PRINT FAMILY NAME:________________________

Examination 2
COT 6315 - Formal Languages and Computation Theory
17 June 1999

Instructions:

- **Read all these instructions; failure to follow instructions will result in loss of points.**
- Do not start the examination until instructed to do so.
- This test is closed book, closed notes, but you are permitted one 8.5" by 11" crib sheet (two sides).
- **Answer all questions from Part A (first 4 questions) and then any two questions from Part B (last 3 questions) below.**
- Show your work.
- **Start the answer to each question on a new page** (i.e., do NOT put the answer to more than one question on the same page).
- **Assemble your answers in numerical order of the questions when you submit them.**
- **Leave a 1" square in the upper left corner for a staple.**
- Be sure to include your name on your answer sheets.
- You have **120 minutes** to complete this examination.
- **Read and sign the following statement.** You may write this on your exam and sign it there if you wish to take the exam questions home with you today. Do not discuss this exam with anyone in this course who has not yet taken this exam.

On my honor, I have neither given nor received unauthorized aid on this examination, and I will not discuss the contents of this examination with any student who has not yet taken this examination.

Signed:

Part I

Answer **all four (4)** questions.

1. (10) Show that PCP is decidable over a unary alphabet, e.g., $\Sigma = \{0\}$.

2. (15) Consider a pair of Turing machines F and M, where F takes input from M, such that

   (a) F produces a copy of F based on the output of M when M has the string X as its input;

   (b) F produces a copy of M based on the output of M when M has the string Y for its input;

   (c) M produces nothing if neither X nor Y is input;

   (d) F produces nothing if nothing is input.

   Produce two machines satisfying the requirements for F and M.

3. (15) Prove that Multiprocessor Scheduling (MPS) is NP-Complete. An instance of MPS is a finite set T of tasks, a length $l : T \rightarrow \mathcal{N}$, a number $m \in \mathcal{N}$ of processors, and a deadline $D \in \mathcal{N}$. An instance is in the language iff there exists a collection of m subsets of T, $T_1, T_2, \ldots, T_m$ such that

   (a) $T = \bigcup_{i=1}^{m} T_i$

   (b) $\forall\ i \neq j,\ i, j \in [1..m], T_i \cap T_j = \emptyset$

   (c) $\max\{\Sigma_{t \in T_i} l(t) \mid 1 \leq i \leq m\} \leq D$.

   In other words, each T_i is the set of tasks assigned to processor i, where each task is assigned to exactly one processor, and every processor meets the deadline on its task set.

4. (10) An Euler circuit for graph $G = \ <V, E>$ is a circuit such that every edge $e \in E$ is visited exactly once. Classify $EULER-CKT = \{<G> \mid G$ has an Euler circuit$\}$ as a language in P, NP, $NP-C$, or $NP-hard$, and prove your claim(s).

Part II

Answer **any two (2)** questions.

5. (25) Show that a k-headed Turing machine with jumps (a k-JTM) is equivalent in power to a standard TM. A k-JTM has special moves in which it may move any head to the current location of any other head in a single move. Instead of $\{L, R\}$ being the only possible moves for each head, $\{L, R, 1, 2, ..., k\}$ are all possible moves for each head, where a move of j for head i takes head i to the same location head j was in prior to the move. Note that heads i and j may swap positions by moving head i to j and head j to i in the same move, and that moving head i to i is the same as a stationary move (leaving head i in the same location as it was at the start of the move). Begin by formalizing the definition, configurations, and computations of the k-JTM.

6. (25) Are Recursive (Turing Decidable) sets closed under the following operations? If so, prove it (a construction sketch will do with explanation and argument that the constructed machine recognizes exactly the desired language), if not, give a counterexample.

   (a) union

   (b) intersection

   (c) concatenation

   (d) Kleene Closure (*)

   (e) countable union (i.e., is union of a countable set of Recursive sets a Recursive set?)

7. (25) Consider the second most popular data structure next to the stack: the FIFO queue.

   (a) Give a formal definition of a (generic) FIFO-queue automaton (FQA), similar to that of the PDA, including a definition of its configurations, computations, and the language it accepts.

   (b) What is the power of FQA's, i.e., what languages can they recognize? Do these include CFL's or not? Prove your answers.

   (c) Non-determinism of PDAs makes a difference in the languages PDAs can accept. Does it make a similar difference in the languages acceptable by FQAs? If so, how, if not, why not?