

# Homework 2: Operational Semantics for WHILE

CS 252: Advanced Programming Languages  
Prof. Thomas H. Austin  
San José State University

## 1 Introduction

For this assignment, you will implement the semantics for a small imperative language, named WHILE.  
( $e_1; e_2$ )

## 2 Small-step semantics

Most of these rules are fairly straightforward, but there are a couple of points to note with the [SS-WHILE] rule.

**Runtime Syntax:**

$$\sigma \in Store = variable \rightarrow v$$

**Evaluation Rules:**

$$e, \sigma \rightarrow e', \sigma'$$

$$[SS-VAR] \quad \frac{x \in domain(\sigma) \quad \sigma(x) = v}{x, \sigma \rightarrow v, \sigma}$$

$$[SS-ASSIGN] \quad \frac{}{x := v, \sigma \rightarrow v, \sigma[x := v]}$$

$$[SS-OP] \quad \frac{v = v_1 \text{ op } v_2}{v_1 \text{ op } v_2, \sigma \rightarrow v, \sigma}$$

$$[SS-SEQ] \quad \frac{}{v; e, \sigma \rightarrow e, \sigma}$$

$$[SS-IFTRUE] \quad \frac{}{\text{if true then } e_1 \text{ else } e_2, \sigma \rightarrow e_1, \sigma}$$

$$[SS-IFFALSE] \quad \frac{}{\text{if false then } e_1 \text{ else } e_2, \sigma \rightarrow e_2, \sigma}$$

$$[SS-WHILE] \quad \frac{}{\text{while } (e_1) \text{ } e_2, \sigma \rightarrow \text{if } e_1 \text{ then } e_2; \text{while } (e_1) \text{ } e_2 \text{ else false}, \sigma}$$

**Figure 2:** Small-step semantics for WHILE

$e ::=$		<i>Expressions</i>
	$x$	variables/addresses
	$v$	values
	$x := e$	assignment
	$e; e$	sequential expressions
	$e \text{ op } e$	binary operations
	<b>if</b> $e$ <b>then</b> $e$ <b>else</b> $e$	conditional expressions
	<b>while</b> $(e)$ $e$	while expressions
	<b>not</b> $e$	not expressions
	<b>and</b> $(e)$ $e$	and expressions
	<b>or</b> $(e)$ $e$	or expressions
$v ::=$		<i>Values</i>
	$i$	integer values
	$b$	boolean values
$op ::=$	$+ \mid - \mid * \mid / \mid > \mid >= \mid < \mid <=$	<i>Binary operators</i>

**Figure 1:** The WHILE language

**Variable Evaluation Rules:**

$$[\text{SS-VAR}] \quad \frac{x \in \text{domain}(\sigma) \quad \sigma(x) = v}{x, \sigma \rightarrow v, \sigma}$$

**Figure 3:** Variable Small-Step Semantics Evaluation Order Rules

**Set/Assignment Evaluation Rules:**

$$[\text{SS-ASSIGNCONTEXT}] \quad \frac{e, \sigma \rightarrow e', \sigma'}{x := e, \sigma \rightarrow x := e', \sigma'}$$

$$[\text{SS-ASSIGNREDUCTION}] \quad \frac{}{x := v, \sigma \rightarrow v, \sigma[x := v]}$$

**Figure 4:** Set/Assignment Small-Step Semantics Evaluation Order Rules

**Sequence (;) Evaluation Rules:**

$$[\text{SS-SEQCONTEXT}] \quad \frac{e_1, \sigma \rightarrow e'_1, \sigma'}{e_1; e_2, \sigma \rightarrow e'_1; e_2, \sigma'}$$

$$[\text{SS-SEQREDUCTION}] \quad \frac{}{v; e, \sigma \rightarrow e, \sigma'}$$

**Figure 5:** Sequence (;) Evaluation Order Rules

**Conditional Statement (if) Evaluation Rules:**

[SS-IFCONTEXT]	$\frac{e_1, \sigma \rightarrow e'_1, \sigma'}{\text{if } e_1 \text{ then } e_2 \text{ else } e_3, \sigma \rightarrow \text{if } e'_1 \text{ then } e_2 \text{ else } e_3, \sigma'}$
[SS-IFTRUEREDUCTION]	$\frac{}{\text{if true then } e_1 \text{ else } e_2, \sigma \rightarrow e_1, \sigma}$
[SS-IFFALSEREDUCTION]	$\frac{}{\text{if false then } e_1 \text{ else } e_2, \sigma \rightarrow e_2, \sigma}$

