**The Australian National University**
**School of Computing**
Dirk Pattinson and Victor Rivera

Semester 2, 2021
Practical Session 5

# Foundations of Computation

*The practical contains a number of exercises designed for the students to practice the course content. During the practical session, the tutor will work through some of these exercises while students will be responsible for completing the remaining exercises in their own time. There is no expectation that all the exercises will be covered in the practical session.*

Covers: Lecture Material Week 5

At the end of this tutorial, you will be able to prove the partial correctness (loop-less) of imperative programs using Hoare Logic.

**Exercise 1** **Hoare Logic: Semantics**

Determine the truth value of the following Hoare triples and give your reasoning. You are *not* required to use Hoare Logic to prove these!

1. $\{j = a\}$ `j:=j+1` $\{a = j + 1\}$

   False. Consider the case of $a = j = 0$; the precondition is satisfied, but after the assignment $a = 0$ and $j = 1$ and the postcondition is false.

2. $\{i > j\}$ `j:=i+1; i:=j+1` $\{i > j\}$

   **Solution.** True. Regardless of the initial state, the computation ends by making $i = j+1$ and so $i > j$ is satisfied.

3. $\{i \neq j\}$ `if i>j then m:=i-j else m:=j-i` $\{m > 0\}$

   **Solution.** True. If $i > j$ then $m$ will be assigned a positive quantity. The other possibility is that $i < j$, given that the precondition holds, and a



**Exercise 2** **Assignment Axiom**

Prove each of the following. The first one has been done for you in detail, but you can just giv
and their justifications.



1. $\{i = 5\}$ `a` $:=$ `i` $+ 2\ \{(a = 7) \wedge (i = 5)\}$

   We apply the 'copy-and-replace' assignment axiom. The precondition this generates doesn't look exactly like the one we are asked for, but it is equivalent to it:

   1. $\{(i + 2 = 7) \wedge (i = 5)\}$ `a` $:=$ `i` $+ 2\ \{(a = 7) \wedge (i = 5)\}$ (Assignment)
   2. $(i + 2 = 7) \wedge (i = 5) \leftrightarrow i = 5$ (Logic)
   3. $\{i = 5\}$ `a` $:=$ `i` $+ 2\ \{(a = 7) \wedge (i = 5)\}$ (Precondition Equivalence, 1, 2)

   We could have used precondition strengthening instead of precondition equivalence here.

2. $\{(i = 5) \wedge (a = 3)\}$ `a:=i+2` $\{a = 7\}$

   **Solution.** Again we start with the assignment axiom, but this time the precondition we generate is *not* equivalent to the one we want - it is too *weak*. Hence we must invoke precondition strengthening:

   1. $\{i + 2 = 7\}$ `a` $:=$ `i` $+ 2\ \{a = 7\}$ (Assignment)
   2. $(i = 5\ \wedge\ a = 3)\ \rightarrow\ (i + 2 = 7)$ (Basic Maths. & Logic)
   3. $\{(i = 5) \wedge (a = 3)\}$ `a` $:=$ `i` $+ 2\ \{a = 7\}$ (Precondition Strengthening, 1, 2)

3. $\{i = a - 1\}$ `i:=i+2` $\{i = a + 1\}$

   **Solution.** Assignment axiom, then precondition equivalence, once again.

**Exercise 3** **Control Structures**

Prove each of the following. The first one has been done for you. You need only give the formal proof steps and their justifications.

1. $\{a > b\}$ `m:=1; n:=a-b` $\{m * n > 0\}$
   Start at the end of the program, and apply the assignment axiom to the command $\texttt{n} := \texttt{a} - \texttt{b}$ and the given post-condition. This generates the precondition $m * (a - b) > 0$. Apply the assignment axiom again with this new intermediate assertion and $\texttt{m} := \texttt{1}$, rearrange the precondition we generate from this to get $a > b$, then use the sequencing rule to stick the proof together.

   Formally:

   1. $\{m * (a - b) > 0\}\ \texttt{n} := \texttt{a} - \texttt{b}\ \{m * n > 0\}$ (Assignment)
   2. $\{1 * (a - b) > 0\}\ \texttt{m} := \texttt{1}\ \{m * (a - b) > 0\}$ (Assignment)
   3. $1 * (a - b) > 0 \leftrightarrow a > b$ (Logic)
   4. $\{a > b\}\ \texttt{m} := \texttt{1}\ \{m * (a - b) > 0\}$ (Precondition Equivalence, 2, 3)
   5. $\{a > b\}\ \texttt{m} := \texttt{1}; \texttt{n} := \texttt{a} - \texttt{b}\ \{m * n > 0\}$ (Sequence, 4, 1)

