# Imperative Programming

Dr. Liam O'Connor
University of Edinburgh LFCS
UNSW, Term 3 2020

# Imperative Programming



imperō



**Definition**

*Imperative programming* is where programs are describ *atements* or commands to manipulate mutable *state* or cause ex *ects*.



*States* may take the form of a *mapping* from vari 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.

## Factorial Example in BASIC (1964)

## Dijkstra (1968)



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

## Factorial Example in Pascal (1970)

# Syntax

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



## Grammar

**St** ... *hing*



... *nment*

... *ration*

| `if` **Expr** `then` **Stmt** `e` ... *al*

| `while` **Expr** `do` **Stmt** ... *p*

**Stmt** ; **Stmt** ... *g*



**Expr** ::= ⟨*Arithmetic expressions*⟩

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** $m$ **var** $n$ **var** $i$



**while** $i < N$ **do**
$i := i + 1;$
$m := m \times i$
**od**



$i$
**od**

## Static Semantics

**Types?**

## Static Semantics

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

**Scopes?**

## 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?**

# 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!

$$U, V \vdash s\ \mathbf{o}$$

Note: $V \subseteq U$

# 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!



$$U; V \vdash s\ \mathbf{ok}$$

$U$: Set of declared free variables

Note: $V \subseteq U$

## 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!



$U; V \vdash s$ **ok**

Set of declared free variables

Note: $V \subseteq U$

## 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!

$$U; V \vdash s\ \mathbf{ok}$$

Set of declared free variables

Note: $V \subseteq U$

## 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!



$$U ; V \vdash s\ \mathbf{o}$$



Set of declared free variables

Set of definitely written to free variables

Note: $V \subseteq U$

## Static Semantics Rules



$$\frac{}{U; V \quad \texttt{skip}\ \textbf{ok}} \rightsquigarrow$$

## Static Semantics Rules



$$\frac{}{U; V \quad \texttt{skip}\ \mathbf{ok} \rightsquigarrow} \qquad \frac{}{U; V \quad x := e\ \mathbf{ok} \rightsquigarrow}$$

## Static Semantics Rules



$$\frac{}{U; V \vdash \texttt{skip}\ \textbf{ok} \rightsquigarrow} \qquad \frac{x \in U}{U; V \vdash x\ \texttt{:=}\ e\ \textbf{ok} \rightsquigarrow}$$

## Static Semantics Rules



$$\frac{}{U; V \quad \texttt{skip}\ \mathbf{ok} \leadsto} \qquad \frac{x \in U \qquad \mathrm{FV}(e) \subseteq V}{U; V \quad x \texttt{ := } e\ \mathbf{ok} \leadsto}$$

## Static Semantics Rules

$$\frac{}{U; V \quad \texttt{skip}\ \textbf{ok} \leadsto}$$

$$\frac{x \in U \qquad \mathrm{FV}(e) \subseteq V}{U; V \quad x := e\ \textbf{ok} \leadsto \quad x}$$

## Static Semantics Rules



$$\frac{}{U; V \vdash \texttt{skip}\ \mathbf{ok} \rightsquigarrow \emptyset} \qquad \frac{x \in U \quad \mathrm{FV}(e) \subseteq V}{U; V \vdash x := e\ \mathbf{ok} \rightsquigarrow \{x\}}$$

# Static Semantics Rules



$$\frac{}{U; V \quad \texttt{skip}\ \textbf{ok} \leadsto} \qquad \frac{x \in U \quad \mathrm{FV}(e) \subseteq V}{U; V \quad x \texttt{ := } e\ \textbf{ok} \leadsto \quad x}$$

## Static Semantics Rules



$$\frac{}{U; V \quad \texttt{skip}\ \textbf{ok} \rightsquigarrow} \qquad \frac{x \in U \quad \mathrm{FV}(e) \subseteq V}{U; V \quad x \texttt{ := } e\ \textbf{ok} \rightsquigarrow \quad x}$$

## Static Semantics Rules



$$\frac{}{U; V \vdash \texttt{skip}\ \mathbf{ok} \rightsquigarrow \emptyset} \qquad \frac{x \in U \quad \mathrm{FV}(e) \subseteq V}{U; V \vdash x \texttt{ := } e\ \mathbf{ok} \rightsquigarrow \{x\}}$$



$$\frac{}{U; V \vdash \texttt{if}\ e\ \texttt{then}\ s_1\ \texttt{else}\ s_2}$$

# Static Semantics Rules



$$\frac{}{U; V \quad \texttt{skip}\ \mathbf{ok} \rightsquigarrow} \qquad \frac{x \in U \quad \mathrm{FV}(e) \subseteq V}{U; V \quad x\ \texttt{:=}\ e\ \mathbf{ok} \rightsquigarrow \quad x}$$



$$\frac{\mathrm{FV}(e) \subseteq V}{U; V \vdash \texttt{if}\ e\ \texttt{then}\ s_1\ \texttt{else}\ s_2}$$

