1. A CNF formula φ is in k-CNF form if each clause of φ has exactly k literals. The k-SAT problem is to decide if a given k-CNF formula is satisfiable. In this problem we will reduce 3-SAT to 4-SAT. Note that we have seen in lecture how to reduce 4-SAT (and more generally SAT) to 3-SAT.

    - Suppose φ is a 3-CNF formula. Consider the following reduction where we add a new variable u and replace each clause $c = (\ell_1 \vee \ell_2 \vee \ell_3)$ by a new clause $c' = (\ell_1 \vee \ell_2 \vee \ell_3 \vee u)$. Note that we are using the same variable for each clause. Let φ' be the new formula obtained from φ via this recution. Prove that φ' is satisfiable if φ is satisfiable. Given an example to show that φ' is satisfiable but φ is not satisfiable.
    - Obtain a correct reduction by altering the preceding one and prove that φ' is satisfiable if and only if φ is satisfiable.

2. A path P in a directed graph G is called a Hamiltonian path if it contains all the vertices of G. The Hamiltonian Path problem is the following: given G, does G contain a Hamiltonian path? The Longest s-t Path problem is the following: given a directed graph $G = (V, E)$ two nodes $s, t \in V$ and an integer k, is there a simple path of length at least k from s to t in G?

    - Assuming that you have a black box algorithm for the Longest s-t Path problem describe a polynomial-time algorithm for the Hamiltonian Path problem.
    - Did you use a mapping reduction for the preceding part? If not, give a mapping reduction from the Hamiltonian Path problem to the Longest s-t Path problem. That is, given G your reduction should output a graph $G' = (V', E')$, two nodes $s, t \in V'$ and an integer k such that G' has an s-t simple path of length at least k if and only if G has a Hamiltonian Path.

3. Self-reduction. We focus on decision problems even when the underlying problem we are interested in is an optimization problem. For most problems of interest we can in fact show that a polynomial-time algorithm for the decision problem also implies a polynomial-time algorithm for the corresponding optimization problem. To illustrate this consider the maximum independent set (MIS) problem.

    - Suppose you are given a algorithm that given a graph H and integer ℓ outputs whether H has an independent set of size at least ℓ. Using this algorithm as a *black box*, describe a polynomial time algorithm that given a graph G and integer k outputs an independent set of size k in G if it has one. Note that you can use the black box algorithm more than once. *Hint:* What happens if you remove a vertex v and the independent set size does not decrease? What if it does?
    - How would you efficiently find a maximum independent set in a given graph G using the black box?