Practice Problems - Lecture 16 - Fri Feb 17

Nano1 Semantics

What is the next step in the reduction of each of these terms? Prove it by building the derivation tree using our rules for the operational semantics of Nano1.

1) `0 + (2 + (1 + 3))`
2) `let x = 5 in x + let x = 3 in x`

NanoB Semantics

Let's consider another very simple language (call it NanoB) where we just have true, false, and a conditional expression (but no integers). Here is the syntax:

```
e ::= v                              -- values
    | if e1 then e2 else e3          -- conditionals

v ::= true
    | false
```

Here are the operational semantics rules:

```
                              e1 => e1'
[If]        -------------------------------------------------
             if e1 then e2 else e3 => if e1' then e2 else e3

[If-True]   if true then e2 else e3 => e2

[If-False]  if false then e2 else e3 => e3
```

3) What is the next step in the reduction of each of this term? Write down the derivation tree.

   `if (if false then true else false) then false else true`

4) Notice that in our operational semantics above the guard is evaluated first and at most one of the branches is ever evaluated. Suppose we wanted to change our operational semantics so that the then and else branches are evaluated *before* the guard is evaluated. Write a set of reduction rules that give an implementation of this semantics. What kind of design decisions do you have to make if you want these semantics to be deterministic?