Context Free Grammar For Input of a Calculator

Student Name 1st: Md. Ariful Islam, Student Name 2nd : Partha Dep Sarkar, Student Name 3rd: Rifayet Azam Talukder

Student ID 1: 2016-1-60-019, Student ID 2: 2016-1-60-042, Student ID 3: 2016-1-60-

Department of Computer Science and Engineering, East West University, Dhaka-1212, Bangladesh.

Email: arifewucse2016@gmail.com

***Abstract*—: The goal of the study is to learn about the parser generator, how a parse tree is generated and how the grammar creates a parse tree. We have tried to implement some basic features of context free grammar to see the working procedure of the parser generator for input of a calculator.**

# **Introduction**

Having some working knowledge of compiler and its design is must for a programmer. In our project we have tried to display whatever we have learned about compiler. Our project is about ‘Context Free Grammar For Input of a Calculator’. We have used ANTLR tool to implement our grammar for calculating mathematical operation. We have created our own grammar and dictionary to complete the task.

# **Related work**

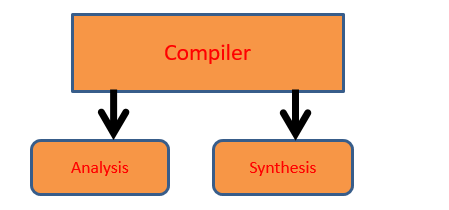
Calculating mathematical operation is not a new thing. We have developed a context free grammar which calculates almost all of mathematical operation. We can calculate summation, multiplying, division, exponential and log base operator.

# **Compiler Task**

Any compiler must complete two major tasks. Those are:

i)**Analysis** of the source program

ii)**Synthesis** of a machine-language program



# **Context Free Grammar**

# A context-free grammar (CFG) is a set of recursive rewriting rules (or *productions*) used to generate patterns of strings. design

A CFG consists of the following components:

* a set of *terminal symbols*, which are the characters of the alphabet that appear in the strings generated by the grammar.
* a set of *nonterminal symbols*, which are placeholders for patterns of terminal symbols that can be generated by the nonterminal symbols.
* a set of *productions*, which are rules for replacing (or rewriting) nonterminal symbols (on the left side of the production) in a string with other nonterminal or terminal symbols (on the right side of the production).
* a *start symbol*, which is a special nonterminal symbol that appears in the initial string generated by the grammar.
* To generate a string of terminal symbols from a CFG, we:
* Begin with a string consisting of the start symbol;
* Apply one of the productions with the start symbol on the left hand size, replacing the start symbol with the right hand side of the production.
* Repeat the process of selecting non terminal symbols in the string, and replacing them with the right hand side of some corresponding production, until all nonterminal have been replaced by terminal symbols.

# **OUR GRAMMAR**

grammar experiment;

start: (e NL)\* ;

e: e '^' e| e ( '\*' | '/' ) e | e ( '+' | '-' ) e | '(' e ')' | f e |INT ;

f: 'log' | 'ln' | 'e' | 'sqrt' |'sin' | 'cos' | 'tan' ;

NL : [\r\n]+ ;

INT : [0-9]+ ;

# **Proposed Work**

The grammar program file (experiment.g4) is executed using the ANTLR tool and the generated tokens files are run with java and it generates the class files. After executing the java file an input of an English sentence is required. After the input the java program generates a parse tree and shows that tree using Graphical User Interface (GUI). That parse tree will contain attributes of the given string. A simple demonstration of our work is presented below. The input are:

Frist example:4+5\*8^2/3+(log(10)) , second example: 19+42\*78^5/7-(log(10)\*sin90).

Example 1:4+5\*8^2/3+(log(10))

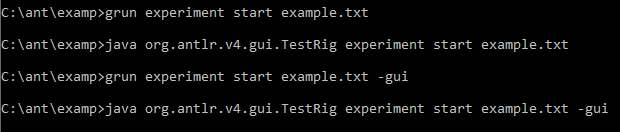
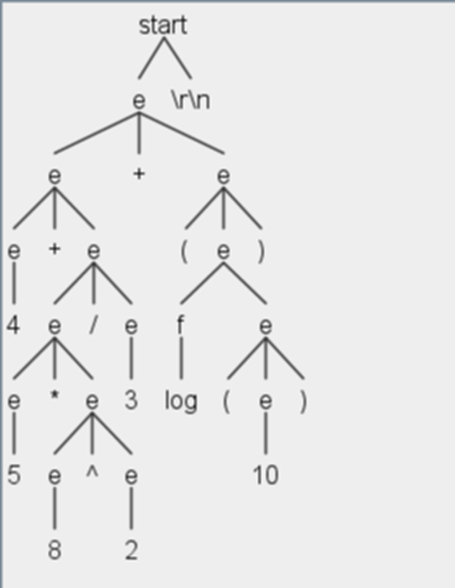


Figure1: User command for output

Figure2: parse tree for example 1.

Example2: 19+42\*78^5/7-(log(10)\*sin90)

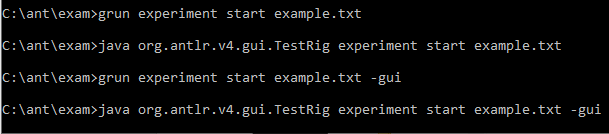


Figure3: User command for output

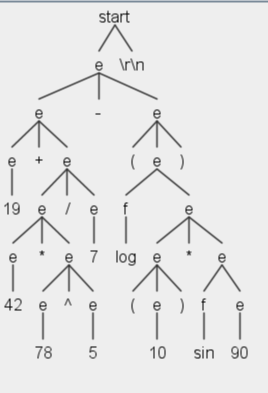


Figure4: Parse tree for example 2.

# **Conclusion**

Our developed a grammar that can calculate all the mathematical operation for any inputs. We can add more mathematical function into this grammar so we can find out more calculation from the given inputs. After all it is an interesting and learnable project for us