CSCE 330 Fall 2003
QUIZ 3
Assigned Tuesday, 03-09-09

1. Syntax diagrams are a notational variant of EBNF. True or false?

   Answer: True

2. What follows is a (slightly edited) long quote from Andrew Cumming's "Introduction to Functional Programming" (http://www.dcs.napier.ac.uk/course-notes/sml/introfp.htm). What is the word that I substituted with *mysterious*?

   > The *mysterious* language community is excessively dour. The *mysterious* ascetics forbid themselves facilities which less pious programmers regard as standard . When using *mysterious* languages we do away with notions such as variables and reassignments. This allows us to define programs which may be subjected to analysis much more easily. When a value is assigned it does not change during the execution of the program. This property is referential transparency. There is no state corresponding to the global variables of a traditional language or the instances of objects in an object oriented language. When a definition is made it sticks. Reassignment does not take place. Getting used to this and finding alternatives to the traditional structures such as loops which require reassignment is one of the hardest tasks for a programmer "converting" from a traditional language. The line
   >
   > `x := x+1;`
   >
   > may appear in ... [an imperative] language and is understood to indicate that 'box' or 'location' referred to as 'x' has its contents incremented at this stage. We do not admit such concepts. 'x' is 'x' and 'x+1' is one more than x; the one may not be changed into the other. A program without a state is a simpler thing—it is easier to write the code and easier to reason about the code once written. It is harder to write poor code.
   >
   > *Mysterious* languages are considered, by their devotees, to be higher level than [imperative] languages. *Mysterious* languages are regarded as declarative rather than imperative. Ordinary [imperative] languages such as Pascal, C (including flavours such as C++) and assembly instruct the computer on how to solve a problem. A declarative language is one in which the programmer declares what the problem is; the execution of the program is

a low level concern. This is an attitude shared with the logic language community (Prolog people).

Answer: functional