Proofs are Programs

IMP - Simple imperative programs

Simple Imperative Programs

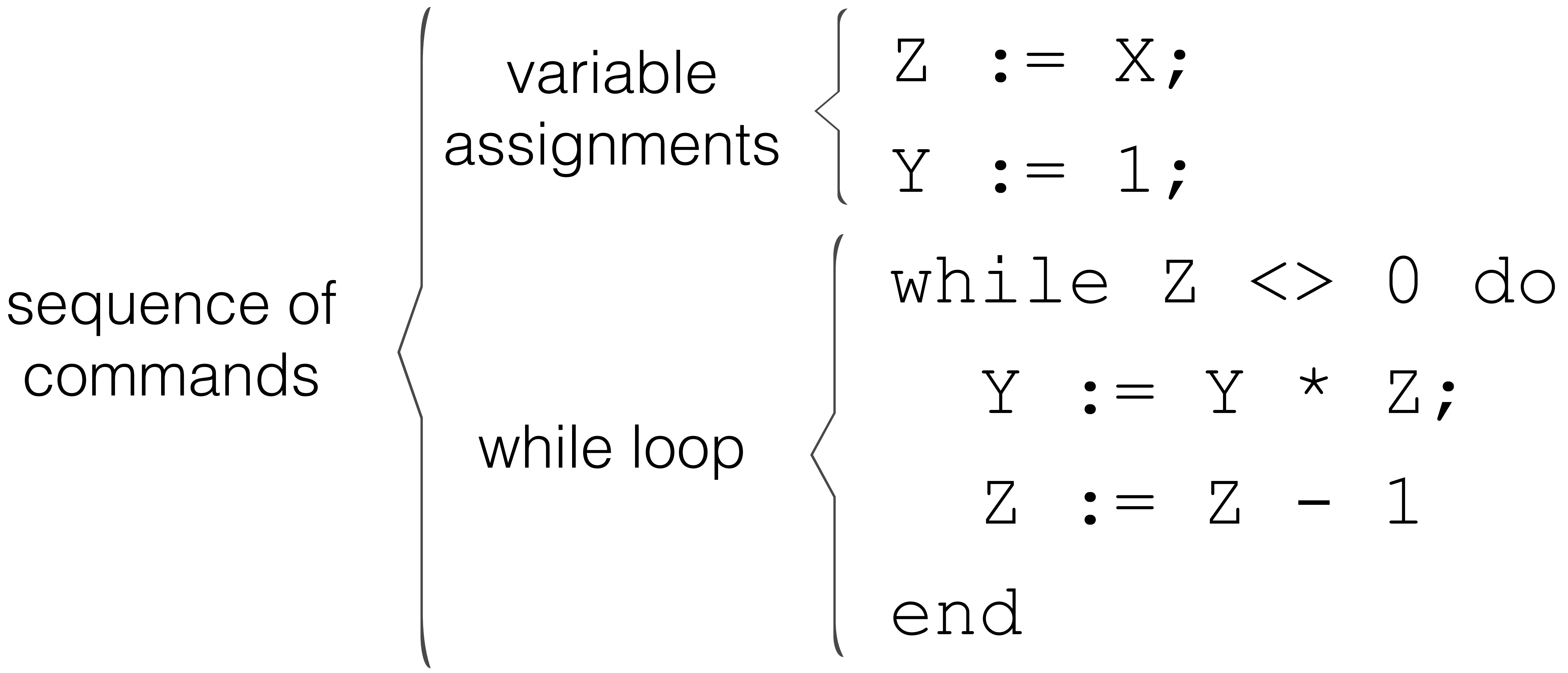
```
Z := X;
Y := 1;
while Z <> 0 do
  Y := Y * Z;
  Z := Z - 1
end
```

Simple Imperative Programs

sequence of commands

```
Z := X;
Y := 1;
while Z <> 0 do
  Y := Y * Z;
  Z := Z - 1
end
```

