Two hours

UNIVERSITY OF MANCHESTER
SCHOOL OF COMPUTER SCIENCE

Understanding Programming Languages

Date: Thursday 26th January 2012
Time: 14:00 - 16:00

Please answer any THREE Questions from the FIVE questions provided

This is a CLOSED book examination

The use of electronic calculators is NOT permitted

[PTO]
All questions on this paper refer to the While language whose syntax and semantics are given as follows.

A Abstract Machine and Compiler for While

\[ a ::= \quad n \mid x \mid a_1 + a_2 \mid a_1 \cdot a_2 \mid a_1 - a_2 \mid \]
\[ b ::= \quad \text{true} \mid \text{false} \mid a_1 = a_2 \mid a_1 \leq a_2 \mid -b \mid b_1 \land b_2 \]
\[ S ::= \quad x := a \mid \text{skip} \mid S_1 ; S_2 \mid \text{if } b \text{ then } S_1 \text{ else } S_2 \mid \text{while } b \text{ do } S \]

Table 1: Syntax of While

<table>
<thead>
<tr>
<th>Expression</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\langle PUSH - n : c, e, s \rangle</td>
<td>\langle c, N \cdot n \rangle : e, s</td>
</tr>
<tr>
<td>\langle ADD : c, z_1 : z_2 : e, s \rangle</td>
<td>\langle c, (z_1 + z_2) : e, s \rangle if ( z_1, z_2 \in \mathbb{Z} )</td>
</tr>
<tr>
<td>\langle MULT : c, z_1 : z_2 : e, s \rangle</td>
<td>\langle c, (z_1 \cdot z_2) : e, s \rangle if ( z_1, z_2 \in \mathbb{Z} )</td>
</tr>
<tr>
<td>\langle SUB : c, z_1 : z_2 : e, s \rangle</td>
<td>\langle c, (z_1 - z_2) : e, s \rangle if ( z_1, z_2 \in \mathbb{Z} )</td>
</tr>
<tr>
<td>\langle TRUE : c, e, s \rangle</td>
<td>\langle c, \text{true} : e, s \rangle</td>
</tr>
<tr>
<td>\langle FALSE : c, e, s \rangle</td>
<td>\langle c, \text{false} : e, s \rangle</td>
</tr>
<tr>
<td>\langle EQ : c, z_1 : z_2 : e, s \rangle</td>
<td>\langle c, (z_1 = z_2) : e, s \rangle if ( z_1, z_2 \in \mathbb{Z} )</td>
</tr>
<tr>
<td>\langle LE : c, z_1 : z_2 : e, s \rangle</td>
<td>\langle c, (z_1 \leq z_2) : e, s \rangle if ( z_1, z_2 \in \mathbb{Z} )</td>
</tr>
<tr>
<td>\langle AND : c, t_1 : t_2 : e, s \rangle</td>
<td>\langle c, \text{true} : e, s \rangle if ( t_1 = \text{true} ) and ( t_2 = \text{true} )</td>
</tr>
<tr>
<td>\langle NEG : c, t : e, s \rangle</td>
<td>\langle c, \text{false} : e, s \rangle if ( t = \text{false} )</td>
</tr>
<tr>
<td>\langle FETCH - x : c, e, s \rangle</td>
<td>\langle c, (s \cdot x) : e, s \rangle</td>
</tr>
<tr>
<td>\langle STORE - x : c, e, s \rangle</td>
<td>\langle c, e, s[x \rightarrow z] \rangle if ( z \in \mathbb{Z} )</td>
</tr>
<tr>
<td>\langle NOOP : c, e, s \rangle</td>
<td>\langle c, e, s \rangle</td>
</tr>
<tr>
<td>\langle BRANCH(c_1, c_2) : c, t : e, s \rangle</td>
<td>\langle c_1 : c, \text{false} : e, s \rangle if ( t = \text{false} )</td>
</tr>
<tr>
<td>\langle LOOP(c_1, c_2) : c, e, s \rangle</td>
<td>\langle c_1 : \text{BRANCH}(c_2 : \text{LOOP}(c_1, c_2), \text{NOOP}) : c, e, s \rangle</td>
</tr>
</tbody>
</table>

