CS720
Logical Foundations of Computer Science
Lecture 16: Program verification (part 3)
Tiago Cogumbreiro
Equiv.v

Due Thursday October 25, 11:59pm EST
Imp.v

Due Friday October 26, 11:59pm EST
Hoare.v, HoareAsLogic.v, Hoare2.v

Due Thursday November 1, 11:59pm EST
Summary

- Axiomatic Hoare Logic
- Program verification using Hoare logic
On the strength of propositions

Recall the rule for consequence

\[ P \implies P' \quad \{P'\} \ c \ \{Q'\} \quad Q' \implies Q \]

\[ \{P\} \ c \ \{Q\} \]

We mentioned that we can **strengthen** the pre-condition and **weaken** the post-condition.

1. *Strengthening a pre-condition* \((P \implies P')\) means having **more assumptions** to reach the same goal.
2. *Weakening a post-condition* \((Q' \implies Q)\) means having **fewer goals** to prove.
Stronger and weaker statements

- When you think of the strength of propositions, think of \( \implies \) as \( \geq \).
- We say that \( P \) is (strictly) stronger than \( Q \) if \( P \implies Q \) (and \( \neg(Q \implies P) \))
  That is, (strict-)strength corresponds to (strict-)implication.

Between \( x = 3 \land y = 10 \) and \( x = 3 \), which is stronger than the other?
Stronger and weaker statements

- When you think of the strength of propositions, think of $\implies$ as $\geq$.
- We say that $P$ is (strictly) stronger than $Q$ if $P \implies Q$ (and $\neg(Q \implies P)$)
  
  That is, (strict-)strength corresponds to (strict-)implication.

Between $x = 3 \land y = 10$ and $x = 3$, which is stronger than the other?

- $x = 3 \land y = 10$ is stronger than $x = 3$, which is weaker
Weakest pre-condition


The *weakest-pre-condition* of a program \( c \) and a post-condition \( \{ Q \} \), is such that we can always prove \( Q \).

**Definition** \( \text{wp} \ (c:\text{com}) \ (Q:\text{Assertion}) : \text{Assertion} := \)

\[
\text{fun } s \Rightarrow \forall s', c / s \setminus s' \rightarrow Q \ s'.
\]

1. **Theorem** (WP is the pre-condition of any program): \( \{ \text{wp}(c, Q) \} c \{ Q \} \)

2. **Theorem** (WP is the weakest pre-condition): If \( \{ P \} c \{ Q \} \), then \( \{ P \} \rightarrow \{ \text{wp}(c, Q) \} \).
Axiomatic Hoare Logic

HoareAsLogic.v
Hoare Logic Theory

\{P\} \text{SKIP} \{P\} \quad (\text{H-skip})

\{P\} \ c_1 \ \{Q\} \quad \{Q\} \ c_2 \ \{R\} \quad (\text{H-seq})

\{P\} \ c \ \{Q\} \quad \text{IF} \ b \ \text{THEN} \ c_1 \ \text{ELSE} \ c_2 \ \text{FI} \ \{Q\} \quad (\text{H-if})

\{P\} \ \text{WHILE} \ b \ \text{DO} \ c \ \text{END} \ \{P\} \wedge \neg b \quad (\text{H-while})
Hoare Logic as an Axiomatic Logic

The theorems in Slide 6 are necessary and sufficient to show that a Hoare's triple holds. We can use the theorems as axioms (rules) and encode Hoare's Logic axiomatically!

- **Necessary** condition (soundness): \( \text{hoare\_proof}(P, c, Q) \rightarrow \{P\} \ c \ \{Q\} \)
- **Sufficient** condition (completeness): \( \{P\} \ c \ \{Q\} \rightarrow \text{hoare\_proof}(P, c, Q) \)

Inductive hoare\_proof : Assertion \rightarrow com \rightarrow Assertion \rightarrow Type :=

