AlmostC Almost Compiler User Manual

Overview

This compiler is used to compile programs written in the AlmostC programming language. This programming language was designed and developed by Professor Erik Steinmetz. This compiler was written by Maxwell Herron. The compiler does not actually run the program, it simply generates the MIPS assembly language code that is then able to run the program in your preferred MIPS simulator.

Usage

Using the compiler is quite simple. All you have to do is run the jar file Compiler.jar and pass in your AlmostC file as a command line argument. To do this, you enter the command `java -jar Compiler.jar path/to/your/file.ac`. If the file is a valid AlmostC file and produces no errors, then four files will be generated in the directory you ran this command in: `syntax_tree_output.st`, `symbol_table_output.st`, `semantic_analyzer_tree.st`, and `program.asm`. The syntax tree output is simply a stringified representation of the syntax tree generated by the input program. The symbol table output is a table of all the identifier names, along with the kind of identifier i.e. variable, function, etc. The semantic analyzer tree output is similar to the syntax tree output, but it displays the data type of nodes in the tree where it is appropriate. Finally the program file is what you will then be loading into your preferred MIPS simulator, as it is the runnable version of your input file.

Common Errors

This compiler contains errors that will break the compilation process, along with errors that simply disallow your program to be converted into a legal assembly program. The runtime errors can occur in the scanning and parsing phase of compilation, while the errors that do not break the process occur in semantic analysis of the program. Below is the listing of specific errors.

- BadCharacterException - This occurs in the scanning phase of compilation. This occurs when an illegal character is detected in the program. The compiler will tell you what the character is, and also the row and column that it appears in.

  Example: `Error BadCharacterException: Illegal char: '#' found. at line 1 column 4`

- ParserException - This occurs in the parsing phase of compilation. This occurs when the grammar of the language is not properly followed. The compiler will let you know the line and column of the character that triggered the exception.

  Example: `Error Factor at line 1 column 9`

- Undeclared variable - This occurs during the semantic analysis phase of compilation. This will not break the program, but it will make it so that an output assembly file is not generated. It is caused by the use of an undeclared variable.

  Example: `Undeclared variable: foo.  Please declare this variable.`

- Type mismatch - This occurs during the semantic analysis phase of compilation. This will not break the program, but it will make it so that an output assembly file is not generated. It is caused by the mismatched assignment of a float to an int.

  Example: `Type mismatch detected: left node is of type: int`

  `Expression is of type: float`