Tichafa Andrew Rinomhota

This my report on the Part1 of our project where we had to write a lexical analyzer for the subset of the language TIPS, which is in itself the subset of the Pascal Programming Language. I used C++ and flex for Part1 the lexical analyzer returns the lexemes in the input program to a parser for further analysis.

The two rules from my rules.l file that I choose to explain are SEMICOLON and IDENT.

My first rule matches exactly one character: the semicolon (;). When the lexer meets the semicolon character, it returns the token TOK\_SEMICOLON, which is used to indicate the end of a statement in the TIPS language. Here is the code for my rules.l file.

";" { return TOK\_SEMICOLON; }

My second rule matches an infinite number of lexemes. It matches any string of characters that start with an Upper Case letter, and then followed by seven additional characters which can be upper letters or numbers. The code is here below.

PROGRAM { return TOK\_PROGRAM; }

The development process of this project I looked at the table of the tips lexemes. So, I started with keywords, datatype specifier lexemes, punctuation, operator and lastly abstraction lexemes. I ran tests on every sample input files given. I did this through the terminal using the command “./tips\_lex input1.in | diff - input1.correct” .