Simple Imperative Programs



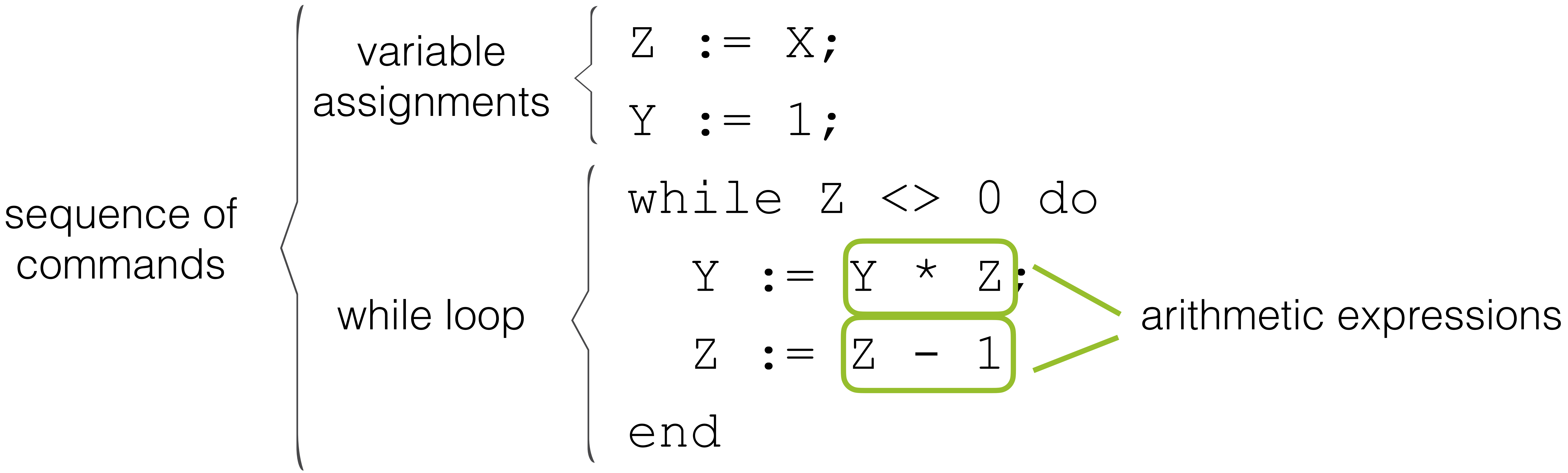
sequence of commands

variable assignments

while loop

```
Z := X;
Y := 1;
while Z <> 0 do
  Y := Y * Z;
  Z := Z - 1
end
```

Simple Imperative Programs



sequence of commands

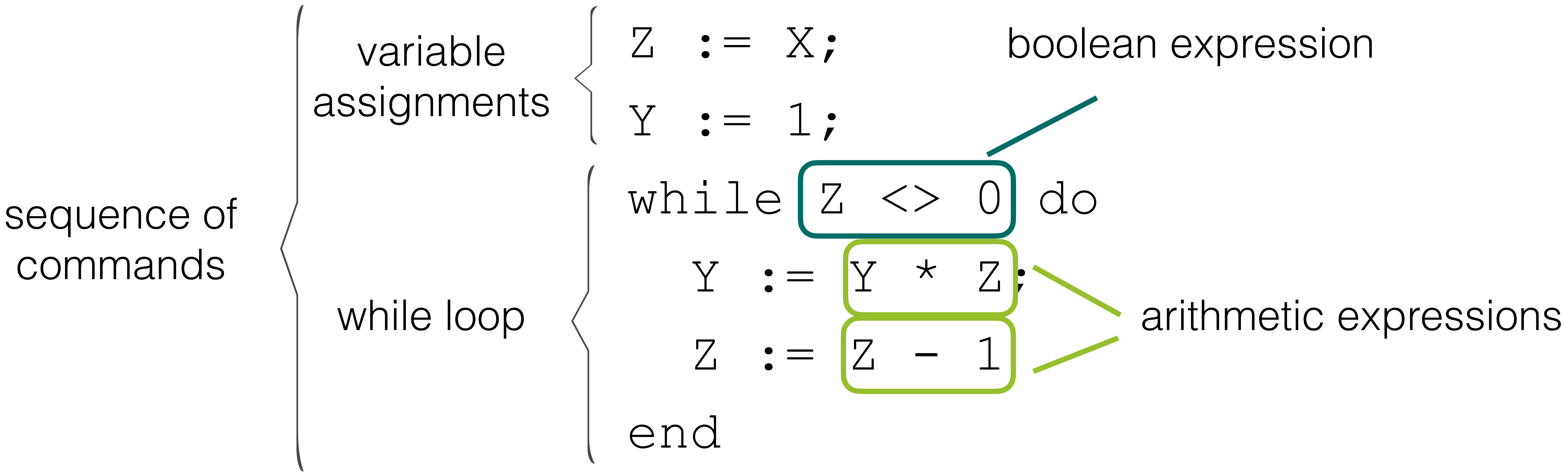
variable assignments

while loop

```
Z := X;
Y := 1;
while Z <> 0 do
  Y := Y * Z;
  Z := Z - 1
end
```

arithmetic expressions

Simple Imperative Programs



```
Z := X;
Y := 1;
while Z <> 0 do
   Y := Y * Z;
   Z := Z - 1
end
```

sequence of commands

variable assignments

while loop

boolean expression

arithmetic expressions

Expressions

Arithmetic Expressions (BNF)

