Semantics and typing of break and continue

Programming Paradigms · Session 10 · lab

The language WHILE of block 2

Expressions	$e ::= n \mid b \mid x \mid e \; op \; e$
Statements	$s ::= x \leftarrow e \mid \mathsf{if}\ e\ \mathsf{then}\ s\ \mathsf{else}\ s \mid \mathsf{while}\ e\ \mathsf{do}\ s \mid s; s \mid \mathsf{skip}$
Values	$v ::= n \mid b$
Types	$\tau ::= \mathsf{int} \mid \mathsf{bool}$

Here n is an integer constant, $b \in \{\mathsf{true}, \mathsf{false}\}$ and $op \in \{+, -, \times, <, >, =\}$. An environment E is a finite map from variables to values, and a typing environment Γ a finite map from variables to types.

Big-step semantics. $E, e \twoheadrightarrow v$: in the environment E, the expression e has the value v. $E, s \twoheadrightarrow E'$: in the environment E, the run of the statement s ends, in the environment E'.

$$\frac{}{E, n \twoheadrightarrow n}\ (\textsc{Int}) \qquad \frac{}{E, b \twoheadrightarrow b}\ (\textsc{Bool}) \qquad \frac{x \in \mathrm{dom}(E)}{E, x \twoheadrightarrow E(x)}\ (\textsc{Var})$$

$$\frac{E, e_1 \twoheadrightarrow n_1 \quad E, e_2 \twoheadrightarrow n_2 \quad n = n_1 + n_2}{E, e_1 + e_2 \twoheadrightarrow n}\ (\textsc{Plus})$$

$$\frac{}{E, \mathsf{skip} \twoheadrightarrow E}\ (\textsc{Skip}) \qquad \frac{E, e \twoheadrightarrow v}{E, x \leftarrow e \twoheadrightarrow E\{x \mapsto v\}}\ (\textsc{Assign}) \qquad \frac{E, s_1 \twoheadrightarrow E_1 \quad E_1, s_2 \twoheadrightarrow E_2}{E, s_1; s_2 \twoheadrightarrow E_2}\ (\textsc{Seq})$$

$$\frac{E, e \twoheadrightarrow \mathsf{true} \quad E, s_1 \twoheadrightarrow E_1}{E, \mathsf{if}\ e\ \mathsf{then}\ s_1\ \mathsf{else}\ s_2 \twoheadrightarrow E_1}\ (\textsc{IfTrue}) \qquad \frac{E, e \twoheadrightarrow \mathsf{false} \quad E, s_2 \twoheadrightarrow E_2}{E, \mathsf{if}\ e\ \mathsf{then}\ s_1\ \mathsf{else}\ s_2 \twoheadrightarrow E_2}\ (\textsc{IfFalse})$$

$$\frac{E, e \twoheadrightarrow \mathsf{false}}{E, \mathsf{while}\ e\ \mathsf{do}\ s \twoheadrightarrow E}\ (\textsc{WhileFalse})$$

$$\frac{E, e \twoheadrightarrow \mathsf{true} \quad E, s \twoheadrightarrow E_1 \quad E_1, \mathsf{while}\ e\ \mathsf{do}\ s \twoheadrightarrow E_2}{E, \mathsf{while}\ e\ \mathsf{do}\ s \twoheadrightarrow E_2}\ (\textsc{WhileTrue})$$

Typing. $\Gamma \vdash e : \tau$: in the typing environment Γ, the expression e has the type τ. $\Gamma \vdash s$: in the typing environment Γ, the statement s is well typed.

