**Complexity Theory**
Easter 2013
Suggested Exercises 3

1. **Self-Reducibility**. *Self-reducibility* refers to the property of some problems in $L \in \mathsf{NP}$, where the problem of finding a *witness* for the membership of an input $x$ in $L$ can be reduced to the decision problem for $L$. This question asks you to give such arguments in two specific instances.

   (a) Show that, given an oracle (i.e. a black box) for deciding whether a given graph $G = (V, E)$ is Hamiltonian, there is a polynomial-time algorithm that, on input $G$, outputs a Hamiltonian cycle in $G$ if one exists.

   (b) Show that, given an oracle for deciding whether a given graph $G$ is 3-colourable, there is a polynomial-time algorithm that, on input $G$, produces a valid 3-colouring of $G$ if one exists.

2. Show that a language $L$ is in $\mathsf{co\text{-}NP}$ if, and only if, there is a nondeterministic Turing machine $M$ and a polynomial $p$ such that $M$ halts in time $p(n)$ for all inputs of length $x$, and $L$ is exactly the set of strings $x$ such that *all* computations of $M$ on input $x$ end in an accepting state.

3. Define a *strong* nondeterministic Turing machine as one where each computation has three possible outcomes: accept, reject or maybe. If $M$ is such a machine, we say that it accepts $L$, if for every $x \in L$, every computation path of $M$ on $x$ ends in either accept or maybe, with at least one accept *and* for $x \notin L$, every computation path of $M$ on $x$ ends in reject or maybe, with at least one reject.

   Show that if $L$ is decided by a strong nondeterministic Turing machine running in polynomial time, then $L \in \mathsf{NP} \cap \mathsf{co\text{-}NP}$.

4. We saw in the lectures that if there is a one-way function, then there is a language $L$ in $\mathsf{UP}$ that is not in $\mathsf{P}$. Suppose that the $\mathsf{RSA}$ function described in the lecture notes (page 38) is a one-way function. What is the language $L$ that can then be proved to be in $\mathsf{UP} \setminus \mathsf{P}$?

5. Consider the algorithm presented in the lecture which establishes that $\mathsf{Reachability}$ is in $\mathsf{SPACE}((\log n)^2)$. What is the time complexity of this algorithm?

   Can you generalise the time bound to the entire complexity class? That is, give a class of functions $F$, such that

$$\mathsf{SPACE}((\log n)^2) \subseteq \bigcup_{f \in F} \mathsf{TIME}(f)$$

6. Show that, for every nondeterministic machine $M$ which uses $O(\log n)$ work space, there is a machine $R$ with three tapes (`input`, `work` and `output`) which works as follows. On input $x$, $R$ produces on its output tape a description of the configuration graph for $M, x$, and $R$ uses $O(\log |x|)$ space on its work tape.

   Explain why this means that if Reachability is in L, then L = NL.

7. Consider the language $L$ in the alphabet $\{a, b\}$ given by $L = \{a^n b^n \mid n \in \mathbb{N}\}$. $L \notin \mathsf{SPACE}(c)$ for any constant $c$. Why?

8. On page 39 of the notes, a number of functions are listed as being constructible. Show that this is the case by giving, for each one, a description of an appropriate Turing machine.

   Prove that if $f$ and $g$ are constructible functions and $f(n) \geq n$, then so are $f(g)$, $f + g$, $f \cdot g$ and $2^f$.

9. For any constructible function $f$, and any language $L \in \mathsf{NTIME}(f(n))$, there is a nondeterministic machine $M$ that accepts $L$ and all of whose computations terminate in time $O(f(n))$ for all inputs of length $n$. Give a detailed argument for this statement, describing how $M$ might be obtained from a machine accepting $L$ in time $f(n)$.

10. In the lecture, a proof of the Time Hierarchy Theorem was sketched. Give a similar argument for the following Space Hierarchy Theorem:

    > **Space Hierarchy.** For every constructible function $f$, there is a language in $\mathsf{S}PACE(f(n) \cdot \log f(n))$ that is not in $\mathsf{S}PACE(f(n))$.

11. Show that, if $\mathsf{SPACE}((\log n)^2) \subseteq \mathsf{P}$, then $\mathsf{L} \neq \mathsf{P}$. (Hint: use the Space Hierarchy Theorem from Exercise 3 above.)

12. POLYLOGSPACE is the complexity class

$$\bigcup_k \mathsf{SPACE}((\log n)^k).$$

    (a) Show that, for any $k$, if $A \in \mathsf{SPACE}((\log n)^k)$ and $B \leq_L A$, then $B \in \mathsf{SPACE}((\log n)^k)$.

    (b) Show that there are no POLYLOGSPACE-complete problems with respect to $\leq_L$. (Hint: use (a) and the space hierarchy theorem).

    (c) Which of the following might be true: P $\subseteq$ POLYLOGSPACE, P $\supseteq$ POLYLOGSPACE, P = POLYLOGSPACE?