

IMP: a simple imperative language

CS 152 (Spring 2021)

Harvard University

Tuesday, February 9, 2021

IMP syntax

- ▶ arithmetic expressions

$$a \in \mathbf{Aexp}$$

- ▶ boolean expressions

$$b \in \mathbf{Bexp}$$

- ▶ commands

$$c \in \mathbf{Com}$$

IMP syntax

$a ::= x \mid n \mid a_1 + a_2 \mid a_1 \times a_2$
 $b ::= \mathbf{true} \mid \mathbf{false} \mid a_1 < a_2$
 $c ::= \mathbf{skip} \mid x := a \mid c_1; c_2$
 $\mid \mathbf{if } b \mathbf{ then } c_1 \mathbf{ else } c_2$
 $\mid \mathbf{while } b \mathbf{ do } c$

Small-step operational semantics

- ▶ configurations of the form:
 $\langle c, \sigma \rangle$, $\langle b, \sigma \rangle$, and $\langle a, \sigma \rangle$,
where σ is a store.
- ▶ final configurations of the form:
 $\langle \mathbf{skip}, \sigma \rangle$,
 $\langle \mathbf{true}, \sigma \rangle$, $\langle \mathbf{false}, \sigma \rangle$,
 $\langle n, \sigma \rangle$.

3 different small-step operational semantics relations

$$\longrightarrow_{\mathbf{Com}} \subseteq \mathbf{Com} \times \mathbf{Store} \times \mathbf{Com} \times \mathbf{Store}$$

$$\longrightarrow_{\mathbf{Bexp}} \subseteq \mathbf{Bexp} \times \mathbf{Store} \times \mathbf{Bexp} \times \mathbf{Store}$$

$$\longrightarrow_{\mathbf{Aexp}} \subseteq \mathbf{Aexp} \times \mathbf{Store} \times \mathbf{Aexp} \times \mathbf{Store}$$

Arithmetic expressions (1/2)

$$\frac{}{\langle x, \sigma \rangle \longrightarrow \langle n, \sigma \rangle} \text{ where } n = \sigma(x)$$

$$\frac{\langle a_1, \sigma \rangle \longrightarrow \langle a'_1, \sigma \rangle}{\langle a_1 + a_2, \sigma \rangle \longrightarrow \langle a'_1 + a_2, \sigma \rangle}$$
$$\frac{\langle a_2, \sigma \rangle \longrightarrow \langle a'_2, \sigma \rangle}{\langle n + a_2, \sigma \rangle \longrightarrow \langle n + a'_2, \sigma \rangle}$$

$$\frac{}{\langle n + m, \sigma \rangle \longrightarrow \langle p, \sigma \rangle} \text{ where } p = n + m$$

Arithmetic expressions (2/2)

$$\frac{\frac{\frac{\langle a_1, \sigma \rangle \longrightarrow \langle a'_1, \sigma \rangle}{\langle a_1 \times a_2, \sigma \rangle \longrightarrow \langle a'_1 \times a_2, \sigma \rangle} \quad \langle a_2, \sigma \rangle \longrightarrow \langle a'_2, \sigma \rangle}{\langle n \times a_2, \sigma \rangle \longrightarrow \langle n \times a'_2, \sigma \rangle}}{\langle n \times m, \sigma \rangle \longrightarrow \langle p, \sigma \rangle} \text{ where } p = n \times m$$

Boolean expressions

$$\frac{\langle a_1, \sigma \rangle \longrightarrow \langle a'_1, \sigma \rangle}{\langle a_1 \langle a_2, \sigma \rangle \longrightarrow \langle a'_1 \langle a_2, \sigma \rangle \rangle}$$
$$\frac{\langle a_2, \sigma \rangle \longrightarrow \langle a'_2, \sigma \rangle}{\langle n \langle a_2, \sigma \rangle \longrightarrow \langle n \langle a'_2, \sigma \rangle \rangle}$$

$$\frac{}{\langle n \langle m, \sigma \rangle \longrightarrow \langle \mathbf{true}, \sigma \rangle} \text{ where } n < m$$

$$\frac{}{\langle n \langle m, \sigma \rangle \longrightarrow \langle \mathbf{false}, \sigma \rangle} \text{ where } n \geq m$$

