How to use the compiler:

Once the program has launched, the user will be prompted to input a number to perform a specific operation. Upon selection, the user must select a sample program which will then be executed for the following operations:

|  |  |
| --- | --- |
| 1. – Display Parse Tree/JSON | Displays the program’s parse tree structure in Java Swing and outputs the tree in JSON in the console window. |
| 2. – Search Parse Tree | User has to input a valid lexer rule to find the number of occurrences. Invalid lexer rules will be outputted as well. User input can be chained so that multiple rules can be searched. |
| 3. – Display Subtrees | User has to input a valid lexer rule to find all subtrees of a rule. They will then be displayed in Java Swing. |
| 4. – Execute Program in JavaScript | Displays the program in JavaScript in the console. Furthermore it will deploy that code in JavaScript and run in it in a web browser. |
| Any other number | Terminate the program. |

How to use LinguaNova

|  |  |
| --- | --- |
| Variable Initialisation | SCOPE($ | % | #) + DATATYPE(String | Boolean | Decimal) + VARNAME  Example: #String hello |
| Global Variables | Can only exist outside a block. There are two types of global scopes:  Generic Globals($) and Constant Globals(%)  Global variables must terminate with a semicolon (;) |
| Local Variable | Local variables only exist inside a block. They are denoted with the Local Scope(#) |
| Variable Assignment | (SCOPE + DATATYPE +) VARNAME + “<-“ + VALUE  Example: #String hello <- “Hello World” |
| Array | (SCOPE + DATATYPE +) VARNAME<SIZE> + “<-“ + “<”VALUES” “,”>” |
| Main | It is compulsory that every program must have one main body.  It uses the following signature:  $Main init(): |
| Block Indentation | Use two whitespaces to indent the code.  Example:  $Main init():  output(“Hello World”) |
| Block Terminator | Indent [end] after a new line (does not apply to main) |
| Input | Precondition: A string variable must be initialised.  To perform an input, use the following method:  input(VARNAME) |
| Output | To perform an output, use the following method:  output(STRING or VARNAME) *(separate with “,” to concatenate)* |
| Negation | Use negate() with equals(), Booleans or Boolean operators |
| Comparisons | Compare two decimals with the following methods:  less(), less\_equal(), greater(), greater\_equal(), equals() |
| Boolean operators | Use and() or or() |
| If else | verify(CONDITION):  statement  [end]  divert:  statement  [end] |
| Switch Case | trigger(VARNAME):  choice VALUE:  statement  [end]  standard:  statement  [end]  [end] |

|  |  |
| --- | --- |
| For loop | step(#Decimal i <- VALUE / COMPARATOR / INCDEC):  statement  [end]  Example:  step(#Decimal i <- 0.00 / less(i, array.length) / inc(i)):  output(array<i>)  [end] |
| Foreach loop | stepthru(DATATYPE VARNAME in ARRAY):  statement *(should contain ARRAY<VARNAME>)*  [end] |
| While loop | interval(CONDITION):  statement  [end] |
| Do While loop | initiate:  statement  interval(CONDITION) [end] |
| Function call | “call” + FUNCNAME(ARG) *(separate with “,” to add more arguments)* |
| Function body | SCOPE + “Func” + FUNCNAME(DATATYPE ARG):  statement  [end]  Example:  $Func helloworld(String hiworld):  output(hiworld)  [end]  Note: Functions can also have a local scope when it should exist inside a function. |
| Return | return a value using “return” before terminating a function |

Sample Programs

|  |  |
| --- | --- |
| assessment1.txt | Open “andyle-program-explanation.txt” inside the Assessment 1 folder for further explanations. |
| baphomet.txt | Uses a do while loop to double a number until it reaches 666.00. |
| helloworld.txt | Outputs “helloworld” |
| sciencelegends.txt | Iterates through an array to sequentially output strings. |
| stackoverflow.txt | Causes an infinite loop. |
| userinput.txt | Allows the user to alter this program in eclipse as a means of executing in JavaScript |