2. $\{s = 2^i\}$ `i:=i+1; s:=s*2` $\{s = 2^i\}$

   **Solution.**

   1. $\{s * 2 = 2^i\}\ \texttt{s} := \texttt{s} * \texttt{2}\ \{s = 2^i\}$ (Assignment)
   2. $\{s * 2 = 2^{i+1}\}\ \texttt{i} := \texttt{i} + \texttt{1}\ \{s * 2 = 2^i\}$ (Assignment)
   3. $s * 2 = 2^{i+1} \leftrightarrow s = 2^i$ (Logic)
   4. $\{s = 2^i\}\ \texttt{i} := \texttt{i} + \texttt{1}\ \{s * 2 = 2^i\}$ (Precondition Equivalence, 2, 3)
   5. $\{s = 2^i\}\ \texttt{i} := \texttt{i} + \texttt{1}; \texttt{s} := \texttt{s} * \texttt{2}\ \{s = 2^i\}$ (Sequence, 1, 4)

   LHS of Step 3 is valid because we can divide both sides of the equation by 2. Note also that we have proved that $(s = 2^i)$ is an invariant for this code.



3. $\{True\}$ `if i<j then min:=i else min:=j`

   **Solution.** We obviously n atch for the top line of this rule:



   1. $\{(i \leq i) \wedge (i \leq j)\}\ \texttt{min} := \texttt{i}\ \{(min \leq i) \wedge (min \leq j)$ (Assignment)
   2. $(True \wedge (i < j)) \rightarrow ((i \leq i) \wedge (i \leq j))$ (Basic Maths.)
   3. $\{True \wedge (i < j)\}\ \texttt{min} := \texttt{i}\ \{(min \leq i) \wedge (min \leq j$ on Strengthening, 1, 2)
   4. $\{(j \leq i) \wedge (j \leq j)\}\ \texttt{min} := \texttt{j}\ \{(min \leq i) \wedge (min \leq$ (Assignment)
   5. $True \wedge \neg(i < j) \leftrightarrow (j \leq i) \wedge (j \leq j)$ (Logic)
   6. $\{True \wedge \neg(i < j)\}\ \texttt{min} := \texttt{j}\ \{(min \leq i) \wedge (min \leq j)\}$ (Precondition Equivalence, 4, 5)
   7. $\{True\}$ `if i<j then min:=i else min:=j` $\{(min \leq i) \wedge (min \leq j)\}$ (Condition, 3, 6)



**Exercise 4** **More on Semantics**

Determine the truth value of the following Hoare triples and give your reasoning. You are *not* required to use Hoare Logic to prove these!

1. $\{i = j\}$ `i:=j+i` $\{i > j\}$

   **Solution.** False. Consider the case that $i$ is negative. For example, if $i = j = -1$ initially, then the postcondition will be false.

2. $\{i = 3 * j\}$ `if i>j then m:=i-j else m:=j-i` $\{m - 2 * j = 0\}$

   **Solution.** False. If $i = -3$ and $j = -1$ then $m$ will end up as 2, and so the postcondition will not hold.

3. $\{a = 0\}$ `while x>a do x:=x-1` $\{x = 0\}$

   **Solution.** False. If initially $x = -1$, or any other negative number, then the loop terminates with no iterations and $x$ is unchanged.

**Exercise 5** **More on Assignment Axiom**

Prove each of the following.

1. $\{i = 5\}$ `a:=i+2` $\{(a = 7) \wedge (i > 0)\}$

   **Solution.** This follows by the same reasoning as Exercise 2.1 above - the assignment axiom, then precondition equivalence. Alternatively, we could apply *postcondition weakening* to the result of Exercise 2.1.

2. $\{a = 7\}$ `i:=i+2` $\{a = 7\}$

   **Solution.** This question is a trivial assignment axiom one-liner. The importance of the example is that you often want to know that some variable's value is not affected by the execution of some given code. This is a sort of ***invariant***.

   Formally:

   1. $\{a = 7\}$ `i` $:=$ `i` $+ 2$ $\{a = 7\}$ (Assignment)

3. $\{True\}$ `a:=i+2` $\{a = i + 2\}$

   **Solution.** Assignment axiom and precondition equivalence again. The only thing that is a little unusual is the constant *True* as the precondition, so discuss what this means (that if the code fragment terminates then it does so fulfilling the postcondition, whatever the opening state might have been). What might it mean if *False* was the precondition?

