Stack based Compiler

Objective

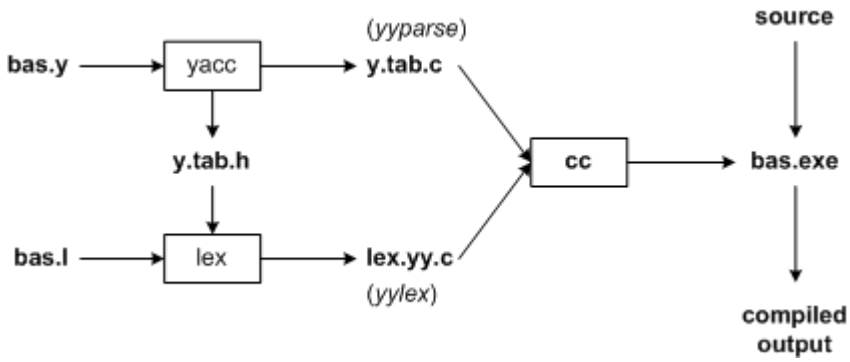
To build a Stack based Compiler.

Description

Lex and yacc are tools used to generate lexical analyzers and parsers.

- Lex : Lex reads the patterns and generates C code for a lexical analyzer or scanner. The lexical analyzer matches strings in the input, based on the patterns, and converts the strings to tokens. Tokens are numerical representations of strings, and simplify processing.

  When the lexical analyzer finds identifiers in the input stream it enters them in a symbol table. The symbol table may also contain other information such as data type (integer or real) and location of each variable in memory. All subsequent references to identifiers refer to the appropriate symbol table index.
- Yacc : Yacc will read the grammar and generate C code for a syntax analyzer or parser. The syntax analyzer uses grammar rules that allow it to analyze tokens from the lexical analyzer and create a syntax tree. The syntax tree imposes a hierarchical structure the tokens.



Building a Compiler with Lex/Yacc

Team Members

Mayank Tiwari (11BCE1016)
Divyansh Jain (11BCE1085)
Zeeshan Vohra (11BCE1106)