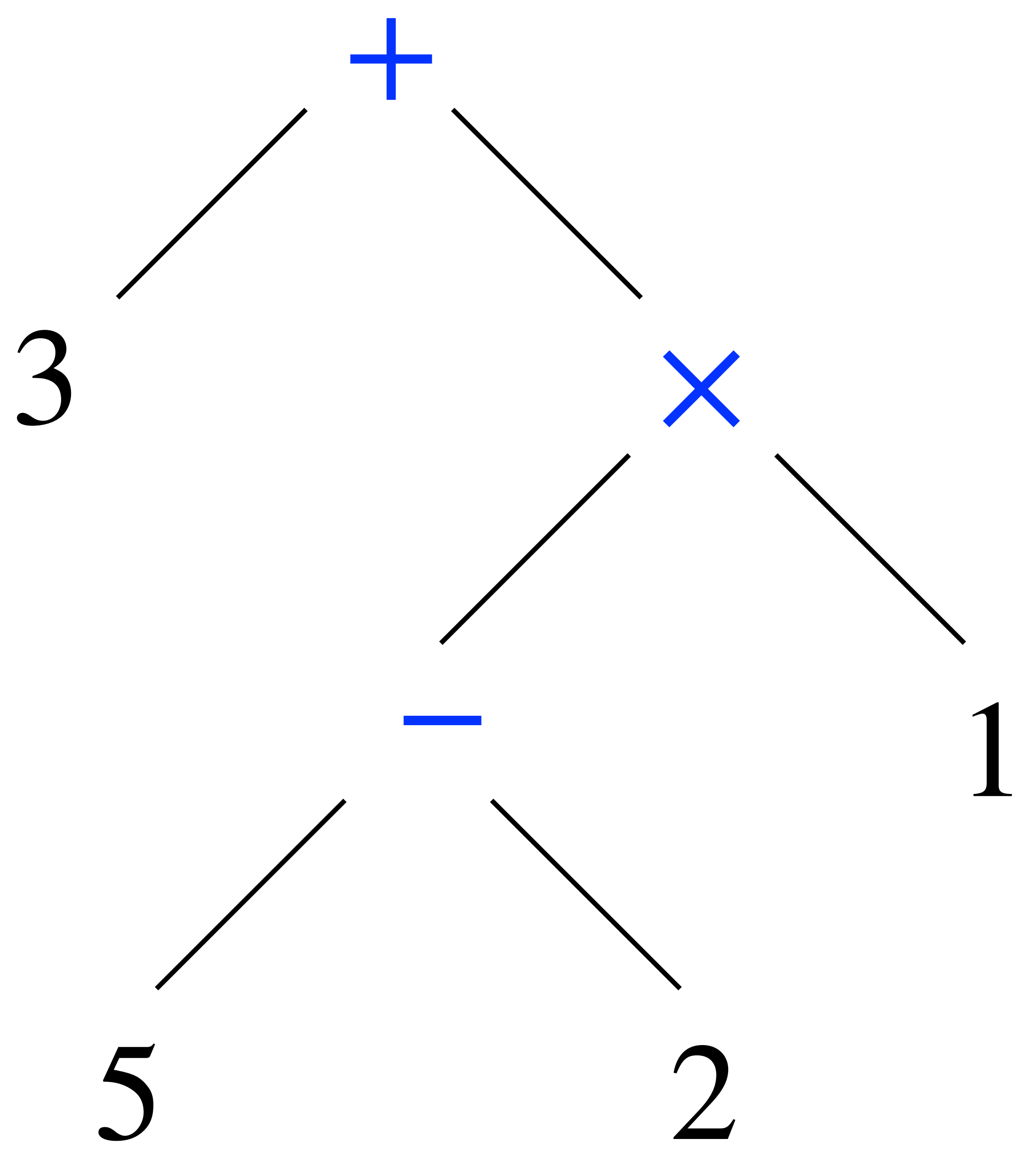
$$
\begin{aligned}
a &:= n \\
&\quad | \; a + a \\
&\quad | \; a - a \\
&\quad | \; a \times a
\end{aligned}
$$

$$n \in \mathbb{N}$$

Arithmetic Expressions (BNF)

$$\begin{aligned} a :=\ & n \\ & \mid a + a \\ & \mid a - a \\ & \mid a \times a \end{aligned}$$

