condition of the coloring. We will start with a vertex v for this. Add three new vertices to the graph a, b, c to the graph G with a triangle between them. Any 3-coloring will have to*

assign three different colors to a, b and c. Why? Now construct three different graphs G_a, G_b, G_c, where we add only the edge (a, v), (b, v), (c, v) into G respectively. Note that if G had a 3-coloring, then what can you say about these three graphs having 3-coloring? Use the subroutine. This will give you a color to fix for the vertex v and proceed to the next vertex.