January 22, 1992 Name: ____________________

CS 342 — Principles of Programming Languages

Pre-test

This test does *not* enter into your final grade. Its purpose is to diagnose what prerequisite material you may need to review.

section	score	comments
Grammars		
Computability		
Math		
Computer Arch.		

1 Theory of Computing

1.1 Grammars

1. Consider the following grammar.

```
expression → value | variable
    | ( operator expression* )
    | ( set expression expression )
    | ( if expression expression expression )
    | ( while expression expression )
    | ( begin expression+ )
value → 1 | 2 | 3
variable → x | y | z
operator → + | * | - | / | = | < | > | print
```

(a) For each of the following, write "legal" if the expression is legal according to the above grammar, and "not legal" otherwise.

i. `(print x)`

ii. `begin x := 1 ; end`

iii. `(begin expression expression)`

iv. `(begin (set x 1) x)`

v. `(while x (set x 2) (set y 3))`

(b) Write 7 expressions in the above grammar, one for each of the productions of expression, making each expression longer than the previous one if possible. Check to make sure that each expression is legal according to the grammar. For example, your first answer might be "1". Please write each expression on a separate line.

1.2 Computability

1. Can a Pascal compiler tell you when your program will get into an infinite loop? Why or why not?

2. Suppose we eliminated the while loop and recursion from Pascal. Would it be possible to write a universal Turing machine (i.e., a program that simulates a universal Turing machine) in this restricted version of Pascal? Why or why not?

1.3 Math

1. Recall that the factorial function maps a positive integer n to n times $n-1$ factorial and it maps 0 to 1. For example, the factorial of 4 is 24.

   (a) What is the domain of the factorial function?

   (b) Briefly explain why the factorial of -3 is undefined.

   (c) What is the range of the factorial function.

2. The following English sentences are to be translated into the first-order predicate calculus. Translate the sentences as literally as possible, so that the structure of the predicate corresponds directly to the structure of the English sentence.

   (a) If Y and $Q(x)$, then Z.

   (b) For all X, X is an element of S.

   (c) There is no X in L such that $L < Z$.

   Translate the following into English, and make your translation similar in structure to the predicate calculus.

   (a) $\forall X.(X \in T) \Rightarrow (X \in S)$

   (b) $(Q \wedge P \wedge R) \Rightarrow Z$

3. Negate and simplify the following formulas. For example, the negation of X is $\neg X$.

   (a) $(n \div m = 0) \vee n = m$

   (b) $(Q(X) \wedge R(X, Y)) \Rightarrow P(X, Y)$

2 Computer Architecture

1. How are Pascal (or C) pointers represented in a typical computer?

2. In describing programming langauges, such as Pascal, we often speak of "the stack". What is this stack? That is, what purpose does it serve?

3. In Pascal if `PTR` is a pointer variable, executing `new PTR` allocates space for `PTR` and puts a pointer to that space in `PTR`. Where is the space to which `PTR` points allocated?