Commands (1/3)

$$\frac{\langle a, \sigma \rangle \longrightarrow \langle a', \sigma \rangle}{\langle x := a, \sigma \rangle \longrightarrow \langle x := a', \sigma \rangle}$$

$$\langle x := n, \sigma \rangle \longrightarrow \langle \mathbf{skip}, \sigma[x \mapsto n] \rangle$$

$$\frac{\langle c_1, \sigma \rangle \longrightarrow \langle c'_1, \sigma' \rangle}{\langle c_1; c_2, \sigma \rangle \longrightarrow \langle c'_1; c_2, \sigma' \rangle}$$

$$\langle \mathbf{skip}; c_2, \sigma \rangle \longrightarrow \langle c_2, \sigma \rangle$$

Commands (2/3)

$$\frac{\langle b, \sigma \rangle \longrightarrow \langle b', \sigma \rangle}{\langle \text{if } b \text{ then } c_1 \text{ else } c_2, \sigma \rangle \longrightarrow \langle \text{if } b' \text{ then } c_1 \text{ else } c_2, \sigma \rangle}$$

$$\langle \text{if true then } c_1 \text{ else } c_2, \sigma \rangle \longrightarrow \langle c_1, \sigma \rangle$$

$$\langle \text{if false then } c_1 \text{ else } c_2, \sigma \rangle \longrightarrow \langle c_2, \sigma \rangle$$

Commands (3/3)

$\langle \text{while } b \text{ do } c, \sigma \rangle \longrightarrow$
 $\langle \text{if } b \text{ then } (c; \text{while } b \text{ do } c) \text{ else skip}, \sigma \rangle$

Small-step execution

$\langle \text{foo} := 3; \text{while } \text{foo} < 4 \text{ do } \text{foo} := \text{foo} + 5, \sigma \rangle$
 $\rightarrow \langle \text{skip}; \text{while } \text{foo} < 4 \text{ do } \text{foo} := \text{foo} + 5, \sigma' \rangle$ where $\sigma' = \sigma[\text{foo} \mapsto 3]$
 $\rightarrow \langle \text{while } \text{foo} < 4 \text{ do } \text{foo} := \text{foo} + 5, \sigma' \rangle$
 $\rightarrow \langle \text{if } \text{foo} < 4 \text{ then } (\text{foo} := \text{foo} + 5; W) \text{ else skip}, \sigma' \rangle$
 $\rightarrow \langle \text{if } 3 < 4 \text{ then } (\text{foo} := \text{foo} + 5; W) \text{ else skip}, \sigma' \rangle$
 $\rightarrow \langle \text{if true then } (\text{foo} := \text{foo} + 5; W) \text{ else skip}, \sigma' \rangle$
 $\rightarrow \langle \text{foo} := \text{foo} + 5; \text{while } \text{foo} < 4 \text{ do } \text{foo} := \text{foo} + 5, \sigma' \rangle$
 $\rightarrow \langle \text{foo} := 3 + 5; \text{while } \text{foo} < 4 \text{ do } \text{foo} := \text{foo} + 5, \sigma' \rangle$
 $\rightarrow \langle \text{foo} := 8; \text{while } \text{foo} < 4 \text{ do } \text{foo} := \text{foo} + 5, \sigma' \rangle$
 $\rightarrow \langle \text{skip}; \text{while } \text{foo} < 4 \text{ do } \text{foo} := \text{foo} + 5, \sigma'' \rangle$ where $\sigma'' = \sigma'[\text{foo} \mapsto 8]$
 $\rightarrow \langle \text{while } \text{foo} < 4 \text{ do } \text{foo} := \text{foo} + 5, \sigma'' \rangle$
 $\rightarrow \langle \text{if } \text{foo} < 4 \text{ then } (\text{foo} := \text{foo} + 5; W) \text{ else skip}, \sigma'' \rangle$
 $\rightarrow \langle \text{if } 8 < 4 \text{ then } (\text{foo} := \text{foo} + 5; W) \text{ else skip}, \sigma'' \rangle$
 $\rightarrow \langle \text{if false then } (\text{foo} := \text{foo} + 5; W) \text{ else skip}, \sigma'' \rangle$
 $\rightarrow \langle \text{skip}, \sigma'' \rangle$

(where W is an abbreviation for the while loop **while** $\text{foo} < 4$ **do** $\text{foo} := \text{foo} + 5$).

