Assignment: Extending the continuation-passing interpreter
CSC 312, “Programming language implementation”
Department of Computer Science · Grinnell College
September 30, 2015

This assignment is an elaboration of exercise 5.11 from the Friedman and Wand book (page 153).

The LETREC language, which the authors use as a basis for their explanation of continuation passing, has no expressions with side effects and so has no need for begin-expressions.

However, exercise 3.15 (on page 74 of the Friedman and Wand book) proposes the addition of a print-expression with a side effect: Evaluating the print-expression outputs the expressed value of its subexpression. (The value of the print-expression itself is always 1.) We can postulate the following simple syntax for print-expressions:

<expression> ::= print <expression>

with print as a new keyword.

We now have a use for begin-expressions, for which the syntax is specified in exercise 4.4 on page 108 of the text; in more conventional Backus-Naur form, it is

<expression> ::= begin <expression> {; <expression>}* end

We have used expressions with this structure in all of the languages from chapter 4 of the text.

The assignment is to write a continuation-passing interpreter for an extension of LETREC—let’s call it ‘SE-LETREC’—by extending the interpreter for LETREC that the authors develop in section 5.1 of the text, making the changes that are needed to accommodate print-expressions and begin-expressions.

Hint: One way to get started is to consider which, if any, of the subexpressions in print- and begin-expressions are in tail position and which are in positions that grow the control context. You’ll need to define a new variant of the continuation datatype for each kind of subexpression that grows the control context.

For reference and testing, the directory /home/stone/courses/languages/code/SE-LETREC on MathLAN contains a full implementation of SE-LETREC that does not use continuation passing. You may use the scanner and parser given there for your interpreter—only the semantics of the language needs to be changed.

This assignment will be due on Monday, October 5. Please submit your solution in hard copy at the beginning of that day’s class session.