Program Verification

What is a "correct program"?

Program Verification

What is a “correct program”?
It has to terminate.

Program Verification

What is a “correct program”?
It has to terminate.
Assuming it terminates, its ending state should match some stated objective.

Program Verification

What is a "correct program"?
It has to terminate. Provably impossible to detect (for all programs)
Assuming it terminates, its ending state should match some stated objective.

How do we specify what a program is supposed to do?

Program Verification

What is a "correct program"?
It has to terminate. Provably impossible to detect (for all programs)
Assuming it terminates, its ending state should match some stated objective.

How do we specify what a program is supposed to do?
initial assertion: conjunction of propositions about the initial variables used in the program

Program Verification

What is a "correct program"?
It has to terminate. Provably impossible to detect (for all programs)
Assuming it terminates, its ending state should match some stated objective.

How do we specify what a program is supposed to do?
initial assertion: conjunction of propositions about the initial variables used in the program
final assertion: conjunction of propositions about the final state of the program

Program Verification

What is a "correct program"?
It has to terminate. Provably impossible to detect (for all programs)
Assuming it terminates, its ending state should match some stated objective.

How do we specify what a program is supposed to do?
initial assertion: conjunction of propositions about the initial variables used in the program
final assertion: conjunction of propositions about the final state of the program
Note: There are many programs that are "correct" for the same criteria. The ends justify the means.

Program Verification

What is a "correct program"?
It has to terminate. Provably impossible to detect (for all programs)
Assuming it terminates, its ending state should match some stated objective.

How do we specify what a program is supposed to do?
initial assertion: conjunction of propositions about the initial variables used in the program
final assertion: conjunction of propositions about the final state of the program
Note: There are many programs that are "correct" for the same criteria. The ends justify the means.

What kind of programs will we consider?

Program Verification

What is a "correct program"?
It has to terminate. Provably impossible to detect (for all programs)
Assuming it terminates, its ending state should match some stated objective.

How do we specify what a program is supposed to do?
initial assertion: conjunction of propositions about the initial variables used in the program
final assertion: conjunction of propositions about the final state of the program
Note: There are many programs that are "correct" for the same criteria. The ends justify the means.

What kind of programs will we consider?
We will look at:

- Assignment statements
- Sequences of statements
- Conditional statements (If B then S1 else S2)
- iteration statements (while loops)

Hoare Triples, Assignments, and Sequences

Hoare Triple: $p\{S\}q$:

If p is true for initial state of code S, and S terminates,
then q is true about the final state.

Hoare Triples, Assignments, and Sequences

Hoare Triple: $p\{S\}q$:
If p is true for initial state of code S, and S terminates,
then q is true about the final state.
p is the pre-condition
q is the post-condition.

Hoare Triples, Assignments, and Sequences

Hoare Triple: $p\{S\}q$:
If p is true for initial state of code S, and S terminates,
then q is true about the final state.
p is the pre-condition
q is the post-condition.

Assignment operator:
$p(e)\{v \Leftarrow e\}p(v)$

Hoare Triples, Assignments, and Sequences

Hoare Triple: $p\{S\}q$:
If p is true for initial state of code S, and S terminates,
then q is true about the final state.
p is the pre-condition
q is the post-condition.

Assignment operator:
$p(e)\{v \Leftarrow e\}p(v)$
$ODD(y)\{x \Leftarrow y + 2\}ODD(x)$

Hoare Triples, Assignments, and Sequences

Hoare Triple: $p\{S\}q$:
If p is true for initial state of code S, and S terminates,
then q is true about the final state.
p is the pre-condition
q is the post-condition.

Assignment operator:
$p(e)\{v \Leftarrow e\}p(v)$
$ODD(y)\{x \Leftarrow y + 2\}ODD(x)$
$ODD(x)\{x \Leftarrow x + 1\}Even(x)$

Hoare Triples, Assignments, and Sequences

Hoare Triple: $p\{S\}q$:
If p is true for initial state of code S, and S terminates,
then q is true about the final state.
p is the pre-condition
q is the post-condition.