$$\frac{}{\Gamma \vdash n : \mathsf{int}}\ (\text{T-}\textsc{Int}) \qquad \frac{}{\Gamma \vdash b : \mathsf{bool}}\ (\text{T-}\textsc{Bool}) \qquad \frac{x \in \mathrm{dom}(\Gamma)}{\Gamma \vdash x : \Gamma(x)}\ (\text{T-}\textsc{Var})$$

$$\frac{\Gamma \vdash e_1 : \mathsf{int} \quad \Gamma \vdash e_2 : \mathsf{int}}{\Gamma \vdash e_1 + e_2 : \mathsf{int}}\ (\text{T-}\textsc{Plus}) \qquad \frac{\Gamma \vdash e_1 : \mathsf{int} \quad \Gamma \vdash e_2 : \mathsf{int}}{\Gamma \vdash e_1 < e_2 : \mathsf{bool}}\ (\text{T-}\textsc{Less})$$

$$\frac{}{\Gamma \vdash \mathsf{skip}}\ (\text{T-}\textsc{Skip}) \qquad \frac{\Gamma \vdash s_1 \quad \Gamma \vdash s_2}{\Gamma \vdash s_1; s_2}\ (\text{T-}\textsc{Seq}) \qquad \frac{\Gamma \vdash x : \tau \quad \Gamma \vdash e : \tau}{\Gamma \vdash x \leftarrow e}\ (\text{T-}\textsc{Assign})$$

$$\frac{\Gamma \vdash e : \mathsf{bool} \quad \Gamma \vdash s_1 \quad \Gamma \vdash s_2}{\Gamma \vdash \mathsf{if}\ e\ \mathsf{then}\ s_1\ \mathsf{else}\ s_2}\ (\text{T-}\textsc{If}) \qquad \frac{\Gamma \vdash e : \mathsf{bool} \quad \Gamma \vdash s}{\Gamma \vdash \mathsf{while}\ e\ \mathsf{do}\ s}\ (\text{T-}\textsc{While})$$

Conventions. The operators $-$ and $\times$ have the rules of $+$, (PLUS) and (T-PLUS), each with its own operation on integers. The operators $<$, $>$ and $=$ have the typing rule (T-LESS), and an evaluation rule of the form of (PLUS) whose value is the Boolean true when the comparison of n_1 and n_2 holds, and false otherwise. The environment $E\{x \mapsto v\}$ maps x to v and every other variable as E does. The sequence associates to the right: $s_1; s_2; s_3$ is $s_1; (s_2; s_3)$. A branch or a loop body that is itself a sequence stands in parentheses.

A *derivation* of a judgment is a tree: the judgment stands at the root, every node is the conclusion of an instance of one of the rules, and the node's children are the premises of that instance. Each step of a derivation carries the name of its rule.

The extension: two statements and three outcomes

The lab extends WHILE with two statements:

$$s \ ::= \ \cdots \mid \mathsf{break} \mid \mathsf{continue}$$

The statement break ends the loop whose body is running: the run goes on after that loop. The statement continue ends the current run of the body: the run goes on with the next evaluation of the guard of that loop. Expressions, values and environments are unchanged, and so is the judgment $E, e \twoheadrightarrow v$.

Both statements interrupt the statement that contains them. The judgment for statements therefore records how a run ends, with an *outcome* ω:

$$E, s \twoheadrightarrow (E', \omega) \qquad\qquad \omega \ ::= \ \mathsf{normal} \mid \mathsf{broken} \mid \mathsf{continued}$$

It reads: in the environment E, the run of s ends, in the environment E' and with the outcome ω. The outcome broken says that the run of s ended on a break that no loop inside s has caught, and continued says the same of a continue. An outcome is a part of the judgment, not a value of the language.

Four rules of the extended semantics are given. They are the rules (SKIP), (ASSIGN), (IFTRUE) and (IFFALSE) of block 2, with an outcome:

$$\frac{}{E, \mathsf{skip} \twoheadrightarrow (E, \mathsf{normal})}\ (\textsc{Skip}) \qquad \frac{E, e \twoheadrightarrow v}{E, x \leftarrow e \twoheadrightarrow (E\{x \mapsto v\}, \mathsf{normal})}\ (\textsc{Assign})$$

$$\frac{E, e \twoheadrightarrow \mathsf{true} \quad E, s_1 \twoheadrightarrow (E_1, \omega)}{E, \mathsf{if}\ e\ \mathsf{then}\ s_1\ \mathsf{else}\ s_2 \twoheadrightarrow (E_1, \omega)}\ (\textsc{IfTrue}) \qquad \frac{E, e \twoheadrightarrow \mathsf{false} \quad E, s_2 \twoheadrightarrow (E_2, \omega)}{E, \mathsf{if}\ e\ \mathsf{then}\ s_1\ \mathsf{else}\ s_2 \twoheadrightarrow (E_2, \omega)}\ (\textsc{IfFalse})$$

The exercises of the first part build the other rules: those of break, of continue, of the sequence and of the loop.