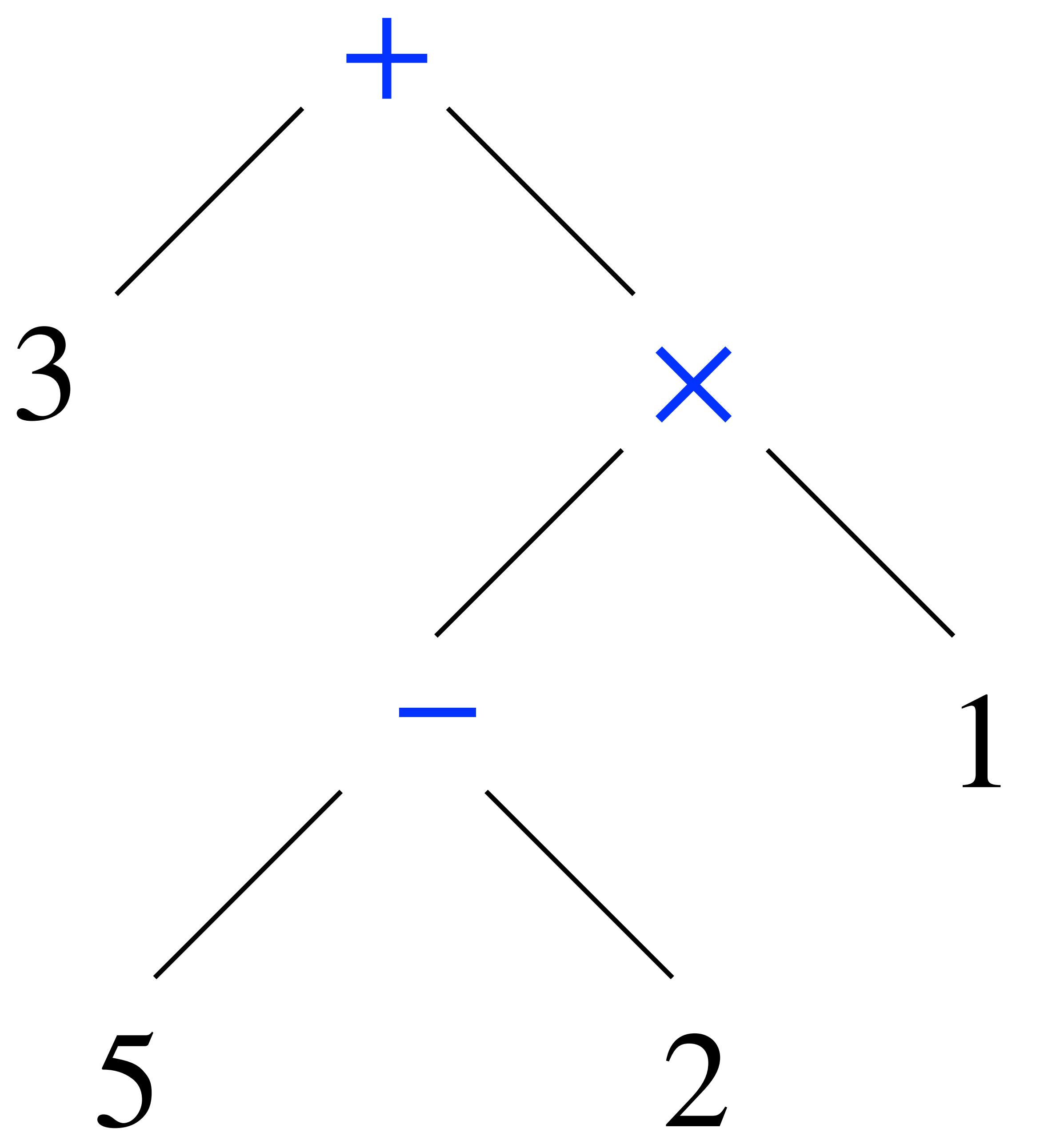
$$n \in \mathbb{N}$$



Arithmetic Expressions (BNF)

$$
\begin{aligned}
a :=\ & n \\
& \mid a + a \\
& \mid a - a \\
& \mid a \times a
\end{aligned}
$$

$$n \in \mathbb{N}$$



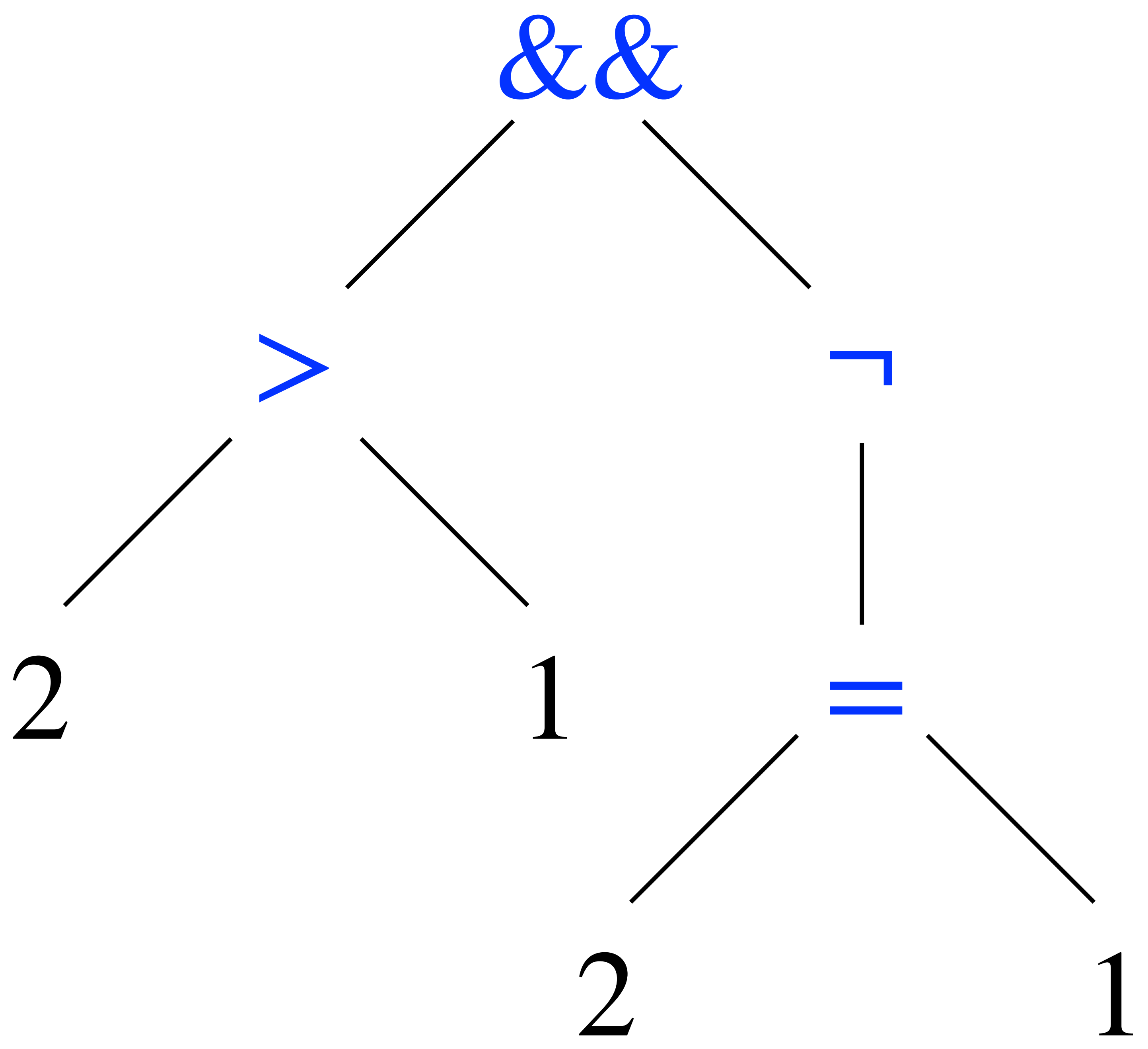
$$3 + (5 - 2) \times 1$$

Boolean Expressions

$$
\begin{aligned}
b := {} & \textsf{true} \\
& | \,\textsf{false} \\
& | \, a = a \\
& | \, a \neq a \\
& | \, a \leq a \\
& | \, a > a \\
& | \, \neg b \\
& | \, b \&\& b
\end{aligned}
$$