## Static Semantics Rules



$$\frac{}{U; V \quad \texttt{skip}\ \textbf{ok} \leadsto} \qquad \frac{x \in U \quad \mathrm{FV}(e) \subseteq V}{U; V \quad x\ \texttt{:=}\ e\ \textbf{ok} \leadsto \quad x}$$



$$\frac{\mathrm{FV}(e) \subseteq V \qquad U; V \vdash s_1 \qquad\qquad W_1}{U; V \vdash \texttt{if}\ e\ \texttt{then}\ s_1\ \texttt{else}\ s_2}$$

# Static Semantics Rules



$$\frac{}{U; V \quad \texttt{skip}\ \textbf{ok} \rightsquigarrow}$$

$$\frac{x \in U \quad \mathrm{FV}(e) \subseteq V}{U; V \quad x\ \texttt{:=}\ e\ \textbf{ok} \rightsquigarrow \quad x}$$



$$\frac{\mathrm{FV}(e) \subseteq V \qquad U; V \vdash s_1 \qquad W_1 \qquad U; V \quad s_2 \qquad W_2}{U; V \vdash \texttt{if}\ e\ \texttt{then}\ s_1\ \texttt{else}\ s_2}$$

# Static Semantics Rules



$$\frac{}{U; V \quad \texttt{skip}\ \textbf{ok} \rightsquigarrow} \qquad \frac{x \in U \quad \mathrm{FV}(e) \subseteq V}{U; V \quad x\ \texttt{:=}\ e\ \textbf{ok} \rightsquigarrow \quad x}$$



$$\frac{}{}$$

$$\frac{\mathrm{FV}(e) \subseteq V \qquad U; V \vdash s_1 \qquad W_1 \qquad U; V \quad s_2 \qquad W_2}{U; V \vdash \texttt{if}\ e\ \texttt{then}\ s_1\ \texttt{else}\ s_2}$$

# Static Semantics Rules



$$\frac{}{U; V \quad \texttt{skip}\ \textbf{ok} \rightsquigarrow} \qquad \frac{x \in U \quad \mathrm{FV}(e) \subseteq V}{U; V \quad x \texttt{ := } e\ \textbf{ok} \rightsquigarrow \quad x}$$



$$\frac{\mathrm{FV}(e) \subseteq V \qquad U; V \vdash s_1 \qquad W_1 \qquad U; V \quad s_2 \qquad W_2}{U; V \vdash \texttt{if } e \texttt{ then } s_1 \texttt{ else } s_2}$$



$$\frac{}{U; V \vdash \texttt{while } e \texttt{ do } s \texttt{ od}\ \textbf{ok}}$$

# Static Semantics Rules



$$\frac{}{U; V \quad \texttt{skip}\ \mathbf{ok} \rightsquigarrow} \qquad \frac{x \in U \quad \mathrm{FV}(e) \subseteq V}{U; V \quad x\ \texttt{:=}\ e\ \mathbf{ok} \rightsquigarrow \quad x}$$



$$\frac{\mathrm{FV}(e) \subseteq V \qquad U; V \vdash s_1 \qquad W_1 \qquad U; V \quad s_2 \qquad W_2}{U; V \vdash \texttt{if}\ e\ \texttt{then}\ s_1\ \texttt{else}\ s_2}$$



$$\frac{\mathrm{FV}(e) \subseteq V}{U; V \vdash \texttt{while}\ e\ \texttt{do}\ s\ \texttt{od}\ \mathbf{ok}}$$

## Static Semantics Rules



$$\frac{}{U; V \quad \texttt{skip}\ \mathbf{ok} \leadsto} \qquad \frac{x \in U \quad \mathrm{FV}(e) \subseteq V}{U; V \quad x := e\ \mathbf{ok} \leadsto \quad x}$$



$$\frac{\mathrm{FV}(e) \subseteq V \qquad U; V \vdash s_1 \qquad W_1 \qquad U; V \quad s_2 \qquad W_2}{U; V \vdash \texttt{if}\ e\ \texttt{then}\ s_1\ \texttt{else}\ s_2}$$



$$\frac{\mathrm{FV}(e) \subseteq V \qquad U; V}{U; V \vdash \texttt{while}\ e\ \texttt{do}\ s\ \texttt{od}\ \mathbf{ok}}$$

# Static Semantics Rules



$$\frac{}{U; V \quad \texttt{skip}\ \mathbf{ok} \rightsquigarrow} \qquad \frac{x \in U \quad \mathrm{FV}(e) \subseteq V}{U; V \quad x\ \texttt{:=}\ e\ \mathbf{ok} \rightsquigarrow \quad x}$$



$$\frac{\mathrm{FV}(e) \subseteq V \qquad U; V \vdash s_1 \qquad W_1 \qquad U; V \quad s_2 \qquad W_2}{U; V \vdash \texttt{if}\ e\ \texttt{then}\ s_1\ \texttt{else}\ s_2}$$



$$\frac{\mathrm{FV}(e) \subseteq V \qquad U; V}{U; V \vdash \texttt{while}\ e\ \texttt{do}\ s\ \texttt{od}\ \mathbf{ok} \qquad \emptyset}$$

