Final Exam

This test is three hours long. You may use the assigned text (Sipser's "Introduction to the Theory of Computation") and your notes (including class handouts, problem solutions, etc.). You may not use other materials.

!!! Be sure to write your full name **here and** on each page !!!

Name: ______________________________

Problem 1: **Short-Answer Questions: (3 points each)**

1. Suppose L is a language and M is a probabilistic polynomial-time decider such that
   - $Pr\left[M(x) \text{ accepts } | x \in L\right] \leq \frac{1}{10}$, and
   - $Pr\left[M(x) \text{ accepts } | x \notin L\right] \geq \frac{9}{10}$

   Is $L \in \mathcal{BPP}$? (Explain briefly.)

2. Suppose the language L is the union of the sublanguages $L_1, L_2, \ldots$, where the number of such sublanguages may be finite or infinite, and where each language L_i is known to be regular. How hard can it be to recognize L? (What is the worst it could be, given these constraints?)

3. Due to manufacturing limitations, the Turing machines made by ACME Computer are limited to having a tape with at most 2^n tape squares when the input has size n. Is the Halting Problem for an ACME TM decidable? (Explain briefly.)

4. It is true that if $P = NP$, then $PRIMES \in NP$. Can you explain (briefly) why?

5. True or false: it is impossible for a deterministic encryption scheme to be semantically secure. (Explain briefly.)

Name:

Problem 2: **(15 points)** The operation of *shuffle* is important in the study of concurrent systems. If x, $y \in \Sigma^*$, then $x \,||\, y$ is the set of all strings that can be made by shuffling x and y together like a deck of cards. For example,

$$ab \,||\, cd = \{abcd, acbd, acdb, cabd, cadb, cdab\}$$

The shuffle of two languages A and B, denoted $A \,||\, B$, is the set of strings obtained by shuffling a string from A with a string from B:

$$A \,||\, B = \bigcup_{\substack{x \in A \\ y \in B}} x \,||\, y$$

Prove that if A and B are regular, then $A \,||\, B$ is regular also. (Hint: can you design an FA that recognizes $A \,||\, B$?)

Name: ________________________________

Problem 3: **(15 points)** One of the following two sets is Turing-recognizable, and the other is not. Which one is which? Prove your answer to be correct. (Hint: can you prove that they can't both be recognizable?)

- $A = \{\langle M \rangle \ : \ M$ accepts at most 481 distinct inputs$\}$
- $B = \{\langle M \rangle \ : \ M$ accepts more than 481 distinct inputs$\}$

Name:

Problem 4: **(15 points)** Consider the *FEEDBACK VERTEX SET* problem:

> Given a *directed* graph $G = (V, E)$ and a number $K \leq |V|$, is there a subset $F \subseteq V$ such that $|F| \leq K$ and F contains a vertex from every directed cycle in G?

Show that the *FEEDBACK VERTEX SET* problem is $\mathcal{NP}$-complete. (Hint: Consider *VC*.)

Name: ____

Problem 5: **(15 points)** Show that if $\mathcal{P} = \mathcal{NP}$, then there exists a polynomial-time algorithm that takes in an instance $\langle G, K\rangle$ of the *FEEDBACK VERTEX SET* problem (as defined in the previous problem of this final) and returns either

- A feedback vertex set of size at most K, if one exists, or
- A special character $\perp$ otherwise.

Name:

Problem 6: **(15 points)** Suppose $L \in \mathcal{BPP}$ and M is a probabilistic polynomial-time decider so that

- $Pr\left[M(x) \text{ accepts } | x \in L\right] \geq \frac{9}{10}$, and
- $Pr\left[M(x) \text{ accepts } | x \notin L\right] \leq \frac{1}{10}$

(Here the probabilities, as usual, depend only on the internal coin flips of the machine M.)

Suppose that on any input x, the machine M never flips more than $\log(1 + |x|)$ coins (i.e., never uses more than $\log(1 + |x|)$ bits of randomness). Show that $L \in \mathcal{P}$.

Name:

Problem 7: **(15 points)** Suppose we have a public-key algorithm, a triple of polynomial-time algorithms (G, E, D), such that

- G is a randomized key-generation algorithm that produces key pairs (PK, SK) when given as input a "security parameter" k (in unary, as 1^k, as is usual),
- E is a randomized encryption algorithm that takes in a public key PK and a message $M \in \{0,1\}^*$, and produces a ciphertext $C = E(PK, M)$ in $\{0,1\}^*$.
- D is a decryption algorithm that takes in an alleged ciphertext C in $\{0,1\}^*$ and a secret key SK and produces either
  - M, if C was produced by $E(PK, M)$ (for the PK associated with SK), or
  - A special output $\perp$ otherwise.

We define a public-key algorithm (G, E, D) to be "simply secure" (a simpler notion than "semantically secure") if for all messages M_0, M_1 in $\{0,1\}^*$, for all probabilistic polynomial-time adversaries A, for all sufficiently large security paramters k,

$$\Pr\left[(PK, SK) \leftarrow G(1^k); b \leftarrow \{0,1\}; C \leftarrow E(PK, M_b); d \leftarrow A(PK, C) : d = b\right] < \frac{1}{2} + \frac{1}{k}.$$

(Here $b \leftarrow \{0,1\}$ means that b is chosen uniformly at random from the set $\{0,1\}$.)

Suppose further that there was an adversary that was able at least two-thirds of the time to decrypt a doubly-encrypted message bit (0 or 1). That is, suppose there exists a probabilistic polynomial-time adversary A' such that for all k,

$$\Pr\left[(PK, SK) \leftarrow G(1^k); b \leftarrow \{0,1\}; C_1 \leftarrow E(PK, b); C_2 \leftarrow E(PK, C_1); d \leftarrow A'(PK, C_2) : d = b\right] \geq \frac{2}{3}$$

Show that the encryption scheme (G, E, D) is not simply secure. (Hint: if you had A', how would you distinguish $E(PK, 0)$ from $E(PK, 1)$?)