Software Testing & Verification 2013/2014
Universiteit Utrecht

2nd Jul. 2014, 13:30 - 16:30, BBL 001

Lecturer: Wishnu Prasetya

You are allowed to bring along the Appendix of the LN.

Part I [3pt (6×0.5)]

For each question, choose *one* correct answer.

1. What is the **weakest** pre-condition of the following statement with respect to the given post-condition?

   $$\{*\ ?\ *\}\ \ \texttt{x} := \texttt{x+y}\ ;\ \texttt{y} := \texttt{x+3}\ \ \{*\ \texttt{xy} = \texttt{0};*\}$$

   (a) $\texttt{x}^2 + \texttt{y}^2 + 2\texttt{xy} + 3\texttt{x} + 3\texttt{y} \quad = \quad 0$

   (b) $2\texttt{x}^2 + 9\texttt{x} + 9 \quad = \quad 0$

   (c) $(\texttt{x+y})(\texttt{x+3}) \quad = \quad 0$

   (d) $(\texttt{x} = 0) \wedge (\texttt{y} = 0)$

2. What is the **weakest** pre-condition of the following statement with respect to the given post-condition?

   $$\{*\ ?\ *\}\quad \texttt{a}[0] := \texttt{a}[0] - \texttt{a}[\texttt{k}]\ \ \{*\ \texttt{a}[\texttt{k}]{=}0\ *\}$$

   (a) $\texttt{a}[\texttt{k}] = 0$

   (b) $\texttt{k} = 0$

   (c) $(\texttt{k}{=}0\ \rightarrow\ \texttt{a}[0]{-}\texttt{a}[\texttt{k}]\ |\ \texttt{a}[\texttt{k}])\ =\ 0$

   (d) $\texttt{a}(0\ \textbf{repby}\ (\texttt{a}\ \textbf{repby}\ 0) - (\texttt{a}\ \textbf{repby}\ \texttt{k}))[\texttt{k}] \quad = \quad 0$

3. Consider the following program to search for a prime number between a and b. It's body is not fully shown: *body* below is some statement, and e is some expression. The parameter a is passed by value, and b by copy-restore. The *body* is known to modify a and b.

$$\texttt{find}(\texttt{a} : \texttt{int},\ \texttt{OUT b} : \texttt{int})\ :\ \texttt{bool}\ \{\ body\ ;\ \textbf{return}\ e\ \}$$

Here is the specification of the program:

$$\{*\ \texttt{0<a}\leq\texttt{b}\ *\}$$

$$\texttt{B}_0 := \texttt{b}\ ;\ \texttt{find(a, OUT b)}$$

$$\{*\ (\texttt{return} = (\exists \texttt{x} : \texttt{a}\leq\texttt{x}<\texttt{B}_0 : \texttt{isPrime(x)}))\ \wedge\ (\texttt{return} \Rightarrow \texttt{isPrime(b)})\ *\}$$

Which of the following specifications is a correct reduction of the above specification to the corresponding specification of the program's body?

(a)

$$\{*\ \texttt{0<a}\leq\texttt{b}*\}$$

$$body\ ;\ \texttt{return} := e$$

$$\{*\ (\texttt{return} = (\exists \texttt{x} : \boxed{a}\leq\texttt{x}<\boxed{b} : \texttt{isPrime(x)}))\ \wedge\ (\texttt{return} \Rightarrow \texttt{isPrime}(\boxed{b}))\ *\}$$

(b)

$$\{*\ \texttt{0<a}\leq\texttt{b}*\}$$

$$\texttt{B}_0 := \texttt{b}\ ;\ body\ ;\ \texttt{return} := e$$

$$\{*\ (\texttt{return} = (\exists \texttt{x} : \boxed{a}\leq\texttt{x}<\boxed{B_0} : \texttt{isPrime(x)}))\ \wedge\ (\texttt{return} \Rightarrow \texttt{isPrime}(\boxed{B_0}))\ *\}$$