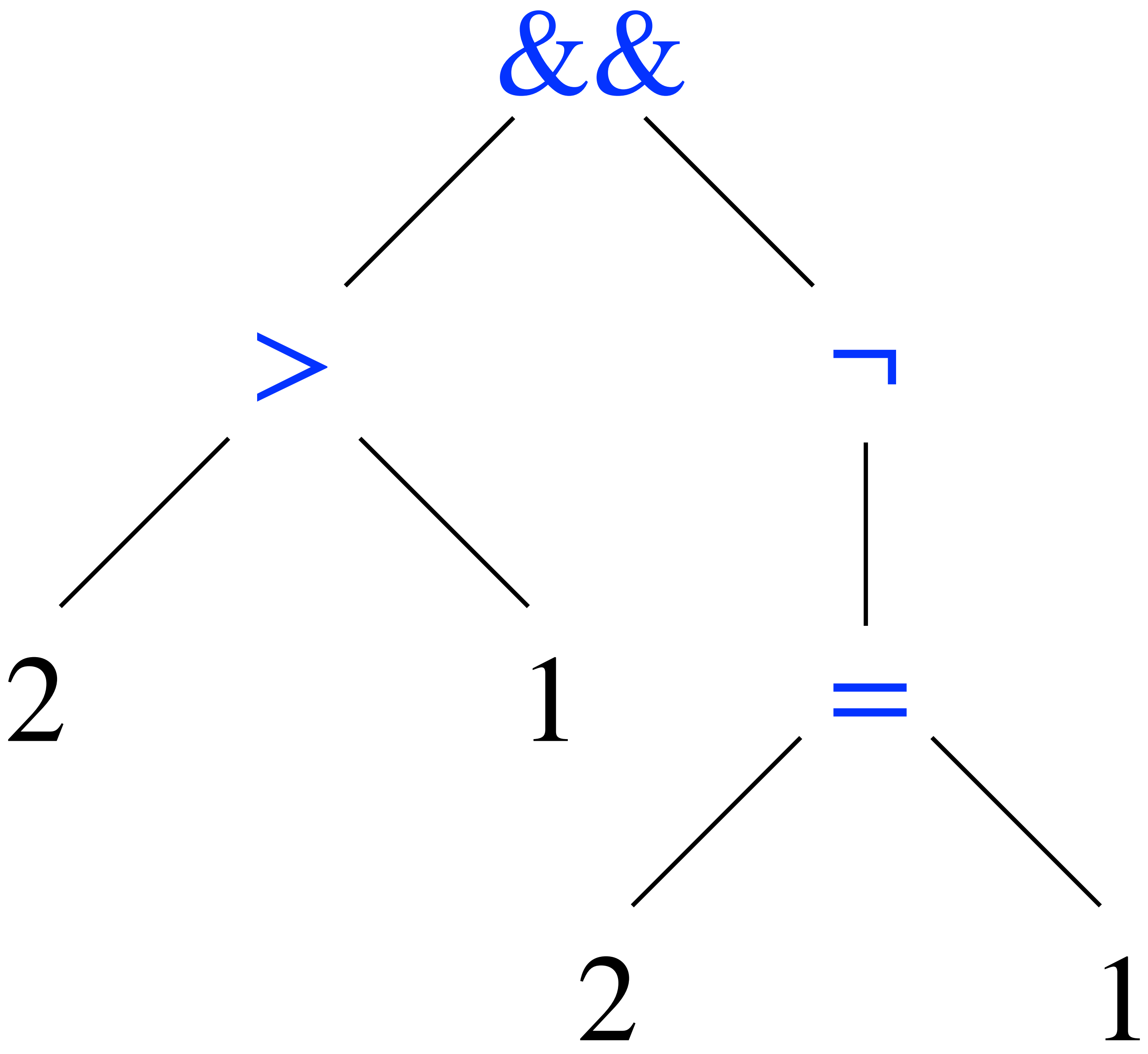
Boolean Expressions

$$
\begin{aligned}
b := \ & \mathsf{true} \\
& \mid \mathsf{false} \\
& \mid a = a \\
& \mid a \neq a \\
& \mid a \leq a \\
& \mid a > a \\
& \mid \neg b \\
& \mid b \&\& b
\end{aligned}
$$



Boolean Expressions

$$
\begin{aligned}
b := \; & \text{true} \\
& \mid \text{false} \\
& \mid a = a \\
& \mid a \neq a \\
& \mid a \leq a \\
& \mid a > a \\
& \mid \neg b \\
& \mid b \&\& b
\end{aligned}
$$



$2 > 1\& \& \neg(2 = 1)$

Evaluating Expressions

Quiz 1

http://etc.ch/VAiu

What does the following expression evaluate to?

```
    aeval (APlus (ANum 3) (AMinus (ANum 4) (ANum 1)))
```

1) `true`

2) `false`

3) 0

4) 3

5) 6

More Tactics… and Tacticals!

Expression Evaluation as a Relation

$e \Rightarrow n$ “expression e evaluates to number n”

$$\frac{}{n \Rightarrow n}$$

$$\frac{e_1 \Rightarrow n_1 \qquad e_2 \Rightarrow n_2}{e_1 + e_2 \Rightarrow n_1 + n_2}$$

$$\frac{e_1 \Rightarrow n_1 \qquad e_2 \Rightarrow n_2}{e_1 \times e_2 \Rightarrow n_1 \times n_2}$$

$$\frac{e_1 \Rightarrow n_1 \qquad e_2 \Rightarrow n_2}{e_1 - e_2 \Rightarrow n_1 - n_2}$$

