Homework 2: Operational Semantics for WHILE

CS 252: Advanced Programming Languages
Student Ji An Lee
San José State University

1 MY ASSIGNMENT

Part 1: Rewrite the operational semantic rules for WHILE in LaTeX to use big-step operational semantics instead. Submit both your LaTeX source and the generated PDF file.

Extend your semantics with features to handle boolean values. **Do not treat these a binary operators.** Specifically, add support for:

- `and`
- `or`
- `not`

The exact behavior of these new features is up to you, but should seem reasonable to most programmers.

Part 2: Once you have your semantics defined, download `WhileInterp.hs` and implement the `evaluate` function, as well as any additional functions you need. Your implementation must be consistent with your operational semantics, *including your extensions for* `and`, `or`, *and* `not`. Also, you may not change any type signatures provided in the file.

Finally, implement the interpreter to match your semantics.

Zip all files together into `hw2.zip` and submit to Canvas.

$e ::=$		*Expressions*
	x	variables/addresses
	v	values
	$x := e$	assignment
	$e; e$	sequential expressions
	$e\ op\ e$	binary operations
	`if` e `then` e `else` e	conditional expressions
	`while` (e) e	while expressions
$v ::=$		*Values*
	i	integer values
	b	boolean values
$op ::=$	$+ \mid - \mid * \mid / \mid > \mid >= \mid < \mid <=$	*Binary operators*

Figure 1: The WHILE language

Evaluation Rules: $\boxed{(e_0, \sigma_0) \Downarrow (e_1, \sigma_1)}$

[BS-VALUE] $$\frac{}{(v, \sigma) \Downarrow (v, \sigma)}$$

[BS-VAR] $$\frac{}{(x_0, \sigma_0) \Downarrow \sigma(x_0)}$$

[BS-ASSIGN] $$\frac{(e_0, \sigma_0) \Downarrow v_0}{(x := e_0, \sigma_0) \Downarrow \sigma(x \mapsto v_0)}$$

[BS-SEQ] $$\frac{(e_0, \sigma_0) \Downarrow \sigma_1 \quad (e_1, \sigma_1) \Downarrow \sigma_2}{(e_0; e_1, \sigma_0) \Downarrow \sigma_2}$$

[BS-OP] $$\frac{(e_0, \sigma_0) \Downarrow n_0 \quad (e_1, \sigma_0) \Downarrow n_1 \quad n_2 = n_0\ op\ n_1}{(e_0\ op\ e_1, \sigma_0) \Downarrow n_2}$$

[BS-IFTRUE] $$\frac{(e_0, \sigma_0) \Downarrow True \quad (e_1, \sigma_0) \Downarrow \sigma_1}{(\texttt{if e}_0\ \texttt{e}_1\ \texttt{e}_2, \sigma_0) \Downarrow \sigma_1}$$

[BS-IFFALSE] $$\frac{(e_0, \sigma_0) \Downarrow False \quad (e_2, \sigma_0) \Downarrow \sigma_1}{(\texttt{if e}_0\ \texttt{e}_1\ \texttt{e}_2, \sigma_0) \Downarrow \sigma_1}$$

[BS-WHILE-FALSE] $$\frac{(e_0, \sigma_0) \Downarrow False}{(while(e_0)e_1, \sigma_0) \Downarrow \sigma_0}$$

[BS-WHILE-TRUE] $$\frac{(e_0, \sigma_0) \Downarrow True \quad (e_1, \sigma_0) \Downarrow \sigma_1 \quad (while(e_0)e_1, \sigma_1) \Downarrow \sigma_2}{(while(e_0)e_1, \sigma_0) \Downarrow \sigma_2}$$

Figure 2: Big-step semantics for WHILE

Continued..

$$[\text{BS-AND-TRUE}] \quad \frac{(e_0, \sigma_0) \Downarrow True \quad (e_1, \sigma_0) \Downarrow b \quad | \quad b \in bool}{(AND\ e_0\ e_1, \sigma_0) \Downarrow b}$$

$$[\text{BS-AND-FALSE}] \quad \frac{(e_0, \sigma_0) \Downarrow False}{(AND\ e_0\ e_1, \sigma_0) \Downarrow False}$$

$$[\text{BS-OR-TRUE}] \quad \frac{(e_0, \sigma_0) \Downarrow True}{(OR\ e_0\ e_1, \sigma_0) \Downarrow True}$$

$$[\text{BS-OR-FALSE}] \quad \frac{(e_0, \sigma_0) \Downarrow False \quad (e_1, \sigma_0) \Downarrow b \quad | \quad b \in bool}{(OR\ e_0\ e_1, \sigma_0) \Downarrow b}$$

$$[\text{BS-NOT-TRUE}] \quad \frac{(e_0, \sigma_0) \Downarrow True}{(NOT\ e_0, \sigma_0) \Downarrow False}$$

$$[\text{BS-NOT-FALSE}] \quad \frac{(e_0, \sigma_0) \Downarrow False}{(NOT\ e_0, \sigma_0) \Downarrow True}$$

Figure 3: Big-step semantics for WHILE