HW3 – Proof Trees and Operational Semantics

CS 476, Fall 2018
Due Sep. 28

1 Instructions

This assignment is to be completed by hand (or in LaTeX if you know how to use it). Submit your answers as a PDF file via Gradescope. If you don't have easy access to a scanner, you can use the one in SEO 1120, the main CS office – the staff will be happy to help you. As always, please don't hesitate to ask for help on Piazza (`https://piazza.com/class/jkh8q52qrh06v`).

2 Operational Semantics of IMP

Here are the operational semantics rules for a simple imperative programming language, using the "hybrid style" of big steps for expressions and small steps for commands.

$$\frac{(n \text{ is a number})}{(n, \sigma) \Downarrow n} \qquad \frac{(b \text{ is a boolean})}{(b, \sigma) \Downarrow b} \qquad \frac{(\sigma(x) = v)}{(x, \sigma) \Downarrow v}$$

$$\frac{(e_1, \sigma) \Downarrow v_1 \qquad (e_2, \sigma) \Downarrow v_2 \qquad (v_1 \oplus v_2 = v)}{(e_1 \oplus e_2, \sigma) \Downarrow v} \text{ where } \oplus \text{ is an arithmetic or boolean operator}$$

$$\frac{(e, \sigma) \Downarrow \text{true} \qquad (e_1, \sigma) \Downarrow v}{(\text{if } e \text{ then } e_1 \text{ else } e_2, \sigma) \Downarrow v} \qquad \frac{(e, \sigma) \Downarrow \text{false} \qquad (e_2, \sigma) \Downarrow v}{(\text{if } e \text{ then } e_1 \text{ else } e_2, \sigma) \Downarrow v}$$

$$\frac{(e, \sigma) \Downarrow v}{(x := e, \sigma) \to (\text{skip}, \sigma[x \mapsto v])} \qquad \frac{(c_1, \sigma) \to (c_1', \sigma')}{(c_1; c_2, \sigma) \to (c_1'; c_2, \sigma')} \qquad \frac{}{(\text{skip}; c_2, \sigma) \to (c_2, \sigma)}$$

3 Problems

There are four problems in all. Each problem is on a separate page. Use as much space as you need for each problem. You can add extra pages if you need to.

1. (6 points) Using the rules above, construct a proof tree showing that $(\mathrm{x} + (2 * \mathrm{y}), \{x = 2, y = 3\}) \Downarrow 8$. In other words, show that $\mathrm{x} + (2 * \mathrm{y})$ evaluates to 8 in the state where $x = 2$ and $y = 3$.

2. (3 points) Construct a proof tree showing that

$$(z := x + (2 * \mathrm{y}); x := \text{if } z = 7 \text{ then } 3 \text{ else } 4, \{x = 2, y = 3\}) \rightarrow$$
$$(\text{skip}; x := \text{if } z = 7 \text{ then } 3 \text{ else } 4, \{x = 2, y = 3, z = 8\})$$

You can write "P1" to stand for the proof tree from the previous problem.

3. (7 points) Construct a proof tree for the next step that

$$(x := \text{if } z = 7 \text{ then } 3 \text{ else } 4, \{x = 2, y = 3, z = 8\})$$

takes.

4. (9 points) Suppose we extended the language with a command "c_1 andthen c_2 if e" that behaves as follows:

   - First, it executes c_1 normally.
   - If e is true in the resulting environment, it then executes c_2.
   - Otherwise, it ignores c_2 and is finished executing.

   In other words, to execute c_1 andthen c_2 if e, first execute c_1 normally, and then execute c_2 only if e is true.

   Give small-step semantic rules for c_1 andthen c_2 if e. Remember that a command becomes "skip" when it is finished executing. As a test case, if you've written your rules correctly,
   $x := 3$ andthen $y := 4$ if $x = 3$ should step to $(\text{skip}, \{x = 3, y = 4\})$ in three small steps.

   Hint: it is probably easiest to define the command using three separate rules.