Computational vs. Relational Definitions

Expression Evaluation as a Relation

$e \Rightarrow n$ “expression e evaluates to number n”

$$\frac{}{n \Rightarrow n} \qquad \frac{e_1 \Rightarrow n_1 \qquad e_2 \Rightarrow n_2}{e_1 + e_2 \Rightarrow n_1 + n_2}$$

$$\frac{e_1 \Rightarrow n_1 \qquad e_2 \Rightarrow n_2}{e_1 \times e_2 \Rightarrow n_1 \times n_2} \qquad \frac{e_1 \Rightarrow n_1 \qquad e_2 \Rightarrow n_2}{e_1 - e_2 \Rightarrow n_1 - n_2}$$

Expression Evaluation as a Relation

$e \Rightarrow n$ “expression e evaluates to number n”

makes relation partial

$$\frac{}{n \Rightarrow n}$$

$$\frac{e_1 \Rightarrow n_1 \qquad e_2 \Rightarrow n_2}{e_1 + e_2 \Rightarrow n_1 + n_2}$$

$$\boxed{\frac{e_1 \Rightarrow n_1 \qquad e_2 \Rightarrow n_2 \qquad n_2 > 0 \qquad n_2 \times n_3 = n_1}{e_1 \div e_2 \Rightarrow n_3}}$$

$$\frac{e_1 \Rightarrow n_1 \qquad e_2 \Rightarrow n_2}{e_1 \times e_2 \Rightarrow n_1 \times n_2}$$

$$\frac{e_1 \Rightarrow n_1 \qquad e_2 \Rightarrow n_2}{e_1 - e_2 \Rightarrow n_1 - n_2}$$

Expression Evaluation as a Relation

$e \Rightarrow n$ “expression e evaluates to number n”

makes relation partial

$$\frac{}{n \Rightarrow n} \qquad \frac{e_1 \Rightarrow n_1 \qquad e_2 \Rightarrow n_2}{e_1 + e_2 \Rightarrow n_1 + n_2} \qquad \boxed{\frac{e_1 \Rightarrow n_1 \quad e_2 \Rightarrow n_2 \quad n_2 > 0 \quad n_2 \times n_3 = n_1}{e_1 \div e_2 \Rightarrow n_3}}$$

$$\frac{e_1 \Rightarrow n_1 \qquad e_2 \Rightarrow n_2}{e_1 \times e_2 \Rightarrow n_1 \times n_2} \qquad \frac{e_1 \Rightarrow n_1 \qquad e_2 \Rightarrow n_2}{e_1 - e_2 \Rightarrow n_1 - n_2} \qquad \boxed{\frac{}{? \Rightarrow n}}$$

makes relation non-deterministic

Expressions with Variables

Arithmetic Expressions with Variables

$$
\begin{aligned}
a :=\ & n \\
& \mid x \\
& \mid a + a \\
& \mid a - a \\
& \mid a \times a \\
n \in\ & \mathbb{N} \\
x \in\ & ?
\end{aligned}
$$

Arithmetic Expressions with Variables

$$
\begin{aligned}
a := {} & n \\
& | \, x \\
& | \, a + a \\
& | \, a - a \\
& | \, a \times a \\
n \in {} & \mathbb{N} \\
x \in {} & ?
\end{aligned}
$$

$\texttt{aeval}\ x = ?$

Arithmetic Expressions with Variables

$$\begin{aligned} a := {} & n \\ & \mid x \\ & \mid a + a \\ & \mid a - a \\ & \mid a \times a \\ n & \in \mathbb{N} \\ x & \in ? \end{aligned}$$

$$\texttt{aeval} \underbrace{\quad}_{\texttt{st : state}} x = ?$$

```
Definition state := total_map nat.

Definition total_map (A : Type) := string -> A.
```

Maps

```
Definition total_map (A : Type) := string -> A.
```

```
t_empty: forall {A:Type}, A -> total_map A

t_update: forall {A:Type}, total_map A -> string -> A -> total_map A
```

Maps

```
Definition total_map (A : Type) := string -> A.
```

```
t_empty: forall {A:Type}, A -> total_map A

t_update: forall {A:Type}, total_map A -> string -> A -> total_map A
```

```
Notation "x '!->' v ';' m" := (t_update m x v)

Notation "'_' '!->' v" := (t_empty v)
```

Maps

```
Definition total_map (A : Type) := string -> A.
```

```
t_empty: forall {A:Type}, A -> total_map A

t_update: forall {A:Type}, total_map A -> string -> A -> total_map A
```

```
Notation "x '!->' v ';' m" := (t_update m x v)

Notation "'_' '!->' v" := (t_empty v)
```

```
Definition examplemap' :=
  ( "bar" !-> true;
    "foo" !-> true;
    _     !-> false
  ).
```

IMP - simple imperative programs

IMP - Syntax

$$
\begin{aligned}
c := {} & \texttt{skip} \\
& |\, x \texttt{:=}\, a \\
& |\, c \texttt{;} c \\
& |\, \texttt{if}\ b\ \texttt{then}\ c\ \texttt{else}\ c\ \texttt{end} \\
& |\, \texttt{while}\ b\ \texttt{do}\ c\ \texttt{end}
\end{aligned}
$$