(c)

$$\{*\ \texttt{0<a}\leq\texttt{b}*\}$$

$$\texttt{A}_0, \texttt{B}_0 := \texttt{a}, \texttt{b}\ ;\ body\ ;\ \texttt{return} := e$$

$$\{*\ (\texttt{return} = (\exists \texttt{x} : \boxed{A_0}\leq\texttt{x}<\boxed{B_0} : \texttt{isPrime(x)}))\ \wedge\ (\texttt{return} \Rightarrow \texttt{isPrime}(\boxed{b}))\ *\}$$

(d)

$$\{*\ \texttt{0<a}\leq\texttt{b}*\}$$

$$\texttt{A}_0, \texttt{B}_0 := \texttt{a}, \texttt{b}\ ;\ body\ ;\ \texttt{return} := e$$

$$\{*\ (\texttt{return} = (\exists \texttt{x} : \boxed{A_0}\leq\texttt{x}<\boxed{b} : \texttt{isPrime(x)}))\ \wedge\ (\texttt{return} \Rightarrow \texttt{isPrime}(\boxed{b}))\ *\}$$

4. Which of the following proofs is correct (according the the proof system of the LN)? Read the steps carefully.

   (a) PROOF
   [A1:] $(\forall x :: P\ x)$
   [A2:] $Q\ x$
   [G:] $(\forall x :: P\ x \wedge Q\ x)$
   1. { $\forall$-elimination on A1 } $P\ x$
   2. { conjunction of 1 and A2 } $P\ x \wedge Q\ x$
   3. { $\forall$-introduction on 2 } $(\forall x :: P\ x \wedge Q\ x)$
   END

   (b) PROOF
   [A1:] $a = b$
   [G:] $a \vee (\exists k :: x[k]) \;=\; b \vee (\exists k :: x[k])$
   1. { $\vee$-introduction } $a \vee (\exists k :: x[k])$
   2. { $\vee$-introduction } $b \vee (\exists k :: x[k])$
   3. { combining 1 and 2 } $a \vee (\exists k :: x[k]) \;=\; b \vee (\exists k :: x[k])$
   END

   (c) PROOF
   [A1:] $(\exists x :: P\ x)$
   [A2:] $Q\ a$
   [G:] $(\exists a :: P\ a \wedge Q\ a)$
   1. { $\exists$-elimination on A1 } $P\ a$
   2. { conjunction of 1 and A2 } $P\ a \wedge Q\ a$
   3. { $\exists$-introduction on 2 } $(\exists a :: P\ a \wedge Q\ a)$
   END

   (d) PROOF
   [A1:] $\neg(\exists x :: P\ x)$
   [A2:] $P\ a$
   [G:] false
   1. { $\exists$-introduction on A2 } $(\exists a :: P\ a)$
   2. { contradiction between A1 and 1 } false
   END

5. A statement S satisfies the following specifications:

   (a) $\{* \ P \ *\}\ \ S\ \ \{* \ Q_1 \ *\}$
   (b) $\{* \ Q_2 \ *\}\ \ S\ \ \{* \ R \ *\}$, where $Q_2 \Rightarrow Q_1$ (note the direction!)

   Which of the folowing specifications is a valid consequence of (a) and (b) above ?

   (a) $\{* \ P \ *\}\ \ S; S\ \ \{* \ R \ *\}$
   (b) $\{* \ P \wedge Q_2 \ *\}\ \ S\ \ \{* \ Q_1 \wedge R \ *\}$
   (c) $\{* \ P \vee Q_2 \ *\}\ \ S\ \ \{* \ Q_1 \wedge R \ *\}$
   (d) $\{* \ P \ *\}\ \ S; S\ \ \{* \ Q_2 \Rightarrow R \ *\}$

6. Consider the loop below; `x` is of type `int` and `even(x)` is a side-effect-free function that checks if x is an even integer.

