The exam will cover everything we have discussed from Chapters 9-10 in the text and all presented materials, but emphasizing the material covered in class. The style of the exam will be very flexible, possibly consisting of fill in the blank, true or false (with justification), multiple choice, matching, short answer (e.g., definitions or listing), and discussion questions. Problems such as identifying a language as recursive or not, or in P or NP, are likely. In general, you should know how to solve the problems given in assignments even though you may not have to actually solve such problems on the exam.

A brief outline of the topics we have covered is described below. (This list is intended to be as complete as possible but may not be all inclusive.)

- **Undecidability.** We introduced the diagonalization language as a non-recursively enumerable language and used this to show the universal language was recursively enumerable but not recursive. A problem is said to be undecidable if it is not recursive. The primary means for proving undecidability of a language L is the reduction of a known undecidable language L’ to L such that the existence of a Turing machine to decide L implies the existence of one for L’. Rice’s Theorem describes a general method of applying reductions. Post’s Correspondence Problem was shown to be undecidable and a good language to reduce to problems involving context-free languages (e.g., ambiguity).

- **Intractable Problems.** The complexity classes P and NP were introduced for deterministic and non-deterministic polynomial time, respectively. Satisfiability of a Boolean formula (SAT) was shown to be NP-complete by Cook’s Theorem, which shows that all problems in NP may be polynomial-time reduced to SAT. SAT was then used to show other problems NP-complete, including Conjunctive Normal Form Satisfiability (CSAT), CSAT with 3 literals per clause, and Node Cover. The significance of NP-completeness is that none of the NP-complete problems is believed to have a solution in P, and if any one of them did, then P would be equal to NP, whereas it is believed that P is a proper subset of NP.
1. A(n) __________________________ is a translation of an instance of one problem P into an instance of another problem P’ such that the solution of P’ implies a solution to P.

2. True or False. The class of languages accepted by nondeterministic Turing machines in polynomial time is the same as the class of languages accepted by deterministic Turing machines in polynomial time.

3. True or False. It is known that P ⊂ NP but it can never be formally shown that the containment is proper or if P = NP.

4. List two (2) properties that are undecidable for recursive languages.

5. For each language below, indicate whether the language is: 1) in P, 2) in NP but probably not in P, or 3) probably not in NP. Give a brief informal justification for your answer. Note that all languages are encoded over the alphabet \{0, 1\}.
   (a) L = \{w | w is a Boolean formula which is not satisfiable\}. For example, (x)(-x), (x+y)(-x+y)(x+-y), etc.
   (b) L = \{<x, y> | x is a list of integers and y is the same list of integers sorted in ascending sequence\}. For example, <(3, 1, 2), (1, 2, 3)>, <(7, 4, 2, 13), (2, 4, 7, 13)>, etc.
   (c) L = \{<P, n> | P is an instance of Post’s Correspondence Problem which has a solution of maximum size n\}. For example, <P_1, 0> is not in L because any solution to P_1 would be at least of size 1, and <P_2, 3> would not be in L if P_2 has a solution of size larger than n or has no solution at all, and would be in L otherwise.

6. For each language below, indicate whether the language is: 1) recursive, 2) recursively enumerable but not recursive, or 3) not recursively enumerable. Give a brief informal justification for your answer. Note that all languages are encoded over the alphabet \{0, 1\}.
   (a) L = \{x#y#z | x, y and z are binary numbers such that x + y = z\}. Examples of strings in L include 1001#011#1111 and 111#1000#1000.
   (b) L = \{<M> | M is a Turing machine which accepts at least 4115 words\}.
   (c) L = \{<M> | M is a Turing machine which accepts no string of length 3\}.