CS411 Preliminary Examination

October 20, 2004

This exam is closed book. There are 9 questions in the exam. Please write your answers in an exam booklet. Make sure you clearly indicate your final answer for each question.

1. [8 pts] Indicate which of the following commands are equivalent:

   (a) while (b) do c
   (b) while (b) do (c;c)
   (c) while (b) do (c; while (b) do c)
   (d) while (b) do (while (b) do (c;c))

2. [12 pts] For each of the following partial correctness assertions, write an appropriate loop invariant that would make it possible to prove its validity:

   (a) [6 pts] \( i = 1 \) while \( i < 100 \) do \( i := i + 1 \) \( \{ i = 100 \} \)
   (b) [6 pts] \( i = 1 \) while \( i < 100 \) do \( i := i + 2 \) \( \{ i = 101 \} \)

3. [8 pts] What are the possible values of \( n \) for which the following partial correctness assertion holds?

   \( \{ x = n \} \ y := x - 1; x := x + 1; y := y * x \ \{ x = y + 2 \} \)

4. [7 pts] Are there any commands \( c \) for which the following Hoare-triple holds? If no, explain why. If yes, show an example.

   \( \{ x > 0 \} \ while (x > 0) do c \ (x > 0) \)

5. [18 pts] Suppose we build an analysis for IMP that identifies pairs of variables whose values are off by one. For this, we use an analysis domain: \( \text{Abs} = \text{Var} \times \text{Var} \rightarrow \{0, 1, ?\} \). The meaning of \( \text{Abs} \) is as follows: given \( a \in \text{Abs} \) and variables \( x \) and \( y \), then \( a(x,y) = 0 \) if \( x = y \); \( a(x,y) = 1 \) if \( x = y + 1 \); and \( a(x,y) = ? \) if the relation between the values of \( x \) and \( y \) is not known.

   (a) [6 pts] What is the most precise information that such an analysis can derive at the end of the following program: \( x := 0; y := 1; z := x + 1 \) ?
   (b) [12 pts] Show the analysis for assignments of the form \( x := y + 1 \). More precisely, given \( a \in \text{Abs} \) before the assignment, show how to compute the analysis information \( a' \) after the assignment. Make sure your analysis result is as accurate as possible.

6. [7 pts] What is the set of free variables of \( \lambda x. z (\lambda y. y x) y \)?

7. [7 pts] What is the result of following substitution: \( (\lambda x. y) (\lambda y. y x) \) \( [x/y] \)?

8. [7 pts] Which is true about the evaluation of the following expression: a) call-by-name is faster; b) call-by-value is faster; or c) they both take the same number of evaluation steps?

   \( (\lambda x. \lambda y. x y) ((\lambda x)(\lambda x)) (\lambda x) \)
9. [26 pts] Consider the following simple stack language STK:

\[
c \in \text{Com} \quad c ::= \text{skip} \mid n \mid x \mid \text{pop} \ x \mid + \mid c_1 ; c_2 \mid \text{if} \ c_1 c_2 \mid \text{loop} \ c
\]

\(n \in \text{Int}\)
\(x \in \text{Var}\)

The execution of the program maintains a store \(S : \text{Var} \rightarrow \text{Int}\) that maps variables to their values, and a stack \(T\) of integers. The empty stack is \(\emptyset\), and \((T : n)\) is a stack obtained by pushing value \(n\) on top of stack \(T\). The following rules describe the semantics of this language:

\[
\begin{align*}
\langle n, T, S \rangle & \rightarrow \langle \text{skip}, T : n, S \rangle \\
\langle x, T, S \rangle & \rightarrow \langle \text{skip}, T : S(x), S \rangle \\
\langle \text{pop} \ x, T : n, S \rangle & \rightarrow \langle \text{skip}, T, S[x \rightarrow n] \rangle \\
\langle +, T : n_1 : n_2, S \rangle & \rightarrow \langle \text{skip}, T : n, S \rangle \\
\langle c_1, T, S \rangle & \rightarrow \langle c'_1, T', S' \rangle \\
\langle c_1 ; c_2, T, S \rangle & \rightarrow \langle c'_1 ; c_2, T', S' \rangle \\
\langle \text{if} \ c_1 c_2, T : n_1 : n_2, S \rangle & \rightarrow \langle c_1, T, S \rangle \\
\langle \text{if} \ c_1 c_2, T : n_1 : n_2, S \rangle & \rightarrow \langle c_2, T, S \rangle
\end{align*}
\]

Final configurations are of the form \(\langle \text{skip}, T, S \rangle\).

(a) [7 pts] Identify all of the error configurations in STK.
(b) [7 pts] Write an error-free STK program that never terminates.
(c) [12 pts] For each of the following IMP commands, write an equivalent STK command (one that yields the same final store as the IMP command):

i. if \((x > 0)\) then \(x := x + 1\) else \text{skip}

ii. while \((x < y)\) do \(x := x + 2\)