

Liam O'Connor

CSE, UNSW (and data61)

Semester 2 2018

# Imperative Programming

imperō

## Definition

*Imperative programming* is where programs are described as a series of *statements* or commands to manipulate mutable *state* or cause externally observable *effects*.

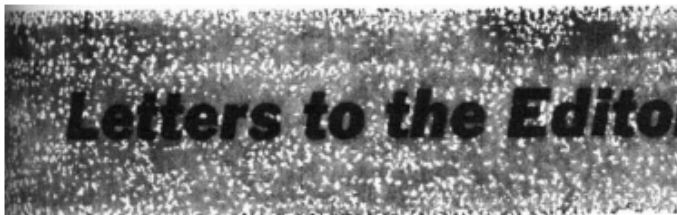
*States* may take the form of a *mapping* from variable names to their values, or even a model of a CPU state with a memory model (for example, in an *assembly language*).

## The Old Days



Early microcomputer languages used a **line numbering** system with **GO TO** statements used to arrange control flow.

# Dijkstra



## Go To Statement Considered Harmful

Key Words and Phrases: go to statement, jump instruction, branch instruction, conditional clause, alternative clause, repetitive clause, program intelligibility, program sequencing

CR Categories: 4.22. 5.23. 5.24

dyna  
call  
we c  
text  
dyna  
\_

The *structured programming* movement brought in *control structures* to mainstream use, such as conditionals and loops.

# Syntax

We're going to specify a language **TinyImp**, based on **structured programming**. The syntax consists of **statements** and **expressions**.

## Grammar

<b>Stmt</b>	<b>::=</b>	skip	<i>Do nothing</i>
		x := <b>Expr</b>	<i>Assignment</i>
		var y · <b>Stmt</b>	<i>Declaration</i>
		if <b>Expr</b> then <b>Stmt</b> else <b>Stmt</b> fi	<i>Conditional</i>
		while <b>Expr</b> do <b>Stmt</b> od	<i>Loop</i>
		<b>Stmt</b> ; <b>Stmt</b>	<i>Sequencing</i>
<b>Expr</b>	<b>::=</b>	⟨ <i>Arithmetic expressions</i> ⟩	

We already know how to make unambiguous **abstract syntax**, so we will use **concrete syntax** in the rules for readability.

# Examples

## Example (Factorial and Fibonacci)

```
var  $i$  ·  
var  $m$  ·  
 $i := 0$ ;  
 $m := 1$ ;  
while  $i < N$  do  
   $i := i + 1$ ;  
   $m := m \times i$   
od
```

```
var  $m$  · var  $n$  · var  $i$  ·  
 $m := 1$ ;  $n := 1$ ;  
 $i := 1$ ;  
while  $i < N$  do  
  var  $t$  ·  $t := m$ ;  
   $m := n$ ;  
   $n := m + t$ ;  
   $i := i + 1$   
od
```

# Static Semantics

**Types?** We only have one type (`int`), so type checking is a wash.

**Scopes?** We have to check that variables are declared before use.

**Anything Else?** We have to check that variables are *initialized* before they are used!



## Static Semantics

**Types?** We only have one type (`int`), so type checking is a wash.

**Scopes?** We have to check that variables are declared before use.

**Anything Else?** We have to check that variables are *initialized* before they are used!

indicates that no uninitialized reads occur

$$U; V \vdash s \text{ ok} \rightsquigarrow W$$

U: set of initially uninitialized variables



## Static Semantics

**Types?** We only have one type (`int`), so type checking is a wash.

**Scopes?** We have to check that variables are declared before use.

**Anything Else?** We have to check that variables are *initialized* before they are used!

Indicates that no errors occur

$U; V \vdash s \text{ ok} \rightsquigarrow W$

## Static Semantics

**Types?** We only have one type (`int`), so type checking is a wash.

**Scopes?** We have to check that variables are declared before use.

**Anything Else?** We have to check that variables are *initialized* before they are used!

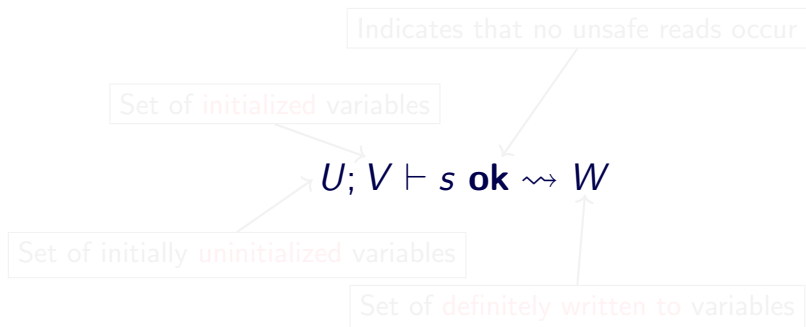
$$U; V \vdash s \text{ ok} \rightsquigarrow W$$

## Static Semantics

**Types?** We only have one type (`int`), so type checking is a wash.

**Scopes?** We have to check that variables are declared before use.

**Anything Else?** We have to check that variables are *initialized* before they are used!

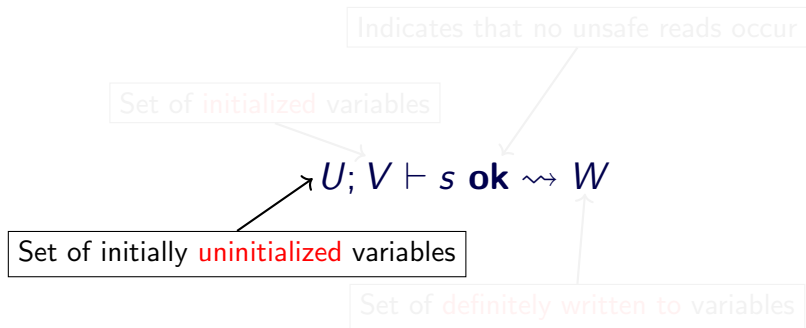


## Static Semantics

**Types?** We only have one type (`int`), so type checking is a wash.

**Scopes?** We have to check that variables are declared before use.

**Anything Else?** We have to check that variables are *initialized* before they are used!

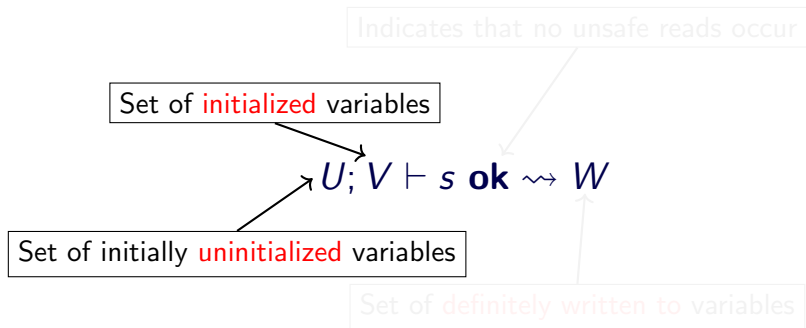


## Static Semantics

**Types?** We only have one type (`int`), so type checking is a wash.

**Scopes?** We have to check that variables are declared before use.

**Anything Else?** We have to check that variables are *initialized* before they are used!

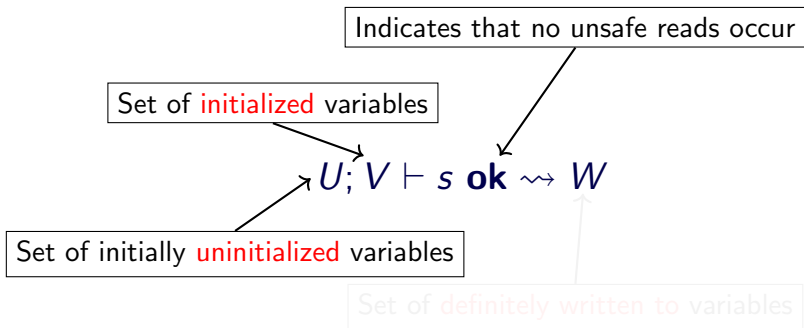


## Static Semantics

**Types?** We only have one type (`int`), so type checking is a wash.

**Scopes?** We have to check that variables are declared before use.

**Anything Else?** We have to check that variables are *initialized* before they are used!

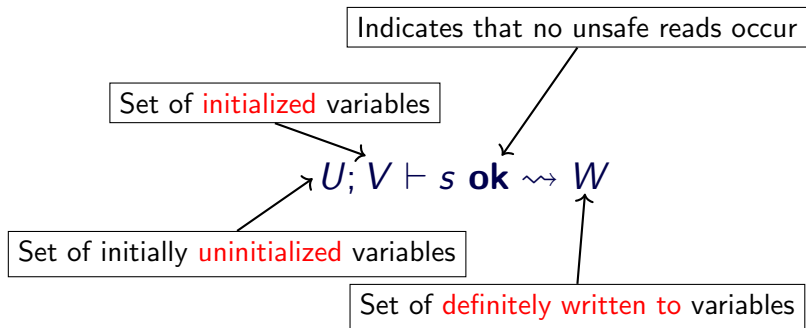


## Static Semantics

**Types?** We only have one type (`int`), so type checking is a wash.

**Scopes?** We have to check that variables are declared before use.

**Anything Else?** We have to check that variables are *initialized* before they are used!



# Static Semantics Rules

---

$$U; V \vdash \text{skip } \mathbf{ok} \rightsquigarrow \emptyset$$

---

$$U; V \vdash x := e \mathbf{ok} \rightsquigarrow$$



# Static Semantics Rules

$$\frac{}{U; V \vdash \text{skip } \mathbf{ok} \rightsquigarrow \emptyset} \quad \frac{x \in U \cup V \quad \text{FV}(e) \cap U = \emptyset}{U; V \vdash x := e \mathbf{ok} \rightsquigarrow \{x\}}$$

$$\frac{}{U; V \vdash \text{var } y \cdot s \mathbf{ok} \rightsquigarrow \{y\}}$$

# Static Semantics Rules

$$\frac{}{U; V \vdash \text{skip } \mathbf{ok} \rightsquigarrow \emptyset} \quad \frac{x \in U \cup V \quad \text{FV}(e) \cap U = \emptyset}{U; V \vdash x := e \mathbf{ok} \rightsquigarrow \{x\}}$$

$$\frac{}{U; V \vdash \text{var } y \cdot s \mathbf{ok} \rightsquigarrow \{y\}}$$

