Computer Science Comprehensive Exam—Fall 1999
Programming Languages

Instructions: Please answer all the questions directly on the exam itself. Answer **all** the questions; you have 90 minutes. Explain answers as fully as possible, give examples if appropriate, define terms.

1. What are the major areas of application for logic programming languages?

2. Consider literals of the same name from different enumerations types. Is this a good idea for the programmer? Why, or why not? Give an example. What possible language design approaches can be used to distinguish the types of literals with the same name?

3. What is static type checking? What is dynamic scoping? Is it possible to do static type checking with dynamic scoping? Why, or why not? Give an example.

4. Explain how passing procedures as arguments is implemented in a block-structured language?

5. Concerning data types, what is representation independence and is it good or bad?

6. What is dynamic dispatch (called dynamic binding by Sebesta) in object-oriented languages? How is it implemented? In designing a program, when would you use dynamic dispatch?

7. What are the differences between the C++ **throw** specification and the Java **throws** specification?

8. What is the type of the ML function `f` below? Describe in a few words what the function does.

```
datatype T = nl | lf of int * T * T;
fun f nl = [] | f (lf (x,l,r)) =  (f l) @ (x :: (f r));
```

9. Formulate in PROLOG the classical syllogism:

   All students hate exams;
   This is an exam;
   Therefore I hate this exam.

10. What is a unifying substitution? Give the most general unifying substitution for each of the following pairs of terms (x, y, and z are variables):

$$\begin{array}{rl} g(x,a) & g(x,a) \\ g(x,y) & g(y,h(a,x)) \\ f(g(a,b),h(x,y)) & f(g(z,b),h(b,b)) \end{array}$$

11. Define "append" of two lists in PROLOG. Define the member function in PROLOG that tests if an element is a member of a list.