Datum

Syntax Analyzer

Building Parse Tree

- In the previous phase, we extracted the tokens from the input program. We pass these tokens into our syntax analyzer to get the parse tree.

- The parse tree is built using a grammar which ensures that the syntax is correct according to our language.

- The grammar is free of ambiguity and shift-reduce conflicts.

Building the Grammar

The structure and flow of the grammar is as follows:

- Assignment Statements
- Variable declarations
- Function declarations
- Conditional Statements
- Loop Statements
- Integration of all sections

Testing

- Testing was done with multiple meaningful codes.
- Each section of grammar was tested individually by trying out various possible cases.
- Multiple syntactically incorrect codes were also used to test the grammar.

Demo for the Grammar

https://drive.google.com/file/d/1l6nSV4ncslJxtgoyPSqjof4FVVCWKTdz/view?usp=sharing

Code Link

https://github.com/ksananth4424/Datum/tree/main/parser

References used

ANSI C Grammar: https://www.lysator.liu.se/c/ANSI-C-grammar-y.html