## Static Semantics Rules



$$\frac{}{U; V \quad \texttt{skip}\ \textbf{ok} \leadsto} \qquad \frac{x \in U \quad \mathrm{FV}(e) \subseteq V}{U; V \quad x \ \texttt{:=}\ e\ \textbf{ok} \leadsto \quad x}$$



$$\frac{}{}$$

$$\frac{\mathrm{FV}(e) \subseteq V \qquad U; V \vdash s_1 \qquad W_1 \qquad U; V \quad s_2 \qquad W_2}{U; V \vdash \texttt{if}\ e\ \texttt{then}\ s_1\ \texttt{else}\ s_2}$$



$$\frac{\mathrm{FV}(e) \subseteq V \qquad U; V}{U; V \vdash \texttt{while}\ e\ \texttt{do}\ s\ \texttt{od}\ \textbf{ok} \quad \emptyset}$$

$$\frac{}{U; V \vdash s_1 ; s_2\ \textbf{ok} \leadsto}$$

## Static Semantics Rules



$$\frac{}{U; V \quad \texttt{skip}\ \textbf{ok} \leadsto} \qquad \frac{x \in U \quad \mathrm{FV}(e) \subseteq V}{U; V \quad x \mathrel{\texttt{:=}} e\ \textbf{ok} \leadsto \quad x}$$



$$\frac{\mathrm{FV}(e) \subseteq V \qquad U; V \vdash s_1 \qquad W_1 \qquad U; V \quad s_2 \qquad W_2}{U; V \vdash \texttt{if}\ e\ \texttt{then}\ s_1\ \texttt{else}\ s_2}$$



$$\frac{\mathrm{FV}(e) \subseteq V \qquad U; V}{U; V \vdash \texttt{while}\ e\ \texttt{do}\ s\ \texttt{od}\ \textbf{ok} \quad \emptyset}$$

$$\frac{U; V \vdash s_1\ \textbf{ok} \leadsto W_1}{U; V \vdash s_1; s_2\ \textbf{ok} \leadsto}$$

# Static Semantics Rules



$$\frac{}{U; V \quad \texttt{skip}\ \mathbf{ok} \rightsquigarrow} \qquad \frac{x \in U \quad \mathrm{FV}(e) \subseteq V}{U; V \quad x\ \texttt{:=}\ e\ \mathbf{ok} \rightsquigarrow \quad x}$$



$$\frac{\mathrm{FV}(e) \subseteq V \qquad U; V \vdash s_1 \qquad W_1 \qquad U; V \quad s_2 \qquad W_2}{U; V \vdash \texttt{if}\ e\ \texttt{then}\ s_1\ \texttt{else}\ s_2}$$



$$\frac{\mathrm{FV}(e) \subseteq V \qquad U; V}{U; V \vdash \texttt{while}\ e\ \texttt{do}\ s\ \texttt{od}\ \mathbf{ok} \qquad \emptyset}$$

$$\frac{U; V \vdash s_1\ \mathbf{ok} \rightsquigarrow W_1 \qquad U; (V \cup W_1) \vdash s_2\ \mathbf{ok} \rightsquigarrow W_2}{U; V \vdash s_1; s_2\ \mathbf{ok} \rightsquigarrow}$$

# Static Semantics Rules



$$\frac{}{U; V \quad \texttt{skip}\ \textbf{ok} \rightsquigarrow} \qquad \frac{x \in U \quad \mathrm{FV}(e) \subseteq V}{U; V \quad x\ \texttt{:=}\ e\ \textbf{ok} \rightsquigarrow \quad x}$$



$$\frac{\mathrm{FV}(e) \subseteq V \qquad U; V \vdash s_1 \qquad W_1 \qquad U; V \quad s_2 \qquad W_2}{U; V \vdash \texttt{if}\ e\ \texttt{then}\ s_1\ \texttt{else}\ s_2}$$



$$\frac{\mathrm{FV}(e) \subseteq V \qquad U; V}{U; V \vdash \texttt{while}\ e\ \texttt{do}\ s\ \texttt{od}\ \textbf{ok} \qquad \emptyset}$$

$$\frac{U; V \vdash s_1\ \textbf{ok} \rightsquigarrow W_1 \qquad U; (V \cup W_1) \vdash s_2\ \textbf{ok} \rightsquigarrow W_2}{U; V \vdash s_1; s_2\ \textbf{ok} \rightsquigarrow W_1 \cup W_2}$$

## Dynamic Semantics

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

## Dynamic Semantics

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



**Evaluable Expressions**:

# 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**:

# 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 fin

# 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 fin



**States**

A *state* is a muta

notation:

- To read a variable $x$ from the state $\sigma$, we writ
- To update an existing variable $x$ to have value e 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.