- **H\_Skip** : forall P, hoare\_proof P (SKIP) P
- **H\_Asgn** : forall Q \ V a, hoare\_proof (assn\_sub V a Q) (V ::= a) Q
- **H\_Seq** : forall P \ c \ Q \ d \ R, hoare\_proof P c Q \rightarrow hoare\_proof Q d R \rightarrow hoare\_proof P c; d R
- **H\_If** : forall P \ Q \ b \ c1 \ c2,
  hoare\_proof (fun st \rightarrow P st \\/ \ bassn \ b \ st) c1 Q \rightarrow
  hoare\_proof (fun st \rightarrow P st \\/ \ ~(bassn \ b \ st)) c2 Q \rightarrow
  hoare\_proof P (IFB b THEN c1 ELSE c2 FI) Q
- **H\_While** : forall P \ b \ c,
  hoare\_proof (fun st \rightarrow P st \\/ \ bassn \ b \ st) c P \rightarrow
  hoare\_proof P (WHILE b DO c END) (fun st \rightarrow P st \\/ \ ~(bassn \ b \ st))
- **H\_Consequence** : forall (P \ Q \ P' : Assertion) c,
  hoare\_proof P' c Q' \rightarrow (forall st, P st \rightarrow P' st) \rightarrow (forall st, Q' st \rightarrow Q st) \rightarrow hoare\_proof P c Q.
Why an Axiomatic Hoare Logic?

- When defining a logic axiomatically, you get the principles of injectivity (inversion) and induction for free.
- For instance, given some evidence, we can reason about how we reached that conclusion (ie, using the rules/constructors).
- In this specific case, it forces us to think more deeply about Hoare's logic (eg, learn about the weakest pre-condition).

Your homework is to prove soundness and completeness!

- Soundness (easy): \( \text{hoare\_proof}(P, c, Q) \rightarrow \{ P \} \ c \ \{ Q \} \)
- Completeness (hard): \( \{ P \} \ c \ \{ Q \} \rightarrow \text{hoare\_proof}(P, c, Q) \)
Exercise

Theorem hoare_proof_complete: forall P c Q, {{P}} c {{Q}} → hoare_proof P c Q.

Proof.

The proof follows by induction on the structure of c.

- At each case our goal is to apply the rule that relates to the term. (when \( c = \text{SKIP} \) we apply H-skip, when \( c = s ::= a \) we apply H-asgn, and so on).

- When applying a rule and it requires a condition \(?P\) we don’t know how to fill, supply the weakest precondition of the post-condition.
Verifying programs

Hoare2.v
Example

What does this algorithm do and how do we specify this algorithm?

\[
X ::= X + Y;;
Y ::= X - Y;;
X ::= X - Y
\]
Example

Pre and post condition

$$\{\{ X = m \land Y = n \}\}$$

\[
X ::= X + Y ; ;
Y ::= X - Y ; ;
X ::= X - Y
\]

$$\{\{ X = n \land Y = m \}\}$$
Fully annotated example

Let us learn how to annotate a program

\[
\begin{align*}
\{\{ X = m \land Y = n \}\} & \implies \\
\{\{ (X + Y) - ((X + Y) - Y) = n \land (X + Y) - Y = m \}\} & \\
& \quad \text{X ::= X + Y;} \\
\{\{ X - (X - Y) = n \land X - Y = m \}\} & \\
& \quad \text{Y ::= X - Y;} \\
\{\{ X - Y = n \land Y = m \}\} & \\
& \quad \text{X ::= X - Y} \\
\{\{ X = n \land Y = m \}\}
\end{align*}
\]
Annotating assignments/sequences

Start from the post-condition and work backwards. The pre-condition of the program must imply the pre-condition of the first instruction.

1. \{\{ X = m \land Y = n \}\} \rightarrow

2. \{\}
   \hspace{1cm} \{\}
   \hspace{1cm} X ::= X + Y;;

3. \{\}
   \hspace{1cm} \{\}
   \hspace{1cm} Y ::= X - Y;;

4. \{\}
   \hspace{1cm} \{\}
   \hspace{1cm} X ::= X - Y

5. \{\{ X = n \land Y = m \}\}
Annotating assignments/sequences

In an assignment you have \( \{\{ P \left[ X \rightarrow A \right] \} \} X ::= a \{\{ P \} \} \), so take the post-condition (5) and replace \( X \) by \( a \).

