CMPUT 415 - Project Checkpoint 2
Program Documentation

Matthew Low
Anthony Galati
Stevan Clement
Mike Bujold

9 November 2011

1 Handling of Lexical Units

The lex program works in the usual way to match tokens, the only interesting part being how it handles comments that are allowed to spread across multiple lines. To do this, the lexer switches into a 'comment' state where it grabs everything until it sees the end of the comment. This state change was deemed necessary for the line counting to continue while observing comments.

So far we are not doing anything with strings between single quotes, just acknowledging them. Garbage characters are collected silently and lex calls an error function which then notifies the user.

2 Syntax Error Reporting Strategies

Syntactic errors were detected by placing the 'error' token in reasonable positions. The yyerror function then uses the information we store holding the line and column position to provide a reasonable message to the user.

Since line endings have little meaning in the PAL language, we don't use the yerrok macro, as there is no way to match an error to a whole line (to do so may mean an error matches the whole program). As such, if we encounter an error, we don't want to output a bunch of other errors that are immediately caused by the first.

3 Resolving Conflicts in the Bison Grammar

The conflicts in the bison grammar were mainly the result of ambiguities created by multiple similar paths to the grammar's atoms. The most serious example of this was how both "type" and "expr", two similar rules, reduced along separate paths to $\text{simple}_{t}ype.Wewereabletoremovethisambiguity$

Another issue with the grammar was the possibility of declaring an array with type anonymous enum. We replaced this rule with two new rules, one of which allows for instantiation of strings and one which allows only previously declared variables as a type.

4 References Cited