## Evaluation Rules

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

## Evaluation Rules

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

$$\frac{}{} \qquad \frac{}{(\sigma_1, s_1) \Downarrow \sigma_2}$$

## Evaluation Rules

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



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

## Evaluation Rules

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

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

## Evaluation Rules

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



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

# Evaluation Rules

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



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

# Evaluation Rules

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



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

# Evaluation Rules

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



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

## Evaluation Rules

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



$$\frac{}{\phantom{x}} \qquad \frac{(\sigma_1, s_1) \Downarrow \sigma_2 \qquad (\sigma_2, s_2) \Downarrow \sigma_3}{\phantom{x}}$$

## Evaluation Rules

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



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





$$\frac{}{(\sigma_1, \texttt{if } e \texttt{ then } s_1 \texttt{ els}}$$

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

# Evaluation Rules

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

$$\frac{}{\phantom{x}} \qquad \frac{(\sigma_1, s_1) \Downarrow \sigma_2 \qquad (\sigma_2, s_2) \Downarrow \sigma_3}{\phantom{x}}$$

$$\frac{\sigma_1 \vdash e \Downarrow v \qquad v \neq 0}{(\sigma_1, \texttt{if } e \texttt{ then } s_1 \texttt{ els}}$$

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

## Evaluation Rules

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



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



$$\frac{\sigma_1 \vdash e \Downarrow v \qquad v \neq 0}{(\sigma_1, \texttt{if}\ e\ \texttt{then}\ s_1\ \texttt{els}}$$



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

# Evaluation Rules

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



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



$$\frac{\sigma_1 \vdash e \Downarrow v \qquad v \neq 0}{(\sigma_1, \texttt{if } e \texttt{ then } s_1 \texttt{ els}}$$



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

## Evaluation Rules

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



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



$$\frac{\sigma_1 \vdash e \Downarrow v \qquad v \neq 0}{(\sigma_1, \texttt{if}\ e\ \texttt{then}\ s_1\ \texttt{els}}$$



$$\frac{\sigma_1 \vdash e \Downarrow 0}{(\sigma_1, \texttt{if}\ e\ \texttt{then}\ s_1\ \texttt{else}\ s_2\ \texttt{fi}) \Downarrow}$$

# Evaluation Rules

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



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



$$\frac{\sigma_1 \vdash e \Downarrow v \qquad v \neq 0}{(\sigma_1, \texttt{if}\ e\ \texttt{then}\ s_1\ \texttt{els}}$$



$$\frac{\sigma_1 \vdash e \Downarrow 0 \qquad (\sigma_1}{(\sigma_1, \texttt{if}\ e\ \texttt{then}\ s_1\ \texttt{else}\ s_2\ \texttt{fi}) \Downarrow \sigma_2}$$

# Evaluation Rules

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



$$\frac{}{\phantom{x}} \qquad \frac{(\sigma_1, s_1) \Downarrow \sigma_2 \qquad (\sigma_2, s_2) \Downarrow \sigma_3}{\phantom{x}}$$



$$\frac{\sigma_1 \vdash e \Downarrow v \qquad v \neq 0}{(\sigma_1, \texttt{if } e \texttt{ then } s_1 \texttt{ els}}$$



$$\frac{\sigma_1 \vdash e \Downarrow 0 \qquad (\sigma_1}{(\sigma_1, \texttt{if } e \texttt{ then } s_1 \texttt{ else } s_2 \texttt{ fi}) \Downarrow \sigma_2}$$

$$\frac{}{(\sigma_1, \texttt{while } e \texttt{ do } s \texttt{ od}) \Downarrow} \qquad \frac{}{(\sigma_1, \texttt{while } e \texttt{ do } s \texttt{ od}) \Downarrow}$$

# Evaluation Rules

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



$$\frac{}{\phantom{x}} \qquad \frac{(\sigma_1, s_1) \Downarrow \sigma_2 \quad (\sigma_2, s_2) \Downarrow \sigma_3}{}$$



$$\frac{\sigma_1 \vdash e \Downarrow v \quad v \neq 0}{(\sigma_1, \texttt{if } e \texttt{ then } s_1 \texttt{ els}}$$



$$\frac{\sigma_1 \vdash e \Downarrow 0 \quad (\sigma_1}{(\sigma_1, \texttt{if } e \texttt{ then } s_1 \texttt{ else } s_2 \texttt{ fi}) \Downarrow \sigma_2}$$

$$\frac{\sigma_1 \vdash e \Downarrow 0}{(\sigma_1, \texttt{while } e \texttt{ do } s \texttt{ od}) \Downarrow} \qquad \frac{}{(\sigma_1, \texttt{while } e \texttt{ do } s \texttt{ od}) \Downarrow}$$