1. \( \{\{ X = m \land Y = n \} \} \rightarrow \)
2. \( \{\} \)
   \( X ::= X + Y ; ; \)
3. \( \{\} \)
   \( Y ::= X - Y ; ; \)
4. \( \{\} \)
   \( X ::= X - Y \)
5. \( \{\{ X = n \land Y = m \} \} \)
Annotating assignments/sequences

In an assignment you have \{\{ P \[ X \rightarrow A \] \}\} X ::= a \{\{ P \}\}, so take the post-condition (5) and replace X by a.

1. \{\{ X = m \land Y = n \} \}
2. \{\{
   X ::= X + Y;;
\}\}
3. \{\{
   Y ::= X - Y;;
\}\}
4. \{\{
   X - Y = n \land Y = m \}
   X ::= X - Y
\}\}
5. \{\{
   X = n \land Y = m \}
\}\}

CS720: Lecture 16  Tiago Cogumbreiro
Annotating assignments/sequences

In an assignment you have \( \{\{ P \left[ X \mid \rightarrow A \right] \} \} \ X ::= a \ \{\{ P \} \} \), so take the post-condition (5) and replace \( X \) by \( a \).

1. \( \{\{ X = m \land Y = n \} \} \rightarrow \)
2. \( \{\{ X ::= X + Y ;; \} \}
3. \( \{\{ X - (X - Y) = n \land X - Y = m \} \}
    Y ::= X - Y ;;
4. \( \{\{ X - Y = n \land Y = m \} \}
    X ::= X - Y
5. \( \{\{ X = n \land Y = m \} \} \)
Annotating assignments/sequences

In an assignment you have \( \{\{ \text{P} \ [ \ X \ |\rightarrow \ A \ ] \} \} \ X ::= \text{a} \ \{\{ \text{P} \}\}, \) so take the post-condition (5) and replace \( X \) by \( \text{a} \).

1. \( \{\{ \ X = m \land Y = n \} \} \rightarrow \)
2. \( \{\{ \ (X + Y) - ((X + Y) - Y) = n \land (X + Y) - Y = m \} \}
    \quad X ::= X + Y;; \)
3. \( \{\{ \ X - (X - Y) = n \land X - Y = m \} \}
    \quad Y ::= X - Y;; \)
4. \( \{\{ \ X - Y = n \land Y = m \} \}
    \quad X ::= X - Y \)
5. \( \{\{ \ X = n \land Y = m \} \} \)

Finally, prove all the consequence-rules, that is show that \( (1) \rightarrow (2) \).
Annotating conditionals

Conditionals follow this structure

\[
\begin{align*}
&\text{IFB } b \text{ THEN} \\
&\{\{ P \land b \} \} \\
&c1 \\
&\{\{ Q \} \} \\
&\text{ELSE} \\
&\{\{ P \land \neg b \} \} \\
&c2 \\
&\{\{ Q \} \} \\
&\text{FI} \\
&\{\{ Q \}\} \\
\end{align*}
\]

\[
\begin{align*}
&\{\{ \text{True} \}\} \\
&\text{IFB } X \leq Y \text{ THEN} \\
&\{\{ \} \} \\
&\{\{ \} \} \\
&Z := Y - X \\
&\text{ELSE} \\
&\{\{ \} \} \\
&\{\{ \} \} \\
&Z := X - Y \\
&\{\{ \} \} \\
&\text{FI} \\
&\{\{ Z + X = Y \lor Z + Y = X \}\}
\end{align*}
\]
Annotating conditionals