# Static Semantics Rules

$$\frac{}{U; V \vdash \text{skip } \mathbf{ok} \rightsquigarrow \emptyset} \quad \frac{x \in U \cup V \quad \text{FV}(e) \cap U = \emptyset}{U; V \vdash x := e \mathbf{ok} \rightsquigarrow \{x\}}$$

$$\frac{}{U; V \vdash \text{var } y \cdot s \mathbf{ok} \rightsquigarrow \{y\}}$$

# Static Semantics Rules

$$\frac{}{U; V \vdash \text{skip } \mathbf{ok} \rightsquigarrow \emptyset} \quad \frac{x \in U \cup V \quad \text{FV}(e) \cap U = \emptyset}{U; V \vdash x := e \mathbf{ok} \rightsquigarrow \{x\}}$$

$$\frac{}{U; V \vdash \text{var } y \cdot s \mathbf{ok} \rightsquigarrow \{y\}}$$

# Static Semantics Rules

$$\frac{}{U; V \vdash \text{skip } \mathbf{ok} \rightsquigarrow \emptyset} \quad \frac{x \in U \cup V \quad \text{FV}(e) \cap U = \emptyset}{U; V \vdash x := e \mathbf{ok} \rightsquigarrow \{x\}}$$

$$\frac{U \cup \{y\}; V \vdash s \mathbf{ok} \rightsquigarrow W}{U; V \vdash \text{var } y \cdot s \mathbf{ok} \rightsquigarrow W \setminus \{y\}}$$

---

$$U; V \vdash \text{if } e \text{ then } s_1 \text{ else } s_2 \text{ fi } \mathbf{ok} \rightsquigarrow$$

# Static Semantics Rules

$$\frac{}{U; V \vdash \text{skip } \mathbf{ok} \rightsquigarrow \emptyset} \quad \frac{x \in U \cup V \quad \text{FV}(e) \cap U = \emptyset}{U; V \vdash x := e \mathbf{ok} \rightsquigarrow \{x\}}$$
$$\frac{U \cup \{y\}; V \vdash s \mathbf{ok} \rightsquigarrow W}{U; V \vdash \text{var } y \cdot s \mathbf{ok} \rightsquigarrow W \setminus \{y\}}$$

---

$U; V \vdash \text{if } e \text{ then } s_1 \text{ else } s_2 \text{ fi } \mathbf{ok} \rightsquigarrow$

# Static Semantics Rules

$$\frac{}{U; V \vdash \text{skip } \mathbf{ok} \rightsquigarrow \emptyset} \quad \frac{x \in U \cup V \quad \text{FV}(e) \cap U = \emptyset}{U; V \vdash x := e \mathbf{ok} \rightsquigarrow \{x\}}$$
$$\frac{U \cup \{y\}; V \vdash s \mathbf{ok} \rightsquigarrow W}{U; V \vdash \text{var } y \cdot s \mathbf{ok} \rightsquigarrow W \setminus \{y\}}$$

---

$U; V \vdash \text{if } e \text{ then } s_1 \text{ else } s_2 \text{ fi } \mathbf{ok} \rightsquigarrow$

## Static Semantics Rules

$$\frac{}{U; V \vdash \text{skip } \mathbf{ok} \rightsquigarrow \emptyset} \quad \frac{x \in U \cup V \quad \text{FV}(e) \cap U = \emptyset}{U; V \vdash x := e \mathbf{ok} \rightsquigarrow \{x\}}$$

$$\frac{U \cup \{y\}; V \vdash s \mathbf{ok} \rightsquigarrow W}{U; V \vdash \text{var } y \cdot s \mathbf{ok} \rightsquigarrow W \setminus \{y\}}$$

$$\frac{\text{FV}(e) \subseteq V \quad U; V \vdash s_1 \mathbf{ok} \rightsquigarrow W_1 \quad U; V \vdash s_2 \mathbf{ok} \rightsquigarrow W_2}{U; V \vdash \text{if } e \text{ then } s_1 \text{ else } s_2 \text{ fi } \mathbf{ok} \rightsquigarrow W_1 \cap W_2}$$

$$\frac{}{U; V \vdash \text{while } e \text{ do } s \text{ od } \mathbf{ok} \rightsquigarrow \dots}$$



# Static Semantics Rules

$$\frac{}{U; V \vdash \text{skip } \mathbf{ok} \rightsquigarrow \emptyset} \quad \frac{x \in U \cup V \quad \text{FV}(e) \cap U = \emptyset}{U; V \vdash x := e \mathbf{ok} \rightsquigarrow \{x\}}$$

$$\frac{U \cup \{y\}; V \vdash s \mathbf{ok} \rightsquigarrow W}{U; V \vdash \text{var } y \cdot s \mathbf{ok} \rightsquigarrow W \setminus \{y\}}$$

$$\frac{\text{FV}(e) \subseteq V \quad U; V \vdash s_1 \mathbf{ok} \rightsquigarrow W_1 \quad U; V \vdash s_2 \mathbf{ok} \rightsquigarrow W_2}{U; V \vdash \text{if } e \text{ then } s_1 \text{ else } s_2 \text{ fi } \mathbf{ok} \rightsquigarrow W_1 \cap W_2}$$

$$\frac{}{U; V \vdash \text{while } e \text{ do } s \text{ od } \mathbf{ok} \rightsquigarrow \dots}$$

## Static Semantics Rules

$$\frac{}{U; V \vdash \text{skip } \mathbf{ok} \rightsquigarrow \emptyset} \quad \frac{x \in U \cup V \quad \text{FV}(e) \cap U = \emptyset}{U; V \vdash x := e \mathbf{ok} \rightsquigarrow \{x\}}$$
$$\frac{U \cup \{y\}; V \vdash s \mathbf{ok} \rightsquigarrow W}{U; V \vdash \text{var } y \cdot s \mathbf{ok} \rightsquigarrow W \setminus \{y\}}$$
$$\frac{\text{FV}(e) \subseteq V \quad U; V \vdash s_1 \mathbf{ok} \rightsquigarrow W_1 \quad U; V \vdash s_2 \mathbf{ok} \rightsquigarrow W_2}{U; V \vdash \text{if } e \text{ then } s_1 \text{ else } s_2 \text{ fi } \mathbf{ok} \rightsquigarrow W_1 \cap W_2}$$
$$\frac{}{U; V \vdash \text{while } e \text{ do } s \text{ od } \mathbf{ok} \rightsquigarrow \dots}$$

## Static Semantics Rules

$$\begin{array}{c}
 \frac{}{U; V \vdash \text{skip } \mathbf{ok} \rightsquigarrow \emptyset} \quad \frac{x \in U \cup V \quad \text{FV}(e) \cap U = \emptyset}{U; V \vdash x := e \mathbf{ok} \rightsquigarrow \{x\}} \\
 \\
 \frac{U \cup \{y\}; V \vdash s \mathbf{ok} \rightsquigarrow W}{U; V \vdash \text{var } y \cdot s \mathbf{ok} \rightsquigarrow W \setminus \{y\}} \\
 \\
 \frac{\text{FV}(e) \subseteq V \quad U; V \vdash s_1 \mathbf{ok} \rightsquigarrow W_1 \quad U; V \vdash s_2 \mathbf{ok} \rightsquigarrow W_2}{U; V \vdash \text{if } e \text{ then } s_1 \text{ else } s_2 \text{ fi } \mathbf{ok} \rightsquigarrow W_1 \cap W_2} \\
 \\
 \frac{}{U; V \vdash \text{while } e \text{ do } s \text{ od } \mathbf{ok} \rightsquigarrow \dots}
 \end{array}$$

## Static Semantics Rules

$$\frac{}{U; V \vdash \text{skip } \mathbf{ok} \rightsquigarrow \emptyset} \quad \frac{x \in U \cup V \quad \text{FV}(e) \cap U = \emptyset}{U; V \vdash x := e \mathbf{ok} \rightsquigarrow \{x\}}$$
$$\frac{U \cup \{y\}; V \vdash s \mathbf{ok} \rightsquigarrow W}{U; V \vdash \text{var } y \cdot s \mathbf{ok} \rightsquigarrow W \setminus \{y\}}$$
$$\frac{\text{FV}(e) \subseteq V \quad U; V \vdash s_1 \mathbf{ok} \rightsquigarrow W_1 \quad U; V \vdash s_2 \mathbf{ok} \rightsquigarrow W_2}{U; V \vdash \text{if } e \text{ then } s_1 \text{ else } s_2 \text{ fi } \mathbf{ok} \rightsquigarrow W_1 \cap W_2}$$

$$U; V \vdash \text{while } e \text{ do } s \text{ od } \mathbf{ok} \rightsquigarrow$$

## Static Semantics Rules

$$\begin{array}{c}
 \frac{}{U; V \vdash \text{skip } \mathbf{ok} \rightsquigarrow \emptyset} \quad \frac{x \in U \cup V \quad \text{FV}(e) \cap U = \emptyset}{U; V \vdash x := e \mathbf{ok} \rightsquigarrow \{x\}} \\
 \\
 \frac{U \cup \{y\}; V \vdash s \mathbf{ok} \rightsquigarrow W}{U; V \vdash \text{var } y \cdot s \mathbf{ok} \rightsquigarrow W \setminus \{y\}} \\
 \\
 \frac{\text{FV}(e) \subseteq V \quad U; V \vdash s_1 \mathbf{ok} \rightsquigarrow W_1 \quad U; V \vdash s_2 \mathbf{ok} \rightsquigarrow W_2}{U; V \vdash \text{if } e \text{ then } s_1 \text{ else } s_2 \text{ fi } \mathbf{ok} \rightsquigarrow W_1 \cap W_2} \\
 \\
 \frac{\text{FV}(e) \subseteq V \quad U; V \vdash s \mathbf{ok} \rightsquigarrow W}{U; V \vdash \text{while } e \text{ do } s \text{ od } \mathbf{ok} \rightsquigarrow \emptyset} \\
 \\
 \hline
 U; V \vdash s_1; s_2 \mathbf{ok} \rightsquigarrow
 \end{array}$$

