Alec Thilenius

Lab 3 Writeup for CSCI 3155

1

```
const innerFunction = function(x) { return x + 1; };
const outerFunction = function(x) { return innerFunction(x); };
outerFunction(123);
```

This requires that the scope of outer function container the entire scope of inner function. That is, it requires in the outer funciton that the ‘variable’ innerFunction be captured (substituted).

3-C

Yes, because the search rules (Search*) strictly define it as such, going from left to right. In english this means when we get an expression like e1 + e2, we evaluate e1 first, all the way down to a terminal before going on to e2 (aka right to left).

4

We first consider the entire expression, e going to e’

$$\frac{e1{-}{>}e1'}{e1\ bop\ e2\ {-}{>}\ e1'\ bop\ e2}$$

We then must define that e2 can only be evaluated once e1 has reached a terminal, aka once it’s a number.

$$\frac{e2{-}{>}e2'\ bop\ \{+,-,*,<\}}{n1\ bop\ e2\ {-}{>}\ n1\ bot\ e2'}$$

Lastly we perform the binary operation on the two terminals (the two numbers)

$$\frac{n'{=}n1{+}n2}{n1{+}n2{-}{>}n'}$$

To reverse the evaluation order all be need to do is evaluate e2 first, ensure e1 can’t be evaluated until e2 hits a terminal and then evaluate the binary operation on the two terminals. We only need to change the first and second

$$\frac{e2{-}{>}e2'}{e1\ bop\ e2\ {-}{>}\ e1\ bop\ e2'}$$

$$\frac{e1{-}{>}e1'\ bop\ \{+,-,*,<\}}{e1\ bop\ n2\ {-}{>}\ e1'\ bot\ n2}$$

5-A

The most common case is binary AND (&&) or two expressions.

‘`if (false && complexOperation()) {...`’

If the first expression is false, then we need not evaluate the second expression as there is no possible value it could be that would make the overall operation true.

5-B

Yes. This is explicitly stated in

$$\frac{}{\mathit{false}\ \&\&\ \mathit{e2} \ -\!\!> \mathit{false}}$$

as e2 never goes to e2’