**New Phrases**

- We say a language is in \( P \) if there exists a polynomial time algorithm that decides the language.
- We say a problem is in \( NP \) if there exists a polynomial time verifier TM \( V \) such that for all \( x \in \Sigma^* \), \( x \) is in \( L \) if and only if there exists a polynomial length certificate \( P \) such that \( V(x, p) = 1 \).
- A problem \( A \) reduces in polynomial time to a problem \( B \) if, given an algorithm to solve \( B \), we can use it to solve \( A \) in polynomial time. If this is the case, we write this as \( A \leq_P B \).
- A problem \( Y \) is \( NP \)-hard if for every problem \( X \in NP \), \( X \leq_P Y \).
- A problem is \( NP \)-complete if it is both in \( NP \) and \( NP \)-hard.

**NP is Not Not Polynomial**

Show that \( P \) is contained in \( NP \).

**No Privacy**

DOUBLE-CLIQUE: Given a graph \( G = (V, E) \) and a natural number \( k \), does \( G \) contain two vertex-disjoint cliques of size \( k \) each?

Show DOUBLE-CLIQUE is NP-Complete.

**No Pun**

VERTEX-COVER: Given a graph \( G = (V, E) \), and a natural number \( k \), does there exist a subset \( U \subseteq V \) with \( |U| \leq k \) such that every edge \( e \in E \) has at least one of its endpoints in \( U \)?

Show VERTEX-COVER is NP-complete. (Hint: Reduce from 3SAT)

**Never Pausing**

(a) Prove that the Halting Problem is \( NP \)-hard.

(b) (Bonus) Consider the HALTS-KINDA-SOON problem: Given a turing machine \( T \) and an input \( x \), does it halt in \( 2^{|x|} \) steps?

Show that HALTS-KINDA-SOON is not in \( P \). That is, for any positive \( k \), HALTS-KINDA-SOON is not solvable in time \( O(n^k) \).