## Static Semantics Rules

$$\begin{array}{c}
 \frac{}{U; V \vdash \text{skip } \mathbf{ok} \rightsquigarrow \emptyset} \quad \frac{x \in U \cup V \quad \text{FV}(e) \cap U = \emptyset}{U; V \vdash x := e \mathbf{ok} \rightsquigarrow \{x\}} \\
 \\
 \frac{U \cup \{y\}; V \vdash s \mathbf{ok} \rightsquigarrow W}{U; V \vdash \text{var } y \cdot s \mathbf{ok} \rightsquigarrow W \setminus \{y\}} \\
 \\
 \frac{\text{FV}(e) \subseteq V \quad U; V \vdash s_1 \mathbf{ok} \rightsquigarrow W_1 \quad U; V \vdash s_2 \mathbf{ok} \rightsquigarrow W_2}{U; V \vdash \text{if } e \text{ then } s_1 \text{ else } s_2 \text{ fi } \mathbf{ok} \rightsquigarrow W_1 \cap W_2} \\
 \\
 \frac{\text{FV}(e) \subseteq V \quad U; V \vdash s \mathbf{ok} \rightsquigarrow W}{U; V \vdash \text{while } e \text{ do } s \text{ od } \mathbf{ok} \rightsquigarrow \emptyset} \\
 \\
 \hline
 U; V \vdash s_1; s_2 \mathbf{ok} \rightsquigarrow
 \end{array}$$

## Static Semantics Rules

$$\frac{}{U; V \vdash \text{skip } \mathbf{ok} \rightsquigarrow \emptyset} \quad \frac{x \in U \cup V \quad \text{FV}(e) \cap U = \emptyset}{U; V \vdash x := e \mathbf{ok} \rightsquigarrow \{x\}}$$

$$\frac{U \cup \{y\}; V \vdash s \mathbf{ok} \rightsquigarrow W}{U; V \vdash \text{var } y \cdot s \mathbf{ok} \rightsquigarrow W \setminus \{y\}}$$

$$\frac{\text{FV}(e) \subseteq V \quad U; V \vdash s_1 \mathbf{ok} \rightsquigarrow W_1 \quad U; V \vdash s_2 \mathbf{ok} \rightsquigarrow W_2}{U; V \vdash \text{if } e \text{ then } s_1 \text{ else } s_2 \text{ fi } \mathbf{ok} \rightsquigarrow W_1 \cap W_2}$$

$$\frac{\text{FV}(e) \subseteq V \quad U; V \vdash s \mathbf{ok} \rightsquigarrow W}{U; V \vdash \text{while } e \text{ do } s \text{ od } \mathbf{ok} \rightsquigarrow \emptyset}$$

---


$$U; V \vdash s_1; s_2 \mathbf{ok} \rightsquigarrow$$

## Static Semantics Rules

$$\frac{}{U; V \vdash \text{skip } \mathbf{ok} \rightsquigarrow \emptyset} \quad \frac{x \in U \cup V \quad \text{FV}(e) \cap U = \emptyset}{U; V \vdash x := e \mathbf{ok} \rightsquigarrow \{x\}}$$

$$\frac{U \cup \{y\}; V \vdash s \mathbf{ok} \rightsquigarrow W}{U; V \vdash \text{var } y \cdot s \mathbf{ok} \rightsquigarrow W \setminus \{y\}}$$

$$\frac{\text{FV}(e) \subseteq V \quad U; V \vdash s_1 \mathbf{ok} \rightsquigarrow W_1 \quad U; V \vdash s_2 \mathbf{ok} \rightsquigarrow W_2}{U; V \vdash \text{if } e \text{ then } s_1 \text{ else } s_2 \text{ fi } \mathbf{ok} \rightsquigarrow W_1 \cap W_2}$$

$$\frac{\text{FV}(e) \subseteq V \quad U; V \vdash s \mathbf{ok} \rightsquigarrow W}{U; V \vdash \text{while } e \text{ do } s \text{ od } \mathbf{ok} \rightsquigarrow \emptyset}$$

---


$$U; V \vdash s_1; s_2 \mathbf{ok} \rightsquigarrow$$



## Static Semantics Rules

$$\begin{array}{c}
 \frac{}{U; V \vdash \text{skip } \mathbf{ok} \rightsquigarrow \emptyset} \quad \frac{x \in U \cup V \quad \text{FV}(e) \cap U = \emptyset}{U; V \vdash x := e \mathbf{ok} \rightsquigarrow \{x\}} \\
 \\
 \frac{U \cup \{y\}; V \vdash s \mathbf{ok} \rightsquigarrow W}{U; V \vdash \text{var } y \cdot s \mathbf{ok} \rightsquigarrow W \setminus \{y\}} \\
 \\
 \frac{\text{FV}(e) \subseteq V \quad U; V \vdash s_1 \mathbf{ok} \rightsquigarrow W_1 \quad U; V \vdash s_2 \mathbf{ok} \rightsquigarrow W_2}{U; V \vdash \text{if } e \text{ then } s_1 \text{ else } s_2 \text{ fi } \mathbf{ok} \rightsquigarrow W_1 \cap W_2} \\
 \\
 \frac{\text{FV}(e) \subseteq V \quad U; V \vdash s \mathbf{ok} \rightsquigarrow W}{U; V \vdash \text{while } e \text{ do } s \text{ od } \mathbf{ok} \rightsquigarrow \emptyset} \\
 \\
 \frac{U; V \vdash s_1 \mathbf{ok} \rightsquigarrow W_1 \quad (U \setminus W_1); (V \cup W_1) \vdash s_2 \mathbf{ok} \rightsquigarrow W_2}{U; V \vdash s_1; s_2 \mathbf{ok} \rightsquigarrow W_1 \cup W_2}
 \end{array}$$

## Static Semantics Rules

$$\begin{array}{c}
 \frac{}{U; V \vdash \text{skip } \mathbf{ok} \rightsquigarrow \emptyset} \quad \frac{x \in U \cup V \quad \text{FV}(e) \cap U = \emptyset}{U; V \vdash x := e \mathbf{ok} \rightsquigarrow \{x\}} \\
 \\
 \frac{U \cup \{y\}; V \vdash s \mathbf{ok} \rightsquigarrow W}{U; V \vdash \text{var } y \cdot s \mathbf{ok} \rightsquigarrow W \setminus \{y\}} \\
 \\
 \frac{\text{FV}(e) \subseteq V \quad U; V \vdash s_1 \mathbf{ok} \rightsquigarrow W_1 \quad U; V \vdash s_2 \mathbf{ok} \rightsquigarrow W_2}{U; V \vdash \text{if } e \text{ then } s_1 \text{ else } s_2 \text{ fi } \mathbf{ok} \rightsquigarrow W_1 \cap W_2} \\
 \\
 \frac{\text{FV}(e) \subseteq V \quad U; V \vdash s \mathbf{ok} \rightsquigarrow W}{U; V \vdash \text{while } e \text{ do } s \text{ od } \mathbf{ok} \rightsquigarrow \emptyset} \\
 \\
 \frac{U; V \vdash s_1 \mathbf{ok} \rightsquigarrow W_1 \quad (U \setminus W_1); (V \cup W_1) \vdash s_2 \mathbf{ok} \rightsquigarrow W_2}{U; V \vdash s_1; s_2 \mathbf{ok} \rightsquigarrow W_1 \cup W_2}
 \end{array}$$

## Static Semantics Rules

$$\begin{array}{c}
 \frac{}{U; V \vdash \text{skip } \mathbf{ok} \rightsquigarrow \emptyset} \quad \frac{x \in U \cup V \quad \text{FV}(e) \cap U = \emptyset}{U; V \vdash x := e \mathbf{ok} \rightsquigarrow \{x\}} \\
 \frac{U \cup \{y\}; V \vdash s \mathbf{ok} \rightsquigarrow W}{U; V \vdash \text{var } y \cdot s \mathbf{ok} \rightsquigarrow W \setminus \{y\}} \\
 \frac{\text{FV}(e) \subseteq V \quad U; V \vdash s_1 \mathbf{ok} \rightsquigarrow W_1 \quad U; V \vdash s_2 \mathbf{ok} \rightsquigarrow W_2}{U; V \vdash \text{if } e \text{ then } s_1 \text{ else } s_2 \text{ fi } \mathbf{ok} \rightsquigarrow W_1 \cap W_2} \\
 \frac{\text{FV}(e) \subseteq V \quad U; V \vdash s \mathbf{ok} \rightsquigarrow W}{U; V \vdash \text{while } e \text{ do } s \text{ od } \mathbf{ok} \rightsquigarrow \emptyset} \\
 \frac{U; V \vdash s_1 \mathbf{ok} \rightsquigarrow W_1 \quad (U \setminus W_1); (V \cup W_1) \vdash s_2 \mathbf{ok} \rightsquigarrow W_2}{U; V \vdash s_1; s_2 \mathbf{ok} \rightsquigarrow W_1 \cup W_2}
 \end{array}$$

## Static Semantics Rules

$$\begin{array}{c}
 \frac{}{U; V \vdash \text{skip } \mathbf{ok} \rightsquigarrow \emptyset} \quad \frac{x \in U \cup V \quad \text{FV}(e) \cap U = \emptyset}{U; V \vdash x := e \mathbf{ok} \rightsquigarrow \{x\}} \\
 \\
 \frac{U \cup \{y\}; V \vdash s \mathbf{ok} \rightsquigarrow W}{U; V \vdash \text{var } y \cdot s \mathbf{ok} \rightsquigarrow W \setminus \{y\}} \\
 \\
 \frac{\text{FV}(e) \subseteq V \quad U; V \vdash s_1 \mathbf{ok} \rightsquigarrow W_1 \quad U; V \vdash s_2 \mathbf{ok} \rightsquigarrow W_2}{U; V \vdash \text{if } e \text{ then } s_1 \text{ else } s_2 \text{ fi } \mathbf{ok} \rightsquigarrow W_1 \cap W_2} \\
 \\
 \frac{\text{FV}(e) \subseteq V \quad U; V \vdash s \mathbf{ok} \rightsquigarrow W}{U; V \vdash \text{while } e \text{ do } s \text{ od } \mathbf{ok} \rightsquigarrow \emptyset} \\
 \\
 \frac{U; V \vdash s_1 \mathbf{ok} \rightsquigarrow W_1 \quad (U \setminus W_1); (V \cup W_1) \vdash s_2 \mathbf{ok} \rightsquigarrow W_2}{U; V \vdash s_1; s_2 \mathbf{ok} \rightsquigarrow W_1 \cup W_2}
 \end{array}$$

# Dynamic Semantics

We will use **big-step** operational semantics. **What are the sets of evaluable expressions and values here?**

**Evaluable Expressions:** A pair containing a **statement** to execute and a **state**  $\sigma$ .

**Values:** The final **state** that results from executing the statement.

## States

A **state** is a mutable mapping from variables to their values. We use the following notation:

- To **read** a variable  $x$  from the state  $\sigma$ , we write  $\sigma(x)$ .
- To **update** an existing variable  $x$  to have value  $v$  inside the state  $\sigma$ , we write  $(\sigma : x \mapsto v)$ .
- To **extend** a state  $\sigma$  with a new, previously undeclared variable  $x$ , we write  $\sigma \cdot x$ . In such a state,  $x$  has undefined value.

