Semantics of Programming Languages 2007
Worksheet2: Operational Semantics

Matthew Hennessy

1. In the lectures, we saw a big-step operational semantics for the language *Exp*. Use it to find a numeral $\mathtt{n}$ such that $(\mathtt{4} + \mathtt{1}) + (\mathtt{2} + \mathtt{2}) \Downarrow \mathtt{n}$.

   You should give the full derivation of the conclusion.

2. In the lectures, the big-step operational semantics was only for addition. Extend the big-step semantics to incorporate *multiplication*.

3. Using your big-step semantics, give a proof that $((\mathtt{3} + \mathtt{2}) \times (\mathtt{1} + \mathtt{4})) \Downarrow \mathtt{25}$.

4. Extend the big-step semantics further to incorporate *subtraction* as an arithmetic operation. Remember that the numerals in the syntax of the language are $\mathtt{0}$, $\mathtt{1}$, $\mathtt{2}$ and so on; that is to say, there are no negative numbers.

   How is an expression such as $(\mathtt{3} - \mathtt{7})$ evaluated in your semantics? Have you made any arbitrary decisions about this? If so, what other options were available?

5. Give the full derivation in the small-step semantics of the first step of evaluation of $((\mathtt{1} + \mathtt{2}) + (\mathtt{4} + \mathtt{3}))$.

   In other words give the full derivation of the step

   $$((\mathtt{1} + \mathtt{2}) + (\mathtt{4} + \mathtt{3})) \to E$$

   for some expression E.

6. Write down all the steps of evaluation needed to reduce the above expression to $\mathtt{10}$.

   Give the full derivation for each of these steps.

7. Find some E such that $E \to ((\mathtt{1} + \mathtt{2}) + \mathtt{7})$. You should give the full derivation of the step.

8. There are in all five expressions E of our language *Exp* (with addition as the only operator) such that $E \to ((\mathtt{1} + \mathtt{2}) + \mathtt{7})$. What are they ?

9. How many steps of evaluation are needed to get a final answer from the expression $((\mathtt{1}+\mathtt{1})+\mathtt{1})$? What about if we add another $\mathtt{1}$? What can you say in general about the number of steps needed to evaluate an expression fully?

10. Extend the denotational semantics of *Exp* to include subtraction. How will you handle expressions which ought to produce negative numbers now? Are the choices available to you any different from the choices you could have made in part 4 above?

11. Here is the abstract syntax for a simple language of boolean expressions:

$$\begin{array}{rcl} B \in \mathit{Bool} & ::= & \texttt{true} \mid \texttt{false} \mid B \& B \\ & \mid & \neg\, B \mid \texttt{if}\ B\ \texttt{then}\ B\ \texttt{else}\ B \end{array}$$

Intuitively every expression evaluates either to `true` or `false`

Give a big step semantics for *Bool*.

12. Show the full derivation in your bigstep semantics of

$$\begin{array}{ll} \neg\,(\texttt{if} & (\texttt{false}\&\texttt{true}) \\ \texttt{then} & (\texttt{if}\ \mathit{true}\ \texttt{then}\ (\texttt{false}\&\texttt{true})\ \texttt{else}\ \texttt{false}) \\ \texttt{else} & \neg\,\texttt{true} \\ &) \end{array}$$

13. Give a small-step semantics for *Bool*.