Assignment operator:
$p(e)\{v \Leftarrow e\}p(v)$
$ODD(y)\{x \Leftarrow y+2\}ODD(x)$
$ODD(x)\{x \Leftarrow x+1\}Even(x)$

Sequencing of statements:
$p\{S_1\}q$
$q\{S_2\}r$

$p\{S_1; S_2\}r$

Examples

Consider the following simple program:
$y \Leftarrow 3; z \Leftarrow x + y$

Examples

Consider the following simple program:
$y \Leftarrow 3; z \Leftarrow x + y$

Claim: If we have the pre-condition $x = 1$, we have the post-condition $z = 4$.

Examples

Consider the following simple program:
$y \Leftarrow 3; z \Leftarrow x + y$

Claim: If we have the pre-condition $x = 1$, we have the post-condition $z = 4$.
$(x = 1)\{y \Leftarrow 3\}(x = 1 \wedge y = 3)$

Examples

Consider the following simple program:
$y \Leftarrow 3; z \Leftarrow x + y$

Claim: If we have the pre-condition $x = 1$, we have the post-condition $z = 4$.
$(x = 1)\{y \Leftarrow 3\}(x = 1 \wedge y = 3)$
$(x = 1 \wedge y = 3)\{z \Leftarrow x + y\}(z = 4)$

Examples

Consider the following simple program:
$y \Leftarrow 3; z \Leftarrow x + y$

Claim: If we have the pre-condition $x = 1$, we have the post-condition $z = 4$.
$(x = 1)\{y \Leftarrow 3\}(x = 1 \wedge y = 3)$
$(x = 1 \wedge y = 3)\{z \Leftarrow x + y\}(z = 4)$

$(x = 1)\{y \Leftarrow 3; z \Leftarrow x + y\}(z = 4)$

Examples

Consider the following simple program:
$y \Leftarrow 3; z \Leftarrow x + y$

Claim: If we have the pre-condition $x = 1$, we have the post-condition $z = 4$.
$(x = 1)\{y \Leftarrow 3\}(x = 1 \wedge y = 3)$
$(x = 1 \wedge y = 3)\{z \Leftarrow x + y\}(z = 4)$

$(x = 1)\{y \Leftarrow 3; z \Leftarrow x + y\}(z = 4)$

Consider the following simple program:
$x \Leftarrow x + 2; y \Leftarrow y + 1$

Examples

Consider the following simple program:
$y \Leftarrow 3; z \Leftarrow x + y$

Claim: If we have the pre-condition $x = 1$, we have the post-condition $z = 4$.
$(x = 1)\{y \Leftarrow 3\}(x = 1 \wedge y = 3)$
$(x = 1 \wedge y = 3)\{z \Leftarrow x + y\}(z = 4)$

$(x = 1)\{y \Leftarrow 3; z \Leftarrow x + y\}(z = 4)$

Consider the following simple program:
$x \Leftarrow x + 2; y \Leftarrow y + 1$

Claim: if we have the pre-condtion $x = 2y$, we have the post-condition $x = 2y$.

Examples

Consider the following simple program:
$y \Leftarrow 3; z \Leftarrow x + y$

Claim: If we have the pre-condition $x = 1$, we have the post-condition $z = 4$.
$(x = 1)\{y \Leftarrow 3\}(x = 1 \wedge y = 3)$
$(x = 1 \wedge y = 3)\{z \Leftarrow x + y\}(z = 4)$

$(x = 1)\{y \Leftarrow 3; z \Leftarrow x + y\}(z = 4)$

Consider the following simple program:
$x \Leftarrow x + 2; y \Leftarrow y + 1$

Claim: if we have the pre-condtion $x = 2y$, we have the post-condition $x = 2y$.
$(x = 2y)\{x \Leftarrow x + 2\}(x = 2y + 2)$

Examples

Consider the following simple program:
$y \Leftarrow 3; z \Leftarrow x + y$

Claim: If we have the pre-condition $x = 1$, we have the post-condition $z = 4$.
$(x = 1)\{y \Leftarrow 3\}(x = 1 \wedge y = 3)$
$(x = 1 \wedge y = 3)\{z \Leftarrow x + y\}(z = 4)$

$(x = 1)\{y \Leftarrow 3; z \Leftarrow x + y\}(z = 4)$

Consider the following simple program:
$x \Leftarrow x + 2; y \Leftarrow y + 1$

Claim: if we have the pre-condtion $x = 2y$, we have the post-condition $x = 2y$.
$(x = 2y)\{x \Leftarrow x + 2\}(x = 2y + 2)$
$(x = 2(y + 1))\{y \Leftarrow y + 1\}(x = 2y)$