# Evaluation Rules

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



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



$$\frac{\sigma_1 \vdash e \Downarrow v \qquad v \neq 0}{(\sigma_1, \texttt{if } e \texttt{ then } s_1 \texttt{ els}}$$



$$\frac{\sigma_1 \vdash e \Downarrow 0 \qquad (\sigma_1}{(\sigma_1, \texttt{if } e \texttt{ then } s_1 \texttt{ else } s_2 \texttt{ fi}) \Downarrow \sigma_2}$$

$$\frac{\sigma_1 \vdash e \Downarrow 0}{(\sigma_1, \texttt{while } e \texttt{ do } s \texttt{ od}) \Downarrow \sigma_1} \qquad \frac{}{(\sigma_1, \texttt{while } e \texttt{ do } s \texttt{ od}) \Downarrow}$$

## Evaluation Rules

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



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



$$\frac{\sigma_1 \vdash e \Downarrow v \qquad v \neq 0}{(\sigma_1, \texttt{if}\ e\ \texttt{then}\ s_1\ \texttt{els}}$$



$$\frac{\sigma_1 \vdash e \Downarrow 0 \qquad (\sigma_1}{(\sigma_1, \texttt{if}\ e\ \texttt{then}\ s_1\ \texttt{else}\ s_2\ \texttt{fi}) \Downarrow \sigma_2}$$

$$\frac{\sigma_1 \vdash e \Downarrow 0}{(\sigma_1, \texttt{while}\ e\ \texttt{do}\ s\ \texttt{od}) \Downarrow \sigma_1} \qquad \frac{\sigma_1 \vdash e \Downarrow v \quad v \neq 0}{(\sigma_1, \texttt{while}\ e\ \texttt{do}\ s\ \texttt{od}) \Downarrow}$$

## Evaluation Rules

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



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



$$\frac{\sigma_1 \vdash e \Downarrow v \qquad v \neq 0}{(\sigma_1, \texttt{if}\ e\ \texttt{then}\ s_1\ \texttt{els} \ldots}$$



$$\frac{\sigma_1 \vdash e \Downarrow 0 \qquad (\sigma_1 \ldots}{(\sigma_1, \texttt{if}\ e\ \texttt{then}\ s_1\ \texttt{else}\ s_2\ \texttt{fi}) \Downarrow \sigma_2}$$

$$\frac{\sigma_1 \vdash e \Downarrow 0}{(\sigma_1, \texttt{while}\ e\ \texttt{do}\ s\ \texttt{od}) \Downarrow \sigma_1} \qquad \frac{\sigma_1 \vdash e \Downarrow v \quad v \neq 0 \qquad (\sigma_1, s) \Downarrow \sigma_2 \quad \ldots}{(\sigma_1, \texttt{while}\ e\ \texttt{do}\ s\ \texttt{od}) \Downarrow \ldots}$$

# Evaluation Rules

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



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



$$\frac{\sigma_1 \vdash e \Downarrow v \qquad v \neq 0}{(\sigma_1, \texttt{if}\ e\ \texttt{then}\ s_1\ \texttt{els}}$$



$$\frac{\sigma_1 \vdash e \Downarrow 0 \qquad (\sigma_1}{(\sigma_1, \texttt{if}\ e\ \texttt{then}\ s_1\ \texttt{else}\ s_2\ \texttt{fi}) \Downarrow \sigma_2}$$

$$\frac{\sigma_1 \vdash e \Downarrow 0}{(\sigma_1, \texttt{while}\ e\ \texttt{do}\ s\ \texttt{od}) \Downarrow \sigma_1} \qquad \frac{\sigma_1 \vdash e \Downarrow v \quad v \neq 0 \quad (\sigma_1, s) \Downarrow \sigma_2 \quad (\sigma_2, \texttt{while}\ e\ \texttt{do}\ s\ \texttt{od}) \Downarrow \sigma_3}{(\sigma_1, \texttt{while}\ e\ \texttt{do}\ s\ \texttt{od}) \Downarrow}$$

## Evaluation Rules

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



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



$$\frac{\sigma_1 \vdash e \Downarrow v \qquad v \neq 0}{(\sigma_1, \texttt{if } e \texttt{ then } s_1 \texttt{ els}}$$



$$\frac{\sigma_1 \vdash e \Downarrow 0 \qquad (\sigma_1}{(\sigma_1, \texttt{if } e \texttt{ then } s_1 \texttt{ else } s_2 \texttt{ fi}) \Downarrow \sigma_2}$$

$$\frac{\sigma_1 \vdash e \Downarrow 0}{(\sigma_1, \texttt{while } e \texttt{ do } s \texttt{ od}) \Downarrow \sigma_1} \qquad \frac{\sigma_1 \vdash e \Downarrow v \quad v \neq 0 \qquad (\sigma_1, s) \Downarrow \sigma_2 \quad (\sigma_2, \texttt{while } e \texttt{ do } s \texttt{ od}) \Downarrow \sigma_3}{(\sigma_1, \texttt{while } e \texttt{ do } s \texttt{ od}) \Downarrow \sigma_3}$$

