CS403: Programming Languages

Notes on Building Language Interpreters

Overview

The process of building a language interpreter for a fully functioning language seems intimidating, especially for students who have taken only two or three CS classes. Yet, the whole process can be broken down into five steps, each of which builds simply and logically on the previous step. It is here that you will begin to appreciate the difference between programming and computer science. To hack out an interpreter without benefit of the science in computer science may well be an overwhelming task. With the science, each step can be easily and quickly implemented by intermediate level programmers. The five steps are

- designing
- scanning
- recognizing
- pretty printing
- evaluating

In the *designing* phase, you will develop a grammar for you language. The grammar specifies how you say things, such as defining and calling a function, in your language.

In the *scanning* phase, you will break up source code into the individual words in the language and identify those words as to their types: keywords, variables, operators and so on.

In the *recognizing* phase, you will use your grammar to build a series of parsing functions, which step through the words generated by the scanning phase, in order to determine if the words have been placed in their proper order.

In the *pretty-printing* phase, you will convert the parsing functions implemented in the recognizing phase so that they produce parse trees which reflect the structure of the expressions in the source code. You will then regenerate the input as a check on how well you understand the parse trees you have built.

In the *evaluating* phase, you will modify the pretty printer so that instead of printing the parse trees, you will execute the expressions contained in those parse trees. You will also use a simple data structure to store the values of variables currently in scope.

At this point, your language will be up and running!