Examples

Consider the following simple program:
$y \Leftarrow 3; z \Leftarrow x + y$

Claim: If we have the pre-condition $x = 1$, we have the post-condition $z = 4$.
$(x = 1)\{y \Leftarrow 3\}(x = 1 \wedge y = 3)$
$(x = 1 \wedge y = 3)\{z \Leftarrow x + y\}(z = 4)$

$(x = 1)\{y \Leftarrow 3; z \Leftarrow x + y\}(z = 4)$

Consider the following simple program:
$x \Leftarrow x + 2; y \Leftarrow y + 1$

Claim: if we have the pre-condtion $x = 2y$, we have the post-condition $x = 2y$.
$(x = 2y)\{x \Leftarrow x + 2\}(x = 2y + 2)$
$(x = 2(y + 1))\{y \Leftarrow y + 1\}(x = 2y)$

$(x = 2y)\{x \Leftarrow x + 2; y \Leftarrow y + 1\}(x = 2y)$

Examples

Consider the following simple program:
$y \Leftarrow 3; z \Leftarrow x + y$

Claim: If we have the pre-condition $x = 1$, we have the post-condition $z = 4$.
$(x = 1)\{y \Leftarrow 3\}(x = 1 \wedge y = 3)$
$(x = 1 \wedge y = 3)\{z \Leftarrow x + y\}(z = 4)$

$(x = 1)\{y \Leftarrow 3; z \Leftarrow x + y\}(z = 4)$

Consider the following simple program:
$x \Leftarrow x + 2; y \Leftarrow y + 1$

Claim: if we have the pre-condtion $x = 2y$, we have the post-condition $x = 2y$.
$(x = 2y)\{x \Leftarrow x + 2\}(x = 2y + 2)$
$(x = 2(y + 1))\{y \Leftarrow y + 1\}(x = 2y)$

$(x = 2y)\{x \Leftarrow x + 2; y \Leftarrow y + 1\}(x = 2y)$

This is called an *invariant condition* (or just an invariant)

Branches

If-then:

$(p \wedge B)\{S\}q$

$(p \wedge \neg B) \rightarrow q$

$p\{\text{if B then S}\}q$

Branches

If-then:
$(p \wedge B)\{S\}q$
$(p \wedge \neg B) \rightarrow q$

$p\{\text{if B then S}\}q$

If-then-else:
$(p \wedge B)\{S_1\}q$
$(p \wedge \neg B)\{S_2\}q$

$p\{\text{if B then } S_1 \text{ else } S_2\}q$

Branches

If-then:
$(p \wedge B)\{S\}q$
$(p \wedge \neg B) \rightarrow q$

$p\{\text{if B then S}\}q$

If-then-else:
$(p \wedge B)\{S_1\}q$
$(p \wedge \neg B)\{S_2\}q$

$p\{\text{if B then } S_1 \text{ else } S_2\}q$

Example:
Let $x = 7$, and consider code:
$\{\text{if } y < x \text{ then } y \Leftarrow x\}$
Show $y \geq 7$.

Branches

If-then:
$(p \wedge B)\{S\}q$
$(p \wedge \neg B) \rightarrow q$

$p\{\text{if B then S}\}q$

If-then-else:
$(p \wedge B)\{S_1\}q$
$(p \wedge \neg B)\{S_2\}q$

$p\{\text{if B then } S_1 \text{ else } S_2\}q$

Example:
Let $x = 7$, and consider code:
$\{\text{if } y < x \text{ then } y \Leftarrow x\}$
Show $y \geq 7$.

$(x = 7 \wedge y < x)\{y \Leftarrow x\}(y \geq 7)$

Branches

If-then:
$(p \wedge B)\{S\}q$
$(p \wedge \neg B) \rightarrow q$

$p\{\text{if B then S}\}q$

If-then-else:
$(p \wedge B)\{S_1\}q$
$(p \wedge \neg B)\{S_2\}q$

$p\{\text{if B then } S_1 \text{ else } S_2\}q$

Example:
Let $x = 7$, and consider code:
$\{\text{if } y < x \text{ then } y \Leftarrow x\}$
Show $y \geq 7$.

