Expression Semantics

\[ M : Expression \times State \rightarrow Value \]

Expression = Variable | Value | Binary | Unary

Variable = String id

Value = IntValue | BoolValue | FloatValue | CharValue

Binary = BinaryOp op; Expression term1, term2

Unary = UnaryOp op; Expression term
If the Expression is a Value, then its meaning is the meaning of the Value itself.

If the Expression is a Variable, then its meaning is the Value of the Variable in the current state.

If the Expression is a Binary, then the meaning of each of its operands term1 and term2 is first determined. Then Meaning Rule of Binary Expression determines the meaning of the expression by applying the Operator op to the Value of those two operands.

If the Expression is a Unary, then the meaning of its operand term is first determined. Then Meaning Rule of Unary Expression determines the meaning of the expression by applying the Operator op to the Value of the operand.
Meaning Rule of Binary Expression

✦ If either operand term1 or term2 is undefined, the expression is semantically meaningless.

✦ If the operator is an integer operator, then integer arithmetic add (int+), subtract (int-), or multiply (int*) perform on the integer operands, resulting in an integer result. If the operator is divide (int/), then the result is the same as a mathematical divide with truncation toward zero.

✦ If the operator is a floating point operator, then floating point arithmetic using the IEEE standard is performed on the float operands, resulting in a float result.

✦ If the operator is a relational operator, then the operands are compared with a result of either true or false.

✦ If the operator is a Boolean operator, then
  • The operator && is interpreted as: \( a \land b \equiv \text{if } a \text{ then } b \text{ else false} \)
  • The operator || is interpreted as: \( a \lor b \equiv \text{if } a \text{ then true else } b \)
ApplyBinary : Operator $\times$ Value $\times$ Value $\rightarrow$ value

$\text{ApplyBinary}(\text{Operator } op, \text{Value } v1, \text{Value } v2) = \begin{cases} 
    v1 + v2, & \text{if } op = \text{int}+ \\
    v1 - v2, & \text{if } op = \text{int}- \\
    v1 \times v2, & \text{if } op = \text{int}* \\
    \left\lfloor \frac{v1}{v2} \right\rfloor \times \text{sign}(v1 \times v2), & \text{if } op = \text{int}/ \\
    \ldots
\end{cases}$
Example: $b + 3 \times c$

\[
\{ \langle b, 1 \rangle, \langle c, 3 \rangle \}
\]

\[
M(\text{Expression } e, \text{State state}) = \begin{cases} 
  e, & \text{if } e \text{ is a Value} \\
  \text{state}(e), & \text{if } e \text{ is a Variable} \\
  \text{ApplyBinary}(e.\text{op}, M(e.\text{term1}, state), M(e.\text{term2}, state)), & \text{if } e \text{ is a Binary} \\
  \text{ApplyUnary}(e.\text{op}, M(e.\text{term}, state)), & \text{if } e \text{ is a Unary}
\end{cases}
\]

\[
\text{ApplyBinary}(\text{Operator op, Value } v1, \text{Value } v2) = \begin{cases} 
  v1 + v2, & \text{if } op = \text{int+} \\
  v1 - v2, & \text{if } op = \text{int-} \\
  v1 \times v2, & \text{if } op = \text{int*} \\
  \left\lfloor \frac{v1}{v2} \right\rfloor \times \text{sign}(v1 \times v2), & \text{if } op = \text{int/} \\
  \ldots
\end{cases}
\]
$M(b + 3 \times c, \{\langle b, 1 \rangle, \langle c, 3 \rangle \}) = ApplyBinary(+, A, B)$

$= ApplyBinary(+, 1, 9)$

$= 10$

$A = M(b, \{\langle b, 1 \rangle, \langle c, 3 \rangle \})$

$= 1$

$B = M(3 \times c, \{\langle b, 1 \rangle, \langle c, 3 \rangle \})$

$= ApplyBinary(\times, C, D)$

$= ApplyBinary(\times, 3, 3)$

$= 9$

$C = M(3, \{\langle b, 1 \rangle, \langle c, 3 \rangle \})$

$= 3$

$D = M(c, \{\langle b, 1 \rangle, \langle c, 3 \rangle \})$

$= 3$
Side Effects

A *side effect* occurs during the evaluation of an expression if, in addition to returning a value, the expression *alters the state of the program*

```
x = 1
+++x + x++
```
Solutions to Address Side Effects

\[ M : Expression \times State \to Value \times State \]

- If the Expression is a Binary, the meaning of term1 in the current state is first determined, giving a value \( v_1 \) and a state \( s_1 \). Then the meaning of term2 in state \( s_1 \) is determined, giving a value \( v_2 \) and a state \( s_2 \). Then Meaning Rule of Binary Expression determines the meaning of the expression by applying the Operator \( \oplus \) to the resulting values \( v_1 \) and \( v_2 \) in state \( s_2 \), and the resulting state is \( s_2 \).

By applying this rule, what is the value of the expression, \( ++x + x++ \)?