Big-step rules of the extension

Exercise 1 Rules of break and continue

Given the judgment $E, s \twoheadrightarrow (E', \omega)$, write the rule of break and the rule of continue. Each of the two is an axiom, a rule without a premise.

Exercise 2 Rules of the sequence

Given a sequence $s_1; s_2$, write its three rules, one for each outcome of the run of s_1. In each rule the run of s_1 is a premise.

Exercise 3 Derivation of a run that ends on a break

Given the environment $\{x \mapsto 0\}$ and the statement

$$x \leftarrow 1;\ \mathsf{break};\ x \leftarrow 2,$$

write the derivation of its run with the given rules and the rules of the two exercises above, and write the final environment and the outcome.

Exercise 4 Rules of a loop whose body ends normally

Given a loop while e do s, write two rules: the rule for a guard whose value is false, and the rule for a guard whose value is true and a body whose run ends with the outcome normal. Both follow (WHILEFALSE) and (WHILETRUE) of block 2; what is to decide is the outcome in each premise and in each conclusion.

Exercise 5 Rule of a loop whose body ends on a break

Given a loop $\mathsf{while}\ e\ \mathsf{do}\ s$ whose body ends with the outcome broken, write the rule of the loop. The body runs only when the guard has the value true, and the rule has a premise that says so. The outcome of the loop itself is the point of the rule.

Exercise 6 Rule of a loop whose body ends on a continue

Given a loop $\mathsf{while}\ e\ \mathsf{do}\ s$ whose body ends with the outcome $\mathsf{continued}$, write the rule of the loop. The rest of the body is skipped, and the loop itself goes on.

Exercise 7 Runs of two loops

Given the environment $E_0 = \{x \mapsto 0,\ i \mapsto 0\}$ and the two statements

$$s_a = \mathsf{while}\ i < 2\ \mathsf{do}\ (i \leftarrow i + 1;\ \mathsf{if}\ i = 2\ \mathsf{then}\ \mathsf{break}\ \mathsf{else}\ x \leftarrow x + i)$$
$$s_b = \mathsf{while}\ i < 3\ \mathsf{do}\ (i \leftarrow i + 1;\ \mathsf{if}\ i = 2\ \mathsf{then}\ \mathsf{continue}\ \mathsf{else}\ x \leftarrow x + i)$$

write, for each of them, the run from E_0 as a list of steps, one line for each evaluation of the guard: the rule of the loop that applies and, when the body runs, the environment after the body and the outcome of the body. Write then the final environment and the outcome of each statement. The full derivation tree is not asked.

Typing rules of the extension

A break or a $\mathsf{continue}$ that stands in no loop has no loop to end or to go on with. The type checker rejects such a program. The typing judgment for expressions, $\Gamma \vdash e : \tau$, is unchanged. The typing judgment for statements carries a *flag* ℓ:

$$(\Gamma, \ell) \vdash s \qquad\qquad \ell \in \{\mathsf{true}, \mathsf{false}\}$$

The flag says whether the statement s stands inside a loop. A whole program s is *well typed in* Γ when $(\Gamma, \mathsf{false}) \vdash s$.