**Figure 6:** Conditional (if) Small-Step Semantics Evaluation Order Rules

**while Evaluation Rules:**

[SS-WHILE]	$\frac{}{\text{while } (e_1) \ e_2, \sigma \rightarrow \text{if } e_1 \text{ then } e_2; \text{while } (e_1) \ e_2 \text{ else false}, \sigma}$
------------	---

**Figure 7:** while Small-Step Semantics Evaluation Order Rules

### 3 Boolean Expressions Small-Step Semantics

In this section, I add three new expression types to the WHILE language namely: **not**, **and**, and **or**. The evaluation order rules for each are below.

Do I need the parentheses in the "and" and "or" statements? Is an infix style more typically used?

not Evaluation Rules:

$$[\text{SS-NOTCONTEXT}] \quad \frac{e, \sigma \rightarrow e', \sigma'}{\text{not } e, \sigma \rightarrow \text{not } e', \sigma'}$$

Not sure if I need this. If I do, then why? Why is this not like the "op" case:

$$[\text{SS-NOTREDUCTION}] \quad \frac{}{\text{not } v, \sigma \rightarrow \text{if } v \text{ then false else true}, \sigma}$$

I believe the above rule makes these unnecessary. Would most define as above or like below?

$$[\text{SS-NOTTRUE}] \quad \frac{}{\text{not true}, \sigma \rightarrow \text{false}, \sigma}$$

$$[\text{SS-NOTFALSE}] \quad \frac{}{\text{not false}, \sigma \rightarrow \text{true}, \sigma}$$

**Figure 8:** not Small-Step Semantics Evaluation Order Rules

and Evaluation Rules:

$$[\text{SS-ANDCONTEXT}] \quad \frac{e_1, \sigma \rightarrow e'_1, \sigma'}{\text{and } (e_1) e_2, \sigma \rightarrow \text{and } (e'_1) e_2, \sigma'}$$

$$[\text{SS-ANDREDUCTION}] \quad \frac{}{\text{and } (v) e, \sigma \rightarrow \text{if } v \text{ then } e \text{ else false}, \sigma}$$

Using the above, I think I do not need these:

$$[\text{SS-ANDCONTEXT2}] \quad \frac{e, \sigma \rightarrow e', \sigma'}{\text{and } (v) e, \sigma \rightarrow \text{and } (v) e', \sigma'}$$

$$[\text{SS-ANDALLTRUE}] \quad \frac{}{\text{and } (\text{true}) \text{ true}, \sigma \rightarrow \text{true}, \sigma}$$

$$[\text{SS-ANDFALSE1}] \quad \frac{}{\text{and } (\text{false}) v, \sigma \rightarrow \text{false}, \sigma}$$

$$[\text{SS-ANDFALSE2}] \quad \frac{}{\text{and } (v) \text{ false}, \sigma \rightarrow \text{false}, \sigma}$$

**Figure 9:** and Small-Step Semantics Evaluation Order Rules

or Evaluation Rules: (Is defining "temporary variables" as I did allowed in small step semantics?)

$$[\text{SS-ORREDUCTION}] \quad \frac{e'_1 = \text{not } e_1 \quad e'_2 = \text{not } e_2 \quad e_3 = \text{and } (e'_1) \ e'_2}{\text{or } (e_1) \ e_2, \sigma \rightarrow \text{not } e_3, \sigma}$$

**Figure 10:** or Evaluation Order Rule