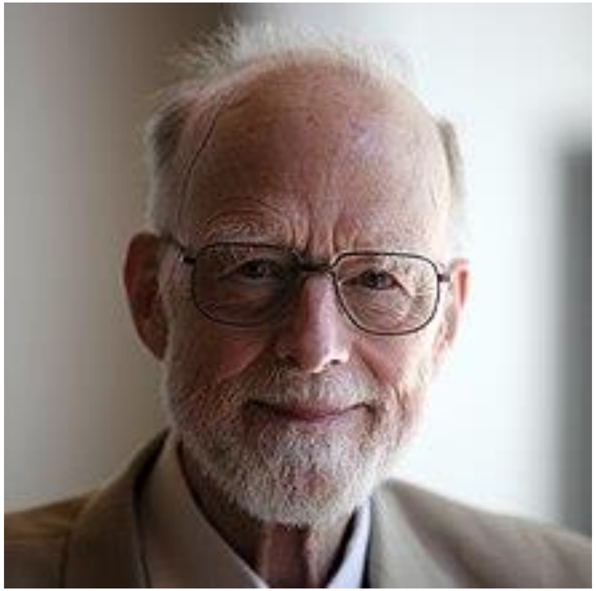
CMSC 433
Programming Language Technologies and Paradigms

Hoare Logic

Hoare logic

Hoare logic (also known as **Floyd–Hoare logic** is a formal system with a set of logical rules for reasoning about the correctness of computer programs.



It is a style of **Axiomatic Semantics.**

Hoare Triple

- The central feature of **Hoare logic** is the **Hoare triple**.

$$\{P\}\ S\ \{Q\}$$

- P is the precondition
- Q is the postcondition
- S is any statement

P and Q are *assertions* and S is a *command*

Hoare Triple Semantics

$$\{P\}\ S\ \{Q\}$$

- If statement S begins execution in a state satisfying assertion P,
- and if S eventually terminates in some final state,
- then that final state will satisfy the assertion Q.

Hoare Triple Examples $\{P\}\ S\ \{Q\}$

- `{x == 0} x := x + 1 {x == 1}`
- `{x = 5} x := x * 2 {x = 10}`
- `{i > j} {i := i+1; j := j+1} { i > j}`
- ```
  {0 <= x <= 15}
  if x < 15 then x:= x+1 else x:=0
  {0 <= x <= 15}
  ```
  ```

Strongest Postconditions

$\{P\}\ S\ \{Q\}$

If {P} S {Q} and for all Q' such that {P} S {Q'}, Q ⇒ Q', then Q is the strongest postcondition of S with respect to P

