Compiler implementation for programming language. The application is able to:

1. convert any input program into a sequence of tokens, and
2. generate a symbol table for all keywords and user-defined variables using the tree structure.

**Lexical Conventions**

1. Key words are:

"compute", "begin", "end", " Real "," Integer ", "Read", "Write", Calculate", "Number", "Results”

1. Operators and punctuation characters include:

+ - \* / % < <= > >= == != && || ! = ; , . ( ) [ ] { }

1. A Boolean constant is either true or false.
2. An integer constant can either be specified in decimal (base 10) or hexadecimal (base 16). A decimal integer is a sequence of at least one decimal digits (0-9)
3. A double constant is a sequence of at least one digit, a period, followed by any sequence of digits, may be none. Thus, .99 is not a valid double but both 0.99 and 99. are valid.
4. A string constant is a string of letters surrounded by double quotes. Except for a newline or a double quotation, strings can include any character. A string must start on a single line and conclude on a single line; it cannot be divided across several lines.
5. Blanks, newlines, and tabs are examples of white space. White space is used to divide tokens, but it is not used elsewhere. White space or a token that is neither a keyword nor an identifier must be used to separate keywords and identifiers.
6. A single-line comment is started by // and extends to the end of the line

For instance a sample input program is as follows:

Program  
begin  
Count:=29.31+(Number-71);  
Number:=Account\*10.99;  
Result:=Number+Account;  
end

The output of the project should be:

Lexeme | ID | Token | Type | Repetitions | Line

Identifiers

Node::token:::id:800, rep:0, token:Identifiers, type:Int, lex:Program, line:1::::

Node::token:::id:801, rep:0, token:Identifiers, type:Int, lex:begin, line:2::::

Keywords

Node::token:::id:600, rep:0, token:Keywords, type:Boolean expression, lex:end, line:6::::

Int Numbers

Float Numbers

Simple Characters

Node::token:::id:59, rep:0, token:Simple Character, type:Semicolon, lex:;, line:3::::