This is a 50 minute, closed book examination—though you may bring a single 8.5x11 inch (or A4) sheet of paper with whatever you want written on it. There are 100 possible points, eight questions, and five pages (be sure you have them all).

Show your work. Partial credit will be given for incorrect answers, provided that the logic behind the answer is clear and shows an understanding of the subject matter. *More complete answers (e.g. proofs) will earn more points.*

This test uses the following abbreviations for special sets:

- SAT = \{< F > : F is a satisfiable boolean formula\}
- circuit-SAT = \{< C > : C is a boolean circuit which outputs a one for some inputs\}
- CVP = \{< C, \overline{t} > : C is a boolean circuit which outputs a one on input \overline{t}\}
- QBF = \{< F > : F is a true boolean formula with quantifiers (\exists \text{ and } \forall)\}
- dGAP = \{< G, s, t > : G is a directed graph with a path from node s to node t\}

Also, we denote relativization with superscripts. For example, \(P^A\) is \(P\) relativized using oracle set \(A\).

**Question 1 (10 points)** Show that if \(f(n) \leq g(n)\) for all \(n\), then \(\text{Dtime}(f) \subseteq \text{Dtime}(g)\).

**Answer** Let \(A = L(M)\) be an arbitrary set in \(\text{Dtime}(f)\). Then \(\text{time}_M(n) \in O(f)\) by definition of \(\text{Dtime}(f)\). So, \(\text{time}_M(n) \in O(g)\), since \(\forall n f(n) \leq g(n)\) implies that \(f \in O(g)\). So, again by definition of \(\text{Dtime}(g)\), \(A \in \text{Dtime}(g)\). Since \(A\) was arbitrary, it follows that \(\text{Dtime}(f) \subseteq \text{Dtime}(g)\), as was to be shown.
Question 2 (12 points) For which complexity classes are the following problems (described on the first page of this examination) complete using $\leq_{\text{log}}$?

- dGAP
  Answer NLOG
- circuit-SAT
  Answer NP
- QBF
  Answer PSPACE
- CVP
  Answer P

Question 3 (15 points) Is it possible that a PSPACE complete problem could be in the class P? What would this imply about P, NP, and PSPACE, and why?

Answer Yes. It would imply that $P = \text{PSPACE}$ and therefore that $P = \text{NP}$ since we know that $P \subseteq \text{NP} \subseteq \text{PSPACE}$.

Question 4 (10 points) What is an interactive proof system? How much workspace is required for a deterministic algorithm to accept a language which has interactive proofs?

Answer An IPS is a pair of algorithms $(P, V)$ such that $V$ is probabilistic and is restricted to run in polynomial time, where $P$ and $V$ share message tapes so they can communicate with each other. One can view an IPS as a pair of interacting algorithms which recognize language membership with high probability.

$\text{IP} = \text{PSPACE}$, so polynomial space is sufficient for a deterministic algorithm which recognizes the same language as an interactive proof system.
Question 5 (10 points) Prove that $\text{PSPACE} = \text{NPSPACE}$ (note: Savitch’s theorem does not prove this directly, but it makes a proof possible).

Answer PSPACE $\subseteq$ NPSPACE is trivial, so we show that NPSPACE $\subseteq$ PSPACE.

Let $A = L(M)$ be an arbitrary set in NPSPACE. Then by definition of NPSPACE there is a constant $k$ such that $\text{space}_M(n) \in O(n^k)$. By Savitch’s theorem, there is some deterministic Turing machine $M'$ such that $L(M') = A$ and $\text{space}_{M'}(n) \in O((\text{time}_M(n))^2)$. But $(\text{time}_M(n))^2 \in O((n^k)^2) = O(n^{2k})$, which is a polynomial. So, by definition of PSPACE, $L(M') = A \in \text{PSPACE}$. Since $A$ was arbitrary, this shows that NPSPACE $\subseteq$ PSPACE as was to be shown.