# Dynamic Semantics

We will use **big-step** operational semantics. **What are the sets of evaluable expressions and values here?**

**Evaluable Expressions:** A pair containing a **statement** to execute and a **state**  $\sigma$ .

**Values:** The final **state** that results from executing the statement.

## States

A **state** is a mutable mapping from variables to their values. We use the following notation:

- To **read** a variable  $x$  from the state  $\sigma$ , we write  $\sigma(x)$ .
- To **update** an existing variable  $x$  to have value  $v$  inside the state  $\sigma$ , we write  $(\sigma : x \mapsto v)$ .
- To **extend** a state  $\sigma$  with a new, previously undeclared variable  $x$ , we write  $\sigma \cdot x$ . In such a state,  $x$  has undefined value.

# Dynamic Semantics

We will use **big-step** operational semantics. **What are the sets of evaluable expressions and values here?**

**Evaluable Expressions:** A pair containing a **statement** to execute and a **state**  $\sigma$ .

**Values:** The final **state** that results from executing the statement.

## States

A **state** is a mutable mapping from variables to their values. We use the following notation:

- To **read** a variable  $x$  from the state  $\sigma$ , we write  $\sigma(x)$ .
- To **update** an existing variable  $x$  to have value  $v$  inside the state  $\sigma$ , we write  $(\sigma : x \mapsto v)$ .
- To **extend** a state  $\sigma$  with a new, previously undeclared variable  $x$ , we write  $\sigma \cdot x$ . In such a state,  $x$  has undefined value.

# Dynamic Semantics

We will use **big-step** operational semantics. **What are the sets of evaluable expressions and values here?**

**Evaluable Expressions:** A pair containing a **statement** to execute and a **state**  $\sigma$ .

**Values:** The final **state** that results from executing the statement.

## States

A **state** is a mutable mapping from variables to their values. We use the following notation:

- To **read** a variable  $x$  from the state  $\sigma$ , we write  $\sigma(x)$ .
- To **update** an existing variable  $x$  to have value  $v$  inside the state  $\sigma$ , we write  $(\sigma : x \mapsto v)$ .
- To **extend** a state  $\sigma$  with a new, previously undeclared variable  $x$ , we write  $\sigma \cdot x$ . In such a state,  $x$  has undefined value.



# Dynamic Semantics

We will use **big-step** operational semantics. What are the sets of **evaluable expressions and values** here?

**Evaluable Expressions:** A pair containing a **statement** to execute and a **state**  $\sigma$ .

**Values:** The final **state** that results from executing the statement.

## States

A **state** is a mutable mapping from variables to their values. We use the following notation:

- To **read** a variable  $x$  from the state  $\sigma$ , we write  $\sigma(x)$ .
- To **update** an existing variable  $x$  to have value  $v$  inside the state  $\sigma$ , we write  $(\sigma : x \mapsto v)$ .
- To **extend** a state  $\sigma$  with a new, previously undeclared variable  $x$ , we write  $\sigma \cdot x$ . In such a state,  $x$  has undefined value.

## Evaluation Rules

We will assume we have defined a relation  $\sigma \vdash e \Downarrow v$  for **arithmetic expressions**, much like in the previous lecture.

$$\overline{(\sigma, \text{skip}) \Downarrow \sigma}$$

$$(\sigma_1, s_1; s_2) \Downarrow$$

## Evaluation Rules

We will assume we have defined a relation  $\sigma \vdash e \Downarrow v$  for **arithmetic expressions**, much like in the previous lecture.

$$\begin{array}{c}
 \frac{}{(\sigma, \text{skip}) \Downarrow \sigma} \quad \frac{(\sigma_1, s_1) \Downarrow \sigma_2 \quad (\sigma_2, s_2) \Downarrow \sigma_3}{(\sigma_1, s_1; s_2) \Downarrow \sigma_3} \\
 \frac{\sigma \vdash e \Downarrow v}{(\sigma, x := e) \Downarrow (\sigma, x \mapsto v)}
 \end{array}$$

## Evaluation Rules

We will assume we have defined a relation  $\sigma \vdash e \Downarrow v$  for **arithmetic expressions**, much like in the previous lecture.

$$\frac{}{(\sigma, \text{skip}) \Downarrow \sigma} \quad \frac{(\sigma_1, s_1) \Downarrow \sigma_2 \quad (\sigma_2, s_2) \Downarrow \sigma_3}{(\sigma_1, s_1; s_2) \Downarrow \sigma_3}$$
$$\frac{\sigma \vdash e \Downarrow v}{(\sigma, x := e) \Downarrow (\sigma, x \mapsto v)}$$

## Evaluation Rules

We will assume we have defined a relation  $\sigma \vdash e \Downarrow v$  for **arithmetic expressions**, much like in the previous lecture.

$$\frac{}{(\sigma, \text{skip}) \Downarrow \sigma} \quad \frac{(\sigma_1, s_1) \Downarrow \sigma_2 \quad (\sigma_2, s_2) \Downarrow \sigma_3}{(\sigma_1, s_1; s_2) \Downarrow \sigma_3}$$
$$\frac{\sigma \vdash e \Downarrow v}{(\sigma, x := e) \Downarrow \sigma[x \mapsto v]}$$

## Evaluation Rules

We will assume we have defined a relation  $\sigma \vdash e \Downarrow v$  for **arithmetic expressions**, much like in the previous lecture.

$$\frac{}{(\sigma, \text{skip}) \Downarrow \sigma} \quad \frac{(\sigma_1, s_1) \Downarrow \sigma_2 \quad (\sigma_2, s_2) \Downarrow \sigma_3}{(\sigma_1, s_1; s_2) \Downarrow \sigma_3}$$
$$\frac{\sigma \vdash e \Downarrow v}{(\sigma, x := e) \Downarrow \sigma}$$

## Evaluation Rules

We will assume we have defined a relation  $\sigma \vdash e \Downarrow v$  for **arithmetic expressions**, much like in the previous lecture.

$$\frac{}{(\sigma, \text{skip}) \Downarrow \sigma} \quad \frac{(\sigma_1, s_1) \Downarrow \sigma_2 \quad (\sigma_2, s_2) \Downarrow \sigma_3}{(\sigma_1, s_1; s_2) \Downarrow \sigma_3}$$
$$\frac{\sigma \vdash e \Downarrow v}{(\sigma, x := e) \Downarrow (\sigma : x \mapsto v)} \quad \frac{}{(\sigma_1, \text{var } x \cdot s) \Downarrow \sigma_2}$$

## Evaluation Rules

We will assume we have defined a relation  $\sigma \vdash e \Downarrow v$  for **arithmetic expressions**, much like in the previous lecture.

$$\frac{}{(\sigma, \text{skip}) \Downarrow \sigma} \quad \frac{(\sigma_1, s_1) \Downarrow \sigma_2 \quad (\sigma_2, s_2) \Downarrow \sigma_3}{(\sigma_1, s_1; s_2) \Downarrow \sigma_3}$$
$$\frac{\sigma \vdash e \Downarrow v}{(\sigma, x := e) \Downarrow (\sigma : x \mapsto v)} \quad \frac{}{(\sigma_1, \text{var } x \cdot s) \Downarrow \sigma_2}$$



## Evaluation Rules

We will assume we have defined a relation  $\sigma \vdash e \Downarrow v$  for **arithmetic expressions**, much like in the previous lecture.

$$\frac{}{(\sigma, \text{skip}) \Downarrow \sigma} \quad \frac{(\sigma_1, s_1) \Downarrow \sigma_2 \quad (\sigma_2, s_2) \Downarrow \sigma_3}{(\sigma_1, s_1; s_2) \Downarrow \sigma_3}$$

$$\frac{\sigma \vdash e \Downarrow v}{(\sigma, x := e) \Downarrow (\sigma : x \mapsto v)} \quad \frac{(\sigma_1 \cdot x, s) \Downarrow (\sigma_2 \cdot x)}{(\sigma_1, \text{var } x \cdot s) \Downarrow \sigma_2}$$

$$\frac{}{(\sigma_1, \text{if } e \text{ then } s_1 \text{ else } s_2 \text{ fi}) \Downarrow \sigma_2}$$

$$\frac{}{(\sigma_1, \text{if } e \text{ then } s_1 \text{ else } s_2 \text{ fi}) \Downarrow \sigma_2}$$

## Evaluation Rules

We will assume we have defined a relation  $\sigma \vdash e \Downarrow v$  for **arithmetic expressions**, much like in the previous lecture.

$$\frac{}{(\sigma, \text{skip}) \Downarrow \sigma} \quad \frac{(\sigma_1, s_1) \Downarrow \sigma_2 \quad (\sigma_2, s_2) \Downarrow \sigma_3}{(\sigma_1, s_1; s_2) \Downarrow \sigma_3}$$

$$\frac{\sigma \vdash e \Downarrow v}{(\sigma, x := e) \Downarrow (\sigma : x \mapsto v)} \quad \frac{(\sigma_1 \cdot x, s) \Downarrow (\sigma_2 \cdot x)}{(\sigma_1, \text{var } x \cdot s) \Downarrow \sigma_2}$$

$$\frac{}{(\sigma_1, \text{if } e \text{ then } s_1 \text{ else } s_2 \text{ fi}) \Downarrow \sigma_2}$$

$$\frac{}{(\sigma_1, \text{if } e \text{ then } s_1 \text{ else } s_2 \text{ fi}) \Downarrow \sigma_2}$$