Large-step operational semantics

$$\Downarrow_{\mathbf{Aexp}} \subseteq \mathbf{Aexp} \times \mathbf{Store} \times \mathbf{Int}$$

$$\Downarrow_{\mathbf{Bexp}} \subseteq \mathbf{Bexp} \times \mathbf{Store} \times \mathbf{Bool}$$

$$\Downarrow_{\mathbf{Com}} \subseteq \mathbf{Com} \times \mathbf{Store} \times \mathbf{Store}$$

Arithmetic expressions

$$\frac{}{\langle n, \sigma \rangle \Downarrow n} \quad \frac{}{\langle x, \sigma \rangle \Downarrow n} \text{ where } \sigma(x) = n$$

$$\frac{\langle a_1, \sigma \rangle \Downarrow n_1 \quad \langle a_2, \sigma \rangle \Downarrow n_2}{\langle a_1 + a_2, \sigma \rangle \Downarrow n} \text{ where } n = n_1 + n_2$$

$$\frac{\langle a_1, \sigma \rangle \Downarrow n_1 \quad \langle a_2, \sigma \rangle \Downarrow n_2}{\langle a_1 \times a_2, \sigma \rangle \Downarrow n} \text{ where } n = n_1 \times n_2$$

Boolean expressions

$\langle \mathbf{true}, \sigma \rangle \Downarrow \mathbf{true}$

$\langle \mathbf{false}, \sigma \rangle \Downarrow \mathbf{false}$

$\langle a_1, \sigma \rangle \Downarrow n_1 \quad \langle a_2, \sigma \rangle \Downarrow n_2$ where $n_1 < n_2$
 $\langle a_1 < a_2, \sigma \rangle \Downarrow \mathbf{true}$

$\langle a_1, \sigma \rangle \Downarrow n_1 \quad \langle a_2, \sigma \rangle \Downarrow n_2$ where $n_1 \geq n_2$
 $\langle a_1 < a_2, \sigma \rangle \Downarrow \mathbf{false}$

Commands (1/2)

$$\text{SKIP} \frac{}{\langle \mathbf{skip}, \sigma \rangle \Downarrow \sigma}$$

$$\text{ASG} \frac{\langle a, \sigma \rangle \Downarrow n}{\langle x := a, \sigma \rangle \Downarrow \sigma[x \mapsto n]}$$

$$\text{SEQ} \frac{\langle c_1, \sigma \rangle \Downarrow \sigma' \quad \langle c_2, \sigma' \rangle \Downarrow \sigma''}{\langle c_1; c_2, \sigma \rangle \Downarrow \sigma''}$$

$$\text{IF-T} \frac{\langle b, \sigma \rangle \Downarrow \mathbf{true} \quad \langle c_1, \sigma \rangle \Downarrow \sigma'}{\langle \mathbf{if } b \mathbf{ then } c_1 \mathbf{ else } c_2, \sigma \rangle \Downarrow \sigma'}$$

$$\text{IF-F} \frac{\langle b, \sigma \rangle \Downarrow \mathbf{false} \quad \langle c_2, \sigma \rangle \Downarrow \sigma'}{\langle \mathbf{if } b \mathbf{ then } c_1 \mathbf{ else } c_2, \sigma \rangle \Downarrow \sigma'}$$

Commands (2/2)

$$\text{WHILE-F} \frac{\langle b, \sigma \rangle \Downarrow \mathbf{false}}{\langle \mathbf{while } b \mathbf{ do } c, \sigma \rangle \Downarrow \sigma}$$

$$\text{WHILE-T} \frac{\begin{array}{l} \langle b, \sigma \rangle \Downarrow \mathbf{true} \quad \langle c, \sigma \rangle \Downarrow \sigma' \\ \langle \mathbf{while } b \mathbf{ do } c, \sigma' \rangle \Downarrow \sigma'' \end{array}}{\langle \mathbf{while } b \mathbf{ do } c, \sigma \rangle \Downarrow \sigma''}$$

Command equivalence

The small-step operational semantics suggest that the loop **while** b **do** c should be equivalent to the command **if** b **then** (c ; **while** b **do** c) **else skip**. Can we show that this indeed the case when the language is defined using the above large-step evaluation?

Equivalence of commands

Two commands c and c' are equivalent (written $c \sim c'$) if, for any stores σ and σ' , we have

$$\langle c, \sigma \rangle \Downarrow \sigma' \iff \langle c', \sigma \rangle \Downarrow \sigma'.$$

Theorem

For all $b \in \mathbf{Bexp}$ and $c \in \mathbf{Com}$ we have

while b do $c \sim \text{if } b \text{ then } (c; \text{while } b \text{ do } c) \text{ else skip.}$

Proof

Let W be an abbreviation for **while** b **do** c . We want to show that for all stores σ, σ' , we have:

$$\langle W, \sigma \rangle \Downarrow \sigma' \iff \mathbf{if\ } b \mathbf{\ then\ } (c; W) \mathbf{\ else\ skip} \Downarrow \sigma'$$

For this, we must show that both directions (\implies and \impliedby) hold. We'll show only direction \implies ; the other is similar.

