CSCI3136
Assignment 8

Instructor: Alex Brodsky

Due: 9:00am, Friday, July 12, 2019

In this assignment we will expand the Splat Evaluator to handle variables. This assignment adds two additional expressions to the ones from the previous assignment: `def` and `let`.

The syntax for `def` is

`def` *id expression*

For example, the expressions

```
def foo = 42
def bar = 7 * 13
```

binds `foo` to the value 42 and binds `bar` to 91, which is the evaluation of `7 * 13`. A `def` consists of two parts, an *id* (symbol), *which is not evaluated*, and an expression, whose evaluation is then bound to the *id*. The evaluation of a `def` expression adds the binding to the current frame of the reference environment and yields its value. The example above yields `42` and `91`.

The syntax for `let` is

`let` *id1 = expression1, id2 = expression2, ...* { *expression ...* }

where the `...` denotes zero or more of the preceding item. For example, the expressions:

```
let x = 42 {
   x + 13
}

let x = 42, y = 13 {
   x + y
   let z = 7 {
      x + 13
      x - y - z
   }
}

let  x = 2 * 21 {
   let x = 7 {
      x + 13
   }
   x + x
}
```

evaluate to 55, 22, and 84, respectively. Specifically, `let` comprises 2 parts. The first part is a list of bindings where each binding consists of an *id* and an expression, and the bindings are separated

by commas. The second part is the body of the let, which is enclosed in braces and contains one or more expressions that are evaluated using the bindings defined in the first part. The evaluation of the `let` expression is the evaluation of the last expression in the body. The bindings are only active inside the body of the `let` expression.

In the example above, the first `let` expression binds x to 42 and evaluates `x + 13` to yield 55.

The second `let` expression binds x to 42 and y to 13, evaluates `x + y` and then evaluates the next expression, which is also a `let` expression. This expression binds z to 7 and then evaluates the expressions `x + 13` and `x - y - z`. The evaluation of the latter expression is also the evaluation of the overall expression.

Lastly, the third expression binds x to the evaluation of `2 * 21` and evaluates two expressions. The first expression is also a `let` expression, which rebinds x to 7, and evaluates `x + 13`. Then, the second expression `x + x` is evaluated, using the outer binding of x, yielding 55 as the final result.

1. **[10 marks]** Starting with either your own solution to Assignment 7 or the provided solution (`splat.2.py`), add the following productions to your parser/evaluator:

$$\begin{aligned}
EXPR &\rightarrow DEF \\
&\rightarrow LET \\
DEF &\rightarrow \text{'}\texttt{def}\text{'}\ SYMBOL\ \text{'='}\ EXPR \\
LET &\rightarrow \text{'}\texttt{let}\text{'}\ V_LIST\ BODY \\
V_LIST &\rightarrow SYMBOL\ \text{'='}\ EXPR\ V_TAIL \\
V_TAIL &\rightarrow \epsilon \\
&\rightarrow \text{','}\ V_LIST \\
BODY &\rightarrow \text{'\{'}\ E_LIST\ \text{'\}'}
\end{aligned}$$

2. **[40 marks]** Implement the evaluation of the `def` and `let` expressions in your evaluator.

   Suggestions:

   - You will likely need additional inherited attributes and semantic rules to pass the reference environment down the tree.
   - You should use a linked reference environment instead of a single global reference environment.
   - Each frame of the reference environment can be implemented with a map (hashtable).
   - A `def` should add a binding to the current frame of the reference environment.
   - When evaluating a `let expression`, a new frame should be created and linked into the reference environment. When the `let` expression ends, the frame can be removed.
   - Evaluation of symbols is done by looking up the symbol in the reference environment and the result of evaluation is the value that the symbol is bound to.

   A set of test cases is provided for you to test your evaluator. A test script `test.sh` is provided to run the tests. Since the choice of language is up to you, you must provide a standard script called `runme.sh` to run your interpreter, just like in the previous assignment.

   To submit this part of the assignment please use Brightspace.

CSCI3136: Assignment 8

Summer 2019

Student Name	Login ID	Student Number	Student Signature

	Mark	
Question 1		/10
Question 2		/40
Functionality	/20	
Structure	/20	
Total		**/50**

Comments:

Assignments are due by 9:00am on the due date. Assignments *must* be submitted electronically via Brightspace. Plagiarism in assignment answers will not be tolerated. By submitting their answers to this assignment, the authors named above declare that its content is their original work and that they did not use any sources for its preparation other than the class notes, the textbook, and ones explicitly acknowledged in the answers. Any suspected act of plagiarism will be reported to the Facultys Academic Integrity Officer and possibly to the Senate Discipline Committee. The penalty for academic dishonesty may range from failing the course to expulsion from the university, in accordance with Dalhousie University's regulations regarding academic integrity.