## Evaluation Rules

We will assume we have defined a relation  $\sigma \vdash e \Downarrow v$  for **arithmetic expressions**, much like in the previous lecture.

$$\frac{}{(\sigma, \text{skip}) \Downarrow \sigma} \quad \frac{(\sigma_1, s_1) \Downarrow \sigma_2 \quad (\sigma_2, s_2) \Downarrow \sigma_3}{(\sigma_1, s_1; s_2) \Downarrow \sigma_3}$$

$$\frac{\sigma \vdash e \Downarrow v}{(\sigma, x := e) \Downarrow (\sigma : x \mapsto v)} \quad \frac{(\sigma_1 \cdot x, s) \Downarrow (\sigma_2 \cdot x)}{(\sigma_1, \text{var } x \cdot s) \Downarrow \sigma_2}$$

$$\frac{}{(\sigma_1, \text{if } e \text{ then } s_1 \text{ else } s_2 \text{ fi}) \Downarrow \sigma_2}$$

$$\frac{}{(\sigma_1, \text{if } e \text{ then } s_1 \text{ else } s_2 \text{ fi}) \Downarrow \sigma_2}$$

## Evaluation Rules

We will assume we have defined a relation  $\sigma \vdash e \Downarrow v$  for **arithmetic expressions**, much like in the previous lecture.

$$\begin{array}{c}
 \frac{}{(\sigma, \text{skip}) \Downarrow \sigma} \quad \frac{(\sigma_1, s_1) \Downarrow \sigma_2 \quad (\sigma_2, s_2) \Downarrow \sigma_3}{(\sigma_1, s_1; s_2) \Downarrow \sigma_3} \\
 \\
 \frac{\sigma \vdash e \Downarrow v}{(\sigma, x := e) \Downarrow (\sigma : x \mapsto v)} \quad \frac{(\sigma_1 \cdot x, s) \Downarrow (\sigma_2 \cdot x)}{(\sigma_1, \text{var } x \cdot s) \Downarrow \sigma_2} \\
 \\
 \frac{\sigma_1 \vdash e \Downarrow v \quad v \neq 0 \quad (\sigma_1, s_1) \Downarrow \sigma_2}{(\sigma_1, \text{if } e \text{ then } s_1 \text{ else } s_2 \text{ fi}) \Downarrow \sigma_2} \\
 \\
 \frac{\sigma_1 \vdash e \Downarrow 0 \quad (\sigma_1, s_2) \Downarrow \sigma_2}{(\sigma_1, \text{if } e \text{ then } s_1 \text{ else } s_2 \text{ fi}) \Downarrow \sigma_2}
 \end{array}$$

$$(\sigma_1, \text{while } e \text{ do } s \text{ od}) \Downarrow$$

$$(\sigma_1, \text{while } e \text{ do } s \text{ od}) \Downarrow$$

## Evaluation Rules

We will assume we have defined a relation  $\sigma \vdash e \Downarrow v$  for **arithmetic expressions**, much like in the previous lecture.

$$\begin{array}{c}
 \frac{}{(\sigma, \text{skip}) \Downarrow \sigma} \quad \frac{(\sigma_1, s_1) \Downarrow \sigma_2 \quad (\sigma_2, s_2) \Downarrow \sigma_3}{(\sigma_1, s_1; s_2) \Downarrow \sigma_3} \\
 \\
 \frac{\sigma \vdash e \Downarrow v}{(\sigma, x := e) \Downarrow (\sigma : x \mapsto v)} \quad \frac{(\sigma_1 \cdot x, s) \Downarrow (\sigma_2 \cdot x)}{(\sigma_1, \text{var } x \cdot s) \Downarrow \sigma_2} \\
 \\
 \frac{\sigma_1 \vdash e \Downarrow v \quad v \neq 0 \quad (\sigma_1, s_1) \Downarrow \sigma_2}{(\sigma_1, \text{if } e \text{ then } s_1 \text{ else } s_2 \text{ fi}) \Downarrow \sigma_2} \\
 \\
 \frac{\sigma_1 \vdash e \Downarrow 0 \quad (\sigma_1, s_2) \Downarrow \sigma_2}{(\sigma_1, \text{if } e \text{ then } s_1 \text{ else } s_2 \text{ fi}) \Downarrow \sigma_2}
 \end{array}$$

$$(\sigma_1, \text{while } e \text{ do } s \text{ od}) \Downarrow$$

$$(\sigma_1, \text{while } e \text{ do } s \text{ od}) \Downarrow$$

## Evaluation Rules

We will assume we have defined a relation  $\sigma \vdash e \Downarrow v$  for **arithmetic expressions**, much like in the previous lecture.

$$\begin{array}{c}
 \frac{}{(\sigma, \text{skip}) \Downarrow \sigma} \quad \frac{(\sigma_1, s_1) \Downarrow \sigma_2 \quad (\sigma_2, s_2) \Downarrow \sigma_3}{(\sigma_1, s_1; s_2) \Downarrow \sigma_3} \\
 \\
 \frac{\sigma \vdash e \Downarrow v}{(\sigma, x := e) \Downarrow (\sigma : x \mapsto v)} \quad \frac{(\sigma_1 \cdot x, s) \Downarrow (\sigma_2 \cdot x)}{(\sigma_1, \text{var } x \cdot s) \Downarrow \sigma_2} \\
 \\
 \frac{\sigma_1 \vdash e \Downarrow v \quad v \neq 0 \quad (\sigma_1, s_1) \Downarrow \sigma_2}{(\sigma_1, \text{if } e \text{ then } s_1 \text{ else } s_2 \text{ fi}) \Downarrow \sigma_2} \\
 \\
 \frac{\sigma_1 \vdash e \Downarrow 0 \quad (\sigma_1, s_2) \Downarrow \sigma_2}{(\sigma_1, \text{if } e \text{ then } s_1 \text{ else } s_2 \text{ fi}) \Downarrow \sigma_2}
 \end{array}$$

$$(\sigma_1, \text{while } e \text{ do } s \text{ od}) \Downarrow$$

$$(\sigma_1, \text{while } e \text{ do } s \text{ od}) \Downarrow$$

## Evaluation Rules

We will assume we have defined a relation  $\sigma \vdash e \Downarrow v$  for **arithmetic expressions**, much like in the previous lecture.

$$\begin{array}{c}
 \frac{}{(\sigma, \text{skip}) \Downarrow \sigma} \quad \frac{(\sigma_1, s_1) \Downarrow \sigma_2 \quad (\sigma_2, s_2) \Downarrow \sigma_3}{(\sigma_1, s_1; s_2) \Downarrow \sigma_3} \\
 \\
 \frac{\sigma \vdash e \Downarrow v}{(\sigma, x := e) \Downarrow (\sigma : x \mapsto v)} \quad \frac{(\sigma_1 \cdot x, s) \Downarrow (\sigma_2 \cdot x)}{(\sigma_1, \text{var } x \cdot s) \Downarrow \sigma_2} \\
 \\
 \frac{\sigma_1 \vdash e \Downarrow v \quad v \neq 0 \quad (\sigma_1, s_1) \Downarrow \sigma_2}{(\sigma_1, \text{if } e \text{ then } s_1 \text{ else } s_2 \text{ fi}) \Downarrow \sigma_2} \\
 \\
 \frac{\sigma_1 \vdash e \Downarrow 0 \quad (\sigma_1, s_2) \Downarrow \sigma_2}{(\sigma_1, \text{if } e \text{ then } s_1 \text{ else } s_2 \text{ fi}) \Downarrow \sigma_2}
 \end{array}$$

$$(\sigma_1, \text{while } e \text{ do } s \text{ od}) \Downarrow$$

$$(\sigma_1, \text{while } e \text{ do } s \text{ od}) \Downarrow$$

## Evaluation Rules

We will assume we have defined a relation  $\sigma \vdash e \Downarrow v$  for **arithmetic expressions**, much like in the previous lecture.

$$\begin{array}{c}
 \frac{}{(\sigma, \text{skip}) \Downarrow \sigma} \quad \frac{(\sigma_1, s_1) \Downarrow \sigma_2 \quad (\sigma_2, s_2) \Downarrow \sigma_3}{(\sigma_1, s_1; s_2) \Downarrow \sigma_3} \\
 \\
 \frac{\sigma \vdash e \Downarrow v}{(\sigma, x := e) \Downarrow (\sigma : x \mapsto v)} \quad \frac{(\sigma_1 \cdot x, s) \Downarrow (\sigma_2 \cdot x)}{(\sigma_1, \text{var } x \cdot s) \Downarrow \sigma_2} \\
 \\
 \frac{\sigma_1 \vdash e \Downarrow v \quad v \neq 0 \quad (\sigma_1, s_1) \Downarrow \sigma_2}{(\sigma_1, \text{if } e \text{ then } s_1 \text{ else } s_2 \text{ fi}) \Downarrow \sigma_2} \\
 \\
 \frac{\sigma_1 \vdash e \Downarrow 0 \quad (\sigma_1, s_2) \Downarrow \sigma_2}{(\sigma_1, \text{if } e \text{ then } s_1 \text{ else } s_2 \text{ fi}) \Downarrow \sigma_2}
 \end{array}$$

$$(\sigma_1, \text{while } e \text{ do } s \text{ od}) \Downarrow$$

$$(\sigma_1, \text{while } e \text{ do } s \text{ od}) \Downarrow$$



## Evaluation Rules

We will assume we have defined a relation  $\sigma \vdash e \Downarrow v$  for **arithmetic expressions**, much like in the previous lecture.