Assume that σ and σ' are stores such that $\langle W, \sigma \rangle \Downarrow \sigma'$. It means that there is some derivation that proves for this fact. Inspecting the evaluation rules, we see that there are two possible rules whose conclusions match this fact: **WHILE-F** and **WHILE-T**. We analyze each of them in turn.

Case WHILE-F (1/2)

The derivation must look like the following.

$$\text{WHILE-F} \frac{\begin{array}{c} :^1 \\ \hline \langle b, \sigma \rangle \Downarrow \mathbf{false} \end{array}}{\langle W, \sigma \rangle \Downarrow \sigma}$$

Here, we use $:^1$ to refer to the derivation of $\langle b, \sigma \rangle \Downarrow \mathbf{false}$. Note that in this case, $\sigma' = \sigma$.

Case WHILE-F (2/2)

We can use \vdash^1 to derive a proof tree showing that the evaluation of **if** b **then** $(c; W)$ **else skip** yields the same final state σ :

$$\text{IF-F} \frac{\frac{\vdash^1}{\langle b, \sigma \rangle \Downarrow \text{false}} \quad \text{SKIP} \frac{}{\langle \text{skip}, \sigma \rangle \Downarrow \sigma}}{\langle \text{if } b \text{ then } (c; W) \text{ else skip}, \sigma \rangle \Downarrow \sigma}$$

Case WHILE-T (1/2)

In this case, the derivation has the following form.

$$\text{WHILE-T} \frac{\frac{\frac{\text{:2}}{\langle b, \sigma \rangle \Downarrow \mathbf{true}}}{\text{:3}} \quad \frac{\text{:4}}{\langle W, \sigma'' \rangle \Downarrow \sigma'}}{\langle c, \sigma \rangle \Downarrow \sigma''}}{\langle W, \sigma \rangle \Downarrow \sigma'}$$

Case WHILE-T (2/2)

We can use subderivations \vdash^2 , \vdash^3 , and \vdash^4 to show that the evaluation of **if** b **then** $(c; W)$ **else skip** yields the same final state σ .

$$\text{IF-T} \frac{\text{SEQ} \frac{\frac{\frac{\vdash^2}{\langle b, \sigma \rangle \Downarrow \mathbf{true}}}{\vdash^3} \langle c, \sigma \rangle \Downarrow \sigma''}{\vdash^4} \langle W, \sigma'' \rangle \Downarrow \sigma'}{\langle c; W, \sigma \rangle \Downarrow \sigma'}}{\langle \mathbf{if} \ b \ \mathbf{then} \ (c; W) \ \mathbf{else} \ \mathbf{skip}, \sigma \rangle \Downarrow \sigma'}}$$

Break

- ▶ Contrast the design of While in small-step and large-step. Can one style be used for the other? Can you mix small-step and large-step?
- ▶ How do you prove that **while true do skip** never terminates? In small-step? In large-step?
- ▶ Sketch proof for large-step determinism of commands:

For all commands $c \in \mathbf{Com}$ and stores

$\sigma, \sigma_1, \sigma_2 \in \mathbf{Store}$,

if $\langle c, \sigma \rangle \Downarrow \sigma_1$ and $\langle c, \sigma \rangle \Downarrow \sigma_2$ then

$\sigma_1 = \sigma_2$.

Scratchpad

Alternative large-step rule for While

$$\frac{\langle \mathbf{if } b \mathbf{ then } (c; \mathbf{while } b \mathbf{ do } c) \mathbf{ else skip}, \sigma \rangle \Downarrow \sigma'}{\langle \mathbf{while } b \mathbf{ do } c, \sigma \rangle \Downarrow \sigma'}$$

Determinism

For all commands $c \in \mathbf{Com}$ and stores

$\sigma, \sigma_1, \sigma_2 \in \mathbf{Store}$,

if $\langle c, \sigma \rangle \Downarrow \sigma_1$ and $\langle c, \sigma \rangle \Downarrow \sigma_2$ then $\sigma_1 = \sigma_2$.

Determinism

For all commands $c \in \mathbf{Com}$ and stores

$\sigma, \sigma_1, \sigma_2 \in \mathbf{Store}$,

if $\langle c, \sigma \rangle \Downarrow \sigma_1$ and $\langle c, \sigma \rangle \Downarrow \sigma_2$ then $\sigma_1 = \sigma_2$.

