Theory of Computations,
Final Exam for the course
CS 6315/CS 5315, Spring 2025

Name: _____

You can use up to 10 handwritten pages of notes.

1. *Primitive recursive and mu-recursive functions:*

1a. Why do we need to study primitive-recursive and mu-recursive functions in the first place? What programming concepts do they correspond to?

1b. Translate, step-by-step, the following double loop into a mu-recursive expression:

```
int s = a;
while(s <= a)
   {for(int i = 1; i <= a; i++){
       {s = s * b;}
   }
```

You can use the multiplication function is this description. What is the result of this program when a = 2 and b = 4?

1c. Prove, from scratch, that the function (a % b) + a is primitive recursive.

1d. Is Kolmorogov complexity primitive recursive? mu-recursive? Explain your answers.

2. *Computable sets, computable functions, and Turing machines:*

2a. Why do we need to study decidable and recursively enumerable (r.e.) sets?

2b. Is the union of two r.e. sets always r.e.? Is the complement to a r.e. sets always r.e.? In both cases, if yes, prove it, if no, provide a counterexample.

2c. Prove that it is not possible, given a program that always halts, to check whether this program always computes n^3.

2d. Julius Cesar encoded his message by replacing each letter with the next one. Design a Turing machine that would help decode his messages: given a word in an alphabet {a,b,c}, this machine should change b's to a's, c's to b's, and a's to c's. For example, for s = abba, it should return caac. Explain why in a Turing machine (and in most computers) binary numbers are represented starting with the least significant digit.

3. *NP-hardness:*

3a-d. Reduce the satisfiability problem for the formula (~a ∨ c) & (b ∨ c) & (a ∨ b ∨ ~c) to:

a) 3-coloring,
b) clique,
c) subset sum problem, and
d) interval computations.

3e. Reduce the satisfiability problem for the formula (a ∨ ~b ∨ ~c ∨ ~d) & (a ∨ ~b ∨ ~c) to 3-SAT.

3f. In proofs of what results are the reductions from 3a-e used? What do we gain by proving that a problem is NP-complete?

3g. Use the definitions of the classes P, NP, NP-hard, and NP-complete to describe, for each of these four classes, whether this class contains addition, exact change problem, 2-SAT, and interval computations. Explain your answers.

3h. Give two examples of why the current definition of a feasible algorithm is not perfect:

- an example when an algorithm is feasible according to this definition but not practically feasible, and
- an example when an algorithm is not feasible according to this definition but is practically feasible.

These examples should be different what what we had so far.

3i. Use the general algorithm to find a satisfying vector for the following 2-SAT formula

$$(a \vee b) \,\&\, (b \vee \neg c) \,\&\, (\neg c \vee \neg a) \,\&\, (\neg a \vee \neg b) \,\&\, (\neg b \vee c) \,\&\, (a \vee \neg c) \,\&\, (\neg c \vee a).$$

4. *Parallelization:*

4a. Does the matrix addition problem belong to the class NC? Explain your answer.

4b. If we parallelize and take into account communication time, what is the fastest that we can compute the sum of the two matrices? Explain your answer.

4c. Where are similar arguments used in the proof that satisfiability is NP-hard?

4d. What are the physical assumptions behind these arguments? Give two examples of how non-standard physics -- in which these assumptions are not satisfied -- can be used to solve NP-hard problems in polynomial time.

5. *How to solve NP-hard problems?*

5a. Suppose that a probabilistic algorithm for checking the program correctness misses the program's mistake half of the time. How many times do we need to repeat this algorithm to achieve reliability of 99%? And why do we need probabilistic algorithms in the first place?

5b. Apply both greedy algorithms to solve the following particular case of the knapsack problem: we have 4 objects with weights 200, 300, 400, and 500, values 10, 12, 12, and 15, and the overall weight of 600. Why do we need to use these algorithms, if they do not produce optimal results?

5c. Give an example of how quantum computing can speed up computations.

6. *Beyond NP: polynomial hierarchy:*

6a. Which class of the polynomial hierarchy contains optimization problems? The problem of winning in 2 moves? Explain your answers.

6b. Which class of polynomial hierarchy contains $\Pi_3^{\Sigma_3}$?

6c. (For extra credit) Give an example of an oracle A for which $P^A = NP^A$. Explain your answer.

7. *Projects:*

7a. Briefly describe your project for this class.

7b. Briefly describe someone else's project for this class.