Programming Language Comprehensive, August 21, 1997

INSTRUCTIONS. Do all 6 questions. You have 4 hours.

Question 1: Give a formal proof (or natural deduction style proof-tree) that the following specification is partially correct:

$$\{\text{true}\}\ \text{N} := 14518\ ;\ \textbf{while}\ \text{N} > 0\ \textbf{do}\ \text{N} := \text{N - } 17\ \{\text{N} = 0\ \}\ .$$

Note that $14518 = 17 * 854$.

Question 2: Suppose we have a programming language **P** and we provide a denotational semantics for it. Suppose we then go on to provide a formal system of axioms and rules. Discuss the relationship between the denotational semantics and the formal system of axioms and rules. Give an example of the denotation of the local definition block from Tennent's simple applicative language. What is the extensionality rule of inference? Argue that it is sound. (For less credit, discuss the soundness of the rule of inference for **while** coommands in the simple imperative language.)

Question 3:

a) Formulate an intuitively correct definition of

$$[\![\, \exists\, \text{Y(B)}\,]\!]\sigma$$

where B is a boolean expression. ('$\exists$ Y' means 'there exists Y'.)

b) Recall the form of the assignment axioms:

$$\{[\text{B}](\text{X} \mapsto \text{E})\}\ \text{X} := \text{E}\ \{\text{B}\}\,.$$

These axioms have a "backward" direction to them. In contrast, here is a form of "forward" axiom for assignment statements:

$$\{\text{B}\}\ \text{X} := \text{E}\ \{\exists\, \text{Y}\ ([\text{B}](\text{X} \mapsto \text{Y})\ \text{and}\ \text{X} = [\text{E}](\text{X} \mapsto \text{Y})\}$$

where Y is a variable that does not occur in B and does not occur in E.
Assume that B is a boolean expression and show that this axiom is valid.
HINT: The 'Y' in the post condition is the "old" value of X, i.e. the value of 'X' before X := E is executed.

Question 4: Calculate and simplify as much as you can by following the computational steps of a *plausible interpreter* for the simple imperative language:

$$[\![\ \textbf{letrec}\ \text{h} :\ \textbf{val[nat]} \to \textbf{val[nat]}\ \textbf{be}\ \lambda\, \text{n:}\textbf{val[nat]}.\textbf{if}\ \text{n} = 0\ \textbf{then}\ 1\ \textbf{else}\ \text{g(h(n-1))}\ \textbf{in}\ \text{h(2)}\ ]\!]\ \text{u}$$

Show all steps.

Question 5: Give an example of a syntactical programming language construct which is NOT context-free, and briefly sketch an explanation for your example.

Question 6:

a) Define *domain*. If you have in my mind a particular book or paper for this definition, tell which one.

b) A *directed set* is a nonempty subset A of a partially ordered set D with the property that every two elements in A have an upper bound in A. A partially ordered set D is *complete* if every directed subset of D has a least upper bound in D. Denote the least upper bound of a set A by $\bigsqcup A$. We call a partially ordered set which is complete a dcpo. (This stands for *directed-complete partial order*.) Let D and E be dcpo's. A function $f : D \longrightarrow E$ is a *compact* iff $\{f(a) \mid a \in A\}$ is directed and $f(\bigsqcup A) \sqsubseteq \bigsqcup\{f(a) \mid a \in A\}$ for all directed subsets of D. Give an example of a function which is compact, but not monotone. You must specify f, D and E.