The exam will cover *everything* we have discussed on logic programming, axiomatic semantics, static semantics, continuation semantics, and domain-specific languages. The style of the exam will be very flexible, possibly consisting of fill in the blank, true or false (possibly with justification), multiple choice, matching, short answer (e.g., definitions or listing), and discussion questions. You will not be asked to work detailed problems such as writing any actual code (i.e., code in a specific language like Prolog).

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.)

- **Logic Programming**. You should know the general design philosophy and syntactic and semantic flavor of Prolog. You do NOT need to know specific commands. Examples of the most detailed information of a fundamental nature you should know would be how the rule matching works (e.g., execution of rules, instantiation of variables, backtracking, etc.) and the representation of lists.

- **Axiomatic Semantics**. You should know how axiomatic semantics may be used to prove the partial correctness of a program. You need not remember any specific axiomatic rules per se but should know enough about them to be able to explain or use them if they are given to you.

- **Static Semantics**. You should know how static semantics are defined using denotational semantics and how to distinguish static from dynamic semantics in a denotational semantics expression. Implementation techniques for denotational semantics were also discussed.

- **Continuation Semantics**. You should know what a continuation is and how it is used in defining semantics.
1. Fill in the Blank. A(n) ________________ is a condition which is true upon entry to a loop every time the loop is entered, and is still true at the termination of the loop.

2. True or False. In the semantics of an imperative language, the environment is typically part of the static semantics, while the memory store is typically part of the dynamic semantics.

3. Consider the Prolog program which solves the Towers of Hanoi problem. This problem is to move a stack of blocks from one location to another without ever changing the order in which blocks are placed on top of each other. A third stack may be used but only one block may be moved at a time.

   ```prolog
hanoi(N) :- move(N, left, center, right).
move(0, X, Y, Z).
move(N, X, Y, Z) :-
    N > 0, M is N - 1, move(M, X, Z, Y), display([X,Y]), move(M, Z, Y, X).
```

Trace the above program using the query `hanoi(3)`. Show all queries which are generated by the execution and all moves which are printed. You may wish to draw a tree to show this clearly.

4. Consider the axioms of assignment, composition, and loop below:

   **Assignment**
   \[ \{ P \ [E/V] \} \ V := E \ {P} \]

   **Composition**
   \[ \{P\}S_1\{Q\}, \{Q\}S_2\{R\} \quad \therefore \quad \{P\}S_1;S_2\{R\} \]

   **Loop**
   \[ \{P&B\}S\{P\} \quad \therefore \quad \{P\}\text{while } B \text{ loop } S \text{ end loop}\{P&\neg B\} \]

Prove the correctness of the following program segment.

\[ \{n = j \times (j + 1)/2 \land i \geq 0\} \]
while \ j != i loop \ j := j + 1; \ n := n + j; \ end \ loop
\{n = j \times (j + 1)/2 \land i \geq 0 \land \neg(j \neq i)\}

Show all steps used in the proof.