Manager | Keeps team on track
---|---
Recorder | Records decisions
Reporter | Reports to class
Reflector | Assesses team performance

The Rules

**Expression Rules**

\[
\begin{align*}
<i, \sigma> & \Downarrow e \quad \text{CONST, when } i \text{ is an integer.} \\
<u, \sigma> & \Downarrow e \quad \text{VAR, if } u := v \in \sigma. \\
<i_1, \sigma> & \Downarrow e_1 \quad <i_2, \sigma> \Downarrow e_2 \quad \text{ARITH} \\
<i_1 \oplus i_2, \sigma> & \Downarrow e_1 \oplus e_2 \quad <e_1 < e_2, \sigma> \Downarrow e_1 \quad <e_2, \sigma> \Downarrow e_2
\end{align*}
\]

**Boolean Rules**

\[
\begin{align*}
<i, \sigma> & \Downarrow \text{true} \\
<i, \sigma> & \Downarrow \text{false} \\
<u, \sigma> & \Downarrow v \\
<u, \sigma> & \Downarrow \text{true} \\
<u, \sigma> & \Downarrow \text{false}
\end{align*}
\]

**Statement Rules**

\[
\begin{align*}
<i, \sigma> \Downarrow \sigma & \quad \text{SKIP} \\
<i, \sigma> \Downarrow \sigma & \quad \text{ASSIGN} \\
<i, \sigma> \Downarrow \sigma & \quad \text{SEQ} \\
<i, \sigma> \Downarrow \sigma & \quad \text{IF} \\
<i, \sigma> \Downarrow \sigma & \quad \text{WHILE}
\end{align*}
\]
Understanding the Rules

**Problem 1)** Consider the judgment \(< x + y, \{ x := 10, y := 20 \} > \downarrow_e 30 >\). Explain the meaning of the different syntactic components:

- \(< x + y, \{ x := 10, y := 20 \} >\)
- \(\downarrow_e\)

**Problem 2)** There are some rules that come in almost identical pairs, for example, there are two versions of the \(\text{Var}\) rule. What is the difference between them, and why do we need both?

Using the Rules

One use of the rules is to build proof trees. Consider the following proof tree for the initial configuration \(< x + y + z, \{ x := 10, y := 20, z := 12 \}>\):

\[
\frac{< x, \sigma > \downarrow_e 10 \quad \text{Var} \quad < y, \sigma > \downarrow_e 20 \quad \text{Var}}{< x \oplus y, \sigma > \downarrow_e 30 \quad \text{Arith} \quad < z, \sigma > \downarrow_e 30 \quad \text{Arith}}
\]

\[
< 10 + 20 + 30, \sigma > \downarrow_e 60
\]

**Problem 3)** The \(\text{Arith}\) rule has a \(e_1 \oplus e_2\) and \(v_1 \oplus v_2\) in it, but the proof tree only has something corresponding to \(e_1 \oplus e_2\). What happened to \(1 \oplus v_2\)?

**Problem 4)** Suppose we told you that addition were right associative instead of left associative. What would the proof tree look like in that case?
Problem 5) Write the proof tree for the following program.
\[ t := a; a := b; b := t, \{ a := 5, b := 10 \} \]

Problem 6) Write the proof tree for one of the following programs:
- \(<\text{if } x > y \text{ then } m := x - y \text{ else } m := y - x \text{ fi}, \{ x := 10, y := 30 \}\) 
- or 
- \(<\text{while } x > 1 \text{ do } x := x/2 \text{ od}, \{ x := 2 \}\)
Make your own rules!

Problem 7) Write a rule to explain the when B S statement. It executes S only if B is true.

Problem 8) Write a rule for do S while B od. It is like while, but executes S at least one time.
Big Step Semantics Activity--- Reflector’s Report

<table>
<thead>
<tr>
<th>Role</th>
<th>Responsibility</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manager</td>
<td>Keeps team on track</td>
</tr>
<tr>
<td>Recorder</td>
<td>Records decisions</td>
</tr>
<tr>
<td>Reporter</td>
<td>Reports to Class</td>
</tr>
<tr>
<td>Reflector</td>
<td>Assesses team performance</td>
</tr>
</tbody>
</table>

1. What was a strength of your team's performance for this activity?

2. What could you do next time to increase your team's performance?

3. What insights did you have about the activity or your team's interaction today?