```
{x = 5} x := x * 2 {x = 10 || x = 5}
```

```
{x = 10} ⇒{x = 10 || x = 5}
{x = 10} is stronger than {x = 10 || x = 5}
```

```
{x = 5} x := x * 2 {x = 10}
```

Weakest Precondition $\{P\}\ S\ \{Q\}$

If {P} S {Q} and for all P' such that {P'} S {Q}, P' ⇒ P, then P is the weakest precondition wp(S,Q) of S with respect to Q

- `{x = 5 && y = 10}` `z := x / y { z < 1 }`
- `{x < y && y > 0}` `z := x / y { z < 1 }`
- `{y != 0 && x / y < 1}` `z := x / y { z < 1 }`

- All are true, but this one is the most useful because it allows us to invoke the program in the most general condition
- `y != 0 && x / y < 1` is the weakest precondition

Preconditioning Strengthening (Consequence)

$$\frac{P_s \Rightarrow P_w \quad \{P_w\}\ S\ \{Q\}}{\{P_s\}\ S\ \{Q\}}$$

`{x >= 0} x:= x+ 1 {x` $\geq$ `0}`

`x:=2` $\Rightarrow$ `x` $\geq$ `0`

`{x = 2} x:= x+1 {x` $\geq$ `0}`

Postcondition Weakening (Consequence)

$$\frac{\{P\}\ S\ \{Q_s\} \qquad Q_s \Rightarrow Q_w}{\{P_s\}\ S\ \{Q_w\}}$$

- **`{x ≥ 0} x:= x+1{x ≥ 0}`**
- **`x ≥ 0 => x ≥ -10`**
- **`{x ≥ 0} x:= x+1{x ≥ -10}`**

Postcondition is true, but less useful

Practice: More Hoare Triples

Consider the following Hoare triples, which of them are valid?

A. `{ z = y + 1 } x := z * 2 { x = 4 }`

B. `{ y = 7 } x := y + 3 { x > 5 }`

C. `{ false } x := 2 / y { true }`

D. `{ y < 16 } x := y / 2 { x < 8 }`

E. `{true} while true x :=x + 1; {false}`

Practice: More Hoare Triples

A. `{z = y + 1} x := z * 2 {x = 4}`   **Not valid**

B. `{ y = 7 } x := y + 3 { x > 5 }`

C. `{ false } x := 2 / y { true }`

D. `{ y < 16 } x := y / 2 { x < 8 }`

E. `{true} while true x :=x + 1; {false}`

Practice: More Hoare Triples

Consider the following Hoare triples. For which ones can you write a stronger postcondition? (Leave the precondition unchanged, and ensure the resulting triple is still valid)

A. `{ y = 7 } x := y + 3 { x > 5 }`

B. `{ false } x := 2 / y { true }`

C. `{ y < 16 } x := y / 2 { x < 8 }`

Practice: More Hoare Triples

Consider the following Hoare triples. For which ones can you write a stronger postcondition? (Leave the precondition unchanged, and ensure the resulting triple is still valid)

A. `{ y = 7 } x := y + 3 { x > 5 }`
B. `{ false } x := 2 / y { true }`
C. `{ y < 16 } x := y / 2 { x < 8 }`

A. `x = 10`
B. `false`

Practice: More Hoare Triples

Consider the following Hoare triples:

A. `{ y = 7 } x := y + 3 { x > 5 }`

B. `{ false } x := 2 / y { true }`

C. `{ y < 16 } x := y / 2 { x < 8 }`

For which ones can you write a weaker precondition? (Leave the postcondition unchanged, and ensure the resulting triple is still valid)

A. `y > 2`

B. `true`

Hoare Logic Rules

- Assignment:

$$\frac{}{\{\ Q[x := a]\ \}\ \ x := a\ \ \{\ Q\ \}}$$

$$\{\ \ ???\ \ \}\ \ x\ := x+1\ \ \{x\ \le\ N\}$$

Hoare Logic Rules: Assignment

$$\frac{}{\{Q[x := a]\}\ x := a\ \{Q\}}$$

$\{\ ???\ \}\quad x := x+1\ \{x \le N\}$

$\{x \le N[\mathbf{x}/\mathbf{x}+1]\} = \{x+1 \le N\}$

$\{x+1 \le N\}\quad x := x+1\ \{x \le N\}$

Assignment Example

- { P } x := 3 { x+y > 0 }
  - What is the weakest precondition P?
- Assignment rule : { P[e/x] } x := e { P }

Assignment Example

- $\{ P \}\ x := 3\ \{ x+y > 0 \}$
  - What is the weakest precondition P?
- Assignment rule : $\{ P[e/x] \}\ x := e\ \{ P \}$

$(x + y > 0)[3 / x]$

$= 3 + y > 0$

$= y > -3$

- $\{y > -3 \}\ x := 3\ \{ x+y > 0 \}$

Assignment Example

```
{ ??? } x := x + y { x == 1 }
```

If we replace the x in x == 1 with x + y, we get x + y == 1.
It leads to a valid Hoare triple:

```
{ x + y == 1 } x := x + y { x == 1 }
```

Hoare Logic Rules: Skip

Since empty statement blocks don't change the state, they preserve any assertion P:

$$\frac{}{\{\ \mathbf{P}\ \}\ \{\}\ \{\ \mathbf{P}\ \}}$$

Hoare Logic Rules: Sequence

$$\frac{\{\ P\ \}\ \texttt{s1}\ \{\ Q\ \} \qquad \{\ Q\ \}\ \texttt{s2}\ \{\ R\ \}}{\{\ P\ \}\ \texttt{s1; s2}\ \{\ R\ \}}$$

$\{x \geq 0\}$ `x := x + 3; x := 2 * x` $\{x \geq 6\}$

Hoare Logic Rules: Sequence

```
{x ≥ 0}
x := x+ 3;
{???}
x := 2 * x;
{x ≥ 6}
```

Hoare Logic Rules: Sequence

```
{x ≥ 0}

{??? }

x := x + 3;

{2 * x ≥ 6}

x := 2 * x;

{x ≥ 6}
```

Hoare Logic Rules: Sequence

$\{2 * (x+3) \geq 6\} \Rightarrow \{2x + 6 \geq 6\} \Rightarrow$
$\{x \geq 0\}$

```
x := x + 3;
```

$\{2 * x \geq 6\}$

```
x := 2 * x;
```

$\{x \geq 6\}$

Hoare Logic Rules: Conditionals

The same assertion Q holds after executing either of the branches.

$$\frac{\texttt{\{P \&\& b\} s1 \{Q\}} \qquad \texttt{\{P \&\& !b\} s2 \{Q\}}}{\texttt{\{P\} if b \{ s1 \} else \{ s2 \} \{Q\}}}$$

```
{ true }
if x ≤ 0
  { y := 2 }
else
  { y := x + 1 }
{ x ≤ y }
```

Hoare Logic Rules: Conditionals

The same assertion Q holds after executing either of the branches.

$$\frac{\texttt{\{P \&\& b\} s1 \{Q\}} \qquad \texttt{\{P \&\& !b\} s2 \{Q\}}}{\texttt{\{P\} if b \{ s1 \} else \{ s2 \} \{Q\}}}$$

```
{ true }
if x ≤ 0
  { y := 2 }
else
  { y := x + 1 }
{ x ≤ y }
```

```
{true && x ≤ 0} y := 2 {x ≤ y }

{true && !(x ≤ 0)} y := x+1 {x ≤ y }
```

Hoare Logic Rules: Conditionals

The same assertion Q holds after executing either of the branches.

```
{P && b} s1 {Q}     {P && !b} s2 {Q}
─────────────────────────────────────
  {P} if b { s1 } else { s2 } {Q}
```

```
{ true }
if x ≤ 0
  { y := 2 }
else
  { y := x + 1 }
{ x ≤ y }
```

```
{true && x ≤ 0} y := 2 {x ≤ y }
{true && !(x ≤ 0)} y := x+1 {x ≤ y }
```

```
(x ≤ 0 ⇒ y == 2) &&
(!(x ≤ 0) ⇒ y == x + 1)
⇒
x ≤ y
```

Practice: Preconditions/Postconditions

Fill in the missing pre- or post-conditions with predicates that make each Hoare triple valid.

A. { x = y } x := y * 2 {   }
B. {   } x := x + 3 { x = z }
C. {   } x := x + 1; y := y * x { y = 2 * z }
D. {   } if (x > 0) then y := x else y := 0 { y > 0 }

Practice: Preconditions/Postconditions

Fill in the missing pre- or post-conditions with predicates that make each Hoare triple valid.

A. { x = y } x := y * 2 { x = y * 2 }
B. { x+3 = z } x := x + 3 { x = z }
C. { y * (x+1) = 2 * z } x := x + 1; y := y * x { y = 2 * z }
D. { x > 0 } if (x > 0) then y := x else y := 0 { y > 0 }