

CA4003 Compiler Construction ASSIGNMENT1

|  |  |  |
| --- | --- | --- |
| **Student Name** | **Student ID** | **Course Code** |
| Nathan Ndombasi | 13517227 | CASE4 |

Declaration:

*I declare that this material, which I now submit for assessment, is entirely my (our) own work and has not been taken from the work of others, save and to the extent that such work has been cited and acknowledged within the text of my (our) work. I (we) understand that plagiarism, collusion, and copying is a grave and serious offence in the university and accept the penalties that would be imposed should I (we) engage in plagiarism, collusion, or copying. I (we) have read and understood the Assignment Regulations set out in the module documentation. I (we) have identified and included the source of all facts, ideas, opinions and viewpoints of others in the assignment references. Direct quotations from books, journal articles, internet sources, module text, or any other source whatsoever are acknowledged and the source cited are identified in the assignment references. This assignment, or any part of it, has not been previously submitted by me (us) or any other person for assessment on this or any other course of study*

*I (we) have read and understood the referencing guidelines found at http://www.library.dcu.ie/citing&refguide08.pdf and/or recommended in the assignment guidelines. “*

## Aim

The aim of this assignment was to implement a lexical and syntax analyser using Antlr4 for a simple language called CAL.

The syntax and semantics were already given and the test runs were used to make sure the CAL language ran accordingly.

### Test0.cal

Test0 was used to demonstrate how the language was case insensitive. Test0 was also used to demonstrate how it handled comments.

This parsed successfully.

***Test0.cal***

Text

Description automatically generated

### Test1.cal

Test1 was used to demonstrate how the language recognises and uses functions.

This parsed successfully.

***Test1.cal***

A picture containing text

Description automatically generated

### Test2.cal

Test2 was used to demonstrate the different scopes of the language.

This parsed successfully.

***Test2.cal***

Text

Description automatically generated

### Test3.cal

Test3 was the final test and it was a file demonstrating the use of functions.

This test run tested every aspect of the CAL language and at the end, it parsed successfully.

***Test3.cal***

Text

Description automatically generatedText

Description automatically generated

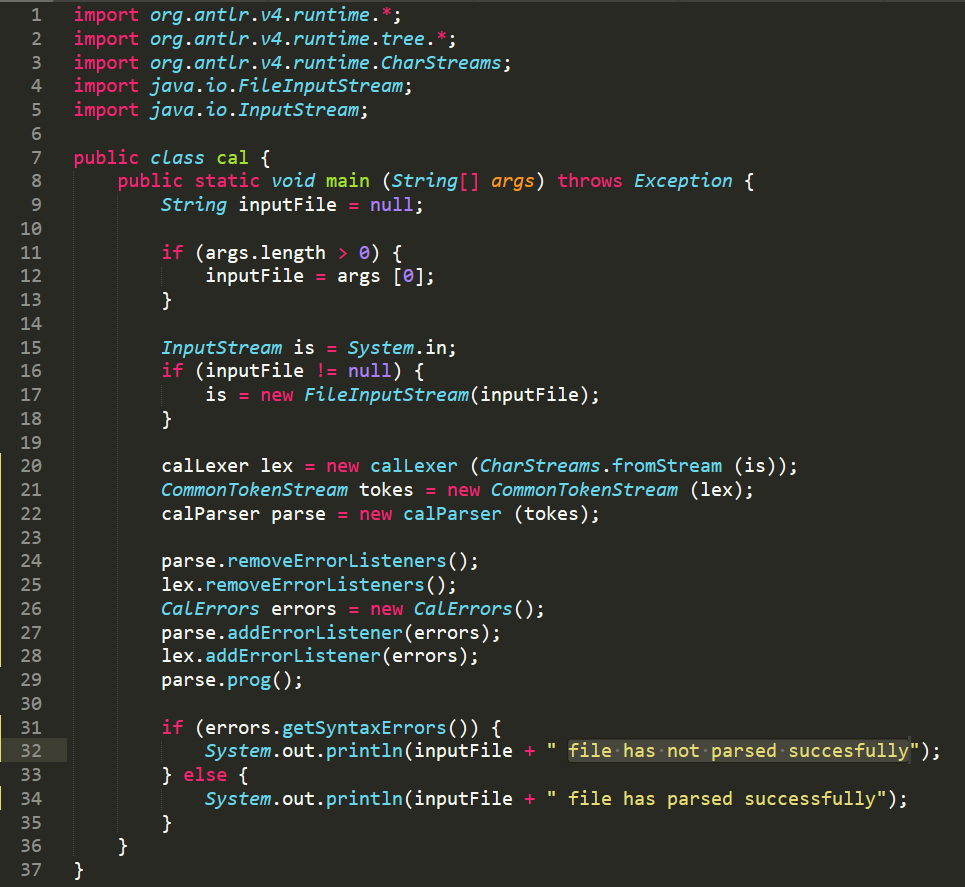
## Cal.java

Firstly I imported the necessary files to run the cal.java. Then I made a String of arrays called args. If the args.length was greater than 0 then we added it into our args list.