Proof Sketch for Determinism

By induction on the derivation of $\langle c, \sigma \rangle \Downarrow \sigma_1$.
The inductive hypothesis P is

$$P(\langle c, \sigma \rangle \Downarrow \sigma_1) = \forall \sigma_2 \in \mathbf{Store}, \\ \text{if } \langle c, \sigma \rangle \Downarrow \sigma_2 \text{ then } \sigma_1 = \sigma_2.$$

We have a derivation for $\langle c, \sigma \rangle \Downarrow \sigma_1$, for some c , σ , and σ_1 . Assume that the inductive hypothesis holds for any subderivation $\langle c', \sigma' \rangle \Downarrow \sigma''$ used in the derivation of $\langle c, \sigma \rangle \Downarrow \sigma_1$.

Assume that for some σ_2 we have $\langle c, \sigma \rangle \Downarrow \sigma_2$.

We need to show that $\sigma_1 = \sigma_2$.

Case IF-T (1/2)

$$\text{IF-T} \frac{\frac{\vdots}{\langle b, \sigma \rangle \Downarrow \mathbf{true}} \quad \frac{\vdots}{\langle c_1, \sigma \rangle \Downarrow \sigma_1}}{\langle \mathbf{if } b \mathbf{ then } c_1 \mathbf{ else } c_2, \sigma \rangle \Downarrow \sigma_1},$$

and we have $c \equiv \mathbf{if } b \mathbf{ then } c_1 \mathbf{ else } c_2$.

The last rule used in the derivation of $\langle c, \sigma \rangle \Downarrow \sigma_2$ must be either IF-T or IF-F (since these are the only rules that can be used to derive a conclusion of the form $\langle \mathbf{if } b \mathbf{ then } c_1 \mathbf{ else } c_2, \sigma \rangle \Downarrow \sigma_2$). But by the determinism of boolean expressions, we must have $\langle b, \sigma \rangle \Downarrow \mathbf{true}$, and so the derivation of $\langle c, \sigma \rangle \Downarrow \sigma_2$ must have the following form...

Case IF-T (2/2)

$$\text{IF-T} \frac{\frac{\vdots}{\langle b, \sigma \rangle \Downarrow \mathbf{true}} \quad \frac{\vdots}{\langle c_1, \sigma \rangle \Downarrow \sigma_2}}{\langle \mathbf{if } b \mathbf{ then } c_1 \mathbf{ else } c_2, \sigma \rangle \Downarrow \sigma_2}$$

The result holds by the inductive hypothesis applied

to the derivation $\frac{\vdots}{\langle c_1, \sigma \rangle \Downarrow \sigma_1}$.

Case WHILE-T (1/3)

Here we have

$$\text{WHILE-T} \frac{\frac{\frac{\vdots}{\langle b, \sigma \rangle \Downarrow \mathbf{true}}{\vdots}}{\langle c_1, \sigma \rangle \Downarrow \sigma'}}{\vdots}}{\langle c, \sigma' \rangle \Downarrow \sigma_1} \frac{}{\langle \mathbf{while } b \mathbf{ do } c_1, \sigma \rangle \Downarrow \sigma_1 ,}$$

and we have $c \equiv \mathbf{while } b \mathbf{ do } c_1$. The last rule used in the derivation of $\langle c, \sigma \rangle \Downarrow \sigma_2$ must also be WHILE-T (by the determinism of boolean expressions), and so we have...

Case WHILE-T (2/3)

$$\text{WHILE-T} \frac{\frac{\frac{\vdots}{\langle b, \sigma \rangle \Downarrow \mathbf{true}}{\vdots}}{\langle c_1, \sigma \rangle \Downarrow \sigma''}}{\langle c, \sigma'' \rangle \Downarrow \sigma_2}}{\langle \mathbf{while} \ b \ \mathbf{do} \ c_1, \sigma \rangle \Downarrow \sigma_2} .$$

By the inductive hypothesis applied to the

derivation $\frac{\vdots}{\langle c_1, \sigma \rangle \Downarrow \sigma'}$, we have $\sigma' = \sigma'' \dots$

Case WHILE-T (3/3)

By another application of the inductive hypothesis,

⋮

to the derivation $\frac{}{\langle c, \sigma' \rangle \Downarrow \sigma_1}$, we have $\sigma_1 = \sigma_2$ and the result holds.

Comment on Case WHILE-T

Even though the command $c \equiv \mathbf{while} \ b \ \mathbf{do} \ c_1$ appears in the derivation of $\langle \mathbf{while} \ b \ \mathbf{do} \ c_1, \sigma \rangle \Downarrow \sigma_1$, we do not run in to problems, as the induction is over the *derivation*, not over the structure of the command.