Carefully read the questions.
On the last quiz, we asked you to show that 2-SCHED \( \leq_{p}^{m} \) 2-SAT, by providing a polynomial-time computable function that maps problems in 2-SCHED and turns them into problems in 2-SAT, and maps problems that aren’t in 2-SCHED to something that isn’t in 2-SAT. Show that you can map 2-COLOR to a language containing only one string, \{\epsilon\}.

Write down the reduction here, from 2-COLOR to \{\epsilon\} using \( \leq_{p}^{m} \) notation:

Write down the domain and range of the function, f, that does this mapping:

Now, write down a polynomial-time algorithm that, when given as input a graph that is 2-COLORable, outputs something in the set \{\epsilon\}, and when given input that isn’t such a graph, outputs something not in the set \{\epsilon\}. By writing such an algorithm that always completes, you are proving the existence of a polynomial-time computable function f that creates such a mapping, and therefore proving the reduction:
In class, we showed a way of turning 3-SAT problems into 3-COLOR problems; the output, a graph, could be colored according to the rules of 3-COLOR if and only if the 3-SAT formula had a satisfying assignment.

Here, let us define CLIQUE to be the following problem: given a graph $G$, and some number $k$, is there a set of $k$ vertices in $G$ which are each connected to the others by an edge (known as a clique)? More formally, $\text{CLIQUE} = \{<G,k> | G \text{ has a complete subgraph of size } k\}$.

Formally express the reduction, using $\leq_{m}^{P}$, that shows 3-SAT is no more powerful than CLIQUE.

Write down the domain and range of a function $f$ which will be the polynomial map for the reduction stated above:

We want to prove that there’s some assignment of variables that makes every clause true, if there’s some clique in the output graph, so we want a clique to imply that there’s an assignment which makes every clause true. The graph has three nodes per clause, one for every literal; describe a way to connect them, AND define the value of $k$, so that this function works as the reduction.
coNP is the class of all languages whose complements are in NP, so a language is in coNP if its complement is in NP. More generally, NP and coNP are classes of languages (or sets of languages), and, formally, if C is a class of languages, coC is another class of languages, such that $L \in \text{coC} \iff L^c \in C$ (for $L^c$ being the complement of the language L).

Prove that, for any two classes C and D, if $C \subseteq D$, then $\text{coC} \subseteq \text{coD}$.