Scheme Compiler

Jackson Schuetzle (jss270)
Ben Smith (bxs566)
Shreyhan Lakhina (sxl1951)

Our group is choosing option 2 which is to **modify the compiler for PG-4 to support the Scheme programming language.** Creating a potentially useful optimization pass after just being introduced to the concept seems a bit optimistic. Therefore, we applied concepts that our team was more familiar with, but in a much more challenging language. Scheme is a purely functional language, so the grammar and parsing trees look much different than a basic subset of C. Functional languages also create an opportunity for many memory-saving optimizations on the state, e.g. tail-call optimizations.

So far, our group has decided on the subset of the Scheme grammar that our compiler will cover. Here is an example subset that we found and will adapt to our project. We want our compiler to handle

- Variable and function definitions (through *define* and *let*)
- Mathematical and logical expressions
- List operators (*car*, *cdr*, *cons*)
- Lambda functions
- Control-Flow statements (*if-else* and *cond*)

Scheme has one of the most simple grammars of any programming language, so the grammar isn't very complicated. Rather, the complexity comes from generating the IR to manage memory properly. The functional property of Scheme means parameter passing and scoping rules must be defined, which adds complexity to the project.

A flex file has been created for the grammar subset that our group chose. The file has been included in the zip submission for the proposal.

Our team plans to test the compiler against very simple programs first in order to test the correctness of single constructs in the grammar. Once we pass the first stage of testing, our goal is to run the compiler on an entire Scheme project that was created for CSDS 345 Programming Language Concepts. This project's use of the Scheme language is very diverse and covers a wide range of the grammar that we intend to implement. We currently don't have tests created for the lexer yet because we are still deciding what testing tools we want to use.