I learned a lot by doing this lab, I had a lot of fun trying to make things more efficient, although this did take a bit of extra time. I am still a bit confused as to how to do the MapFirst functions without just doing recursion. Most of the cases were pretty easy, I think the hardest were call and function.

- This assignment was a little stressful because we had to re-upload the new lab and go through the handout/original code to see what had changed.

- The syntax for function calls, and library functions for lists and maps was very hard to get a grip on (and there’s still a lot I’m a little unsure about when it comes to the syntax).

- A single small not operator in the wrong spot meant 7 cases didn’t work. Just finding that. The rest was quite enjoyable.

- I really had trouble understanding Call, Function and Object.

- The Function calls and substitution were a little more tricky than the rest.

- The hardest part was using the scala libraries.

- It was difficult some times because I would know what the code was supposed to do and I knew how it had to do it but I struggled with how to do it.

- It’s part of the “algebraic” learning curve.

- I really like type systems and this is the first time I’ve implemented type inference; I was pleasantly surprised at how it simplified program evaluation.

- It would help if the instructions included more details as to what needs to be implemented. I felt like I had to run the program just to see what would break in order to know what I needed to implement next.

- It would be easier if the instructions at least had a list of the missing parts in step, substitute, etc.

- Really liking the functional programming though.

- I feel much more comfortable working with formal semantics after this assignment. But knowing how to implement them is still quite challenging.

- The auto tester has very good reliability for this assignment but was down 20 min before the deadline which was frustrating.

Agreed, we’ll keep doing our best with this.

"The List": null, vor, fun wl const/ur, fun wl ref, name, interface, cast

Questions

1) substitute function to use it
2) Dollar
3) Big Picture
4) Imperative Updates
DoCallVar

\[ a \in \text{dom}(M) \]

\[ \frac{m}{(\text{fun} \ (\text{var} \ x : \ T) \ e_1) \ (v_2)} \rightarrow \]

\[ \langle M[a \rightarrow v_2], e_1[\#a/x] \rangle \]
Call-by-Name

\[
\text{const } f = \text{function (name } x \text{) } 1 \\
f( \text{dumbFib}(200742))
\]

\[
(1)[ \text{dumbFib}(200742) ] / x ]
\]

\[
= 1
\]

\[
\text{const } f = \text{function (name } x \text{) } x + x \\
f( \text{dumbFib}(200742))
\]

\[
\rightarrow (x + x)[ \text{dumbFib}(200742) / x ]
\]

\[
= \text{dumbFib}(\ldots) + \text{dumbFib}(\ldots)
\]
e^{c_{\text{sub}}/x}