$(x = 7 \wedge y < x)\{y \Leftarrow x\}(y \geq 7)$
$(x = 7 \wedge y \geq x) \rightarrow (y \geq 7)$

Branches

If-then:
$(p \wedge B)\{S\}q$
$(p \wedge \neg B) \rightarrow q$

$p\{\text{if B then S}\}q$

If-then-else:
$(p \wedge B)\{S_1\}q$
$(p \wedge \neg B)\{S_2\}q$

$p\{\text{if B then } S_1 \text{ else } S_2\}q$

Example:
Let $x = 7$, and consider code:
$\{\text{if } y < x \text{ then } y \Leftarrow x\}$
Show $y \geq 7$.

$(x = 7 \wedge y < x)\{y \Leftarrow x\}(y \geq 7)$
$(x = 7 \wedge y \geq x) \rightarrow (y \geq 7)$
$(x = 7)\{\text{if } y < x \text{ then } y \Leftarrow x\}(y \geq 7)$

While loops

While loop:
$(p \wedge B)\{S\}p$

$$\frac{(p \wedge B)\{S\}p}{p\{\text{while } B \text{ do } S\}(p \wedge \neg B)}$$

While loops

While loop:
$(p \wedge B)\{S\}p$

$p\{\text{while } B \text{ do } S\}(p \wedge \neg B)$

We call p a loop invariant.

While loops

While loop:

$$\frac{(p \land B)\{S\}p}{p\{\text{while } B \text{ do } S\}(p \land \neg B)}$$

We call p a loop invariant.
Note that the correctness of this inference rule technically requires a mathematical induction.

Example: $n!$

$$\frac{(p \wedge B)\{S\}p}{p\{\text{while } B \text{ do } S\}(p \wedge \neg B)}$$

Example: $n!$

$$\frac{(p \wedge B)\{S\}p}{p\{\text{while } B \text{ do } S\}(p \wedge \neg B)}$$

Find a loop invariant for the following program, and prove the program computes $n!$

$i \Leftarrow 1;$
$f \Leftarrow 1;$
while $i < n$ do
  $i \Leftarrow i + 1$
  $f \Leftarrow f \cdot i$

Example: $n!$

$$\frac{(p \wedge B)\{S\}p}{p\{\text{while } B \text{ do } S\}(p \wedge \neg B)}$$

Find a loop invariant for the following program, and prove the program computes $n!$

$i \Leftarrow 1;$
$f \Leftarrow 1;$
while $i < n$ do
    $i \Leftarrow i + 1$
    $f \Leftarrow f \cdot i$

$p = (f = i! \wedge i \leq n)$ (Loop invariant)

Example: $n!$

$$\frac{(p \wedge B)\{S\}p}{p\{\text{while } B \text{ do } S\}(p \wedge \neg B)}$$

Find a loop invariant for the following program, and prove the program computes $n!$

$i \Leftarrow 1;$
$f \Leftarrow 1;$
while $i < n$ do
    $i \Leftarrow i + 1$
    $f \Leftarrow f \cdot i$

$p = (f = i! \wedge i \leq n)$ (Loop invariant)
$B = (i < n)$ (Branch condition)

Example: $n!$

$$\frac{(p \wedge B)\{S\}p}{p\{\text{while } B \text{ do } S\}(p \wedge \neg B)}$$

Find a loop invariant for the following program, and prove the program computes $n!$

$i \Leftarrow 1;$
$f \Leftarrow 1;$
while $i < n$ do
    $i \Leftarrow i + 1$
    $f \Leftarrow f \cdot i$

$p = (f = i! \wedge i \leq n)$ (Loop invariant)
$B = (i < n)$ (Branch condition)

$$(f = i! \wedge i \leq n) \quad \{\text{while } (i < n) \text{ do } S\} \quad (f = i! \wedge i \leq n) \wedge (i \geq n)$$

Example: $n!$

$$\frac{(p \wedge B)\{S\}p}{p\{\text{while } B \text{ do } S\}(p \wedge \neg B)}$$

Find a loop invariant for the following program, and prove the program computes $n!$

$i \Leftarrow 1;$
$f \Leftarrow 1;$
while $i < n$ do
    $i \Leftarrow i + 1$
    $f \Leftarrow f \cdot i$

$p = (f = i! \wedge i \leq n)$ (Loop invariant)
$B = (i < n)$ (Branch condition)

$$(f = i! \wedge i \leq n) \quad \{\text{while } (i < n) \text{ do } S\} \quad (f = i! \wedge i \leq n) \wedge (i \geq n)$$
$$\equiv (f = i! \wedge i = n)$$