$$\begin{array}{c}
 \frac{}{(\sigma, \text{skip}) \Downarrow \sigma} \quad \frac{(\sigma_1, s_1) \Downarrow \sigma_2 \quad (\sigma_2, s_2) \Downarrow \sigma_3}{(\sigma_1, s_1; s_2) \Downarrow \sigma_3} \\
 \\
 \frac{\sigma \vdash e \Downarrow v}{(\sigma, x := e) \Downarrow (\sigma : x \mapsto v)} \quad \frac{(\sigma_1 \cdot x, s) \Downarrow (\sigma_2 \cdot x)}{(\sigma_1, \text{var } x \cdot s) \Downarrow \sigma_2} \\
 \\
 \frac{\sigma_1 \vdash e \Downarrow v \quad v \neq 0 \quad (\sigma_1, s_1) \Downarrow \sigma_2}{(\sigma_1, \text{if } e \text{ then } s_1 \text{ else } s_2 \text{ fi}) \Downarrow \sigma_2} \\
 \\
 \frac{\sigma_1 \vdash e \Downarrow 0 \quad (\sigma_1, s_2) \Downarrow \sigma_2}{(\sigma_1, \text{if } e \text{ then } s_1 \text{ else } s_2 \text{ fi}) \Downarrow \sigma_2}
 \end{array}$$

$$(\sigma_1, \text{while } e \text{ do } s \text{ od}) \Downarrow$$

$$(\sigma_1, \text{while } e \text{ do } s \text{ od}) \Downarrow$$

## Evaluation Rules

We will assume we have defined a relation  $\sigma \vdash e \Downarrow v$  for **arithmetic expressions**, much like in the previous lecture.

$$\begin{array}{c}
 \frac{}{(\sigma, \text{skip}) \Downarrow \sigma} \quad \frac{(\sigma_1, s_1) \Downarrow \sigma_2 \quad (\sigma_2, s_2) \Downarrow \sigma_3}{(\sigma_1, s_1; s_2) \Downarrow \sigma_3} \\
 \\
 \frac{\sigma \vdash e \Downarrow v}{(\sigma, x := e) \Downarrow (\sigma : x \mapsto v)} \quad \frac{(\sigma_1 \cdot x, s) \Downarrow (\sigma_2 \cdot x)}{(\sigma_1, \text{var } x \cdot s) \Downarrow \sigma_2} \\
 \\
 \frac{\sigma_1 \vdash e \Downarrow v \quad v \neq 0 \quad (\sigma_1, s_1) \Downarrow \sigma_2}{(\sigma_1, \text{if } e \text{ then } s_1 \text{ else } s_2 \text{ fi}) \Downarrow \sigma_2} \\
 \\
 \frac{\sigma_1 \vdash e \Downarrow 0 \quad (\sigma_1, s_2) \Downarrow \sigma_2}{(\sigma_1, \text{if } e \text{ then } s_1 \text{ else } s_2 \text{ fi}) \Downarrow \sigma_2} \\
 \\
 \frac{\sigma_1 \vdash e \Downarrow 0}{(\sigma_1, \text{while } e \text{ do } s \text{ od}) \Downarrow \sigma_1} \quad \frac{\sigma_1 \vdash e \Downarrow v \quad v \neq 0 \quad (\sigma_1, s) \Downarrow \sigma_2 \quad (\sigma_2, \text{while } e \text{ do } s \text{ od}) \Downarrow \sigma_3}{(\sigma_1, \text{while } e \text{ do } s \text{ od}) \Downarrow \sigma_3}
 \end{array}$$

## Evaluation Rules

We will assume we have defined a relation  $\sigma \vdash e \Downarrow v$  for **arithmetic expressions**, much like in the previous lecture.

$$\begin{array}{c}
 \frac{}{(\sigma, \text{skip}) \Downarrow \sigma} \quad \frac{(\sigma_1, s_1) \Downarrow \sigma_2 \quad (\sigma_2, s_2) \Downarrow \sigma_3}{(\sigma_1, s_1; s_2) \Downarrow \sigma_3} \\
 \\
 \frac{\sigma \vdash e \Downarrow v}{(\sigma, x := e) \Downarrow (\sigma : x \mapsto v)} \quad \frac{(\sigma_1 \cdot x, s) \Downarrow (\sigma_2 \cdot x)}{(\sigma_1, \text{var } x \cdot s) \Downarrow \sigma_2} \\
 \\
 \frac{\sigma_1 \vdash e \Downarrow v \quad v \neq 0 \quad (\sigma_1, s_1) \Downarrow \sigma_2}{(\sigma_1, \text{if } e \text{ then } s_1 \text{ else } s_2 \text{ fi}) \Downarrow \sigma_2} \\
 \\
 \frac{\sigma_1 \vdash e \Downarrow 0 \quad (\sigma_1, s_2) \Downarrow \sigma_2}{(\sigma_1, \text{if } e \text{ then } s_1 \text{ else } s_2 \text{ fi}) \Downarrow \sigma_2} \\
 \\
 \frac{\sigma_1 \vdash e \Downarrow 0}{(\sigma_1, \text{while } e \text{ do } s \text{ od}) \Downarrow \sigma_1} \quad \frac{\sigma_1 \vdash e \Downarrow v \quad v \neq 0 \quad (\sigma_1, s) \Downarrow \sigma_2 \quad (\sigma_2, \text{while } e \text{ do } s \text{ od}) \Downarrow \sigma_3}{(\sigma_1, \text{while } e \text{ do } s \text{ od}) \Downarrow \sigma_3}
 \end{array}$$

## Evaluation Rules

We will assume we have defined a relation  $\sigma \vdash e \Downarrow v$  for **arithmetic expressions**, much like in the previous lecture.

$$\begin{array}{c}
 \frac{}{(\sigma, \text{skip}) \Downarrow \sigma} \quad \frac{(\sigma_1, s_1) \Downarrow \sigma_2 \quad (\sigma_2, s_2) \Downarrow \sigma_3}{(\sigma_1, s_1; s_2) \Downarrow \sigma_3} \\
 \\
 \frac{\sigma \vdash e \Downarrow v}{(\sigma, x := e) \Downarrow (\sigma : x \mapsto v)} \quad \frac{(\sigma_1 \cdot x, s) \Downarrow (\sigma_2 \cdot x)}{(\sigma_1, \text{var } x \cdot s) \Downarrow \sigma_2} \\
 \\
 \frac{\sigma_1 \vdash e \Downarrow v \quad v \neq 0 \quad (\sigma_1, s_1) \Downarrow \sigma_2}{(\sigma_1, \text{if } e \text{ then } s_1 \text{ else } s_2 \text{ fi}) \Downarrow \sigma_2} \\
 \\
 \frac{\sigma_1 \vdash e \Downarrow 0 \quad (\sigma_1, s_2) \Downarrow \sigma_2}{(\sigma_1, \text{if } e \text{ then } s_1 \text{ else } s_2 \text{ fi}) \Downarrow \sigma_2} \\
 \\
 \frac{\sigma_1 \vdash e \Downarrow 0}{(\sigma_1, \text{while } e \text{ do } s \text{ od}) \Downarrow \sigma_1} \quad \frac{\sigma_1 \vdash e \Downarrow v \quad v \neq 0 \quad (\sigma_1, s) \Downarrow \sigma_2 \quad (\sigma_2, \text{while } e \text{ do } s \text{ od}) \Downarrow \sigma_3}{(\sigma_1, \text{while } e \text{ do } s \text{ od}) \Downarrow \sigma_3}
 \end{array}$$

## Evaluation Rules

We will assume we have defined a relation  $\sigma \vdash e \Downarrow v$  for **arithmetic expressions**, much like in the previous lecture.

$$\begin{array}{c}
 \frac{}{(\sigma, \text{skip}) \Downarrow \sigma} \quad \frac{(\sigma_1, s_1) \Downarrow \sigma_2 \quad (\sigma_2, s_2) \Downarrow \sigma_3}{(\sigma_1, s_1; s_2) \Downarrow \sigma_3} \\
 \\
 \frac{\sigma \vdash e \Downarrow v}{(\sigma, x := e) \Downarrow (\sigma : x \mapsto v)} \quad \frac{(\sigma_1 \cdot x, s) \Downarrow (\sigma_2 \cdot x)}{(\sigma_1, \text{var } x \cdot s) \Downarrow \sigma_2} \\
 \\
 \frac{\sigma_1 \vdash e \Downarrow v \quad v \neq 0 \quad (\sigma_1, s_1) \Downarrow \sigma_2}{(\sigma_1, \text{if } e \text{ then } s_1 \text{ else } s_2 \text{ fi}) \Downarrow \sigma_2} \\
 \\
 \frac{\sigma_1 \vdash e \Downarrow 0 \quad (\sigma_1, s_2) \Downarrow \sigma_2}{(\sigma_1, \text{if } e \text{ then } s_1 \text{ else } s_2 \text{ fi}) \Downarrow \sigma_2} \\
 \\
 \frac{\sigma_1 \vdash e \Downarrow 0}{(\sigma_1, \text{while } e \text{ do } s \text{ od}) \Downarrow \sigma_1} \quad \frac{\sigma_1 \vdash e \Downarrow v \quad v \neq 0 \quad (\sigma_1, s) \Downarrow \sigma_2 \quad (\sigma_2, \text{while } e \text{ do } s \text{ od}) \Downarrow \sigma_3}{(\sigma_1, \text{while } e \text{ do } s \text{ od}) \Downarrow \sigma_3}
 \end{array}$$

## Evaluation Rules

We will assume we have defined a relation  $\sigma \vdash e \Downarrow v$  for **arithmetic expressions**, much like in the previous lecture.

$$\begin{array}{c}
 \frac{}{(\sigma, \text{skip}) \Downarrow \sigma} \quad \frac{(\sigma_1, s_1) \Downarrow \sigma_2 \quad (\sigma_2, s_2) \Downarrow \sigma_3}{(\sigma_1, s_1; s_2) \Downarrow \sigma_3} \\
 \\
 \frac{\sigma \vdash e \Downarrow v}{(\sigma, x := e) \Downarrow (\sigma : x \mapsto v)} \quad \frac{(\sigma_1 \cdot x, s) \Downarrow (\sigma_2 \cdot x)}{(\sigma_1, \text{var } x \cdot s) \Downarrow \sigma_2} \\
 \\
 \frac{\sigma_1 \vdash e \Downarrow v \quad v \neq 0 \quad (\sigma_1, s_1) \Downarrow \sigma_2}{(\sigma_1, \text{if } e \text{ then } s_1 \text{ else } s_2 \text{ fi}) \Downarrow \sigma_2} \\
 \\
 \frac{\sigma_1 \vdash e \Downarrow 0 \quad (\sigma_1, s_2) \Downarrow \sigma_2}{(\sigma_1, \text{if } e \text{ then } s_1 \text{ else } s_2 \text{ fi}) \Downarrow \sigma_2} \\
 \\
 \frac{\sigma_1 \vdash e \Downarrow 0}{(\sigma_1, \text{while } e \text{ do } s \text{ od}) \Downarrow \sigma_1} \quad \frac{\sigma_1 \vdash e \Downarrow v \quad v \neq 0 \quad (\sigma_1, s) \Downarrow \sigma_2 \quad (\sigma_2, \text{while } e \text{ do } s \text{ od}) \Downarrow \sigma_3}{(\sigma_1, \text{while } e \text{ do } s \text{ od}) \Downarrow \sigma_3}
 \end{array}$$

