Lab 10: Dynamically Scoped Semantics

```
<prog> ::= ε | <stmt>; <prog>
<stmt> ::= <var> = <num>
        |  <var> = $<var>
        |  <var>() { <prog> }
        |  <var>
```

What states do these programs terminate in?

```
f() { x = $y; };
y = 2;
g() { y = $x; };
f; g;
```

$\{f \mapsto x = \$y;\ \ y \mapsto 2;\ \ g \mapsto y = \$x;\ \ x \mapsto 2\}$

```
F(){Y=$X;};G(){X=0;};G;F;X=1;F;
```

$\{F \mapsto Y = \$X;\ G \mapsto X = 0;\ X \ \mapsto\ 1\,;\ Y \ \mapsto\ 1\}$

- A **prog** is a sequence of **stmts**, executed in order
- A **stmt** can assign a **var** to an **int** literal
  or assign a var to the (**int**) value of a **var**,
  or declare a **fun**, or invoke a **fun**

- An **env** is a finite map from **vars** to **values**
- A **value** is either an **int** or a **fun body** (i.e. a **prog**)
- **Config** takes the form $\langle \mathcal{E}, p \rangle$ for **env** $\mathcal{E}$ and **prog** p

Write the small-step rules for this **dynamically-scoped** imperative language.

- Judgments should take the form $\langle \mathcal{E}, p \rangle \longrightarrow \langle \mathcal{E}', p' \rangle$

Write the big-step semantics as well.

- Judgments should take the form $\langle \mathcal{E}, p \rangle \Downarrow \mathcal{E}'$

With time remaining, implement in OCaml.
(Skeleton provided in repo, as well as solution)

Small-Step Semantics

```
<prog> ::= ε | <stmt>; <prog>
<stmt> ::= <var> = <num>
         |  <var> = $<var>
         |  <var>() { <prog> }
         |  <var>
```

$$\frac{}{\langle \mathcal{E},\ x = n; p\rangle \longrightarrow \langle \mathcal{E}[x \mapsto n],\ p\rangle}\ \text{AssnInt}$$

$$\frac{\mathcal{E}(x') = n \in \mathbb{Z}}{\langle \mathcal{E},\ x = \$x'; p\rangle \longrightarrow \langle \mathcal{E}[x \mapsto n],\ p\rangle}\ \text{AssnVar}$$

$$\frac{}{\langle \mathcal{E},\ f(\,)\ \{p'\}; p\rangle \longrightarrow \langle \mathcal{E}[f \mapsto p'],\ p\rangle}\ \text{Decl}$$

$$\frac{}{\langle \mathcal{E},\ f; p\rangle \longrightarrow \langle \mathcal{E},\ \mathcal{E}(f)\ p\rangle}\ \text{Call (alternative)}$$

$$\frac{\mathcal{E}(f) = p' \in \text{prog}}{\langle \mathcal{E},\ f; p\rangle \longrightarrow \langle \mathcal{E},\ p'\ p\rangle}\ \text{Call}$$

Big-Step Semantics

```
<prog> ::= ε | <stmt>; <prog>
<stmt> ::= <var> = <num>
        |  <var> = $<var>
        |  <var>() { <prog> }
        |  <var>
```

$$\frac{}{\langle \mathcal{E},\, \varepsilon\rangle \Downarrow \mathcal{E}}\ \text{Skip}$$

$$\frac{\langle \mathcal{E}[x \mapsto n],\, p\rangle \Downarrow \mathcal{E}'}{\langle \mathcal{E},\, x = n; p\rangle \Downarrow \mathcal{E}'}\ \text{AssnInt}$$

$$\frac{\mathcal{E}(x') = n \in \mathbb{Z} \qquad \langle \mathcal{E}[x \mapsto n],\, p\rangle \Downarrow \mathcal{E}'}{\langle \mathcal{E},\, x = \$x'; p\rangle \Downarrow \mathcal{E}'}\ \text{AssnVar}$$

$$\frac{\langle \mathcal{E}[f \mapsto p'],\, p\rangle \Downarrow \mathcal{E}'}{\langle \mathcal{E},\, f()\,\{p'\}; p\rangle \Downarrow \mathcal{E}'}\ \text{Decl}$$

$$\frac{\mathcal{E}(f) = p' \in \text{prog} \qquad \langle \mathcal{E},\, p'\, p\rangle \Downarrow \mathcal{E}'}{\langle \mathcal{E},\, f; p\rangle \Downarrow \mathcal{E}'}\ \text{Call}$$

$$\frac{\begin{array}{c}\mathcal{E}(f) = p' \in \text{prog} \\ \langle \mathcal{E},\, p'\rangle \Downarrow \mathcal{E}' \qquad \langle \mathcal{E}',\, p\rangle \Downarrow \mathcal{E}''\end{array}}{\langle \mathcal{E},\, f; p\rangle \Downarrow \mathcal{E}''}\ \text{Call (alternative)}$$