Exercise 1: P-choice
A language \( L \) is said \( P \)-peek \((L \in Pp)\) if there is a function \( f : \{0,1\}^* \times \{0,1\}^* \rightarrow \{0,1\}^* \) computable in polynomial time such that \( \forall x, y \in \{0,1\}^* : \)
- \( f(x, y) \in \{x, y\} \)
- if \( x \in L \) or \( y \in L \) then \( f(x, y) \in L \)

\( f \) is called the peeking function for \( L \).

1. Show that \( P \subseteq Pp \)
2. Show that \( Pp \) is closed under complementary
3. Show that if there exist \( L \) \( NP \)-hard in \( Pp \), then \( P = NP \)
4. Let \( r \in [0, 1] \) a real number, we define \( L_r \) as the set of words \( b = b_1...b_n \in \{0,1\}^* \) such that \( 0, b_1...b_n \leq r \). Show that \( L_r \in Pp \)
5. Deduce that there exist a non-recursive language in \( Pp \)

Solution:
1. Let there be \( A \in P \). We set \( f(x, y) = x \) if \( x \in A \), and \( f(x, y) = y \) otherwise.
2. Let there be \( A \in Pp \) through \( f \). Then, define \( f'(x, y) = y \) if \( f(x, y) = x \) and \( f'(x, y) = x \) otherwise. \( f' \) is then a peeking function for \( A^c \):
   - if \( x \in A^c \) and \( y \in A^c \), then \( f'(x, y) = y \in A^c \)
   - if \( x \in A^c \) and \( y \in A \), then \( f(x, y) = y \) and \( f'(x, y) = x \in A^c \)
   - if \( x \in A \) and \( y \in A^c \), then \( f(x, y) = x \) and \( f'(x, y) = y \in A^c \) contains a language which is undecidable.
3. Let there be \( A \in Pp \) through \( f \) and \( g \) a reduction from \( SAT \) to \( A \). Here is a polynomial algorithm for \( SAT \) on input \( \phi \) with \( n \) variables, where we denote \( \phi_0 \) (resp. \( \phi_1 \)) the formula \( \phi \) in which the first variable is set to 0 (resp. 1).

   For \( i \) from 1 to \( n \) do
   if \( f(g(\phi_0), g(\phi_1)) = g(\phi_0) \) then \( \phi \leftarrow \phi_0 \)
   else \( \phi \leftarrow \phi_1 \)
   Accept iff \( \phi = \top \)

4. \( f(x, y) = \min(x, y) \) is a valid selection function for \( L_r \)
5. \( Pp \) is not countable as it contains \( L_r \) for any \( r \in [0, 1] \). Thus, \( Pp \) contains a language which is not decidable.

Exercise 2: Complete problems for levels of \( PH \)
Show that the following problem is \( \Sigma_k^p \)-complete (under polynomial time reductions).
\( \Sigma_k^{QBF} : \)
- INPUT: A quantified boolean formula \( \psi := \exists X_1 \forall X_2 \exists...Q_k X_k \phi(X_1, ..., X_k) \), where \( X_1, ...X_k \) are \( k \) disjoint sets of variables, \( Q_k \) is the quantifier \( \forall \) if \( k \) is even, and the quantifier \( \exists \) if \( k \) is odd, \( \phi \) is a boolean formula over variables \( X_1 \cup ... \cup X_k \);
- QUESTION: is the input formula true?
Define a similar problem $\Pi_k \text{QBF}$ such that $\Pi_k \text{QBF}$ is $\Pi_k^P$-complete.

**Solution:**

- If we are given some $X_1, ..., X_k$, we can check in polynomial time if $\phi(X_1, ..., X_k)$ is true. Thus, it is in $\Sigma_k^P$.
- Let there be $A \in \Sigma_k^P$. A can be expressed as follows:

  $$x \in A \iff \exists y_1 \in \{0,1\}^{p(x)} \forall y_2 \in \{0,1\}^{q(x)} Q_k y_k \in \{0,1\}^{p(x)}(x, y_1, ..., y_k) \in B$$

  with $B \in P$.

Let us assume that $Q_k = \exists$, the other case can be done in a similar fashion. Now, $
\exists y_k \in \{0,1\}^{p(x)}(x, y_1, ..., y_k) \in B$ is in $\text{NP}$, so by Cook’s theorem, we have $\phi$ such that:

$$\exists y_k \in \{0,1\}^{p(x)}(x, y_1, ..., y_k) \in B \iff \exists z, \phi(x, y_1, ..., y_k, z)$$

By inspecting Cook’s proof, we can modify $\phi$ such that the input tape $x, y_1, ..., y_{k-1}$ appear as variables in $\phi$. We thus have

$$\exists y_k \in \{0,1\}^{p(x)}(x, y_1, ..., y_k) \in B \iff \exists z, \phi(x, y_1, ..., y_{k-1}, z)$$

And finally:

$$x \in A \iff \exists y_1, \forall y_2, \forall y_{k-1} \exists z, \phi(x, y_1, ..., y_{k-1}, z)$$

**Exercise 3: Oracle machines**

Let $O$ be a language. A Turing machine with oracle $O$ is a Turing machine with a special additional read/write tape, called the oracle tape, and three special states: $q_{\text{query}}, q_{\text{yes}}, q_{\text{no}}$. Whenever the machine enters the state $q_{\text{query}}$, with some word $w$ written on the oracle tape, it moves in one step to the state $q_{\text{yes}}$ or $q_{\text{no}}$ depending on whether $w \in O$.