Conditionals follow this structure

\[
\begin{align*}
\text{IFB } b \text{ THEN} \\
\quad \{\{ P \land b \} \} \\
\quad \{\{ Q \} \} \\
\quad c1 \\
\text{ELSE} \\
\quad \{\{ P \land \neg b \} \} \\
\quad \{\{ Q \} \} \\
\quad c2 \\
\text{FI} \\
\{\{ Q \} \}
\end{align*}
\]

\[
\begin{align*}
\text{IFB } \ X \leq \ Y \text{ THEN} \\
\quad \{\{ \text{True} \} \} \\
\quad \{\{ \} \} \\
\quad \{\{ \} \} \\
\quad Z := Y - X \\
\quad \{\{ Z + X = Y \lor Z + Y = X \} \} \\
\quad \{\{ \} \} \\
\quad \text{ELSE} \\
\quad \{\{ \} \} \\
\quad \{\{ \} \} \\
\quad Z := X - Y \\
\quad \{\{ Z + X = Y \lor Z + Y = X \} \} \\
\quad \{\{ \} \} \\
\text{FI} \\
\{\{ Z + X = Y \lor Z + Y = X \} \}
\end{align*}
\]
Annotating conditionals

Conditionals follow this structure

```plaintext
{{ P }}
IFB b THEN
  {{ P ∧ b }}
c1
  {{ Q }}
ELSE
  {{ P ∧ ¬b }}
c2
  {{ Q }}
FI
  {{ Q }}

{{ True}}
IFB X ≤ Y THEN
  {{ True ∧ X ≤ Y }} →
  Z ::= Y - X
  {{ Z + X = Y ∨ Z + Y = X}}
ELSE
  {{ True ∧ ¬ (X ≤ Y) }} →
  Z ::= X - Y
  {{ Z + X = Y ∨ Z + Y = X}}
FI
  {{ Z + X = Y ∨ Z + Y = X}}
```
Annotating conditionals

Conditionals follow this structure

\[
\begin{align*}
\{\{ P \}\} \\
\text{IFB } b \text{ THEN} \\
\{\{ P \wedge b \}\} \\
c1 \\
\{\{ Q \}\} \\
\text{ELSE} \\
\{\{ P \wedge \neg b \}\} \\
c2 \\
\{\{ Q \}\} \\
\text{FI} \\
\{\{ Q \}\}
\end{align*}
\]
Annotating loops

```plaintext
WHILE b DO
  {P \land b}
  c
  {P}
END
{P \land \neg b}

{{ True }} \implies
{{ X := m; }}
{{ Y := 0; }}

WHILE n \leq X DO
  {n \times Y + X = m \land X < n }
  X := X - n;
  Y := Y + 1
END
```
Annotating loops

\[
\begin{align*}
\{\{ P \}\} & \quad \text{WHILE } b \text{ DO} \\
\{\{ P \land b \}\} & \quad c \\
\{\{ P \}\} & \quad \text{END} \\
\{\{ P \land \neg b \}\} & \\
\end{align*}
\]

\[
\begin{align*}
\{\{ \text{True} \}\} & \quad \rightarrow \\
\{\{ \text{True} \}\} & \quad \rightarrow \\
\{\{ X \equiv m;\} & \quad \{\{ X \equiv m;\} \\
\{\{ Y \equiv 0;\} & \quad \{\{ Y \equiv 0;\} \\
\{\{ n \times Y + X = m \} & \quad \{\{ n \times Y + X = m \} \\
\text{WHILE } n \leq X \text{ DO} & \quad \text{WHILE } n \leq X \text{ DO} \\
\{\{ \} & \quad \{\{ \\
\{\{ X \equiv X - n;\} & \quad \{\{ X \equiv X - n;\} \\
\{\{ Y \equiv Y + 1 & \quad \{\{ Y \equiv Y + 1 \\
\{\{ n \times Y + X = m \} & \quad \{\{ n \times Y + X = m \} \\
\text{END} & \quad \text{END} \\
\{\{ n \times Y + X = m \land X < n \} & \quad \{\{ n \times Y + X = m \land X < n \}
\end{align*}
\]
Annotating loops

{{ \{ P \} \}}
WHILE b DO
  {{ \{ P \land b \} \}}
    c
  {{ \{ P \} \}}
END
{{ \{ P \land \neg b \} \}}