Example: $n!$

$$\frac{(p \wedge B)\{S\}p}{p\{\text{while } B \text{ do } S\}(p \wedge \neg B)}$$

Find a loop invariant for the following program, and prove the program computes $n!$

$i \Leftarrow 1;$
$f \Leftarrow 1;$
while $i < n$ do
    $i \Leftarrow i + 1$
    $f \Leftarrow f \cdot i$

$p = (f = i! \wedge i \leq n)$ (Loop invariant)
$B = (i < n)$ (Branch condition)

$$(f = i! \wedge i \leq n) \quad \{\text{while } (i < n) \text{ do } S\} \quad (f = i! \wedge i \leq n) \wedge (i \geq n)$$
$$\equiv (f = i! \wedge i = n) \equiv f = n!$$

Example: $n!$

$$\frac{(p \wedge B)\{S\}p}{p\{\text{while } B \text{ do } S\}(p \wedge \neg B)}$$

Find a loop invariant for the following program, and prove the program computes $n!$

$i \Leftarrow 1;$
$f \Leftarrow 1;$
while $i < n$ do
    $i \Leftarrow i + 1$
    $f \Leftarrow f \cdot i$

$p = (f = i! \wedge i \leq n)$ (Loop invariant)
$B = (i < n)$ (Branch condition)

$$(f = i! \wedge i \leq n \wedge i < n) \quad \{i \Leftarrow i + 1; f \Leftarrow f \cdot i\} \quad (f = i! \wedge i \leq n)$$

$$(f = i! \wedge i \leq n) \quad \{\text{while } (i < n) \text{ do } S\} \quad (f = i! \wedge i \leq n) \wedge (i \geq n)$$

$$\equiv (f = i! \wedge i = n) \equiv f = n!$$

Example: $n!$

$$\frac{(p \wedge B)\{S\}p}{p\{\text{while } B \text{ do } S\}(p \wedge \neg B)}$$

Find a loop invariant for the following program, and prove the program computes $n!$

$i \Leftarrow 1;$
$f \Leftarrow 1;$
while $i < n$ do
    $i \Leftarrow i + 1$
    $f \Leftarrow f \cdot i$

$p = (f = i! \wedge i \leq n)$ (Loop invariant)
$B = (i < n)$ (Branch condition)

$(f = i! \wedge i \leq n \wedge i < n)$ $\{i \Leftarrow i + 1\}$ $(f = (i-1)! \wedge i \leq n)$

$(f = i! \wedge i \leq n \wedge i < n)$ $\{i \Leftarrow i + 1; f \Leftarrow f \cdot i\}$ $(f = i! \wedge i \leq n)$
$(f = i! \wedge i \leq n)$ $\{\text{while } (i < n) \text{ do } S\}$ $(f = i! \wedge i \leq n) \wedge (i \geq n)$
$\equiv (f = i! \wedge i = n) \equiv f = n!$

Example: $n!$

$$\frac{(p \wedge B)\{S\}p}{p\{\text{while } B \text{ do } S\}(p \wedge \neg B)}$$

Find a loop invariant for the following program, and prove the program computes $n!$

$i \Leftarrow 1;$
$f \Leftarrow 1;$
while $i < n$ do
    $i \Leftarrow i + 1$
    $f \Leftarrow f \cdot i$

$p = (f = i! \wedge i \leq n)$ (Loop invariant)
$B = (i < n)$ (Branch condition)

$$
\begin{array}{rll}
(f = i! \wedge i \leq n \wedge i < n) & \{i \Leftarrow i + 1\} & (f = (i-1)! \wedge i \leq n) \\
(f = (i-1)! \wedge i \leq n) & \{f \Leftarrow f \cdot i\} & (f = i! \wedge i \leq n) \\
(f = i! \wedge i \leq n \wedge i < n) & \{i \Leftarrow i + 1; f \Leftarrow f \cdot i\} & (f = i! \wedge i \leq n) \\
(f = i! \wedge i \leq n) & \{\text{while } (i < n) \text{ do } S\} & (f = i! \wedge i \leq n) \wedge (i \geq n) \\
 & & \equiv (f = i! \wedge i = n) \equiv f = n!
\end{array}
$$