CSC 526, Spring 2020, Final Project

Purpose: Language Features

Due: 11:59pm, Wednesday, April 15, 2020

Get the assignment code

These instructions assume that your course git repository is set up. Change into your course repository directory and enter the following commands.

```
git fetch assignments
git checkout assignments/master -- final_project
git add final_project
git commit -a
```

This will copy the `final_project` directory into your working directory, start tracking the files in the `final_project` directory, and commit those files to your local git repository.

Assignment Description

The purpose of this assignment is to augment the previous assignments with some additional features of your choice. The features additions should be at a sufficient level of implementation difficulty on par with previous projects.

Preset Final Project

This section details a preset final project that would meet the criteria. The syntax is augmented with the following expressions:

```
expr :=
    ...
    | (while <expr> <expr>)
    | (begin <expr list>)
    | (:= <id> <expr>)
    | (= <expr> <expr>)
```

- `while`: a while loop evaluates the condition (the first expression), then evaluates the body if the condition evaluates to `true`. This is repeated until the condition evaluates to `false`. The entire while loop evaluates to `false`.
- `begin`: the begin is for a sequence of expressions. The expressions are evaluated in sequential order. The result of evaluating the last expression is the result of evaluating the entire sequence.
- `<id> := <expr>`: the assignment expression changes the value stored on the stack for a given variable. The expression evaluates to the assigned value.
- `=`: the `=` operator is for structural equality. If the operands are tuples, then perform an elementwise comparison between the two tuples. Otherwise, perform regular value comparison. Note: this can be partially implemented in the `runtime.c` file.

These features should have appropriate runtime type checking. For example, the condition of the while loop should be a Boolean expression.

Here is an example program that uses some of the new features:

```
(
(define main (input)
  (let ((sum 0))
    (begin
      (while (> input 0)
        (begin
          (:= sum (+ sum input))
          (:= input (- input 1))))
      sum)))
)
```

Other Options

Here is a list of features that are acceptable for the final project that are not necessarily for the faint-of-heart:

- Tail call optimization
- Garbage collection
- Register allocator

Turning in the Assignment

To turn in the assignment execute the following git commands from within your repository:

```
git add <file>
git commit -a
git push origin master
```

where a `git add <file>` command is needed for every file that is required for building the assignment executable. Failure to add any required files will result in a failing grade for the