{{ \{ True \} \} \implies}
{{ \{ \} \}}
X ::= m;;
{{ \{ \} \}}
Y ::= 0;;
{{ \{ n \cdot Y + X = m \} \}}
WHILE n \leq X DO
  {{ \{ n \cdot Y + X = m \land n \leq X \} \} \implies}
  {{ \{ \} \}}
    X ::= X - n;;
  {{ \{ \} \}}
    Y ::= Y + 1
{{ \{ n \cdot Y + X = m \} \}}
END
{{ \{ n \cdot Y + X = m \land X < n \} \}}
Annotating loops

```latex
\{\{ P \}\} 
\textbf{WHILE} b \textbf{DO} 
\{\{ P \land b \}\} 
\textbf{c} 
\{\{ P \}\} 
\textbf{END} 
\{\{ P \land \neg b \}\}
```

```latex
\{\{ \text{True} \}\} \implies 
\{\{ \}\} 
X := m;; 
\{\{ \}\} 
Y := 0;; 
\{\{ n \ast Y + X = m \}\} 
\textbf{WHILE} n \leq X \textbf{DO} 
\{\{ n \ast Y + X = m \land n \leq X \}\} \implies 
\{\{ \}\} 
X := X - n;; 
\{\{ n \ast (Y + 1) + X = m \}\} 
Y := Y + 1 
\{\{ n \ast Y + X = m \}\} 
\textbf{END} 
\{\{ n \ast Y + X = m \land X < n \}\}
```
Annotating loops

\[
\{ \{ P \} \} \\
\text{WHILE } b \text{ DO} \\
\quad \{ \{ P \land b \} \} \\
\quad c \\
\quad \{ \{ P \} \} \\
\text{END} \\
\{ \{ P \land \neg b \} \} \\
\]

\[
\{ \{ \text{True} \} \} \\
\{ \{ \} \} \\
X ::= m;; \\
\{ \{ \} \} \\
Y ::= 0;; \\
\{ \{ n \times Y + X = m \} \} \\
\text{WHILE } n \leq X \text{ DO} \\
\quad \{ \{ n \times Y + X = m \land n \leq X \} \} \implies \\
\quad \{ \{ n \times (Y + 1) + (X - n) = m \} \} \\
\quad X ::= X - n;; \\
\quad \{ \{ n \times (Y + 1) + X = m \} \} \\
\quad Y ::= Y + 1 \\
\quad \{ \{ n \times Y + X = m \} \} \\
\text{END} \\
\{ \{ n \times Y + X = m \land X < n \} \} \\
\]
Annotating loops

```plaintext
{{ P }}
WHILE b DO
  {{ P ∧ b }}
  c
  {{ P }}
END
{{ P ∧ ¬b }}
```

```plaintext
{{ True }} →
{{
  X ::= m;;
  {{ n * 0 + X = m }}
  Y ::= 0;;
  {{ n * Y + X = m }}
  WHILE n ≤ X DO
    {{ n * Y + X = m ∧ n ≤ X }}
    {{ n * (Y + 1) + (X - n) = m }}
    X ::= X - n;;
    {{ n * (Y + 1) + X = m }}
    Y ::= Y + 1
    {{ n * Y + X = m }}
  END
  {{ n * Y + X = m ∧ X < n }}
}}
```
Annotating loops

```plaintext
{{ P }}
WHILE b DO
  {{ P ∧ b }}
  c
  {{ P }}
END
{{ P ∧ ¬b }}
```

```plaintext
{{ True }} ➞
{{ n * 0 + m = m }}
  X ::= m;;
{{ n * 0 + X = m }}
  Y ::= 0;;
{{ n * Y + X = m }}
  WHILE n ≤ X DO
{{ n * Y + X = m ∧ n ≤ X }} ➞
{{ n * (Y + 1) + (X - n) = m }}
  X ::= X - n;;
{{ n * (Y + 1) + X = m }}
  Y ::= Y + 1
{{ n * Y + X = m }}
END
{{ n * Y + X = m ∧ X < n }}
```
Loop invariants

Finding the loop invariant $P$ is undecidable!

