Open book, open notes, closed neighbors, 170 minutes (at pace for 130 minutes), plus time for uploading. The exam has six problems and totals 180 pts., subdivided as shown. Problem (6) involves a choice: (6a) XOR (6b). Show your work—this may help for partial credit.

Notation: As always $E = \text{DTIME}[2^{O(n)}]$, while $\text{EXP} = \text{DTIME}[2^{n^{O(1)}}]$ and $\text{NEXP} = \text{NTIME}[2^{n^{O(1)}}]$, and the names of other complexity classes are either completely standard or explained in the questions themselves. The alphabet $\Sigma$ over which languages are encoded is immaterial; you are always welcome to consider $\Sigma = \{0, 1\}$ or $\Sigma = \{0, 1, \#\}$. The alphabet $\Gamma$ used as the worktape alphabet of Turing machines may, however, be much larger. The tupling notation $\langle x, y \rangle$ or $\langle x, y, z \rangle$ may be considered either as giving the strings $x\#y$ and $x\#y\#z$ or as the application of pairing functions. The choice of tupling scheme is not intended to matter, and any language using $\langle \cdot, \cdot \rangle$ notation may be assumed nonregular unless it is $\emptyset$ or $\Sigma^*$.

You may cite any major relevant theorem or fact or definition covered in the course without needing to give a justification (unless one is specifically asked for). The following diagram conveys “current knowledge” in complexity theory and is referenced by problems (1)–(3):
(1) (44 pts.)

Classify each of the following languages $L_1, \ldots, L_{11}$ according to whether it is

(a) regular;
(b) in deterministic logspace but not regular;
(c) in $P$ and not known to be—or believed not to be—in deterministic logspace;
(d) in $NP$ and strongly believed not to belong to $co-NP$;
(e) in $co-NP$ and strongly believed not to belong to $NP$;
(f) in $PSPACE$ and not believed to be in $NP$ or in $co-NP$.

(g) decidable but known to be not in $PSPACE$;
(h) c.e. but not decidable;
(i) co-c.e. but not c.e.; or
(j) neither c.e. nor co-c.e.

There is a unique best answer for each language. Not all answers (a)–(j) need to be used. Note that $0^n$ is a way of presenting the number $n$ by a string of length $n$. The languages are:

1. $L_1 = \{ \langle M, 0^n \rangle : M$ accepts all strings $x$ of length $> n \}$.
2. $L_2 = \{ \langle M, 0^n \rangle : M$ fails to halt on all strings $x$ of length $> n \}$.
3. $L_3 = \{ \langle M, 0^n \rangle : M$ accepts all strings $x$ of length $= n \}$.
4. $L_4 = \{ \langle M, 0^n \rangle : M$ accepts all strings $x$ of length $= n$ within $n^2$ steps $\}$.
5. $L_5 = \{ \langle M, x, 0^n \rangle : n = |x|$ and $M$ accepts $x$ within $n^2$ steps $\}$.
6. $L_6 = \{ \langle M, x, 0^n \rangle : n = |x|$ and $M$ accepts $x$ within $n^2$ space $\}$.
7. $L_7 = \{ \langle M, 0^n \rangle : M$ is a DFA and $M$ accepts $0^n \}$.
8. $L_8 = \{ \langle N, 0^n \rangle : N$ is an NFA and $N$ accepts $0^n \}$.
9. $L_9 = \{ x : x$ does not match the regular expression $(0 + (101)^*)^* \}$
10. $L_{10} = \{ \langle G, 0^k \rangle :$ there are $k$ nodes in $G$ such that every two are connected by an edge $\}$.
11. $L_{11} = \{ \langle M \rangle : M$ runs in $|\langle M \rangle|^{O(1)}$ space and does not accept $\langle M \rangle \}$.