## 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:



Where $\varphi$ and $\psi$ are logical formulae about state variabl *ns*, and $s$ is a statement. This triple states that if the statement om a starting state satisfying the *precondition* $\varphi$, then th *postcondition* $\psi$:



$$\varphi(\sigma) \wedge (\sigma, s) \Downarrow \sigma' \Rightarrow \psi(\sigma')$$

## Proving Hoare Triples

To prove a Hoare triple like:



$\{\mathrm{True}\}$

**var** $i$ · **var** $m$ ·



$m := m \times$

**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

$$\frac{}{(\sigma, \mathtt{skip}) \Downarrow \sigma} \qquad \frac{}{\{\varphi\}\ \mathtt{skip}}$$



$$\frac{(\sigma_1, s_1) \Downarrow \sigma_2 \qquad (\sigma\ \ , s\ \ ) \quad \sigma}{(\sigma_1, s}$$



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

## Hoare Rules

$$\frac{}{(\sigma, \mathtt{skip}) \Downarrow \sigma} \qquad \frac{}{\{\varphi\}\ \mathtt{skip}\ \{\varphi\}}$$



$$\frac{(\sigma_1, s_1) \Downarrow \sigma_2 \qquad (\sigma\ , s\ )\ \ \ \sigma}{(\sigma_1, s}$$



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

# Hoare Rules

$$\frac{}{(\sigma, \texttt{skip}) \Downarrow \sigma} \qquad \frac{}{\{\varphi\}\ \texttt{skip}\ \{\varphi\}}$$



$$\frac{(\sigma_1, s_1) \Downarrow \sigma_2 \qquad (\sigma\ , s\ )\ \ \sigma}{(\sigma_1, s}$$



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

# Hoare Rules

$$\overline{(\sigma, \texttt{skip}) \Downarrow \sigma} \qquad \overline{\{\varphi\}\ \texttt{skip}\ \{\varphi\}}$$



$$\frac{(\sigma_1, s_1) \Downarrow \sigma_2 \qquad (\sigma \quad, s \quad) \quad \sigma}{(\sigma_1, s} \qquad \frac{\varphi \quad s \quad \alpha \qquad\qquad \alpha \quad s \quad \psi}{}$$



$$\sigma$$

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

## Hoare Rules

$$\frac{}{(\sigma, \texttt{skip}) \Downarrow \sigma} \qquad \frac{}{\{\varphi\}\ \texttt{skip}\ \{\varphi\}}$$



$$\frac{(\sigma_1, s_1) \Downarrow \sigma_2 \qquad (\sigma\ \ , s\ \ )\ \ \ \ \sigma}{(\sigma_1, s} \qquad \frac{\varphi\ \ s\ \ \alpha \qquad \alpha\ \ s\ \ \psi}{}$$



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

## Hoare Rules

$$\frac{}{(\sigma, \texttt{skip}) \Downarrow \sigma} \qquad \frac{}{\{\varphi\}\ \texttt{skip}\ \{\varphi\}}$$



$$\frac{(\sigma_1, s_1) \Downarrow \sigma_2 \qquad (\sigma\ \ , s\ \ )\ \ \ \sigma}{(\sigma_1, s} \qquad \frac{\varphi\ \ s\ \ \alpha \qquad \alpha\ \ s\ \ \psi}{}$$



$$\sigma$$

$$\frac{}{(\sigma, x := e) \Downarrow (\sigma : x \mapsto v)} \qquad \frac{}{\{\varphi[x := e]\}}$$



Continuing on, we can get rules for `if`, and `whi`

$$\frac{}{\{\varphi\}\ \texttt{if}\ e\ \texttt{then}\ s_1\ \texttt{else}\ s_2\ \texttt{fi}\ \{\psi\}} \qquad \frac{}{\{\varphi\}\ \texttt{while}\ e\ \texttt{do}\ s\ \texttt{od}}$$

# Hoare Rules

$$\frac{}{(\sigma, \texttt{skip}) \Downarrow \sigma} \qquad \frac{}{\{\varphi\}\ \texttt{skip}\ \{\varphi\}}$$



$$\frac{(\sigma_1, s_1) \Downarrow \sigma_2 \qquad (\sigma\ ,\ s\ )\quad \sigma}{(\sigma_1, s} \qquad \frac{\varphi\quad s\quad \alpha \qquad\qquad \alpha\quad s\quad \psi}{}$$



$$\frac{\sigma}{(\sigma, x := e) \Downarrow (\sigma : x \mapsto v)} \qquad \frac{}{\{\varphi[x := e]\}}$$