Table 2: Operational Semantics for AM
\( \mathcal{C}A[a] = \text{PUSH} - a \)
\( \mathcal{C}A[x] = \text{FETCH} - x \)
\( \mathcal{C}A[a_1 + a_2] = \mathcal{C}A[a_2] : \mathcal{C}A[a_1] : \text{ADD} \)
\( \mathcal{C}A[a_1 * a_2] = \mathcal{C}A[a_2] : \mathcal{C}A[a_1] : \text{MULT} \)
\( \mathcal{C}A[a_1 - a_2] = \mathcal{C}A[a_2] : \mathcal{C}A[a_1] : \text{SUB} \)

\( \mathcal{C}B[\text{true}] = \text{TRUE} \)
\( \mathcal{C}B[\text{false}] = \text{FALSE} \)
\( \mathcal{C}B[a_1 = a_2] = \mathcal{C}A[a_2] : \mathcal{C}A[a_1] : \text{EQ} \)
\( \mathcal{C}B[a_1 \leq a_2] = \mathcal{C}A[a_2] : \mathcal{C}A[a_1] : \text{LEQ} \)
\( \mathcal{C}B[-b] = \mathcal{C}B[b] : \text{NEG} \)
\( \mathcal{C}B[b_1 \land b_2] = \mathcal{C}B[b_2] : \mathcal{C}B[b_1] : \text{AND} \)

Table 3: Translation of Expressions

\( \mathcal{C}S[x := a] = \mathcal{C}A[a] : \text{STORE} - x \)
\( \mathcal{C}S[\text{skip}] = \text{NOOP} \)
\( \mathcal{C}S[S_1; S_2] = \mathcal{C}S[S_1] : \mathcal{C}S[S_2] \)
\( \mathcal{C}S[\text{if } b \text{ then } S_1 \text{ else } S_2] = \mathcal{C}B[b] : \text{BRANCH} (\mathcal{C}S[S_1], \mathcal{C}S[S_2]) \)
\( \mathcal{C}S[\text{while } b \text{ do } S] = \text{LOOP} (\mathcal{C}B[b], \mathcal{C}S[S]) \)

Table 4: Translation of Statements in While
1. a) Give a natural semantics for the statement part of the While language. (5 marks)

b) Prove that the semantics of While in the Appendix is equivalent to your natural semantics. (10 marks)

c) Extend your natural semantics to handle non-deterministic or statements. (5 marks)

2. a) Give a structural operational semantics for the statement part of the While language. (5 marks)

b) Prove that the semantics of While in the Appendix is equivalent to your structural operational semantics. (10 marks)

c) Extend your structural operational semantics to handle non-deterministic par statements. (5 marks)

3. a) Extend the given abstract machine and compiler to handle abort statements. (5 marks)

b) Extend the given abstract machine and compiler to handle non-deterministic or statements. (5 marks)

c) Extend your abstract machine and compiler to handle for loops. (5 marks)

d) Extend your abstract machine and compiler to handle statements of the form:

\[ \text{do } S \text{ while } b \] (5 marks)
4.  a) Give a denotational semantics for the statement part of the While language.  
    (5 marks)

   b) Prove that the semantics of While in the Appendix is equivalent to your denotational semantics.  
    (10 marks)

   c) Extend your denotational semantics to handle for statements.  
    (5 marks)

5.  a) Give an axiomatic semantics for the statement part of the While language.  
    (5 marks)

   b) Prove that the semantics of While in the Appendix is sound and complete with respect to your axiomatic semantics.  
    (10 marks)

   c) Extend your axiomatic semantics to handle repeat statements.  
    (5 marks)

END OF EXAMINATION