Project 1: Syntax Analyzer

Austin Johns

Professor Nasser Tadayon

CST-405

**Planning**

The planning of this project is done mostly in the initial project of this class. The initial project was to submit a lexical analyzer in a coding language of choice. For the coding language, I chose Python, so I have also completed my parser in Python. The planning of this project has included all of the tokens that are produced using the lexical analyzer. The lexical analyzer initially places the letters and numbers into tokens that can be read by a parser. The parser then produces a parse tree to help the syntax analyzer better understand what is being read into the compiler.

**Implementation**

The implementation of the project begins with the removal of left recursion. Then, the parse table is created based on the .gcupl file grammar. After this, the parse table is implemented into the Python code. The python code reads in the tokens taken from the lexical analyzer and understands how to process the information based on this grammar. Once the grammar and input are processed by the program, a parse tree is created to show an order of operations that exists in the compiler being created.

**Using Analyzer**

Using the analyzer is quite simple. First the user inputs an expression into the lexical analyzer. Here, the lexical analyzer understands what tokens represent each character of the input. Based on the provided grammar, the parser reads in the identified tokens. These tokens are put into a parse tree based on this grammar. The parse tree shows the order in which the tokens of the input are processed, and with what operands are used for the input’s processing.

**Main Purpose**

The main purpose of the parser is to help organize and understand the input provided by the user. The user inputs different numbers, operands, variables, etc. and the parser’s job is to follow explicit rules to show in what way the operands and processing should be carried out. The parser can follow any type of grammar, or rules, and the outcomes may differ per grammar provided.

**Code Skeleton**

**Text

Description automatically generated**

This was the best I could do.

**Execution Screenshots**

**Text

Description automatically generated**

**References**

7.6. Parse Tree¶. (n.d.). Retrieved October 05, 2020, from https://runestone.academy

/runestone/books/published/pythonds/Trees/ParseTree.html