Exercise 1F-2

I resonate the most with Hoare's opinion on dividing a difficult task: "often just as difficult is deciding how to do it, -- how to divide a complex task into simpler subtasks, and to specify the purpose of each part, and define clear, precise, and efficient interfaces between them."

During my previous internship, my project over the summer was to develop a complex feature on the web server. At first, I decided to implement a working version as soon as possible, so that I could have more time in the end to have my collegues critique my work. I did not break down the project into separate subtasks. Instead, I worked on it as a whole to make sure that I could finish it in a timely manner.

Eventually, I did manage to have a working version of the feature soon. However, the code base was mostly unreadable and very hard to maintain. Thus, I decided to discard my progress and started from scratch again. This time, I learned to divide this difficult project into several subtasks and classes that were much simpler and could be more easily commented and maintained. Moreover, having clear and efficient interfaces between the classes also make the system much more stable. It was unfortunately that I had to learn it the hard way, but now I deeply believe that it is very important, as well as difficult, to correctly divide a difficult task.

Exercise 1F-3.

Since there are only integer types, and no float or double type in IMP language, we can just do the division and cast the result back to integer. However, we still need to consider the case where the denominator is 0. In that case, we simply skip the division operation. Let e1 be the numerator, e2 be the denominator. and e3 be the destination for the result.

$$\frac{\langle e_1, \sigma\rangle \Downarrow n_1, \quad \langle e_2, \sigma\rangle \Downarrow n_2 \quad \langle n_2 = 0, \sigma\rangle \Downarrow \text{true}}{\langle \text{if } n_2 = 0 \text{ then } \langle \text{skip}, \sigma\rangle \text{ else } \langle e_3 := n_1/n_2, \sigma\rangle \Downarrow \sigma}$$

$$\frac{\langle e_1, \sigma\rangle \Downarrow n_1, \quad \langle e_2, \sigma\rangle \Downarrow n_2 \quad \langle n_2 = 0, \sigma\rangle \Downarrow \text{false}}{\langle \text{if } n_2 = 0 \text{ then } \langle \text{skip}, \sigma\rangle \text{ else } \langle e_3 := n_1/n_2, \sigma\rangle \Downarrow \sigma[e_3 := n_1/n_2]}$$

Here, we assume that the division operator "/" will cast the value back to integer. We can also add a cast statement (int) if the compiler or interpreter requires.

Exercise 1F-4

let x = e in c :

$$\frac{\langle e, \sigma\rangle \Downarrow n \qquad \langle c, \sigma[x := n]\rangle \Downarrow \sigma'}{\langle \text{let } x = e \text{ in } c, \sigma\rangle \Downarrow \sigma'[x := \sigma[x]]}$$

The main catch is to execute the command c with the updated value of x and then restore back to the original value of x.

Exercise 1F-5.

To account for the let command, we need:

contexts:

$$H ::= \ldots \mid \text{let } x := H \text{ in } c.$$

redexes: Denote the value x evaluates to is n, we have.

$$r ::= \ldots \mid \text{let } x := n \text{ in } c.$$

reduction rules:

$$\langle \text{let } x := n' \text{ in } c, \sigma \rangle \rightarrow \langle tmp := x;\ x := n';\ c;\ x := tmp, \sigma \rangle.$$

1 HW1 (select all pages: your first page has your name and bookkeeping, and all others are anonymous))

- **0 pts** Correct