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

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. $\{\mathtt{x} = 5\}$`while x=5 do (continue; x:=3)` $\{false\}$

[4 marks]

Question 3. *Prove the following:*

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

[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)
```
$\{\mathtt{z} = 2\}$

Explain which function definition and call rules you use, and why.
You may assume the following logical facts:

$$fact(0) = 1$$
$$\forall n.\ fact(n+1) = (n+1) * fact(n)$$

[8 marks]

Question 5. *Using separation logic prove the following*

1. $\{\mathtt{x} \mapsto 5\}$ `[x] := 6` $\{\mathtt{x} \mapsto 6\}$
2. $\{\mathtt{x} = \mathtt{y} \wedge \mathtt{x} \mapsto \mathtt{z}\}$ `x := [x]` $\{\mathtt{y} \mapsto \mathtt{z} \wedge \mathtt{x} = \mathtt{z}\}$

Give full outlines and make it clear each rule that you use. *[6 marks]*

Question 6. *Verify using separation logic*

$\{list(\mathtt{x})\}$`disposelist`$(\mathtt{x})\{empty\}$ $\vdash$ $\{list(\mathtt{x})\}$

```
local t in
t := [x];
dispose x;
if t != 0 then disposelist(t)
```

$\{empty\}$

[6 marks]

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

1. $\{\mathtt{x} \mapsto \_\}$ `[x] := 5 || [x] := 5` $\{\mathtt{x} \mapsto 5\}$
2. $\{\mathtt{x} \mapsto \_ \wedge f \neq g\}$

```
if f = 1 then [x] := 5
||
if g = 1 then [x] := 5
```

$\{\mathtt{x} \mapsto \_\}$

[8 marks]

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

$\{\mathtt{x} = 0\}$

```
x := x + 3 || x := 5
```

$\{\mathtt{x} = 8 \vee \mathtt{x} = 5\}$

[10 marks]

Part B

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

$$\frac{\{P_1\}\,\mathtt{C}\,\{Q_1\} \quad \{P_2\}\,\mathtt{C}\,\{Q_2\}}{\{P_1 \wedge P_2\}\,\mathtt{C}\,\{Q_1 \wedge Q_2\}}$$

Prove the conjunction rule is sound for sequential Hoare logic. That is, show:

$$\models \{P_1\}\,\mathtt{C}\,\{Q_1\} \;\wedge\; \models \{P_2\}\,\mathtt{C}\,\{Q_2\} \implies \models \{P_1 \wedge P_2\}\,\mathtt{C}\,\{Q_1 \wedge Q_2\}$$

(b) The disjunction rule is:

$$\frac{\{P_1\}\,\mathtt{C}\,\{Q_1\} \quad \{P_2\}\,\mathtt{C}\,\{Q_2\}}{\{P_1 \vee P_2\}\,\mathtt{C}\,\{Q_1 \vee Q_2\}}$$

Prove the disjunction rule is sound for sequential Hoare logic.

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

(a)

$\{true\}$
```
r := x;
d := 0;
while r >= y do
  r := r - y;
  d := d + 1
```
$\{\mathtt{x} = (\mathtt{d} \times \mathtt{y}) + \mathtt{r} \wedge \mathtt{r} < \mathtt{y}\}$

(b)

$\{list(\mathtt{x})\}$
```
h := new;
t := h;
while true do
  p := t
  v := [x+1];
  [p+1] := v;
  x := [x];
  if x==0 then break;
  t := new;
  [p] := t;
[p] := 0;
```
$\{list(\mathtt{x}) * list(\mathtt{h})\}$

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

$\{\mathtt{x} = 0\}$
```
if x=0 then x := x + 1
||
if x=0 then x := x + 2
||
if x=0 then x := x + 1
```
$\{\mathtt{x} \in \{1, 2, 3, 4\}\}$

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:*

$$C ::= \; x := E \mid C; C \mid \texttt{if}\ B\ \texttt{then}\ C\ \texttt{else}\ C \mid \texttt{while}\ B\ \texttt{do}\ C \\ \texttt{break}\ n \mid \texttt{continue}\ n$$

The command `break 1` *breaks out of the directly enclosing loop. The command* `break` n *break out of* n *enclosing loops. The command* `continue 1` *restarts the directly enclosing loop. The command* `continue` n *restarts the* n*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 `while-do` *and* `break` n *and* `continue` n*. [Hint: Consider extending the context to carry multiple* `break` *and* `continue` *contexts for each level of loop.]*

(b) Verify the following program with your extended logic

$\{true\}$
```
 while true do
   while true do
     (x:=3; break 2)
```
$\{\mathtt{x} = 3\}$