## Evaluation Rules

We will assume we have defined a relation  $\sigma \vdash e \Downarrow v$  for **arithmetic expressions**, much like in the previous lecture.

$$\begin{array}{c}
 \frac{}{(\sigma, \text{skip}) \Downarrow \sigma} \quad \frac{(\sigma_1, s_1) \Downarrow \sigma_2 \quad (\sigma_2, s_2) \Downarrow \sigma_3}{(\sigma_1, s_1; s_2) \Downarrow \sigma_3} \\
 \\
 \frac{\sigma \vdash e \Downarrow v}{(\sigma, x := e) \Downarrow (\sigma : x \mapsto v)} \quad \frac{(\sigma_1 \cdot x, s) \Downarrow (\sigma_2 \cdot x)}{(\sigma_1, \text{var } x \cdot s) \Downarrow \sigma_2} \\
 \\
 \frac{\sigma_1 \vdash e \Downarrow v \quad v \neq 0 \quad (\sigma_1, s_1) \Downarrow \sigma_2}{(\sigma_1, \text{if } e \text{ then } s_1 \text{ else } s_2 \text{ fi}) \Downarrow \sigma_2} \\
 \\
 \frac{\sigma_1 \vdash e \Downarrow 0 \quad (\sigma_1, s_2) \Downarrow \sigma_2}{(\sigma_1, \text{if } e \text{ then } s_1 \text{ else } s_2 \text{ fi}) \Downarrow \sigma_2} \\
 \\
 \frac{\sigma_1 \vdash e \Downarrow 0}{(\sigma_1, \text{while } e \text{ do } s \text{ od}) \Downarrow \sigma_1} \quad \frac{\sigma_1 \vdash e \Downarrow v \quad v \neq 0 \quad (\sigma_1, s) \Downarrow \sigma_2 \quad (\sigma_2, \text{while } e \text{ do } s \text{ od}) \Downarrow \sigma_3}{(\sigma_1, \text{while } e \text{ do } s \text{ od}) \Downarrow \sigma_3}
 \end{array}$$

## Evaluation Rules

We will assume we have defined a relation  $\sigma \vdash e \Downarrow v$  for **arithmetic expressions**, much like in the previous lecture.

$$\begin{array}{c}
 \frac{}{(\sigma, \text{skip}) \Downarrow \sigma} \quad \frac{(\sigma_1, s_1) \Downarrow \sigma_2 \quad (\sigma_2, s_2) \Downarrow \sigma_3}{(\sigma_1, s_1; s_2) \Downarrow \sigma_3} \\
 \\
 \frac{\sigma \vdash e \Downarrow v}{(\sigma, x := e) \Downarrow (\sigma : x \mapsto v)} \quad \frac{(\sigma_1 \cdot x, s) \Downarrow (\sigma_2 \cdot x)}{(\sigma_1, \text{var } x \cdot s) \Downarrow \sigma_2} \\
 \\
 \frac{\sigma_1 \vdash e \Downarrow v \quad v \neq 0 \quad (\sigma_1, s_1) \Downarrow \sigma_2}{(\sigma_1, \text{if } e \text{ then } s_1 \text{ else } s_2 \text{ fi}) \Downarrow \sigma_2} \\
 \\
 \frac{\sigma_1 \vdash e \Downarrow 0 \quad (\sigma_1, s_2) \Downarrow \sigma_2}{(\sigma_1, \text{if } e \text{ then } s_1 \text{ else } s_2 \text{ fi}) \Downarrow \sigma_2} \\
 \\
 \frac{\sigma_1 \vdash e \Downarrow 0}{(\sigma_1, \text{while } e \text{ do } s \text{ od}) \Downarrow \sigma_1} \quad \frac{\sigma_1 \vdash e \Downarrow v \quad v \neq 0 \quad (\sigma_1, s) \Downarrow \sigma_2 \quad (\sigma_2, \text{while } e \text{ do } s \text{ od}) \Downarrow \sigma_3}{(\sigma_1, \text{while } e \text{ do } s \text{ od}) \Downarrow \sigma_3}
 \end{array}$$



## Hoare Logic

To give you a taste of *axiomatic semantics*, and also how *formal verification* works, we are going to define what's called a *Hoare Logic* for TinyImp to allow us to prove properties of our program. We write a *Hoare triple* judgement as:

$$\{\varphi\} s \{\psi\}$$

Where  $\varphi$  and  $\psi$  are logical formulae about state variables, called *assertions*, and  $s$  is a statement. This triple states that if the statement  $s$  successfully evaluates from a starting state satisfying the *precondition*  $\varphi$ , then the result state will satisfy the *postcondition*  $\psi$ .

## Proving Hoare Triples

To prove a hoare triple like:

```
{True}
var i · var m ·
i := 0;
m := 1;
while i < N do
  i := i + 1;
  m := m × i
od
{m = N!}
```

It is undesirable to look at the operational semantics derivations of this whole program to compute what the possible final states are for a given input state. Instead we shall define a set of rules to **prove Hoare triples directly** (called *a proof calculus*).

# Hoare Rules

$$\frac{}{(\sigma, \text{skip}) \Downarrow \sigma}$$

$$\frac{}{\{\varphi\} \text{ skip } \{\varphi\}}$$

$$\frac{(\sigma_1, s_1) \Downarrow \sigma_2 \quad (\sigma_2, s_2) \Downarrow \sigma_3}{(\sigma_1, s_1; s_2) \Downarrow \sigma_3}$$

$$\frac{}{\{\varphi\} s_1; s_2}$$

$$\frac{\sigma \vdash e \Downarrow v}{(\sigma, x := e) \Downarrow (\sigma : x \mapsto v)}$$

$$\frac{}{\{\varphi\} x := e \{\varphi\}}$$

Continuing on, we can get rules for if, and while with a *loop invariant*:

$$\frac{}{\{\varphi\} \text{ if } e \text{ then } s_1 \text{ else } s_2 \text{ fi } \{\varphi\}}$$

$$\frac{}{\{\varphi\} \text{ while } e \text{ do } s \text{ od}}$$

# Hoare Rules

$$\frac{}{(\sigma, \text{skip}) \Downarrow \sigma}$$

$$\frac{}{\{\varphi\} \text{ skip } \{\varphi\}}$$

$$\frac{(\sigma_1, s_1) \Downarrow \sigma_2 \quad (\sigma_2, s_2) \Downarrow \sigma_3}{(\sigma_1, s_1; s_2) \Downarrow \sigma_3}$$

$$\frac{}{\{\varphi\} s_1; s_2}$$

$$\frac{\sigma \vdash e \Downarrow v}{(\sigma, x := e) \Downarrow (\sigma : x \mapsto v)}$$

$$\frac{}{\{\varphi\} x := e \{\varphi\}}$$

Continuing on, we can get rules for if, and while with a *loop invariant*:

$$\frac{}{\{\varphi\} \text{ if } e \text{ then } s_1 \text{ else } s_2 \text{ fi } \{\varphi\}}$$

$$\frac{}{\{\varphi\} \text{ while } e \text{ do } s \text{ od}}$$

# Hoare Rules

$$\frac{}{(\sigma, \text{skip}) \Downarrow \sigma}$$

$$\frac{}{\{\varphi\} \text{ skip } \{\varphi\}}$$

$$\frac{(\sigma_1, s_1) \Downarrow \sigma_2 \quad (\sigma_2, s_2) \Downarrow \sigma_3}{(\sigma_1, s_1; s_2) \Downarrow \sigma_3}$$

$$\frac{\{\varphi\} s_1 \{\alpha\} \quad \{\alpha\} s_2 \{\psi\}}{\{\varphi\} s_1; s_2 \{\psi\}}$$

$$\frac{\sigma \vdash e \Downarrow v}{(\sigma, x := e) \Downarrow (\sigma : x \mapsto v)}$$

$$\frac{}{\{\varphi\} x := e \{\varphi\}}$$

Continuing on, we can get rules for if, and while with a *loop invariant*:

$$\frac{}{\{\varphi\} \text{ if } e \text{ then } s_1 \text{ else } s_2 \text{ fi } \{\varphi\}}$$

$$\frac{}{\{\varphi\} \text{ while } e \text{ do } s \text{ od}}$$

# Hoare Rules

$$\frac{}{(\sigma, \text{skip}) \Downarrow \sigma}$$

$$\frac{}{\{\varphi\} \text{ skip } \{\varphi\}}$$

$$\frac{(\sigma_1, s_1) \Downarrow \sigma_2 \quad (\sigma_2, s_2) \Downarrow \sigma_3}{(\sigma_1, s_1; s_2) \Downarrow \sigma_3}$$

$$\frac{\{\varphi\} s_1 \{\alpha\} \quad \{\alpha\} s_2 \{\psi\}}{\{\varphi\} s_1; s_2 \{\psi\}}$$

$$\frac{\sigma \vdash e \Downarrow v}{(\sigma, x := e) \Downarrow (\sigma : x \mapsto v)}$$

$$\frac{}{\{\varphi\} x := e \{\varphi\}}$$

Continuing on, we can get rules for if, and while with a *loop invariant*:

$$\frac{}{\{\varphi\} \text{ if } e \text{ then } s_1 \text{ else } s_2 \text{ fi } \{\varphi\}}$$

$$\frac{}{\{\varphi\} \text{ while } e \text{ do } s \text{ od}}$$

## Hoare Rules

$$\frac{}{(\sigma, \text{skip}) \Downarrow \sigma}$$

$$\frac{}{\{\varphi\} \text{ skip } \{\varphi\}}$$

$$\frac{(\sigma_1, s_1) \Downarrow \sigma_2 \quad (\sigma_2, s_2) \Downarrow \sigma_3}{(\sigma_1, s_1; s_2) \Downarrow \sigma_3}$$