It depends on what the body of $c$ is and its surrounding conditions:

1. weak enough to be implied by the loop's precondition
2. strong enough to imply the program's postcondition
3. preserved by one iteration of the loop

To read, a survey on the subject:
Loop invariants: analysis, classification, and examples. Furia et al. [10.1145/2506375]
Example

First, fill in the pre-/post-conditions template

\{\{ X = m \land Y = n \}\}\}

\textbf{WHILE} !(X = 0) \textbf{DO}

\begin{align*}
Y & := Y - 1;; \\
X & := X - 1
\end{align*}

\textbf{END}

\{\{ Y = n - m \}\}\}
Example with the template

First, fill in the pre-/post-conditions template

\[
\{\{ X = m \land Y = n \}\} \implies \\
\{\{ I \}\} \\
\text{WHILE } !(X = 0) \text{ DO} \\
\{\{ I \} / \neg !(X = 0) \}\} \\
Y ::= Y - 1;; \\
X ::= X - 1 \\
\{\{ I \}\} \\
\text{END} \\
\{\{ I \} / \neg !(X = 0) \}\} \implies \\
\{\{ Y = n - m \}\}
Example with the template

Second, fill in the assignments

```
{{ X = m ∧ Y = n }}
{{ I }}
WHILE !(X = 0) DO
{{ I \(\text{∧}! (X = 0)\) }}
{{ I [X |\(\rightarrow\) X-1] [Y |\(\rightarrow\) Y-1] }}
Y ::= Y - 1;;
{{ I [X |\(\rightarrow\) X-1] }}
X ::= X - 1
{{ I }}
END
{{ I \(\text{∧}! (X = 0)\) }}
{{ Y = n - m }}
```
Invariant heuristics

**Technique 1:** Use the weakest invariant, that is let $I$ be $\text{True}$.

\[
\begin{align*}
\{\{ X = m \land Y = n \}\} & \implies \\
\{\{ I \}\} & \\
\text{WHILE } !(X = 0) \text{ DO} & \\
& \{\{ I \land \neg (X = 0) \}\} \implies \\
& \{\{ I \land [X|\to X-1] \land [Y|\to Y-1] \}\} \\
& Y := Y - 1;; \\
& \{\{ I \land [X|\to X-1] \}\} \\
& X := X - 1 \\
& \{\{ I \}\} & \\
\text{END} & \\
\{\{ I \land \neg (X = 0) \}\} & \implies \\
\{\{ Y = n - m \}\} & \\
\end{align*}
\]
Invariant heuristics

Technique 1: Use the weakest invariant, that is let $I$ be $\text{True}$.  

\[
\begin{align*}
\{\{ X = m \land Y = n \}\} & \rightarrow \\
\{\{ I \}\} & \\
\text{WHILE } ! (X = 0) \text{ DO} & \\
\{\{ I \land ! (X = 0) \}\} & \rightarrow \\
\{\{ I [X \mapsto X-1] [Y \mapsto Y-1] \}\} & \\
Y & := Y - 1; \\
\{\{ I [X \mapsto X-1] \}\} & \\
X & := X - 1 \\
\{\{ I \}\} & \\
\text{END} & \\
\{\{ I \land \sim ! (X = 0) \}\} & \rightarrow \\
\{\{ Y = n - m \}\} &
\end{align*}
\]

In this example it fails, as $X \neq 0 \rightarrow Y = n - m$ is unprovable!
Invariant heuristics

**Technique 2:** Use the loop's post-condition, that is let I be $Y = n - m$.

\[
\{\{ X = m \land Y = n \} \} \implies \\
\{\{ I \} \}
\]

WHILE $!(X = 0)$ DO
\[
\{\{ I \land !(X = 0) \} \} \implies \\
\{\{ I [X \mapsto X-1] [Y \mapsto Y-1] \} \}
\]

$Y := Y - 1$;
\[
\{\{ I [X \mapsto X-1] \} \}
\]

$X := X - 1$
\[
\{\{ I \} \}
\]

END
\[
\{\{ I \land \neg !(X = 0) \} \} \implies \\
\{\{ Y = n - m \} \}
\]

\[
\{\{ X = m \land Y = n \} \} \implies \\
\{\{ Y = n - m \} \}
\]

