CSCE 330 Fall 2001
MIDTERM EXAM
Friday 01/10/1—Two Pages, Closed Book

1 Short Questions–1 point each; 10 points total

1. What is the name of the family of programming languages whose structure is dictated by the von Neumann computer architecture? **Answer:** Imperative

2. Following up on the previous question, what are the names of the two other main families of programming languages? **Answer:** functional (or applicative) and logic

3. Provide an example of the convergence of software design methodology and programming language design. **Answer:** Information hiding (a methodological principle related to the algorithm design technique "divide and conquer" and the cognitive principle "separation of concerns") is reflected in data abstraction facilities (as embodied in the SIMULA-67 class, the Modula-2 module, the Ada package, the Smalltalk class, the CLU class, the C++ class, the Java class).

4. What does the acronym RUDE stand for? **Answer:** Run Understand Debug Edit

5. The spiral method of software development is characterized by prototyping. True or false? **True**

6. What readability problem is caused by using the same reserved word or symbol to close more than one control statement? (Hint: See Section 3.) **Answer:** The reader must match the closing reserved word to the opening reserved word by counting, rather than by matching a token.

7. Match {form, function} with {semantics, syntax}. **Answer:** (form, syntax), (function, semantics).

8. The following sentence is syntactically correct: "Time flies like green bananas." True or false? **Answer:** True.

9. What are the three components of the state in the denotational semantics approach[1]? **Answer:** mem, i, o

[1] Every question on denotational semantics refers to the simple language described in class.

10. There are three major approaches to describing the semantics of programming languages. List them. **Answer:** axiomatic, denotational, operational.

2 Syntax–14 points

1. (2 points) What does it mean for a (context-free) grammar to be *ambiguous*? **Answer:** The grammar generates a sentence that has two or more distinct parse trees.

2. (8 points) Prove that the following grammar is ambiguous:

```
<S> -> <A>
<A> -> <A> + <A> | <id>
<id> -> a | b | c
```

3. (4 points) Draw a syntax graph corresponding to the following EBNF rule: <ident> ::= letter {letter | digit}* **Answer:** Not provided here.

3 Scope and Type Rules–5 points

Consider the (pseudo-Pascal) program below.

```
program MAIN;
 var X: integer;
 procedure A;
  begin
   write(X)
  end; {of procedure A}
 procedure B;
  var X: integer;
  begin
   X := 10;
   call A
  end; {of procedure B}
 begin {of MAIN}
  X := 5;
   call B
end. {of program MAIN}
```

1. Under static scoping rules, what value of X is printed in procedure A? **Answer:** 5

2. Under dynamic scoping rules, what value of X is printed in procedure A? **Answer:** 10

4 Semantics–15 points

1. (2 points) What is aliasing? **Answer:** The situation in which two identifiers refer to the same program entity. Most typically, the situation in which two variable names refer to the same memory location.

2. (3 points) Complete this sentence: In denotational semantics, the memory function, *mem* is a function from the set of *mysteries1* to the set of *mysteries2*.

3. (5 points) Suppose that you have a `while` loop `while B do SL end`, where B is a Boolean expression and SL is a statement list. Suppose further that you have showed that an assertion I holds before the `while` and that I is invariant upon execution of SL, i.e. $I \wedge B \Rightarrow I$. You want to show that upon termination of the `while`, the assertion Q holds. What do you need to prove? **Answer:** $I \wedge \neg B \rightarrow Q$.

4. (5 points) Describe (very briefly) the semantic difference between commands and expressions. (Hint: Use the assignment statement, which is a command, to illustrate the difference.) **Answer:** A command is executed to update variables or to perform I./O. An expression is evaluated to yield a value. The assignment statement is a command in which the right hand side is an expression whose value becomes the value in the memory location corresponsing to the identifier on the right-hand side.