Then I made a lexer (lex) that was built from the input stream. I declared a stream tokens (tokes) to be taken from the lexer. In the next line I made a parser to be built from the token stream.

. removeErrorListeners allowed me to remove the chosen listener and .addErrorListener allowed me to add the chosen listener to be notified of errors. If I received any errors, then the program will notify me (“file has not parsed succesfully”), otherwise it will print out “file has been successfully parsed”.

***Cal.java***



## CalErrors.java

My CalErrors file is a class that determines how to continue after an error is found.

When there is a syntax error, the void syntaxError() notifies me which parser got the error with *Recognizer and* the *offendingSymbol* tells me which token caused the error.

getSyntaxError() simply returns if it has syntax errors.

***CalErrors.java***

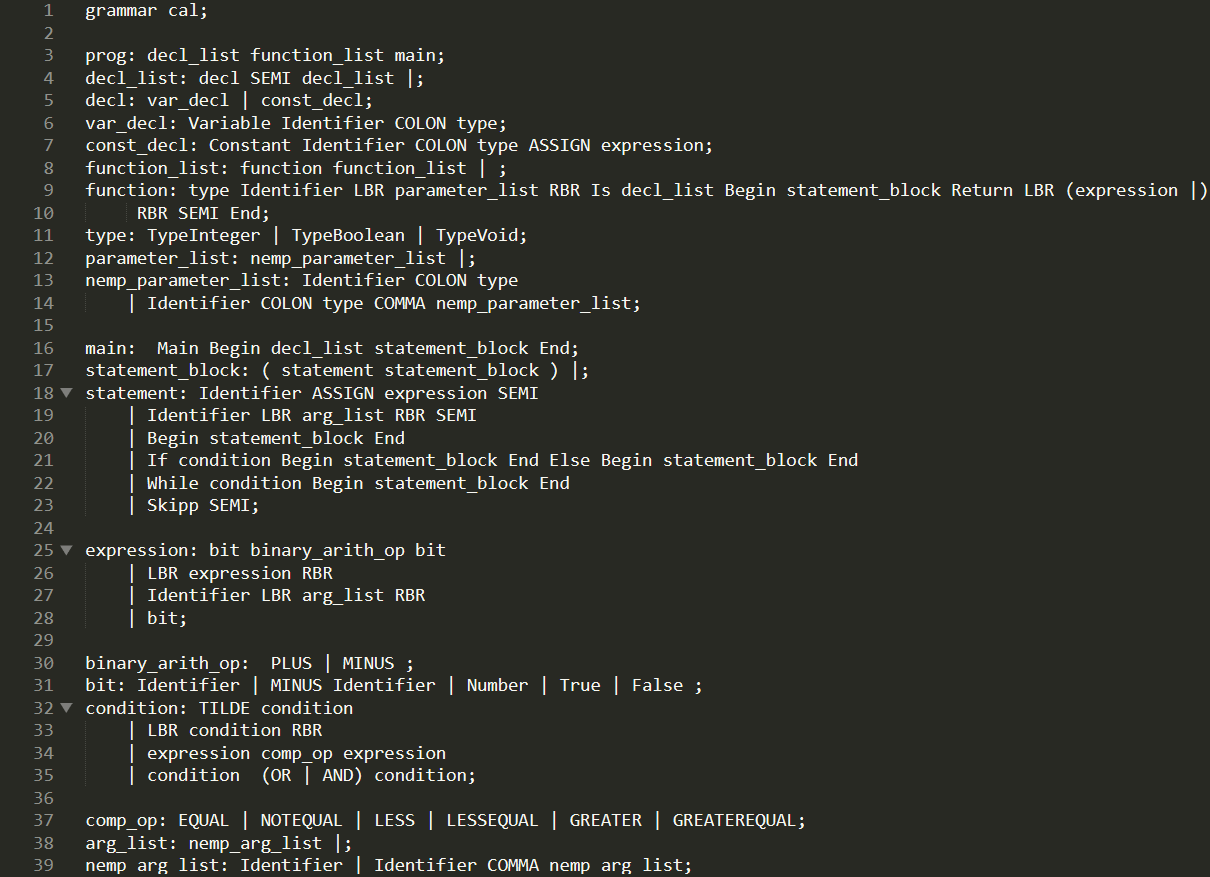
A picture containing sitting

Description automatically generated

## Cal.g4 (Grammar File)

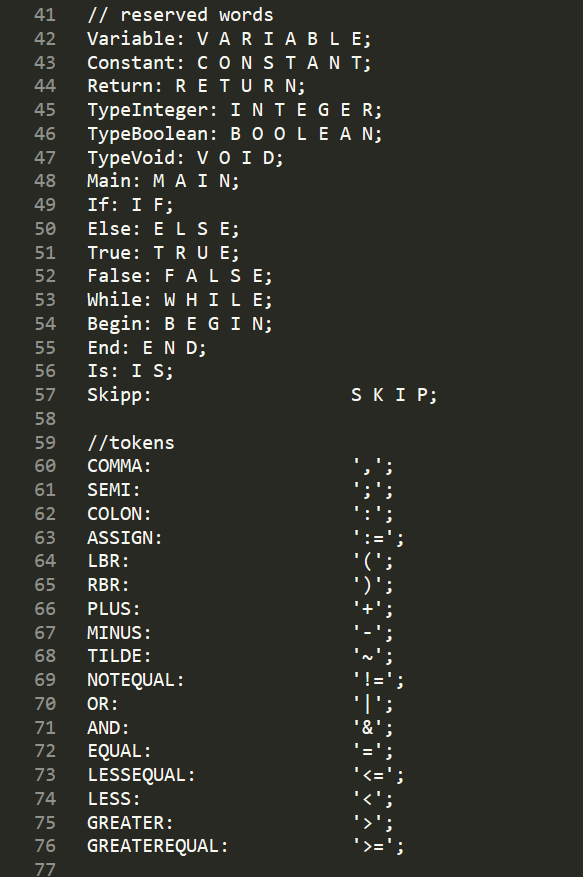
This is the syntax implemented as shown below.

***Syntax of CAL***



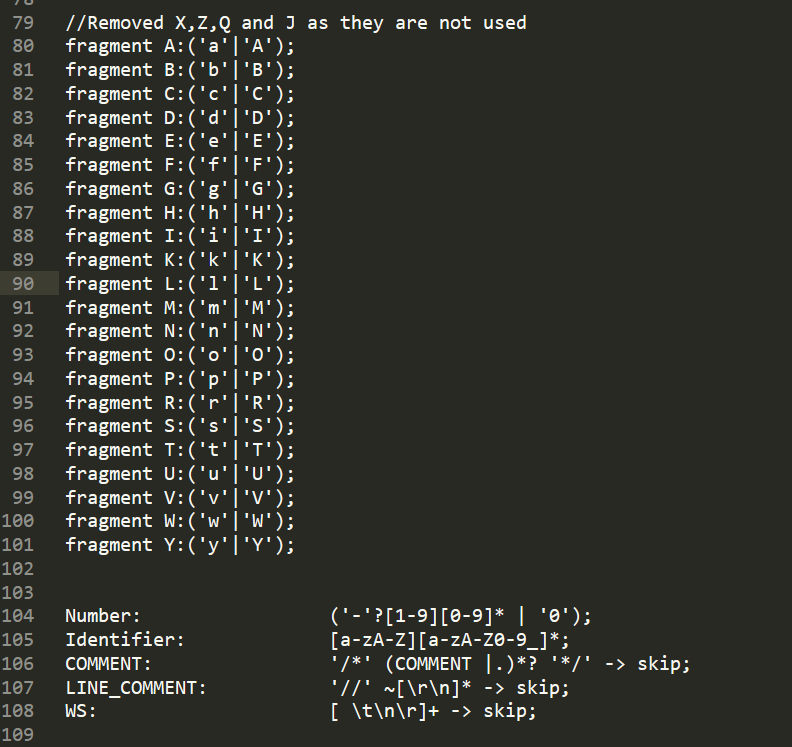
The following image shows my tokens that I assigned and the reserved words for the CAL language

***Reserved words & Tokens of CAL***



Lastly, we have our fragments, numbers, identifiers, comments and whitespace defined.

***Fragments, Numbers, Identifiers, Comments and Whitespace of CAL***



***fin.***