NAME: _____

CSE 431 Sample Final
Introduction to the Theory of Computation

1. Circle True or False below. (All languages below are over some fixed alphabet Σ_0. You do not need to justify your answers.)

   (a) Every Turing decidable language is in $\mathcal{NP}$. ..... T F

   (b) If K is undecidable and $K \leq L$ then L is Turing acceptable. ..... T F

   (c) The complement of every Turing decidable language is Turing decidable. ..... T F

   (d) The intersection of a Turing decidable language and a Turing acceptable language is always Turing decidable. ..... T F

   (e) If A and B are both in $\mathcal{NP}$ then $A \cup B \in \mathcal{NP}$. ..... T F

   (f) If A is $\mathcal{NP}$-hard then any deterministic polynomial time algorithm that solves A would show that $\mathcal{P} = \mathcal{NP}$. ..... T F

   (g) If L is in $\mathcal{NP}$ then L is Turing acceptable but it is not Turing decidable. ..... T F

2. Consider the following list of properties that might apply to the stated language.

   T.r.: The language is Turing recognizable.

   Dec: The language is decidable.

   $\mathcal{NP}$: The language is in $\mathcal{NP}$.

   $\mathcal{P}$: The language is in $\mathcal{P}$.

   Circle all the properties that you are certain are true.
   $\times$ out all the properties that you are certain are false.
   NOTE: You may not be able to do either for some properties.

   (a) $L = \{< M, w > \mid$ Turing machine M runs forever on input $w\ \}$ ..... T.r. Dec $\mathcal{NP}$ $\mathcal{P}$

   (b) $L = \{$Undirected graphs G that are 3-colorable$\}$ ..... T.r. Dec $\mathcal{NP}$ $\mathcal{P}$

   (c) $L = \{$Undirected graphs G that are **not** 3-colorable$\}$ ..... T.r. Dec $\mathcal{NP}$ $\mathcal{P}$

   (d) $L = \{w \mid w$ is the encoding of some Turing machine$\}$ ..... T.r. Dec $\mathcal{NP}$ $\mathcal{P}$

   (e) $L = \{P | P$ is a C program that attempts to execute a divide-by-zero on some input$\}$ ..... T.r. Dec $\mathcal{NP}$ $\mathcal{P}$

   (f) $L = \{(A, b) | A$ is an integer matrix and b is an integer vector such that $Ax = b$ for some vector x of 0's and 1's$\}$ ..... T.r. Dec $\mathcal{NP}$ $\mathcal{P}$

   (g) $L = \{< M > \mid$ Turing machine M accepts regular language$\}$ ..... T.r. Dec $\mathcal{NP}$ $\mathcal{P}$

   (h) $L = \{< M > |$there is some input w on which Turing machine M runs for at most $2|w| + 4$ steps$\}$ ..... T.r. Dec $\mathcal{NP}$ $\mathcal{P}$

3. Suppose that A and B are languages contained in $\{0, 1\}^*$.

   (a) Define what it means for $A \leq_p B$.

   (b) Prove that if $B \in \mathcal{P}$ and $A \leq_p B$ then $A \in \mathcal{P}$.

4. Prove that the following problem is undecidable: Given Turing machines M_1 and M_2, does M_1 accept the complement of the language accepted by M_2?

5. For finite automata, there is an algorithm that will take any DFA M as input and produce an equivalent DFA M' as output that is *the best* equivalent DFA. In this question you will prove that such an algorithm cannot exist for Turing machines:

   We will say that two Turing machines are *equivalent* if and only if they accept the same language. We will say that Turing machine M_1 *is a smaller Turing machine than* M_2 if $< M_1 >$ comes before $< M_2 >$ in the lexicographic ordering on strings over the coding alphabet for machines:

   Suppose that there is some Turing computable function f that on input $< M >$ will output $< M' >$ such that M' is *the smallest* Turing machine equivalent to M.

   (a) Use f to construct an algorithm that will determine whether or not two Turing machines are equivalent.

   (b) Use part (a) to conclude that f cannot be Turing computable.

6. Define the INDEPENDENT-SCHEDULING problem as follows: Its input consists of an undirected graph G and an integer k. The vertices of the graph correspond to tasks that take one time unit each. An edge in the graph indicates that the two tasks it joins cannot be worked on at the same time. A *k-schedule for G* consists of an assignment of each task to a 'time slot' which is one of $1, ..., k$ so that no two tasks joined by an edge in the graph are assigned to the same time slot. An input (G, k) is in INDEPENDENT-SCHEDULING if and only if there is a k-schedule for the input graph G.

   - Prove that INDEPENDENT-SCHEDULING is $\mathcal{NP}$-complete.
     (Hint 1: Use the fact that 3-COLORABILITY is $\mathcal{NP}$-complete.)
     (Hint 2: Notice how similar the notion of 'time slot' is to 'color'.)
   - Even if you don't know how to do all the steps, write down what you would need to show.

7. The *space* used by a k-tape Turing machine M is the maximum number of tape squares it uses on any tape during its computation. This question will relate the time used by a nondeterministic Turing machine and the space that it uses.

   (a) Give a brief outline of the proof that any language $A \in \mathcal{NP}$ may be decided by a deterministic Turing machine that runs in time $T(n) \leq r^{n^d}$ for some constants r and d.

   (b) What (roughly) is the space used by the algorithm that you described in part (a)?

   (c) Use this to conclude that any language in $\mathcal{NP}$ may be decided using only polynomial space.