# Hoare Logic 2

slides by Chris Osborn

## Hoare Triple

$P \{ \ldots code \ldots \} Q$

$$\frac{}{P[e/x] \ \{ x := e \} \ P}$$

$$\frac{P \{ C_1 \} R \quad R \{ C_2 \} Q}{P \ \{ C_1; C_2 \} \ Q}$$

$$\frac{P \wedge b \{ C_1 \} Q \quad P \wedge \neg b \{ C_2 \} Q}{P \ \{ \text{if } b \text{ then } C_1 \text{ else } C_2 \} \ Q}$$

## While Rule

$$\frac{P \wedge b \{ C \} P}{P \{\text{While } b \ C\} \ P \wedge \neg b}$$

(P is a **loop invariant**)

## Rule of Consequence

$$\frac{P \rightarrow P' \quad P' \{ C \} Q' \quad Q' \rightarrow Q}{P \ \{ C \} \ Q}$$

## Sample Proofs

- sum of n
- fibonacci
- list append
- list reverse
- termination

## Sum of n

x = 0 & y = 0 P ≡ x = 1 + ... + y ∧ y ≤ n

{

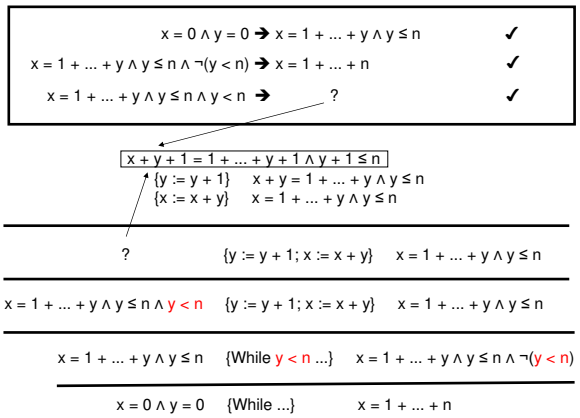
While y < n

y := y + 1;

x := x + y

}

x = 1 + ... + n




## Fibonacci

x = 0 & y = 1 & z = 1 & 1 ≤ n

{

While z < n P ≡ y = fib z ∧ x = fib (z-1) ∧ z ≤ n

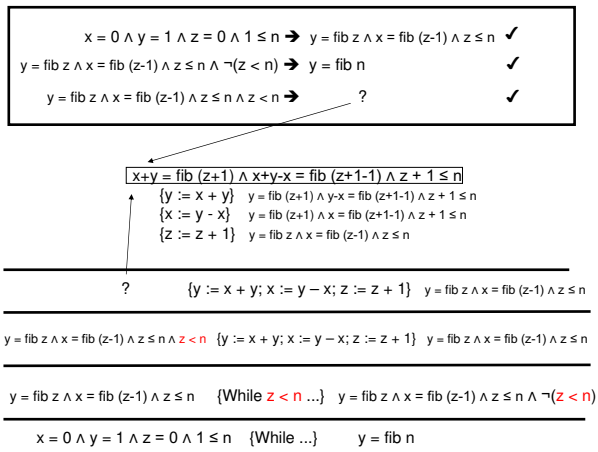
y := x + y;

x := y – x;

z := z + 1

}

y = fib n




## List length

x = lst & y = 0 P ≡ len lst = y + len x

{

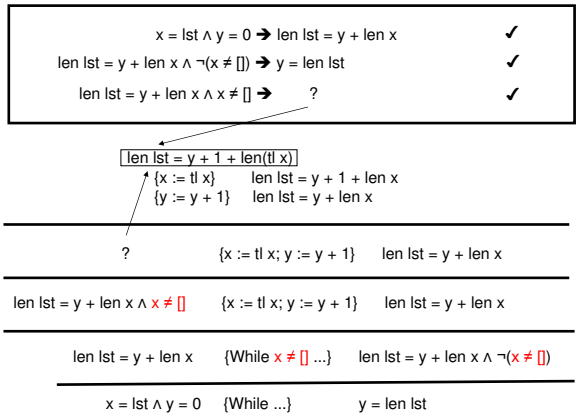
While x ≠ []

x := tl x;

y := y + 1

}

y = len lst



# List reverse

x = lst & y = []  P ≡ lst = rev y @ x

```
{
  While x ≠ []
    y := hd x :: y;
    x := tl x
}
```

y = rev lst