Please write your answers in this form: if $L_{12}$ were the language of the Halting Problem, you could write “12. h” or “12. (g),” but good to add, “c.e. but not co-c.e.” No justifications are needed, but may help for partial credit.
Below are four statements about languages and complexity classes that are commonly disbelieved but are currently unknown to be false. For each one, write down as many other currently-unknown relations among complexity classes and languages that would follow. The other classes you may need to reference include (but are not limited to) \( L \) (i.e., deterministic logspace, also called DLOG), \( \text{NL} \), \( \text{DSPACEx}[(\log n)^2] \), \( \text{P} \), \( \text{RP} \), \( \text{BPP} \), \( \text{BQP} \), co-\( \text{NP} \), DLBA (i.e., deterministic linear space \( \text{DSPACEx}[O(n)] \)), NLBA (i.e., \( \text{NSPACEx}[O(n)] \)), PSPACE, \( \text{E} \), \( \text{EXP} \), and \( \text{NEXP} \). The languages/problems to consider are SAT, TAUT, CVP, GAP, UGAP, TQBF, and FACTORING. The relations to consider are equality (=), inequality (\( \neq \)), containments (\( \subseteq \), \( \supseteq \)), proper containments (\( \subset \), \( \supset \)), and membership or non-membership of particular languages in a class.

(a) GAP \( \in \) L.
(b) TAUT \( \in \) P.
(c) NLBA \( \subseteq \) NP.
(d) BQP = NL.

Eight correct and not-obviously-redundant relations spread among (a)–(d) suffice for full credit, but you are welcome to suggest more. They need not be two apiece.

Define two edges \((u_1, v_1)\) and \((u_2, v_2)\) in an undirected graph \(G = (V, E)\) to be “socially distanced” if neither of \(u_1, v_1\) is connected by an edge to either of \(u_2, v_2\). For example, in the “bowtie graph” shown on the first page, the two edges highlighted at left and right are the only two that are socially distanced. Now consider the following decision problem:

**Instance:** An undirected graph \(G = (V, E)\), an integer \(k \geq 1\).

**Question:** Are there at least \(k\) socially distanced edges in \(E\)?

(a) Write a formal mathematical definition of the language of this decision problem, using standard set-theoretic notation for graphs—without using the words “socially distanced.”

(b) Show that the problem is \(\text{NP}\)-complete. For hardness, you must use a mapping reduction from EXACTLY ONE 3SAT.

(c) For optional offsetting credit (not extra credit), explain why your reduction function \(f\) in (b) fails to be a correct reduction from the “vanilla” 3SAT problem to the above problem.
(4) (24 pts.) True-False with reasons.

For each statement (a)–(d), write out true or false (3 pts.), and then write a brief justification (3 pts.).

(a) It is known that the problem of factoring integers belongs to P if and only if NP = P.

(b) NP has complete languages under polynomial-time many-one reductions (≤p_m), but does not have any complete languages under log-space many-one reductions (≤log_m).

(c) The intersection of two languages in NP always belongs to NP.

(d) If a language A is not in P, then for every Turing machine M such that L(M) = A, and every x ∈ Σ*, the computation M(x) takes an exponential number of steps on the instance x.

(5) (24 pts. total)

Recall the two-qubit quantum circuit used to illustrate entanglement in lecture: Hadamard gate on line 1 followed by CNOT with control on line 1 and target on line 2, with x = |00⟩ as basis-state input. Now let us vary it to be the following:

That is, we added a NOT gate to line 2 and made the CNOT gate have control on line 2 and target on line 1.

(a) Write out the 4 × 4 matrix of H ⊗ X and the matrix of the “upside-down” CNOT gate.

(b) Calculate the quantum state resulting from the input x = |00⟩. Is it entangled?

(6) (24 pts.) Do EXACTLY ONE of the following two problems.

(6a) Calculate a DFA M such that L(M) = (01)* (010)*. First design an NFA, then convert it to a DFA.

(6b) Anti-virus utilities for personal computers work by maintaining a database of “tell-tale substrings” of known viruses, but are often ineffective against new, unknown viruses. A better, “perfect” kind of anti-virus utility would be able to analyze any downloaded software program P and determine whether running P could unleash a virus or not. Using a reduction, explain why no such “perfect” anti-virus utility can ever exist. (You do not need a technical definition of a computer virus to answer this question.)