IMP - Syntax

$c := \texttt{skip}$

$\quad \mid x := a$

$\quad \mid c ; c$

$\quad \mid \texttt{if}\ b\ \texttt{then}\ c\ \texttt{else}\ c\ \texttt{end}$

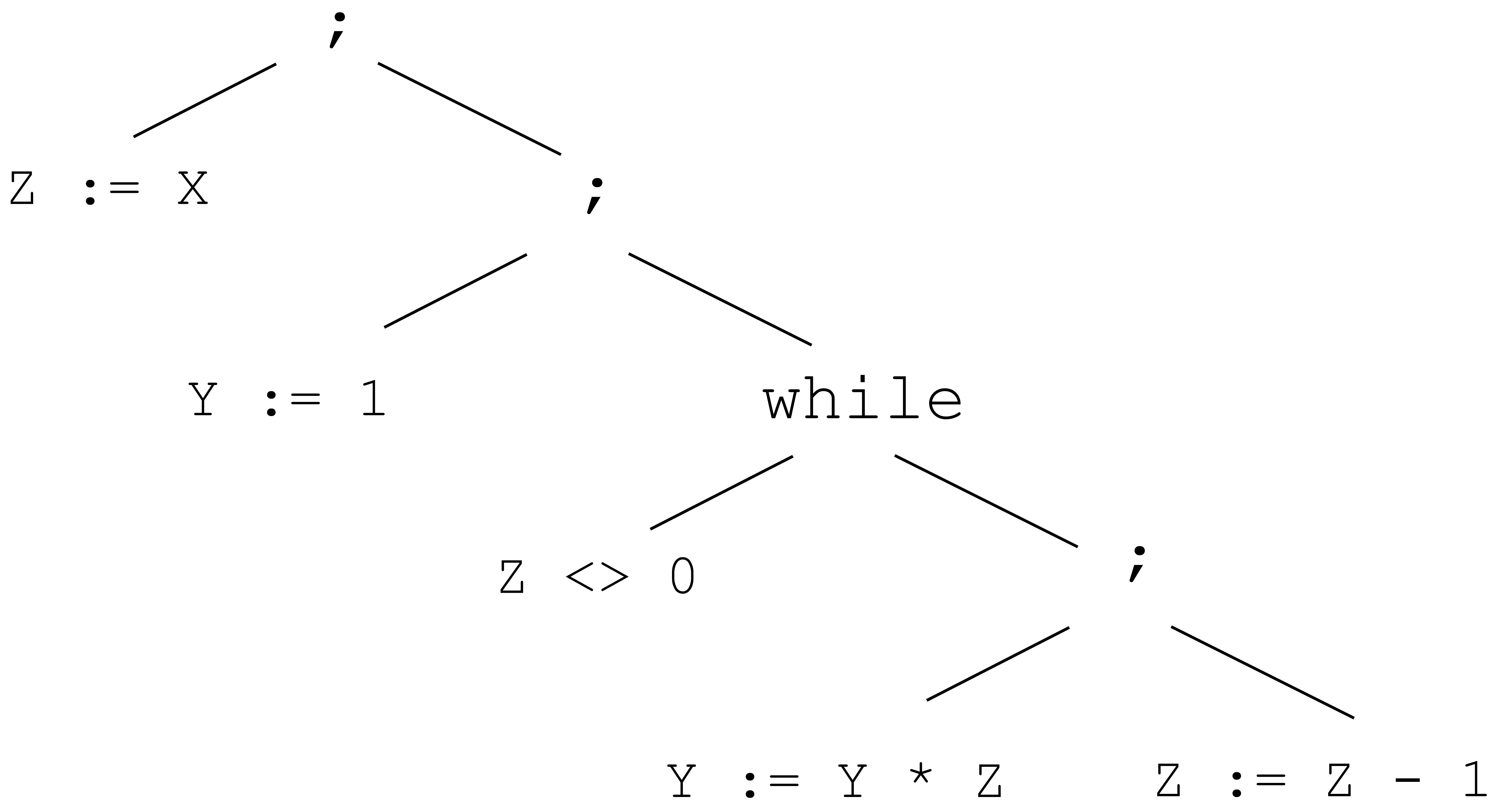
$\quad \mid \texttt{while}\ b\ \texttt{do}\ c\ \texttt{end}$

```
Z := X;
Y := 1;
while Z <> 0 do
  Y := Y * Z;
  Z := Z - 1
end
```

IMP - Syntax

$$
\begin{aligned}
c := {} & \texttt{skip} \\
& |\, x := a \\
& |\, c; c \\
& |\, \texttt{if } b \texttt{ then } c \texttt{ else } c \texttt{ end} \\
& |\, \texttt{while } b \texttt{ do } c \texttt{ end}
\end{aligned}
$$

```
Z := X;
Y := 1;
while Z <> 0 do
  Y := Y * Z;
  Z := Z - 1
end
```



IMP - Semantics

IMP - Semantics

$$\textbf{E\_Skip}\quad \frac{}{\texttt{st} \xrightarrow{\texttt{skip}} \texttt{st}}$$

IMP - Semantics

E_Skip $$\frac{}{\texttt{st} \xrightarrow{\texttt{skip}} \texttt{st}}$$

E_Asgn $$\frac{a \Rightarrow n}{\texttt{st} \xrightarrow{x \texttt{ := } a} \texttt{st}[x \mapsto n]}$$

IMP - Semantics

E_Skip $$\frac{}{\texttt{st} \xrightarrow{\texttt{skip}} \texttt{st}}$$

E_Asgn $$\frac{a \Rightarrow n}{\texttt{st} \xrightarrow{x \,\texttt{:=}\, a} \texttt{st}[x \mapsto n]}$$

E_Seq $$\frac{\texttt{st} \xrightarrow{c_1} \texttt{st}' \qquad \texttt{st}' \xrightarrow{c_2} \texttt{st}''}{\texttt{st} \xrightarrow{c_1\texttt{;}\, c_2} \texttt{st}''}$$

IMP - Semantics

E_Skip
$$\frac{}{\texttt{st} \xrightarrow{\texttt{skip}} \texttt{st}}$$

E_Asgn
$$\frac{a \Rightarrow n}{\texttt{st} \xrightarrow{x\ \texttt{:=}\ a} \texttt{st}[x \mapsto n]}$$

