

Derivation & Formal Verification

•Related processes

- Program derivation
 - Derive program from formal spec
- Formal verification
 - Prove correctness of existing code

•Common approach

- Axiomatic method
 - Program state
 - Pre/post conditions for code segments

Hoare Logic

•Also known as Floyd-Hoare logic

- R. W. Floyd (1967)
- C. A. R. (Tony) Hoare (1969)

•Basic element – “Hoare triple”

- $\{ P \} S \{ Q \}$
 - P = operation precondition
 - S = code block
 - Q = operation postcondition

Hoare Logic – Basic Constructs

•Assignment

- Assign value to a variable (data object)

•Sequence

- Execute two code blocks in succession

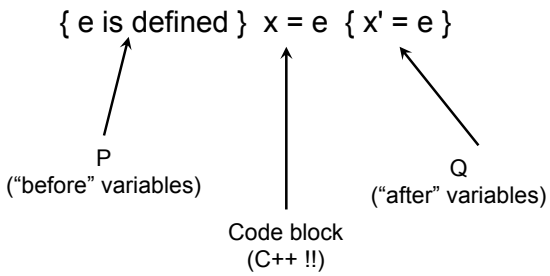
•Selection

- Execute one of two code blocks

•Iteration

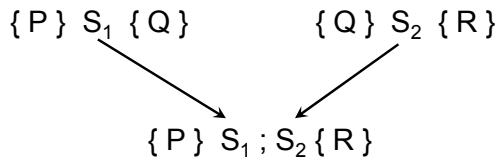
- Execute code block repeatedly

Assignment



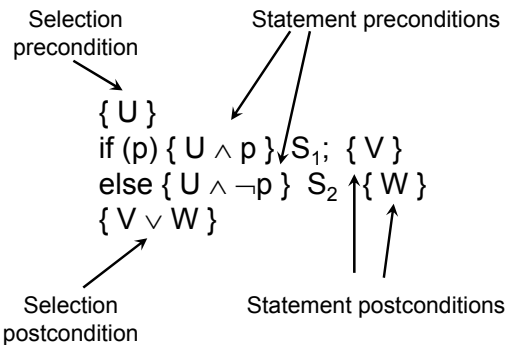
$\{ Q[e/x'] \} x = e \{ Q \}$ ← Goal-oriented form

Sequence



Postcondition of S_1 must be equivalent to
(or at least imply) precondition of S_2

Selection (Textbook Form)



Iteration (Terminating)

{ I }

while (p) { I ∧ p } S { I' ∧ v' < v } ;

{ I ∧ ¬p }

Progress measure (variant)
moves toward limit on each
iteration ($v < k \Rightarrow \neg p$)

Iteration Correctness Conditions

•Initialization

- Invariant is true before loop entry

•Invariance

- Body preserves the invariant

•Progress

- Body of loop decreases the variant

•Boundedness

- Variant below limit implies false guard

•Exit

- Invariant and negated guard imply postcondition

Program Derivation

•Start with

- Given precondition
- Desired postcondition

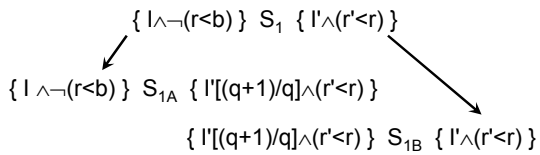
•Choose code blocks

- To achieve desired goal (postcondition)

•How to choose?

- Guessing is good!
 - Based on experience and definition of axioms for basic constructs

Example 2 – Division (2)



$\{ \neg(r < b) \}$ $r=r-b$; $\{ \neg((q+1)/q) \wedge (r' < r) \}$ $\{ \neg((q+1)/q) \wedge (r' < r) \}$ $q=q+1$; $\{ \neg(r' < r) \}$

Example 2 – Division (3)

$a, b, q, r : \mathbb{Z}$ $(a \geq 0) \wedge (b > 0)$ $a = b * q + r$ $r < b$	a is dividend b is divisor q is quotient r is remainder // a and b are set q=0; r=a; while (!(r<b)) { r = r - b; q = q + 1; }
--	---

A Minor Question

$a, b, q, r : \mathbb{Z}$ $(a \geq 0) \wedge (b > 0)$ $a = b * q + r$ $r < b$	a is dividend b is divisor q is quotient r is remainder
--	--

$q=0; r=a;$ $\{ \neg((a \geq 0) \wedge (b > 0) \wedge (a = b * q + r)) \}$ $a=1; b=1; q=1; r=0;$ $\{ \neg(r < b) \Leftrightarrow (a \geq 0) \wedge (b > 0) \wedge (a = b * q + r) \wedge (r < b) \}$

Would this program satisfy the postcondition?
