Exercise 9.1 (P)

(a) Show that $P$ is closed under union, complement, and concatenation.

(b) The complexity class $\text{coP}$ contains all languages $L$ whose complement is in $P$. Formally, $\text{coP} = \{L \mid L \in P\}$. Is $P = \text{coP}$?

Solution:

(a) • **Union:** Let $L_i (i = 1, 2)$ be two languages in $P$ and let $M_i$ be a DTM that accepts $L_i$ in polynomial time $p_i$ where $p_i$. We can construct a DTM $M$ with two tapes that works as follows: on input $w$ copy $w$ to the second tape and simulate $M_1$ on the first tape. If $M_1$ accepts within time $p_1$, accept. Otherwise simulate $M_2$ on the second tape. If it accepts, accept. This can obviously be done in polynomial time: Copying the input takes linear time (running once over the input and then moving the head back) and both simulations can be done in polynomial time (because $L_1, L_2 \in P$). Obviously $M$ accepts a word $w$ iff $w$ in $L_1 \cup L_2$. Since $M$ can be simulated on a single-tape TM with only quadratic overhead, there is an equivalent single-tape TM that accepts the same language in polynomial time.

• **Complement:** Let $L \in P$ be a language that is accepted by a DTM $M$ within time $p$ where $p$ is a polynomial. We can construct a DTM $M'$ that simulates $M$ on its input and accepts if $M$ did not accept within time $p$. Obviously, $M'$ accepts $L$ in polynomial time.

• **Concatenation:** Let $L_1$ and $L_2$ be languages in $P$, and suppose we want to recognise their concatenation. Suppose we are given an input of length $n$. For each $i$ between 1 and $n-1$, test whether positions 1 through $i$ hold a string in $L_1$ and positions $i+1$ to $n$ hold a string in $L_2$. If so, accept; the input is in $L_1L_2$. If the test fails for all $i$, reject the input.

(b) Yes: as usual, we show that $P \subseteq \text{coP}$ and $\text{coP} \subseteq P$.

- $\text{coP} \subseteq P)$ Let $L$ be a language in $P$. Since $P$ is closed under complement, we know that $L^c \in P$. We can conclude that $L = (L^c)^c \in \text{coP}$

- $P \subseteq \text{coP}$ Analogously, let $L$ be a language in $\text{coP}$. Then $L^c$ is in $P$. Since $P$ is closed under complement, $(L^c)^c$ is in $P$ and we can conclude that $L \in P$.

Exercise 9.2 (Reduction)

Given an undirected graph $G := (G, E)$ and an integer number $0 \leq k \leq |G|$, the following $NP$-complete problems have been introduced in the lectures (see 07.pdf, slides 80-82-84):

**Clique:** Does $G$ contain a clique of size at least $k$? That is, there exist a set $C \subseteq G$ so that $(u, v) \in E$ for every $u, v \in C$ ($u \neq v$) and $|C| \geq k$.

\footnote{Wlog, computational time can be supposed monotonically increasing wrt the input's length.}
**IndSet**: Does $G$ contain an independent set whose size is at least $k$? In other words, does $G$ admit a subset $I \subseteq G$ with $|I| \geq k$ and such that there exists no edge $\langle u, v \rangle$ whenever $u, v$ lie in $I$?

**VertexCover**: Does $G$ contain a vertex cover of size at most $k$? That is, is it possible to find a set $C \subseteq G$ so that $|C| \leq k$ and for every edge $\langle u, v \rangle \in E$, $u \in C$ or $v \in C$?

Prove the following statements:

(a) Clique $\leq_P$ IndSet  
**Hint**: consider the complement graph.  
**Solution**: let’s consider the complement graph. By definition, given a graph $G := (G, E)$, we call complement graph of $G$ the graph $G^c := (G, E')$ where $E' := \{\langle u, v \rangle \mid \langle u, v \rangle \notin E\}$. Roughly speaking, an edge in $G^c$ connects two vertices if and only if those vertices are independent in $G$. It is trivial to see that the following relation holds:

\[ X \text{ is a clique of } G \iff X \text{ is an independent set of } G^c. \]

Thus, to prove statement (a) we just need to provide a Turing machine $M$ that converts $G$ into $G^c$ in polynomial time with respect to the size of the graph $|G|$. However, such conversion can be performed at least in $O(|G|^2)$ just by checking for every vertex $u \in G$ which vertex $v \in G$ satisfies $\langle u, v \rangle \notin E$.

As a consequence there exist a function $f_c(\langle G, k \rangle) = \langle G^c, k \rangle$ and $f_c(\langle G, k \rangle) \in IndSet$. The statement is proved.

(b) IndSet $\leq_P$ VertexCover  
**Hint**: consider the relation between vertex covers and independent sets.

**Solution**: let’s prove first a preliminary result. In the above hyphotesis, the following statements are equivalent

\[ X \text{ is an independent set of } G \text{ of size } k \iff G \setminus X \text{ is a vertex cover of } G \text{ of size } |G| - k. \]

$\Rightarrow$ Let $X$ be an independent set and let $\langle u, v \rangle \in E$, then $u$ and $v$ cannot lie both in $X$. That is, either $u$ or $v$ belongs to $G \setminus X$.

$\Leftarrow$ Let’s suppose $G \setminus X$ to be a vertex cover but $X$ not to be an independent set. Accordingly, there must be an edge $\langle u, v \rangle \in E$ so that $u, v \in X$ (and hence $u, v \notin G \setminus X$). This contradicts the definition of vertex cover.

Owing to the above result, we can define the following function $f_c(\langle G, k \rangle) = (G, |V| - k)$. It is easy to see that such conversion can be computed in polynomial time by a Turing machine (at least $f_c \in O(|\langle G, k \rangle|$)). Thus, we can conclude the proof just by observing that the above result implies $\langle G, k \rangle \in IndSet$ iff $f_c(\langle G, k \rangle) \in VertexCover$. 