1. Polynomial Time Exercise.

Let $L$ be a problem in $P$. Let $L'$ be equal to $L$, except on a constant number of instances. Show that $L'$ is also in $P$.

(For example, suppose $L$ is the problem CONN of determining if a graph $G$ is strongly connected. Let $L'$ be defined as equal to: CONN($G$) for all inputs $G$ of size $|G| \geq 100$, and HAMILTONIAN($G$) for inputs $G$ of size $|G| < 100$, where HAMILTONIAN($G$) is the Hamiltonian cycle problem on input $G$. Then this question asserts that $L'$ is polynomial time solvable. Is it?)

2. $\text{SPACE}(n)$ vs. $P$

Show that $\text{SPACE}(n) \neq P$. (Hint: Use the fact that $\text{SPACE}(n) \neq \text{SPACE}(n^2)$. Why is this true?)

3. An Average-Case Time Hierarchy.

Let $f, g : \mathbb{N} \to \mathbb{N}$ be such that $f(n) \log f(n) = o(g(n))$, and $g$ is time-constructible. Show that there is a language $L \in \text{DTIME}(g(n))$ with the following property: For every machine $M$ running in time $f(n)$, there is a constant $\varepsilon_M$ such that for all sufficiently large $n$, $M$ errs in deciding $L$ on at least $\varepsilon_M$ fraction of inputs of length $n$.

Note that the constant $\varepsilon_M$ may depend on the machine.

4. Space-efficient Boolean matrix multiplication and consequences.

Given two $n \times n$ matrices $A, B$ with Boolean entries, their boolean product $A \cdot B$ is the matrix $C$ such that

$$C_{ij} = \bigvee_{k=1}^{n} (A_{ik} \land B_{kj})$$

(a) Give a Logspace algorithm to compute $A \cdot B$ given $A$ and $B$. (Food for thought: How can the algorithm take less space than the output length?)

(b) Given matrix $A$ and integer $k$, give a small space algorithm to compute $A^k$, the $k$-th Boolean power of $A$. How much space does your algorithm use? What well-known theorem follows from this algorithm?

5. Approximation and Inapproximability.

The goal of this question is principally to test your ability to pose and use decision problems to capture computational complexity. A secondary goal is to remind you about NP-completeness. (Warning: The question is longer than your answer might need to be!)

(a) The input to the ASYMMETRIC k-CENTER problem is a directed graph $G = (V, E)$ and an integer $k$. The output should be a subset $S$ containing at most $k$ vertices of
that minimizes the quantity $\max_{x \in V} \{\min_{y \in S} \{d(x, y)\}\}$, where $d(x, y)$ denotes the length of the shortest path from $x$ to $y$ in $G$. (The graph may be assumed to be weighted/unweighted depending on your preference.) Show that the ASYMMETRIC k-CENTER problem is NP-hard to solve by posing an appropriate decision problem, and showing this decision problem to be NP-complete. (Hint: Reduce from Vertex Cover.)

(b) Let $\text{Obj}(S)$ denote the quantity $\max_{x \in V} \{\min_{y \in S} \{d(x, y)\}\}$. An $\alpha$-approximation algorithm for ASYMMETRIC k-CENTER is a polynomial time algorithm that, given $(G, k)$, outputs a set $S$ with $|S| = k$ such that for every $S' \subseteq V$ with $|S'| = k$, it is the case that $\text{Obj}(S) \leq \alpha \text{Obj}(S')$. Show that there exists some $\alpha > 1$ for which an $\alpha$-approximation algorithm for the ASYMMETRIC k-CENTER problem would imply $\text{NP} = \text{P}$. (The larger the $\alpha$ the better.)

(c) A promise problem is a class of “Boolean” computational problems given by a pair of disjoint sets of instances $\Pi = (\Pi_{\text{YES}}, \Pi_{\text{NO}})$. (A standard decision problem is simply the special case where $\Pi_{\text{NO}} = \Pi_{\text{YES}}$). An algorithm $A$ decides a promise problem $\Pi$ if for every $x \in \Pi_{\text{YES}}$, $A(x) = 1$, and for every $x \in \Pi_{\text{NO}}$, $A(x) = 0$. An algorithm $R$ reduces a promise problem $\Pi$ to a promise problem $\Gamma$ if $x \in \Pi_{\text{YES}}$ implies $R(x) \in \Gamma_{\text{YES}}$ and $x \in \Pi_{\text{NO}}$ implies $R(x) \in \Gamma_{\text{NO}}$. Given an integer $c$, describe a promise problem $\Pi$ related to the ASYMMETRIC k-CENTER problem such that the existence of a polynomial time reduction from Vertex Cover to your problem $\Pi$ would rule out the existence of a $c$-approximation algorithm for the ASYMMETRIC k-CENTER problem unless $\text{NP} = \text{P}$. (So you have to describe $\Pi_{\text{YES}}$ and $\Pi_{\text{NO}}$. You dont have to reduce Vertex Cover to this promise problem. What you have to show is that if you assume such a reduction and also a $c$-approximation algorithm for the ASYMMETRIC k-CENTER problem, you get $\text{NP} = \text{P}$.)