Compilers-II(CS3423)

Assignment II - Lexical Analyzer

Name: Aditya Bacharwar

Roll no: Es21btech11003

Compilation Steps and Running Instructions :

1. Main source files : **lex_source_program.l, yacc_source_program.y**
2. Command to Invoke yacc tool on yacc source program : **"yacc -d yacc_source_program.y"**
   - **Yacc** is used to generate parsers for processing and interpreting the syntax of programming languages or other structured data.
   - **-d** ; This is an option that instructs yacc to generate a header file, typically named **y.tab.h**. This header allows lex_source_program to return appropriate tokens to parser.
   - This command also generates a C program(**y.tab.c**) for accurately parsing the input on the basis of given grammer.
3. Command to Invoke flex tool on lex source file : **"flex lex_source_program.l"**
4. Command to compile the lex.yy.c and y.tab.c files using gcc C-compiler : "**gcc lex.yy.c y.tab.c**"
   - **It outputs a parser executable file 'a.out'**
5. **Finally after proper compilation we can run the executable with the input files to parse.**
6. **To run the executable file 'a.out' with input file as command line argument :**
   "./a.out 'relative path to input file' "
   - The parser reads the contents of this file and performs lexical analysis and syntax analysis on it.
   - The outputs two files in **'TP2'** folder : 1) seq_tokens_i.txt 2) parsed_1.parsed
   - **seq_tokens_i.txt** - sequence of tokens generated while doing lexical analysis
   - **parsed_i.parsed** - Corresponding Clike code with each statement labeled with appropriate type of statement label.
   - Example command : ./a.out TPP/1.clike

Known Limitation :

1. Parser Implementation doesn't account for any semantic checks such as type checking, return types etc.
2. In case of errors, Parser only tells about the line no. on which the error has occurred and the last token it got, it doesn't state anything about the tokens it was expecting.
3. Negative integers constants are not parsed and are considered as invalid tokens

Assumptions :

1. The new added keywords: "global" , "local", "declare", "expr", "call", "class", "this" : are assumed to be reserved keyword tokens
2. "add", "sub", "mul", "div", "=", "->" are considered to be operator tokens.
3. "Class_{identifier} is tokenized as ID in seq_of_token file
4. Identifier definition is similar to standard C definition and not that of assignment 1.

{**Note** : 1. while running yacc command, -d flag is necessary as lex file imports "y.tab.h" as a header file}

{ 2. Public test cases has been updated to adhere changes in Assignment specification }