WHILE $!(X = 0)$ DO
\[
\{\{ Y = n - m \land \neg !(X = 0) \} \} \implies \\
\{\{ Y - 1 = n - m \} \}
\]

$Y := Y - 1$;
\[
\{\{ Y = n - m \} \}
\]

$X := X - 1$
\[
\{\{ Y = n - m \} \}
\]

END
\[
\{\{ Y = n - m \land \neg !(X = 0) \} \} \implies \\
\{\{ Y = n - m \} \}
\]
Invariant heuristics

**Technique 2**: Use the loop's post-condition, that is let $I$ be $Y = n - m$.

$$\begin{align*}
\{\{ X = m \land Y = n \}\} \implies \\
\{\{ \ I \}\}
\end{align*}$$

WHILE !(X = 0) DO

$$\begin{align*}
\{\{ \ I \ \land \ !(X = 0) \}\} \implies \\
\{\{ \ I \ [X \mapsto X-1] \ [Y \mapsto Y-1] \}\}
\end{align*}$$

$Y ::= Y - 1;;$

$$\begin{align*}
\{\{ \ I \ [X \mapsto X-1] \}\} \\
X ::= X - 1 \\
\{\{ \ I \}\}
\end{align*}$$

END

$$\begin{align*}
\{\{ \ I \ \land \ \sim \ !(X = 0) \}\} \implies \\
\{\{ \ Y = n - m \}\}
\end{align*}$$

$$\begin{align*}
\{\{ \ Y = n - m \}\} \implies \\
\{\{ \ Y = n - m \}\}
\end{align*}$$

WHILE !(X = 0) DO

$$\begin{align*}
\{\{ \ Y = n - m \ \land \ \!(X = 0) \}\} \implies \\
\{\{ \ Y - 1 = n - m \}\}
\end{align*}$$

$Y ::= Y - 1;;$

$$\begin{align*}
\{\{ \ Y = n - m \ [X \mapsto X-1] \}\} \\
X ::= X - 1 \\
\{\{ \ Y = n - m \}\}
\end{align*}$$

END

$$\begin{align*}
\{\{ \ Y = n - m \ \land \ \sim \ !(X = 0) \}\} \implies \\
\{\{ \ Y = n - m \}\}
\end{align*}$$

In this example it fails, $Y$ changes during the loop, while $m$ and $n$ are constant. Idea: check how the values of $Y$ and $X$ relate to each other (sample their values by executing the program).
Invariant heuristics

**Technique 3:** Sample the variables mentioned in the post-condition and think of what would their value be in the \(i\)-th iteration? Let \(I\) be \(Y - X = n - m\).

\[
\{\{ X = m \land Y = n \}\} \implies \\
\{\{ I \}\} \\
\text{WHILE } !(X = 0) \text{ DO} \\
\{\{ I \land \neg(X = 0) \}\} \implies \\
\{\{ I \iff X \rightarrow X-1 \land Y \rightarrow Y-1 \}\} \\
Y ::= Y - 1; \\
\{\{ I \iff X \rightarrow X-1 \}\} \\
X ::= X - 1 \\
\{\{ I \}\} \\
\text{END} \\
\{\{ I \land \neg(X = 0) \}\} \implies \\
\{\{ Y - X = n - m \}\}
\]

\[
\{\{ X = m \land Y = n \}\} \implies \\
\{\{ Y - X = n - m \}\} \\
\text{WHILE } !(X = 0) \text{ DO} \\
\{\{ Y - X = n - m \land \neg(X = 0) \}\} \implies \\
\{\{ Y - 1) - (X - 1) = n - m \}\} \\
Y ::= Y - 1; \\
\{\{ Y - (X - 1) = n - m \iff X \rightarrow X-1 \}\} \\
X ::= X - 1 \\
\{\{ Y - X = n - m \}\} \\
\text{END} \\
\{\{ Y - X = n - m \land \neg(X = 0) \}\} \implies \\
\{\{ Y = n - m \}\}
\]
Summary

- Axiomatic Hoare Logic
- Program verification using Hoare logic