I have neither given nor received unauthorized aid on this examination, nor do I have reason to believe that anyone else has.

_____

Name (printed)

_____

Signature

C S 355
Principles of Programming Languages
Midterm Exam
Fall 2001

Instructions: This is an closed-book, closed-notes exam.

All work is to be done on these pages. If you need more room for a particular question than is provided, use the backs of these pages but be sure to label your answers clearly. There are a total of 75 points worth of questions on this test. The point values assigned to each question represent a rough indicator of the difficulty and/or level of detail expected. Use your time accordingly.

Your answers, especially where judgments or explanations are requested, should be precise and to the point. Completeness is important, but so is precision. Extraneous material will not be rewarded and may actually cost you points if it gives me the impression that you do not know what part of your own answer is actually important.

1. (5 pts) We have classified languages by paradigm. These paradigms are based on three properties. Name the three properties and give an example of a possible value for each.

2. (6 pts) Define each of the following and give an example:

   Token:

   Lexeme:

3. (6 pts) Many people have suggested that a language with union types cannot be strongly typed.

   (a) Explain why people draw this conclusion.

   (b) How can a language support unions without sacrificing strong typing?

4. (8 pts) An insurance company issues identification codes to all of its policy holders. This number consists of 3 parts, each separated by a colon (:) character:

   - The plan identifier consists of 1 alphabetic letter, followed by two alphanumeric chracters. Alphabetic characters are always upper-case, and an alphabetic character never follows a numeric one.
   - The personal identifier consists of 1 or more numeric digits.
   - The check digit is a single character, consisting of either a numeric digit or the character '@'.

   An example of such an identifer would be "AY2:123:0".

   (a) Give a regular expression describing the set of identification numbers. Use only the 3 fundamental regular expression operations (concatenation, alternation, and Kleene closure). To keep this answer reasonably short, however, you may employ "..." elision for obvious continuations of an established pattern.

   (b) Give a regular expression describing the same set of strings, using the common extensions to the regular expression notation to simplify your answer as much as possible.

5. (10 pts) Use BNF (no extensions) to give a grammar for regular expression notation that is written in terms of the 3 basic regular operations. Assume that Kleene closure has the highest precedence and alternation the lowest. Grouping of expressions inside parentheses should also be permitted.

6. (6 pts) A memory leak occurs when, over time, ever larger portions of the heap are devoted to objects that can no longer be reached from any pointer in the static areas or in an activation record.

   (a) What is the primary cause of memory leaks?

   (b) What can a programming language do to prevent memory leaks?

7. (6 pts) The following EBNF grammar is based on the syntax of statements in Modula 2.

$$
\begin{array}{lll}
\langle S \rangle & ::= & \\
 & | & \texttt{id} = \texttt{expr} \\
 & | & \texttt{if}\,\texttt{expr}\,\texttt{then}\ \langle SL \rangle\ \{\texttt{elsif}\,\texttt{expr}\,\texttt{then}\ \langle SL \rangle\ \}[\texttt{else}\ \langle SL \rangle\ ]\texttt{end} \\
 & | & \texttt{loop}\ \langle SL \rangle\ \texttt{end} \\
 & | & \texttt{while}\,\texttt{expr}\,\texttt{do}\ \langle SL \rangle\ \texttt{end} \\
\langle SL \rangle & ::= & \langle S \rangle\ \{; \langle S \rangle\ \}
\end{array}
$$

`id` and `expr` are tokens for identifiers and expressions, respectively. (For the sake of this problem, we simplify expressions to single tokens.)
Using this grammar, draw a parse tree for `while expr do if expr then S end ; S ; end`

8. (4 pts) For the same language, draw an abstract syntax tree for `if expr then while expr do S end ; S end`

9. (9 pts) Define each of the following:

   type system:

   strongly typed:

   reference semantics:

10. (5 pts) A language allows the use of `elsif` clauses inside an if statement:

```
if (C1) then
   Stmt1;
elsif (C2) then
   Stmt2;
elsif (C3) then
   Stmt3;
  .
  .
  .
elsif (CN) then
   StmtN;
else
   StmtE;
endif
```

(For the sake of this problem, assume that a final `else` clause is always present.
Show how this construct would be translated into assembler/machine code using jumps, conditional jumps, and labels.

11. (5 pts) Consider the following code:

```
x = y;
writeToDisk (x);
writeToDisk (y);
```

where `writeToDisk` writes each byte of its argument into a disk file. After executing this code, the programmer is surprised to discover that the bytes written for the two variables are different.

(a) What is the most likely reason for this difference?

(b) Does this tell you anything about how this language treats assignment?

12. (5 pts) A call-by-value parameter cnn only be used to provide input to a function. A call-by-reference parameter can provide input to or output from a function.
C++ offers an intriguing variation. We can pass a parameter as a "const reference" type, e.g.,

```
void foo (const T& t);
```

in which case it can be used only as input to the function.
Why is this a useful addition to a language that already supports one mechanism (call-by-value) for supplying input parameters?