E_Seq
$$\frac{\texttt{st} \xrightarrow{c_1} \texttt{st}' \qquad \texttt{st}' \xrightarrow{c_2} \texttt{st}''}{\texttt{st} \xrightarrow{c_1\texttt{;}\,c_2} \texttt{st}''}$$

E_IfTrue
$$\frac{b \Rightarrow \texttt{true} \qquad \texttt{st} \xrightarrow{c_1} \texttt{st}'}{\texttt{st} \xrightarrow{\texttt{if}\ b\ \texttt{then}\ c_1\ \texttt{else}\ c_2\ \texttt{end}} \texttt{st}'}$$

E_IfFalse
$$\frac{b \Rightarrow \texttt{false} \qquad \texttt{st} \xrightarrow{c_2} \texttt{st}'}{\texttt{st} \xrightarrow{\texttt{if}\ b\ \texttt{then}\ c_1\ \texttt{else}\ c_2\ \texttt{end}} \texttt{st}'}$$

IMP - Semantics

E_Skip
$$\frac{}{\mathtt{st} \xrightarrow{\mathtt{skip}} \mathtt{st}}$$

E_Asgn
$$\frac{a \Rightarrow n}{\mathtt{st} \xrightarrow{x\ \mathtt{:=}\ a} \mathtt{st}[x \mapsto n]}$$

E_Seq
$$\frac{\mathtt{st} \xrightarrow{c_1} \mathtt{st}' \qquad \mathtt{st}' \xrightarrow{c_2} \mathtt{st}''}{\mathtt{st} \xrightarrow{c_1 \mathtt{;}\, c_2} \mathtt{st}''}$$

E_IfTrue
$$\frac{b \Rightarrow \mathtt{true} \qquad \mathtt{st} \xrightarrow{c_1} \mathtt{st}'}{\mathtt{st} \xrightarrow{\mathtt{if}\ b\ \mathtt{then}\ c_1\ \mathtt{else}\ c_2\ \mathtt{end}} \mathtt{st}'}$$

E_IfFalse
$$\frac{b \Rightarrow \mathtt{false} \qquad \mathtt{st} \xrightarrow{c_2} \mathtt{st}'}{\mathtt{st} \xrightarrow{\mathtt{if}\ b\ \mathtt{then}\ c_1\ \mathtt{else}\ c_2\ \mathtt{end}} \mathtt{st}'}$$

E_WhileTrue
$$\frac{b \Rightarrow \mathtt{true} \qquad \mathtt{st} \xrightarrow{c} \mathtt{st}' \qquad \mathtt{st}' \xrightarrow{\mathtt{while}\ b\ \mathtt{do}\ c\ \mathtt{end}} \mathtt{st}''}{\mathtt{st} \xrightarrow{\mathtt{while}\ b\ \mathtt{do}\ c\ \mathtt{end}} \mathtt{st}''}$$

E_WhileFalse
$$\frac{b \Rightarrow \mathtt{false}}{\mathtt{st} \xrightarrow{\mathtt{while}\ b\ \mathtt{do}\ c\ \mathtt{end}} \mathtt{st}}$$

Quiz 2

http://etc.ch/VAiu

Is the following proposition provable?

```
∀ (c : com) (st st' : state),
    st =[ skip ; c ]=> st' →
    st =[ c ]=> st'
```

1) Yes

2) No

3) Not sure

Quiz 3

http://etc.ch/VAiu

Is the following proposition provable?

```
∀ (c₁ c₂ : com) (st st' : state),
    st =[ c₁ ; c₂ ]=> st' →
    st =[ c₁ ]=> st →
    st =[ c₂ ]=> st'
```

1) Yes

2) No

3) Not sure

Quiz 4

http://etc.ch/VAiu

Is the following proposition provable?

```
∀ (b : bexp) (c : com) (st st' : state),
    st =[ if b then c else c end ]=> st' →
    st =[ c ]=> st'
```

1) Yes

2) No

3) Not sure

Quiz 5

http://etc.ch/VAiu

Is the following proposition provable?

```
    ∀ b : bexp,
      (∀ st, beval st b = true) →
      ∀ (c : com) (st : state),
        ~(∃ st', st =[ while b do c end ]=> st')
```

1) Yes

2) No

3) Not sure

Quiz 6

http://etc.ch/VAiu

Is the following proposition provable?

```
      ∀ (b : bexp) (c : com) (st : state),
         ~(∃ st', st =[ while b do c end ]=> st') →
         ∀ st'', beval st'' b = true
```

1) Yes

2) No

3) Not sure

Summary

Syntax for expressions & programs

$$\begin{array}{l} a := n \\ \quad \mid a + a \\ \quad \mid a - a \\ \quad \mid a \times a \\ n \in \mathbb{N} \end{array} \qquad \begin{array}{l} c := \texttt{skip} \\ \quad \mid x := a \\ \quad \mid c\,;c \\ \quad \mid \texttt{if } b \texttt{ then } c \texttt{ else } c \texttt{ end} \\ \quad \mid \texttt{while } b \texttt{ do } c \texttt{ end} \end{array}$$

Computational evaluation

```
Fixpoint aeval (a : aexp) : nat :=
```

Relational Semantics

$$\frac{b \Rightarrow \texttt{false} \qquad \texttt{st} \xrightarrow{c_2} \texttt{st}'}{\texttt{st} \xrightarrow{\texttt{if } b \texttt{ then } c_1 \texttt{ else } c_2 \texttt{ end}} \texttt{st}'}$$

Tacticals

```
;
try
repeat
```