Mock Test for Software Verification (L19)

Matthew Parkinson
March 12, 2010

Instructions

Answer all questions in Part A.
Answer two questions from Part B.

Part A

Question 1. Prove the following, or say why they are not true:

1. \{x = 5\} x := 3 \{x = 3\}
2. \{\text{even}(x)\} x := x + 1 \{\text{odd}(x)\}
3. \{x = 5\} \text{skip} \{\text{odd}(x)\}

Give complete details of every rule used. Justify each logical implication you required in the rule of consequence. [6 marks]

Question 2. Prove the following:

1. \{true\} while true do skip \{false\}
2. \{x = 5\} while x=5 do (continue; x:=3) \{false\}

[4 marks]

Question 3. Prove the following:

1. \{a[x] = 3\} x := a[x] \{x = 3\}
2. \{true\} y := a[x]; a[y] := y \{a[x] = a[y]\}

[4 marks]

Question 4. Prove the following program:

\{true\}
let f(x) =
  if x = 0 then 1
  else
    local y in y := f(x-1);
    return x*y; in
  z := f(2)
\{z = 2\}
Explain which function definition and call rules you use, and why.
You may assume the following logical facts:

\[
\begin{align*}
\text{fact}(0) &= 1 \\
\forall n \cdot \text{fact}(n+1) &= (n+1) \cdot \text{fact}(n)
\end{align*}
\]

[8 marks]

**Question 5.** Using separation logic prove the following

1. \(\{x \mapsto 5\} \quad \{x \mapsto 6\}\) 
2. \(\{x = y \land x \mapsto z\} \quad \{x \mapsto [x] \land y \mapsto z \land x = z\}\)

Give full outlines and make it clear each rule that you use. 

[6 marks]

**Question 6.** Verify using separation logic

\[
\begin{align*}
\{\text{list}(x)\}\ disposelist(x) \{\text{empty}\} \vdash \{\text{list}(x)\}
\end{align*}
\]

local t in 
\[
\begin{align*}
t &:= [x]; \\
dispose x; \\
\text{if } t \neq 0 \text{ then disposelist}(t) \\
\{\text{empty}\}
\end{align*}
\]

[6 marks]

**Question 7.** Using separation logic, verify the following, or state why it is not possible:

1. \(\{x \mapsto \_\} \quad \{x \mapsto 5\} \lor \{x \mapsto 5\}\)
2. \(\{x \mapsto \_ \land f \neq g\} \quad \text{if } f = 1 \text{ then } \{x \mapsto 5\} \lor \{x \mapsto \_\} \quad \text{if } g = 1 \text{ then } \{x \mapsto 5\}

[8 marks]

**Question 8.** Using the Owicki/Gries method verify

\[
\begin{align*}
\{x = 0\} \\
x &=: x + 3 \lor x = 5 \\
\{x = 8 \lor x = 5\}
\end{align*}
\]

[10 marks]

**Part B**

**Question 9.** (a) The conjunction rule is:

\[
\begin{align*}
\{P_1\} \cap \{Q_1\} \\
\{P_2\} \cap \{Q_2\} \\
\{P_1 \cap P_2\} \cap \{Q_1 \cap Q_2\}
\end{align*}
\]
Prove the conjunction rule is sound for sequential Hoare logic. That is, show:

\[ \vdash \{P_1\} C \{Q_1\} \land \vdash \{P_2\} C \{Q_2\} \implies \vdash \{P_1 \land P_2\} C \{Q_1 \land Q_2\} \]

(b) The disjunction rule is:

\[
\begin{align*}
\{P_1\} C \{Q_1\} \\
\{P_2\} C \{Q_2\} \\
\hline
\{P_1 \lor P_2\} C \{Q_1 \lor Q_2\}
\end{align*}
\]

Prove the disjunction rule is sound for sequential Hoare logic.

**Question 10.** Verify the following programs meet the given specifications:

(a)  
\[
\begin{align*}
\{\text{true}\} & \\
r & := x; \\
d & := 0; \\
\text{while } r \geq y \text{ do} & \\
r & := r - y; \\
d & := d + 1 \\
\{x = (d \times y) + r \land r < y\}
\end{align*}
\]

(b)  
\[
\begin{align*}
\{\text{list}(x)\} & \\
h & := \text{new}; \\
t & := h; \\
\text{while true do} & \\
p & := t; \\
v & := [x+1]; \\
[p+1] & := v; \\
x & := [x]; \\
\text{if } x == 0 \text{ then break} & \\
t & := \text{new}; \\
[p] & := t; \\
[p] & := 0; \\
\{\text{list}(x) \ast \text{list}(h)\}
\end{align*}
\]

**Question 11.** Verify the following program meets its specification:

\[
\begin{align*}
\{x = 0\} & \\
\text{if } x == 0 \text{ then } x & := x + 1 \\
\| & \\
\text{if } x == 0 \text{ then } x & := x + 2 \\
\| & \\
\text{if } x == 0 \text{ then } x & := x + 1 \\
\{x \in \{1, 2, 3, 4\}\}
\end{align*}
\]

You will have to add auxiliary state to the program.  
You may use either Owicki/Gries method or rely-guarantee.
Question 12. Consider the following programming language with loops and rapid exits from the loops using break and continue:

\[ C ::= x := E \mid C; C \mid \text{if } B \text{ then } C \text{ else } C \mid \text{while } B \text{ do } C \mid \text{break } n \mid \text{continue } n \]

The command \textbf{break 1} breaks out of the directly enclosing loop. The command \textbf{break } n \text{ break out of } n \text{ enclosing loops. The command } \textbf{continue 1} \text{ restarts the directly enclosing loop. The command } \textbf{continue } n \text{ restarts the } n\text{th enclosing loop.}

(a) Extend the basic Hoare logic with breaks and continues to deal with additional parameter for the level of loop it applies to. Give rules for both \textbf{while-do} and \textbf{break } n \text{ and } \textbf{continue } n. \text{[Hint: Consider extending the context to carry multiple } \textbf{break} \text{ and } \textbf{continue} \text{ contexts for each level of loop.]} \text{(b) Verify the following program with your extended logic}

\{true\}
\hspace{1em} while true do
\hspace{2em} while true do
\hspace{3em} (x:=3; break 2)
\{x = 3\}