Question 6 (10 points) Show that $P \subseteq \text{PSPACE}$.

Answer Let $A = L(M)$ be an arbitrary set in $P$. Then by definition there is a constant $k$ such that $\text{time}_M(n) \in O(n^k)$. This implies that $M$ can only perform at most $O(n^k)$ changes to the worktape on inputs of size $n$ (it doesn’t have time to do more). So, $\text{space}_M(n) \in O(n^k)$, too (note, $M$ might use considerably less workspace than this, but this is a valid upper bound). So, by definition, $A \in \text{Pspace}$. Since $A$ was arbitrary, this shows that $P \subseteq \text{PSPACE}$.

In less formal terms: an algorithm only has time to do a polynomial number of writes if it’s entire runtime is bounded by a polynomial.

Question 7 (15 points) Suppose that you come up with a proof that $P^A \neq \text{NP}^A$ for some $A \in P$. What would this tell you about $P$ and NP, and why?

Answer This would imply that $P \neq \text{NP}$. This is because $P^A = P$ and $\text{NP}^A = \text{NP}$ when $A \in P$.

First, note that $P^A = P$ for $A \in P$. Let $B = L(M^A)$ be an arbitrary language in $P^A$ ($M$ is an oracle machine, $A$ is the oracle). By definition, $\text{time}_M(n) \in O(n^k)$ for some constant $k$. Since $A \in P$, there is some machine $M'$ such that $A = L(M')$ (without any oracle) and $\text{time}_{M'}(n) \in O(n^l)$ for some constant $l$.

Now, let $M''$ be the same as $M$ except that all queries are replaced by simulations of $M'$. That is, “simulate $M'$ on input $x$” appears in $M''$ whenever “Query $x$” appears in $M$.

Clearly, $L(M'') = L(M) = B$, since $M''$ does the same processing as $M$, even taking the same actions that $M$ would take after a query. But $M''$ makes no
queries, so it is a deterministic machine. Also, $\text{time}_{M^u}(n) \in O(n^{k+1})$ since it replaces at most $O(n^k)$ constant time queries ($M$ cannot make more that this in the allowed time) with subroutine calls each of which take $O(n^k)$ time. But this is a polynomial, so by definition $L(M^u) = L(M) = B \in P$. Since $B$ was arbitrary, this shows that $P^A = P$ for any $A$ such that $A \in P$, as was to be shown.

The argument that $\text{NP}^A = \text{NP}$ when $A \in P$ is very similar. For any arbitrary $B \in \text{NP}^A$, after replacing queries to $A$ with subroutine calls to a polynomial time algorithm to recognize $A$, one has a nondeterministic polynomial time algorithm which recognizes $B$.

**Question 8 (18 points)** What are conflicting relativizations? Give an example, and discuss what the implications are for a question if it has conflicting relativizations. (Hint: Baker-Gill-Solovay.)

**Answer** For a given question, such as “is $P$ equal to $\text{NP}$”, conflicting relativizations are oracles $A$ and $B$ such that the answer to the question is “yes” relative to $A$ and “no” relative to $B$. For example, Baker, Gill, and Solovay shows that there are oracles $A$ and $B$ such that $P^A = \text{NP}^A$ and $P^B \neq \text{NP}^B$, so the $P$ versus $\text{NP}$ question has conflicting relativizations (as does almost every open question in complexity theory).

If a question has conflicting relativizations, then proof techniques which relativize (remain true in the presence of an oracle) cannot possibly answer the question. Suppose you have a question with conflicting relativizations $A$ and $B$. If you prove the answer to the question is “yes”, and your proof relativizes, then your proof will still show that the answer is “yes” relative to $B$—even though the correct answer is “no”. That is, your “proof” will be wrong.

Unfortunately, our standard technique for separating two classes (diagonalization) and our standard technique for bounding the complexity of a set (simulation) both relativize.