CSCI 5535: Homework Assignment 2: Language Design and Implementation

David Baines [*,†,‡]

October 6, 2023

1 Language Design: IMP

1.1. The two judgment forms (one for expressions, the other for functions / commands, respectively e and c in **IMP**, are:

For e:

$$\frac{}{\Gamma \vdash \texttt{addr}[a] : \texttt{num}} \qquad \frac{}{\Gamma \vdash \texttt{num}[n] : \texttt{num}} \qquad \frac{}{\Gamma \vdash \texttt{bool}[b] : \texttt{bool}} \qquad \frac{\Gamma \vdash e_1 : \texttt{num} \qquad \Gamma \vdash e_2 : \texttt{num}}{\Gamma \vdash \texttt{plus}(e_1; e_2) : \texttt{num}}$$

$$\frac{\Gamma \vdash e_1 : \texttt{num} \qquad \Gamma \vdash e_2 : \texttt{num}}{\Gamma \vdash \texttt{times}(e_1; e_2) : \texttt{num}} \qquad \frac{\Gamma \vdash e_1 : \texttt{num} \qquad \Gamma \vdash e_2 : \texttt{num}}{\Gamma \vdash \texttt{eq}(e_1; e_2) : \texttt{bool}} \qquad \frac{\Gamma \vdash e_1 : \texttt{bool} \qquad \Gamma \vdash e_2 : \texttt{bool}}{\Gamma \vdash \texttt{eq}(e_1; e_2) : \texttt{bool}}$$

$$\frac{\Gamma \vdash e_1 : \texttt{num} \qquad \Gamma \vdash e_2 : \texttt{num}}{\Gamma \vdash \texttt{le}(e_1; e_2) : \texttt{bool}} \qquad \frac{\Gamma \vdash e : \texttt{num}}{\Gamma \vdash \texttt{not}(e) : \texttt{bool}} \qquad \frac{\Gamma \vdash e : \texttt{bool}}{\Gamma \vdash \texttt{not}(e) : \texttt{bool}}$$

$$\frac{\Gamma \vdash e_1 : \texttt{bool} \qquad \Gamma \vdash e_2 : \texttt{bool}}{\Gamma \vdash \texttt{and}(e_1; e_2) : \texttt{bool}} \qquad \frac{\Gamma \vdash e_1 : \texttt{bool} \qquad \Gamma \vdash e_2 : \texttt{bool}}{\Gamma \vdash \texttt{or}(e_1; e_2) : \texttt{bool}}$$

For c:

$$\frac{\Gamma \vdash e : \texttt{bool}}{\Gamma \vdash \texttt{not}(e) : \texttt{bool}}$$

1.2. (a) For the evals:

$$\frac{}{\texttt{addr}[a]\ \mathsf{val}} \qquad \frac{}{\texttt{num}[n]\ \mathsf{val}} \qquad \frac{}{\texttt{bool}[b]\ \mathsf{val}}$$

[*] https://courses.cs.cornell.edu/cs412/2004sp/lectures/lec13.pdf
[†] https://csci3155.cs.colorado.edu/csci3155-notes.pdf
[‡] https://www.hedonisticlearning.com/posts/understanding-typing-judgments.html

For "plus()":

$$\frac{n_1 + n_2 = n}{\texttt{plus}(\texttt{num}[n_1];\texttt{num}[n_2]) \longmapsto \texttt{num}[n]} \qquad \frac{e_1 \longmapsto e_1'}{\texttt{plus}(e_1;e_2) \longmapsto \texttt{plus}(e_1';e_2)}$$

$$\frac{e\ \mathsf{val} \qquad e_2 \longmapsto e_2'}{\texttt{plus}(e_1;e_2) \longmapsto \texttt{plus}(e_1;e_2')}$$

For "times()":

$$\frac{n_1 * n_2 = n}{\texttt{times}(\texttt{num}[n_1];\texttt{num}[n_2]) \longmapsto \texttt{num}[n]} \qquad \frac{e_1 \longmapsto e_1'}{\texttt{times}(e_1;e_2) \longmapsto \texttt{times}(e_1';e_2)}$$

$$\frac{e\ \mathsf{val} \qquad e_2 \longmapsto e_2'}{\texttt{times}(e_1;e_2) \longmapsto \texttt{times}(e_1;e_2')}$$