Four typing rules are given. They are the rules of block 2, and each of them passes the flag from its conclusion to each premise on a statement, unchanged:

$$\frac{}{(\Gamma, \ell) \vdash \mathsf{skip}}\ (\text{T-Skip}) \qquad \frac{(\Gamma, \ell) \vdash s_1 \quad (\Gamma, \ell) \vdash s_2}{(\Gamma, \ell) \vdash s_1; s_2}\ (\text{T-Seq}) \qquad \frac{\Gamma \vdash x : \tau \quad \Gamma \vdash e : \tau}{(\Gamma, \ell) \vdash x \leftarrow e}\ (\text{T-Assign})$$

$$\frac{\Gamma \vdash e : \mathsf{bool} \quad (\Gamma, \ell) \vdash s_1 \quad (\Gamma, \ell) \vdash s_2}{(\Gamma, \ell) \vdash \mathsf{if}\ e\ \mathsf{then}\ s_1\ \mathsf{else}\ s_2}\ (\text{T-If})$$

Exercise 8 Typing rules of break and continue

Given the judgment $(\Gamma, \ell) \vdash s$, write the typing rule of break and the typing rule of $\mathsf{continue}$. Explain then how the rules reject break when the flag is false.

Exercise 9 Typing rule of the loop

Given a loop $\mathsf{while}\ e\ \mathsf{do}\ s$, write its typing rule. Write which flag the rule gives to the body s, and which flags it allows in its conclusion, with the reason for each.

Exercise 10 The flag in the rule of the conditional

Given the variant of (T-If) whose premises on s_1 and s_2 use the flag false in place of ℓ,

$$\frac{\Gamma \vdash e : \mathsf{bool} \quad (\Gamma, \mathsf{false}) \vdash s_1 \quad (\Gamma, \mathsf{false}) \vdash s_2}{(\Gamma, \ell) \vdash \mathsf{if}\ e\ \mathsf{then}\ s_1\ \mathsf{else}\ s_2},$$

write a program that the rules with (T-If) accept and that the rules with the variant reject, and explain which of the two rules is the right one.

Exercise 11 Typing of a loop with a break

Given $\Gamma = \{x \mapsto \mathsf{int}\}$ and the statement

$$\mathsf{while}\ x > 0\ \mathsf{do}\ (\mathsf{if}\ x = 1\ \mathsf{then}\ \mathsf{break}\ \mathsf{else}\ x \leftarrow x - 1),$$

decide whether the statement is well typed in Γ. Write the derivation if there is one, and otherwise the premise that fails.

Exercise 12 A conditional outside and inside a loop

Given $\Gamma = \{x \mapsto \mathsf{int}\}$ and the statement $P = \mathsf{if}\ x > 0\ \mathsf{then}\ \mathsf{break}\ \mathsf{else}\ \mathsf{skip}$, decide

(a) whether $(\Gamma, \mathsf{false}) \vdash P$ is derivable;
(b) whether $(\Gamma, \mathsf{false}) \vdash \mathsf{while}\ \mathsf{true}\ \mathsf{do}\ P$ is derivable.

Write for each the rules that apply and, where the derivation fails, the premise that fails.

Exercise 13 Nested loops

Given $\Gamma = \{i \mapsto \mathsf{int},\ j \mapsto \mathsf{int}\}$ and the statement

$$\mathsf{while}\ i < 10\ \mathsf{do}\ \big((\mathsf{while}\ j < 5\ \mathsf{do}\ \mathsf{break});\ i \leftarrow i + 1\big),$$

(a) decide whether the statement is well typed in Γ, and write the flag with which each of its sub-statements is checked;
(b) write which loop the break ends in a run, with the big-step rules that say so;
(c) decide whether the typing judgment tells which loop a break ends.

The property that the typing guarantees

The semantics gives a result to a break that stands in no loop: the run ends with the outcome broken. The typing rules exist to exclude that result. The claim is

$$\text{if } (\Gamma, \mathsf{false}) \vdash s \text{ and } E, s \twoheadrightarrow (E', \omega), \text{ then } \omega = \mathsf{normal}. \qquad (*)$$

Exercise 14 Outcome of a well-typed program

Given the big-step rules and the typing rules of the extension, and the claim $(*)$,

(a) decide whether the claim still holds with the flag true in place of false; write a counterexample if it does not;
(b) write a claim about $(\Gamma, \ell) \vdash s$ that holds for both values of ℓ and that gives $(*)$ when $\ell = \mathsf{false}$;
(c) prove that claim by induction on the derivation of $E, s \twoheadrightarrow (E', \omega)$, one case for each big-step rule. The case of the loop first needs a property of the loop alone: the outcomes with which the run of a loop can end.