We denote by $P^O$ (resp. $\text{NP}^O$) the class of languages decided in polynomial time by a deterministic (resp. non-deterministic) Turing machine with Oracle $O$. Given a complexity class $C$, we define $P^C = \bigcup_{C \subseteq C} P^O$ (and similarly for $\text{NP}$).

1. Prove that for any $C$-complete language $L$, $P^C = P^L$ and $\text{NP}^C = \text{NP}^L$.
2. Show that for any language $L$, $P^L = P^L$ and $\text{NP}^L = \text{NP}^L$.
3. Prove that if $\text{NP} = P^{\text{SAT}}$ then $\text{NP} = \text{coNP}$.

**Solution:**

1. We do the proof for $\text{NP}$. Let $B \in \text{NP}^C$, we have $N$ a polynomial NTM for $B$ with an oracle $C$, $C \in C$. We also have a polynomial reduction $f$ such that: $x \in C \iff f(x) \in A$. We build $N'$ for $B$ with oracle $A$, by simulating $N$ and replacing a call $u \in C$ with a call $f(u) \in A$? $f$ is polynomial, so we are still in $\text{NP}$, which concludes the proof.

2. We simply have to swap the states $q_{\text{yes}}$ and $q_{\text{no}}$ in the computation.

3. $P^{\text{SAT}}$ is a deterministic class, so it is closed by complementation, so if $\text{NP} = P^{\text{SAT}}$, $\text{coNP} = \text{NP}$

4. 

**Exercise 4: Collapse of PH**

1. Prove that if $\Sigma_k^P = \Sigma_{k+1}^P$ for some $k \geq 0$ then $\text{PH} = \Sigma_k^P$. (Remark that this is implied by $P = \text{NP}$).

2. Show that if $\Sigma_k^P = \Pi_k^P$ for some $k$ then $\text{PH} = \Sigma_k^P$ (i.e. $\text{PH}$ collapses).

3. Show that if $\text{PH} = \text{PSPACE}$ then $\text{PH}$ collapses.

4. Do you think there is a polynomial time procedure to convert any QBF formula into a QBF formula with at most 10 variables?
Solution:

1. We assume that $\Sigma^P_k = \Sigma^P_{k+1}$ for some $k \geq 0$, we prove by induction that $\forall t \geq k, \Sigma^P_k = \Sigma^P_j$, for $j = i$, it is directly correct. For $j > i$, $\Sigma^P_j = \text{NP}^\Sigma^P_{j-1} = \text{NP}^\Sigma^P_i$ by induction, and thus $\Sigma^P_j = \Sigma^P_{i+1}$. By hypothesis, we then have $\Sigma^P_j = \Sigma^P_i$.

2. With the previous question, we just have to prove that $\Sigma^P_k = \Sigma^P_{k+1}$.

Let there be $A \in \Sigma^P_{k+1}$. $A$ can be expressed as follows:

$$x \in A \iff \exists y_1 \in \{0, 1\}^{p(x)} \forall y_2 \in \{0, 1\}^{p(x)} \forall y_{k+1} \in \{0, 1\}^{p(x)} (x, y_1, ..., y_{k+1}) \in B$$

with $B \in \mathbb{P}$. On input $(x, y_1)$, decide if $\forall y_2 \in \{0, 1\}^{p(x)} \forall y_{k+1} \in \{0, 1\}^{p(x)} (x, y_1, ..., y_{k+1}) \in B$ is a problem in $\Pi^P_k = \Sigma^P_k$. We can thus rewrite it as, with $C \in \mathbb{P}$:

$$\exists y_2 \in \{0, 1\}^{p(x)} \forall y_{k+1} \in \{0, 1\}^{p(x)} (x, y_1, ..., y_{k+1}) \in C$$

Finally:

$$x \in A \iff \exists y_2, y_{k+1} \in \{0, 1\}^{p(x)} \forall y_{k+1} \in \{0, 1\}^{p(x)} (x, y_1, ..., y_{k+1}) \in B$$

with $B \in \mathbb{P}$. And this is the expression of a problem in $\Sigma^P_k$. Finally, $\Sigma^P_k = \Sigma^P_{k+1}$.

3. If $\text{PH} = \text{PSPACE}$, then QBF is in $\Sigma^P_k$ for some $k$. But QBF is a complete problem for PSPACE, and thus $\text{PH}$. Let there be $B \in \text{PH}$, it can be reduced to $QBF \in \Sigma^P_k$, so $B \in \Sigma^P_k$, and $\text{PH} = \Sigma^P_k$.

4. It is unlikely that $\text{PH}$ collapses, and the statement would imply the previous question.

Exercise 5: Relativization

Show that there is an oracle $O$ such that $\text{P}^O = \text{NP}^O$.

Solution:

We have $\text{PSPACE} \subseteq \text{NP}^{\text{PSPACE}}$, we must show the converse. Let there be $N$ a polynomial NTM with oracle $A \in \text{PSPACE}$. We can simulate $N$ in $\text{PSPACE}$ on input $x$ by:

- enumerate all possible path of $N^A(x)$
- For each of them, compute the oracle calls
- accept if one of the path accepts.

Each path is in polynomial size, thus the enumeration is, and the oracle calls are $\text{PSPACE}$. We do have $\text{NP}^{\text{PSPACE}} \subseteq \text{PSPACE}$.