Continuing on, we can get rules for `if`, and `whi`

$$\frac{\{\varphi \land e\}\ s_1\ \{\psi\}}{\{\varphi\}\ \texttt{if}\ e\ \texttt{then}\ s_1\ \texttt{else}\ s_2\ \texttt{fi}\ \{\psi\}} \qquad \frac{}{\{\varphi\}\ \texttt{while}\ e\ \texttt{do}\ s\ \texttt{od}}$$

## Hoare Rules

$$\frac{}{(\sigma, \texttt{skip}) \Downarrow \sigma} \qquad \frac{}{\{\varphi\}\ \texttt{skip}\ \{\varphi\}}$$



$$\frac{(\sigma_1, s_1) \Downarrow \sigma_2 \qquad (\sigma\ , s\ )\ \ \sigma}{(\sigma_1, s} \qquad \frac{\varphi\ \ s\ \ \alpha \qquad \alpha\ \ s\ \ \psi}{}$$



$$\frac{\sigma}{(\sigma, x := e) \Downarrow (\sigma : x \mapsto v)} \qquad \frac{}{\{\varphi[x := e]\}}$$



Continuing on, we can get rules for `if`, and `whi`

$$\frac{\{\varphi \wedge e\}\ s_1\ \{\psi\} \qquad \{\varphi \wedge \neg e\}\ s_2\ \{\psi\}}{\{\varphi\}\ \texttt{if}\ e\ \texttt{then}\ s_1\ \texttt{else}\ s_2\ \texttt{fi}\ \{\psi\}} \qquad \frac{}{\{\varphi\}\ \texttt{while}\ e\ \texttt{do}\ s\ \texttt{od}}$$

## Hoare Rules

$$\frac{}{(\sigma, \texttt{skip}) \Downarrow \sigma} \qquad \frac{}{\{\varphi\}\ \texttt{skip}\ \{\varphi\}}$$



$$\frac{(\sigma_1, s_1) \Downarrow \sigma_2 \qquad (\sigma\ , s\ )\quad \sigma}{(\sigma_1, s} \qquad \frac{\varphi\quad s\quad \alpha \qquad\qquad \alpha\quad s\quad \psi}{}$$



$$\frac{\sigma}{(\sigma, x := e) \Downarrow (\sigma : x \mapsto v)} \qquad \frac{}{\{\varphi[x := e]\}}$$



Continuing on, we can get rules for `if`, and `whi`

$$\frac{\{\varphi \land e\}\ s_1\ \{\psi\} \qquad \{\varphi \land \neg e\}\ s_2\ \{\psi\}}{\{\varphi\}\ \texttt{if}\ e\ \texttt{then}\ s_1\ \texttt{else}\ s_2\ \texttt{fi}\ \{\psi\}} \qquad \frac{\{\varphi \land e\}\ s\ \{\varphi\}}{\{\varphi\}\ \texttt{while}\ e\ \texttt{do}\ s\ \texttt{od}}$$

# Hoare Rules

$$\frac{}{(\sigma, \texttt{skip}) \Downarrow \sigma} \qquad \frac{}{\{\varphi\}\ \texttt{skip}\ \{\varphi\}}$$



$$\frac{(\sigma_1, s_1) \Downarrow \sigma_2 \qquad (\sigma \;, s\;) \quad \sigma}{(\sigma_1, s} \qquad \frac{\varphi \quad s \quad \alpha \qquad\qquad \alpha \quad s \quad \psi}{}$$



$$\frac{\sigma}{(\sigma, x := e) \Downarrow (\sigma : x \mapsto v)} \qquad \frac{}{\{\varphi[x := e]\}}$$



Continuing on, we can get rules for `if`, and `whi`

$$\frac{\{\varphi \land e\}\ s_1\ \{\psi\} \quad \{\varphi \land \neg e\}\ s_2\ \{\psi\}}{\{\varphi\}\ \texttt{if}\ e\ \texttt{then}\ s_1\ \texttt{else}\ s_2\ \texttt{fi}\ \{\psi\}} \qquad \frac{\{\varphi \land e\}\ s\ \{\varphi\}}{\{\varphi\}\ \texttt{while}\ e\ \texttt{do}\ s\ \texttt{od}\ \{\varphi \land \neg e\}}$$

## Consequence



There is one more rule, called the *rule of consequence*, that we need to insert ordinary logical reasoning i

# Consequence



There is one more rule, called the *rule of consequence*, that we need to insert ordinary logical reasoning i



This is the only rule that is *not* directed entirely by syntax. This mea proof need not look like a derivation tree. Instead we can sprinkle a our program and specially note uses of the consequence rule.

