Thm 7.27 [Cook-Levin]: SAT is in P iff $P = NP$. 
Def 7.29: Language $A$ is **polynomial-time reducible** to language $B$, written $A \leq_p B$, if a polynomial-time computable function $f : \Sigma^* \rightarrow \Sigma^*$ exists such that for every $w$,

$$w \in A \iff f(w) \in B$$

The function $f$ is called **polynomial-time reduction** of $A$ to $B$.

Thm 7.31: If $A \leq_p B$ and $B \in P$, then $A \in P$. 

bec. poly # steps in $f$ \poly # steps in $B$ \poly # steps in $A$ runs in polynomial time
**Thm:** HAMPATH is polynomial-time reducible to LONGESTPATH, where

\[
\text{LONGESTPATH} = \{ \langle G, s, t, k \rangle \mid G \text{ is digraph and there exists a path from } s \text{ to } t \text{ of length } \geq k \}\]
Thm: HAMCYLE is polynomial-time reducible to TSP, where

HAMCYLE = \{ <G> \mid G \text{ is a graph that contains a cycle through all vertices} \}

TSP = \{ <G_w,k> \mid G_w \text{ is a complete weighted graph that contains a cycle through all vertices of length } \leq k} \}
Thm 7.32: 3SAT is polynomial-time reducible to CLIQUE, where

\[ 3\text{SAT} = \{ \langle \phi \rangle \mid \phi \text{ is a satisfiable 3-cnf formula} \} \]

\[
\phi = (x_1 \lor \neg x_2) \land (\neg x_1 \lor x_2 \lor x_3) \land (x_2 \lor x_3) \land (\neg x_1 \lor x_2 \lor x_3)
\]

1. For every clause create vertices corresponding to every variable in the clause.
2. Connect a vertex with all other vertices in the other clauses, except don't connect to own negation.

\[ 3\text{SAT} \leq_p \text{CLIQUE} \]
Def 7.34: A language B is **NP-complete** if it satisfies both conditions:

- B is in NP, and
- every A in NP is polynomial-time reducible to B.
NP-Completeness

Def 7.34: A language $B$ is **NP-complete** if it satisfies both conditions:

- $B$ is in NP, and
- every $A$ in NP is polynomial-time reducible to $B$.

Thm 7.35: If $B$ is NP-complete and $B \in P$, then $P = NP$. 
**Def 7.34:** A language $B$ is **NP-complete** if it satisfies both conditions:

- $B$ is in NP, and
- every $A$ in NP is polynomial-time reducible to $B$.

**Thm 7.36:** If $B$ is NP-complete and $B \leq_p C$ for some $C \in \text{NP}$, then $C$ is NP-complete.
Def 7.34: A language $B$ is **NP-complete** if it satisfies both conditions:
- $B$ is in NP, and
- every $A$ in NP is polynomial-time reducible to $B$.

Thm 7.37 [Cook-Levin]: SAT is NP-complete.

Since we showed $\text{SAT} \leq_{p} \text{CLIQUE}$
then $\text{CLIQUE}$ is NP-complete.

**Note:** a long list of known NP-complete problems.