   Formally:

   1. $\{i + 2 = i + 2\}$ `a` $:=$ `i` $+ 2$ $\{a = i + 2\}$ (Assignment)
   2. $True \leftrightarrow i + 2 = i + 2$ (Logic)
   3. $\{True\}$ `a` $:=$ `i` $+ 2$ $\{a = i + 2\}$ (Precondition Equivalence, 1, 2)

**Exercise 6** **Control Structures (2)**



1. Prove the following Hoare triple: $\{(i > 0) \wedge (j > 0)\}$ `if i<j then min:=i else min:=j` $\{min > 0\}$.

   **Solution.** Follows simila

   Formally:



   1. $\{i > 0\}$ `m` (Assignment)
   2. $i > 0 \wedge j > 0 \wedge i < j \rightarrow i > 0$ (Basic Logic)
   3. $\{i > 0 \wedge j > 0 \wedge i < j\}$ `m` $:=$ `i` $\{min > 0\}$ or Strengthening, 1, 2)
   4. $\{j > 0\}$ `min` $:=$ `j` $\{min > 0\}$ (Assignment)
   5. $i > 0 \wedge j > 0 \wedge \neg(i < j) \rightarrow j > 0$ (Basic Logic)
   6. $\{i > 0 \wedge j > 0 \wedge \neg(i < j)\}$ `min` $:=$ `j` $\{min > 0\}$ (Precondition Strengthening, 4, 5)
   7. $\{i > 0 \wedge j > 0\}$ `if` `i` $<$ `j` `then` `min` $:=$ `i` `else` `min` $:=$ `j` $\{min > 0\}$ (Condition, 3, 6)



2. Consider the following Hoare triple:

   $$\{x = y\}\ \texttt{if (x=0) then x:=y+1 else z:=y+1}\ \{(z = 1) \rightarrow (x = 1)\}$$

   - determine whether it is valid or not
   - if it is not valid, justify why this is the case by giving values for the variables that satisfy the precondition, and argue that the postcondition is *not* true after running the code
   - if it is valid, give a proof in Hoare logic.

   **Solution.**

   (a) This Hoare triple is valid. We give the following Hoare logic proof:

   1 $\{z = 1 \rightarrow y + 1 = 1\}$ `x := y+1` $\{z = 1 \rightarrow x = 1\}$ (Assignment)
   2 $(x = y \wedge x = 0) \rightarrow (z = 1 \rightarrow y + 1 = 1)$ (Basic Logic)
   3 $\{x = y \wedge x = 0\}$ `x := y+1` $\{z = 1 \rightarrow x = 1\}$ (Precondition Strengthening, 1, 2)
   4 $\{y + 1 = 1 \rightarrow x = 1\}$ `z := y+1` $\{z = 1 \rightarrow x = 1\}$ (Assignment)
   5 $(x = y \wedge x \neq 0) \rightarrow (y + 1 = 1 \rightarrow x = 1)$ (Basic Logic)
   6 $\{x = y \wedge x \neq 0\}$ `z := y+1` $\{z = 1 \rightarrow x = 1\}$ (Precondition Strengthening, 4, 5)
   7 $\{x = y\}$ `if (x=0) then x:=y+1 else z:=y+1` $\{(z = 1) \rightarrow (x = 1)\}$ (Conditional, 3, 6)

## Appendix — Hoare Logic Rules

- Assignment:

$$\{Q(e)\}\ \texttt{x} := \texttt{e}\ \{Q(x)\}$$

- Precondition Strengthening:

$$\frac{P_s \ \rightarrow\ P_w \qquad \{P_w\}\ S\ \{Q\}}{\{P_s\}\ S\ \{Q\}}$$

  You can always replace predicates by equivalent predicates,
  i.e. if $P_s \leftrightarrow P_w$; just label your proof step with 'precondition equivalence'.

- Postcondition Weakening:

$$\frac{\{P\}\ S\ \{Q_s\} \qquad Q_s\ \rightarrow\ Q_w}{\{P\}\ S\ \{Q_w\}}$$

  You can always replace predicates by equivalent predicates,
  i.e. if $Q_s \leftrightarrow Q_w$; just label your proof step with 'postcondition equivalence'.

- Sequence:

$$\frac{\{P\}\ S_1\ \{Q\} \qquad \{Q\}\ S_2\ \{R\}}{\{P\}\ S_1; S_2\ \{R\}}$$

- Conditional:

$$\frac{\{P \wedge b\}\ S_1\ \{Q\} \qquad \{P \wedge \neg b\}\ S_2\ \{Q\}}{\{P\}\ \textbf{if}\ b\ \textbf{then}\ S_1\ \textbf{else}\ S_2\ \{Q\}}$$