For "eq()":

$$\frac{n_1 == n_2 \vdash b}{\texttt{eq}(\texttt{num}[n_1];\texttt{num}[n_2]) \longmapsto \texttt{bool}[b]} \qquad \frac{e_1 \longmapsto e_1'}{\texttt{eq}(e_1;e_2) \longmapsto \texttt{eq}(e_1';e_2)} \qquad \frac{e\ \mathsf{val} \qquad e_2 \longmapsto e_2'}{\texttt{eq}(e_1;e_2) \longmapsto \texttt{eq}(e_1;e_2')}$$

For "le()":

$$\frac{n_1 <= n_2 \vdash b}{\texttt{le}(\texttt{num}[n_1];\texttt{num}[n_2]) \longmapsto \texttt{bool}[b]} \qquad \frac{e_1 \longmapsto e_1'}{\texttt{le}(e_1;e_2) \longmapsto \texttt{le}(e_1';e_2)} \qquad \frac{e\ \mathsf{val} \qquad e_2 \longmapsto e_2'}{\texttt{le}(e_1;e_2) \longmapsto \texttt{le}(e_1;e_2')}$$

For "not()":

$$\frac{n_1 !\, n_2 \mid\mid b_1 !\, b_2 \vdash b}{\texttt{not}(\texttt{num}[n_1])\texttt{num}[n_2] \mid\mid \texttt{not}(\texttt{bool}[b_1])\texttt{bool}[b_2] \longmapsto \texttt{bool}[b]} \qquad \frac{e_1 \longmapsto e_1'}{\texttt{not}(e_1)e_2 \longmapsto \texttt{not}(e_1')e_2}$$

$$\frac{e\ \mathsf{val} \qquad e_2 \longmapsto e_2'}{\texttt{not}(e_1)e_2 \longmapsto \texttt{not}(e_1)e_2'}$$

For "and()":

$$\frac{b_1\ \&\&\ b_2 \vdash b}{\texttt{and}(\texttt{bool}[b_1];\texttt{bool}[b_2]) \longmapsto \texttt{bool}[b]} \qquad \frac{e_1 \longmapsto e_1'}{\texttt{not}(e_1)e_2 \longmapsto \texttt{not}(e_1')e_2}$$

$$\frac{e\ \mathsf{val} \qquad e_2 \longmapsto e_2'}{\texttt{not}(e_1)e_2 \longmapsto \texttt{not}(e_1)e_2'}$$

2 Language Implementation: ETPS

See hw02.ml and test_hw02.ml.

3 Final Project Preparation: Pre-Proposal

3.1.

A Syntax of IMP

Typ	τ	$::=$	`num`	`num`	numbers
			`bool`	`bool`	booleans
Exp	e	$::=$	`addr`$[a]$	a	addresses (or "assignables")
			`num`$[n]$	n	numeral
			`bool`$[b]$	b	boolean
			`plus`$(e_1;e_2)$	$e_1 + e_2$	addition
			`times`$(e_1;e_2)$	$e_1 * e_2$	multiplication
			`eq`$(e_1;e_2)$	$e_1 == e_2$	equal
			`le`$(e_1;e_2)$	$e_1 <= e_2$	less-than-or-equal
			`not`(e_1)	$!e_1$	negation
			`and`$(e_1;e_2)$	$e_1 \&\& e_2$	conjunction
			`or`$(e_1;e_2)$	$e_1 \mid\mid e_2$	disjunction
Cmd	c	$::=$	`set`$[a](e)$	$a := e$	assignment
			`skip`	`skip`	skip
			`seq`$(c_1;c_2)$	$c_1; c_2$	sequencing
			`if`$(e;c_1;c_2)$	`if` e `then` c_1 `else` c_2	conditional
			`while`$(e;c_1)$	`while` e `do` c_1	looping
Addr	a				