$$\frac{\{\varphi\} s_1 \{\alpha\} \quad \{\alpha\} s_2 \{\psi\}}{\{\varphi\} s_1; s_2 \{\psi\}}$$

$$\frac{\sigma \vdash e \Downarrow v}{(\sigma, x := e) \Downarrow (\sigma : x \mapsto v)}$$

$$\frac{}{\{\varphi[x := e]\} x := e \{\varphi\}}$$

Continuing on, we can get rules for if, and while with a *loop invariant*:

$$\frac{}{\{\varphi\} \text{ if } e \text{ then } s_1 \text{ else } s_2 \text{ fi } \{\varphi\}}$$

$$\frac{}{\{\varphi\} \text{ while } e \text{ do } s \text{ od}}$$

# Hoare Rules

$$\frac{}{(\sigma, \text{skip}) \Downarrow \sigma}$$

$$\frac{}{\{\varphi\} \text{ skip } \{\varphi\}}$$

$$\frac{(\sigma_1, s_1) \Downarrow \sigma_2 \quad (\sigma_2, s_2) \Downarrow \sigma_3}{(\sigma_1, s_1; s_2) \Downarrow \sigma_3}$$

$$\frac{\{\varphi\} s_1 \{\alpha\} \quad \{\alpha\} s_2 \{\psi\}}{\{\varphi\} s_1; s_2 \{\psi\}}$$

$$\frac{\sigma \vdash e \Downarrow v}{(\sigma, x := e) \Downarrow (\sigma : x \mapsto v)}$$

$$\frac{}{\{\varphi[x := e]\} x := e \{\varphi\}}$$

Continuing on, we can get rules for if, and while with a *loop invariant*:

$$\frac{\{\varphi \wedge e\} s_1 \{\psi\} \quad \{\varphi \wedge \neg e\} s_2 \{\psi\}}{\{\varphi\} \text{ if } e \text{ then } s_1 \text{ else } s_2 \text{ fi } \{\psi\}}$$

$$\frac{\{\varphi \wedge e\} s \{\varphi\}}{\{\varphi\} \text{ while } e \text{ do } s \text{ od } \{\varphi \wedge \neg e\}}$$



## Hoare Rules

$$\frac{}{(\sigma, \text{skip}) \Downarrow \sigma}$$

$$\frac{}{\{\varphi\} \text{ skip } \{\varphi\}}$$

$$\frac{(\sigma_1, s_1) \Downarrow \sigma_2 \quad (\sigma_2, s_2) \Downarrow \sigma_3}{(\sigma_1, s_1; s_2) \Downarrow \sigma_3}$$

$$\frac{\{\varphi\} s_1 \{\alpha\} \quad \{\alpha\} s_2 \{\psi\}}{\{\varphi\} s_1; s_2 \{\psi\}}$$

$$\frac{\sigma \vdash e \Downarrow v}{(\sigma, x := e) \Downarrow (\sigma : x \mapsto v)}$$

$$\frac{}{\{\varphi[x := e]\} x := e \{\varphi\}}$$

Continuing on, we can get rules for if, and while with a *loop invariant*:

$$\frac{\{\varphi \wedge e\} s_1 \{\psi\} \quad \{\varphi \wedge \neg e\} s_2 \{\psi\}}{\{\varphi\} \text{ if } e \text{ then } s_1 \text{ else } s_2 \text{ fi } \{\psi\}}$$

$$\frac{\{\varphi \wedge e\} s \{\varphi\}}{\{\varphi\} \text{ while } e \text{ do } s \text{ od } \{\varphi \wedge \neg e\}}$$

# Hoare Rules

$$\frac{}{(\sigma, \text{skip}) \Downarrow \sigma}$$

$$\frac{}{\{\varphi\} \text{ skip } \{\varphi\}}$$

$$\frac{(\sigma_1, s_1) \Downarrow \sigma_2 \quad (\sigma_2, s_2) \Downarrow \sigma_3}{(\sigma_1, s_1; s_2) \Downarrow \sigma_3}$$

$$\frac{\{\varphi\} s_1 \{\alpha\} \quad \{\alpha\} s_2 \{\psi\}}{\{\varphi\} s_1; s_2 \{\psi\}}$$

$$\frac{\sigma \vdash e \Downarrow v}{(\sigma, x := e) \Downarrow (\sigma : x \mapsto v)}$$

$$\frac{}{\{\varphi[x := e]\} x := e \{\varphi\}}$$

Continuing on, we can get rules for if, and while with a *loop invariant*:

$$\frac{\{\varphi \wedge e\} s_1 \{\psi\} \quad \{\varphi \wedge \neg e\} s_2 \{\psi\}}{\{\varphi\} \text{ if } e \text{ then } s_1 \text{ else } s_2 \text{ fi } \{\psi\}}$$

$$\frac{\{\varphi \wedge e\} s \{\varphi\}}{\{\varphi\} \text{ while } e \text{ do } s \text{ od } \{\varphi \wedge \neg e\}}$$

# Hoare Rules

$$\overline{(\sigma, \text{skip}) \Downarrow \sigma}$$

$$\overline{\{\varphi\} \text{ skip } \{\varphi\}}$$

$$\frac{(\sigma_1, s_1) \Downarrow \sigma_2 \quad (\sigma_2, s_2) \Downarrow \sigma_3}{(\sigma_1, s_1; s_2) \Downarrow \sigma_3}$$

$$\frac{\{\varphi\} s_1 \{\alpha\} \quad \{\alpha\} s_2 \{\psi\}}{\{\varphi\} s_1; s_2 \{\psi\}}$$

$$\frac{\sigma \vdash e \Downarrow v}{(\sigma, x := e) \Downarrow (\sigma : x \mapsto v)}$$

$$\overline{\{\varphi[x := e]\} x := e \{\varphi\}}$$

Continuing on, we can get rules for if, and while with a *loop invariant*:

$$\frac{\{\varphi \wedge e\} s_1 \{\psi\} \quad \{\varphi \wedge \neg e\} s_2 \{\psi\}}{\{\varphi\} \text{ if } e \text{ then } s_1 \text{ else } s_2 \text{ fi } \{\psi\}}$$

$$\frac{\{\varphi \wedge e\} s \{\varphi\}}{\{\varphi\} \text{ while } e \text{ do } s \text{ od } \{\varphi \wedge \neg e\}}$$

# Hoare Rules

$$\frac{}{(\sigma, \text{skip}) \Downarrow \sigma}$$

$$\frac{}{\{\varphi\} \text{ skip } \{\varphi\}}$$

$$\frac{(\sigma_1, s_1) \Downarrow \sigma_2 \quad (\sigma_2, s_2) \Downarrow \sigma_3}{(\sigma_1, s_1; s_2) \Downarrow \sigma_3}$$

$$\frac{\{\varphi\} s_1 \{\alpha\} \quad \{\alpha\} s_2 \{\psi\}}{\{\varphi\} s_1; s_2 \{\psi\}}$$

$$\frac{\sigma \vdash e \Downarrow v}{(\sigma, x := e) \Downarrow (\sigma : x \mapsto v)}$$

$$\frac{}{\{\varphi[x := e]\} x := e \{\varphi\}}$$

Continuing on, we can get rules for if, and while with a *loop invariant*:

$$\frac{\{\varphi \wedge e\} s_1 \{\psi\} \quad \{\varphi \wedge \neg e\} s_2 \{\psi\}}{\{\varphi\} \text{ if } e \text{ then } s_1 \text{ else } s_2 \text{ fi } \{\psi\}}$$

$$\frac{\{\varphi \wedge e\} s \{\varphi\}}{\{\varphi\} \text{ while } e \text{ do } s \text{ od } \{\varphi \wedge \neg e\}}$$

## Consequence

There is one more rule, called the *rule of consequence*, that we need to insert ordinary logical reasoning into our Hoare logic proofs:

$$\frac{\varphi \Rightarrow \alpha \quad \{\alpha\} s \{\beta\} \quad \beta \Rightarrow \psi}{\{\varphi\} s \{\psi\}}$$

This is the only rule that is **not** directed entirely by syntax. This means a Hoare logic proof need not look like a derivation tree. Instead we can sprinkle assertions through our program and specially note uses of the consequence rule. Time to do it on the iPad

$$\frac{\{\varphi \wedge e\} s_1 \{\psi\} \quad \{\varphi \wedge \neg e\} s_2 \{\psi\}}{\{\varphi\} \text{ if } e \text{ then } s_1 \text{ else } s_2 \text{ fi } \{\psi\}}$$

$$\frac{}{\{\varphi[x := e]\} x := e \{\varphi\}}$$

$$\frac{\{\varphi \wedge e\} s \{\varphi\}}{\{\varphi\} \text{ while } e \text{ do } s \text{ od } \{\varphi \wedge \neg e\}}$$

$$\frac{\{\varphi\} s_1 \{\alpha\} \quad \{\alpha\} s_2 \{\psi\}}{\{\varphi\} s_1; s_2 \{\psi\}}$$

## Consequence

There is one more rule, called the *rule of consequence*, that we need to insert ordinary logical reasoning into our Hoare logic proofs:

$$\frac{\varphi \Rightarrow \alpha \quad \{\alpha\} s \{\beta\} \quad \beta \Rightarrow \psi}{\{\varphi\} s \{\psi\}}$$

This is the only rule that is **not** directed entirely by syntax. This means a Hoare logic proof need not look like a derivation tree. Instead we can sprinkle assertions through our program and specially note uses of the consequence rule. Time to do it on the iPad

$$\frac{\{\varphi \wedge e\} s_1 \{\psi\} \quad \{\varphi \wedge \neg e\} s_2 \{\psi\}}{\{\varphi\} \text{ if } e \text{ then } s_1 \text{ else } s_2 \text{ fi } \{\psi\}}$$

$$\frac{}{\{\varphi[x := e]\} x := e \{\varphi\}}$$

$$\frac{\{\varphi \wedge e\} s \{\varphi\}}{\{\varphi\} \text{ while } e \text{ do } s \text{ od } \{\varphi \wedge \neg e\}}$$

$$\frac{\{\varphi\} s_1 \{\alpha\} \quad \{\alpha\} s_2 \{\psi\}}{\{\varphi\} s_1; s_2 \{\psi\}}$$