1. The ***syntax\_analyzer.y*** file is the bison file where the necessary grammars for a C function and the rules for those grammars are described to generate a syntax tree from ***input.txt*** file, and logs the tree structure to an output file named ***my\_log.txt*** file. On the main method of this file, there is a ***yyparse()*** function which is responsible for the parsing and construction of the syntax tree.
2. The ***lex\_analyzer.l*** file is here to generate the tokens.
3. The ***symbol\_info.h*** file which contains the symbol\_info class is designed to store information about symbols in a language, including their names, types, and associated syntax tree nodes.
4. The ***TreeNode.h*** file contains the TreeNode class which is designed to represent nodes in a tree structure. It provides methods for creating nodes, managing child nodes, and traversing the tree structure in a post-traversal order.
5. The ***script.sh*** file is for compiling and running a parser and lexer for syntax analysis.

* Firstly, the yacc command is being used to generate a parser from the grammers defined in the syntax\_analyzer.y file.
* Secondly, the generated parser C file which is y.tab.c is being compiled using g++ compiler after which the parser object file has been generated.
* Thirdly, to generate a lexical analyzer from the rules defined in the lex\_analyzer.l file the flex command is being run which will create a lex.yy.c file.
* After that, for compiling the lex.yy.c file we are using g++ compiler and the scanner object file is being generated.
* Next,using the g++ y.o l.o command, the parser and lexer object files (y.o and l.o) are linked together using the g++ compiler to create an executable program.
* Finally, as the program is ready to be executed, using the command "./a.exe input.txt", the compiled program along with the input.txt file is being run for the syntax analysis process.