   $\{*\ 1 < \mathtt{x} < \mathtt{N}\ *\}$

   `while x<N do { if even(x) then x := 2 * x  else x := x - 1 }`

   $\{*\ \mathtt{even(x)}\ *\}$

   Which of the predicates below is a correct invariant of the loop, that is enough to prove that the above specification is valid, under the partial correctness interpretation?

   (a) $1{<}\mathtt{x} \wedge (\exists \mathtt{x} :: \mathtt{even(x)})$
   (b) $(\mathtt{even(x)} \Rightarrow \mathtt{even(2x)})\ \wedge\ (\neg\mathtt{even(x)} \Rightarrow \mathtt{even(x}-1))$
   (c) $1{<}\mathtt{x}{\leq}\mathtt{N} \wedge \mathtt{even(x)}$
   (d) $\mathtt{x}{\geq}\ \mathtt{N} \Rightarrow \mathtt{even(x)}$

Part II [7pt]

When asked to write a formal proof you need to produce one that is readable, augmented with sufficient comments to explain and convincingly defend your steps. An incomprehensible solution may lose all points.

1. [1.5 pt] **Termination**

   Consider again this program, with the same pre-condition:

   $\{*\ 1 < \mathtt{x} < \mathtt{N}\ *\}$

   `while x<N do { if even(x) then x := 2 * x  else x := x - 1 }`

   Use the Loop Reduction Rule (the inference rule for loop as discussed in the lectures) to prove that this program terminates when executed on the given pre-condition. You only need to prove termination; we do not care in which state the program would terminate.

2. [3 pt] **Loop**

   Here is a program to check if all elements of an array `a[0..N)` are the same.

```
{*  N > 0  *}     // pre-condition

   i := 1 ;
   uniform:= true ;
   while i<N do {
      uniform := uniform ∧ (a[i]=a[0]) ;
      i := i+1
   } ;

{*  uniform  =  (∀k : 0≤k<N : a[k] = a[0])  *}    // post-condition
```

   Give a formal proof that the program is correct. You can skip the termination proof.

3. [1.5 pt] **Adding a break**

   The program from No. 2 can be improved by letting the loop to break when $\texttt{a[i}-1] \neq \texttt{a}[0]$:

```
{*  N > 0  *}     // pre-condition

    i := 1 ;
    uniform:= true ;
    while i<N ∧ [a[i-1]=a[0]] do {
       uniform := uniform ∧ (a[i]=a[0]) ;
       i := i+1
    } ;

{*  uniform  =  (∀k : 0≤k<N : a[k] = a[0])  *}     // post-condition
```

   Give a new formal proof of the loop Exit Condition, that will prove that using the same invariant as in No. 2, the version above will also terminate in the specified post-condition above.

   (You only need to give a new PEC proof)

4. [1 pt] **Program call**

   Consider the following specification of the program P:

   $\{*\ \texttt{y}>0\ *\}\ \ \texttt{Y} := \texttt{y};\ \texttt{P(x:int, OUT y:int)}\ \ \{*\ (\texttt{return}+\texttt{y})/\texttt{Y} > \texttt{x}\ *\}$

   Consider this call to P:

   $\{*\ \texttt{k}>0\ *\}\ \ \texttt{r} := \texttt{P}(\texttt{k}-2, \texttt{k})\ \ \{*\ \texttt{r}+\texttt{k}>0\ *\}$

   To prove the correctness of the call, we first transform the call to the following equivalent statement:

```
{* k>0 *}
{* (1)  ? *}  @x := k-2 ;
{* (2)  ? *}  @y := k ;
{* (3)  ? *}  r := P(@x,@y) ;
{* (4)  ? *}  k := @y ;
{* r+k>0 *}
```

   (a) Fill in the intermediate predicates (1)..(4) above. Calculate them using the weakest-precondition function, and for (3) use the Black Box reduction rule for program call.

   Just give the answers; you do not have to show the calculation.

   (b) Based on your calculation above, is the call correct? Motivate your answer.