## Factorial Example

Let's verify the Factorial program using our Hoare rules:



$\{True\}$

**var** $i$ **var** $m$

**wh**



$i := i + 1;$

$m := m \times i$



**od**

$\{m = N!\}$

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

$\{\varphi\}$

$\{\varphi\}$

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

# Factorial Example

Let's verify the Factorial program using our Hoare rules:



$\{True\}$

**var** $i$ **var** $m$



**wh**

$i := i + 1;$



$m := m \times i$

**od** $\{m = i! \wedge i = N\}$

$\{m = N!\}$

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

$\{\varphi\}$

$\{\varphi\}$

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

# Factorial Example

Let's verify the Factorial program using our Hoare rules:



$\{True\}$

**var** $i$ **var** $m$

$\{m$

**wh**



$i := i + 1;$

$m := m \times i$



**od** $\{m = i! \wedge i = N\}$

$\{m = N!\}$

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

$\{\varphi\}$

$\{\varphi\}$

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

## Factorial Example

Let's verify the Factorial program using our Hoare rules:



$\{True\}$

**var** $i$ **var** $m$

$\{m$

**wh**

$i := i + 1;$

$m := m \times i$

$\{m = i!\}$

**od** $\{m = i! \wedge i = N\}$

$\{m = N!\}$

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

$\{\varphi\}$

$\{\varphi\}$

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

# Factorial Example

Let's verify the Factorial program using our Hoare rules:



$\{True\}$

**var** $i$ **var** $m$

$\{m$

**wh**



$i := i + 1;$



$m := m \times i$

$\{m = i!\}$

**od** $\{m = i! \land i = N\}$

$\{m = N!\}$

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

$\{\varphi\}$

$\{\varphi\}$

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

## Factorial Example

Let's verify the Factorial program using our Hoare rules:



$\{\mathit{True}\}$
**var** $i$ **var** $m$

$\{m$
**wh**

$i := i + 1;$
$\{m \times i = i!\}$
$m := m \times i$
$\{m = i!\}$
**od** $\{m = i! \wedge i = N\}$
$\{m = N!\}$

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

$\{\varphi\}$

$\{\varphi\}$

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

# Factorial Example

Let's verify the Factorial program using our Hoare rules:



$\{True\}$

**var** $i$ **var** $m$

$\{m$

**wh**

$\{m \times (i+1) = (i+1)!\}$

$i := i + 1;$

$\{m \times i = i!\}$

$m := m \times i$

$\{m = i!\}$

**od** $\{m = i! \wedge i = N\}$

$\{m = N!\}$

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



$\{\varphi\}$

$\{\varphi\}$



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

# Factorial Example

Let's verify the Factorial program using our Hoare rules:



$\{True\}$
**var** $i$ **var** $m$

$\{m$
**wh**

$\{m \times (i+1) = (i+1)!\}$
$i := i + 1;$
$\{m \times i = i!\}$
$m := m \times i$
$\{m = i!\}$
**od** $\{m = i! \wedge i = N\}$
$\{m = N!\}$

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



$\{\varphi\}$

$\{\varphi\}$



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

note: $(i+1)! = i! \times (i+1)$

# Factorial Example

Let's verify the Factorial program using our Hoare rules:



$\{\mathit{True}\}$

**var** $i$ **var** $m$

$\{m$

**wh**

$\quad\{m \times (i+1) = (i+1)!\}$

$\quad i := i + 1;$

$\quad\{m \times i = i!\}$

$\quad m := m \times i$

$\quad\{m = i!\}$

**od** $\{m = i! \wedge i = N\}$

$\{m = N!\}$

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

$$\frac{\{\varphi\}}{}$$

$$\{\varphi\}$$

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



note: $(i+1)! = i! \times (i+1)$

# Factorial Example

Let's verify the Factorial program using our Hoare rules:



$\{True\}$

**var** $i$ **var** $m$

$\{1$

$\{m$

**wh**

$\{m \times (i+1) = (i+1)!\}$

$i := i + 1;$

$\{m \times i = i!\}$

$m := m \times i$

$\{m = i!\}$

**od** $\{m = i! \land i = N\}$

$\{m = N!\}$

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



$\{\varphi\}$

$\{\varphi\}$



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

note: $(i+1)! = i! \times (i+1)$

# Factorial Example

Let's verify the Factorial program using our Hoare rules:



$\{True\}$
**var** $i$ **var** $m$

$\{1$
$\{m$



**wh**
$\{m \times (i+1) = (i+1)!\}$
$i := i + 1;$
$\{m \times i = i!\}$
$m := m \times i$
$\{m = i!\}$
**od** $\{m = i! \wedge i = N\}$
$\{m = N!\}$



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

$\{\varphi\}$

$\{\varphi\}$

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

note: $(i+1)! = i! \times (i+1)$

# Factorial Example

Let's verify the Factorial program using our Hoare rules:



$\{True\}$

**var** $i$ **var** $m$

$\{1$

$\{1$

$\{m$

**wh**



$\{m \times (i+1) = (i+1)!\}$

$i := i + 1;$

$\{m \times i = i!\}$

$m := m \times i$



$\{m = i!\}$

**od** $\{m = i! \wedge i = N\}$

$\{m = N!\}$

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

$\{\varphi\}$

$\{\varphi\}$

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

note: $(i+1)! = i! \times (i+1)$