## Syntax Rules:

1. Every Variable must be defined and initialized to be used else the program will generate an error, also must follow the declaration rules EX: INT x = 10, anything else will be flagged as an error and you will be given the line number of the error in the console
2. The first line must start with the keyword program followed by the program name, the program name must be one string i.e. doesn’t contain any spaces
3. To begin using the variables you declared previously you must write the keyword BEGIN, any variables you utilize after begin must be declared and initialized
4. In any arithmetic operation line, the program follows the rule that the operators count must be less than the variables count by one any other case it will be flagged as an error EX: X = X – Y \* m

One of the steps that the program does after the program checks the arithmetic operations line and its all good and clear of errors it puts the line in a python dictionary and the key of that line in the dictionary is the current line number

*Special cases*:

* If you wrote that line in an if or a for statement the key of the arithmetic operations line will be the line number where the If/for statement is written, utilizing that method the program will be able to deal with if/for lines and recognize if the arithmetic operations line belongs to an if/for statement or not and generate the correct code for each case

The program when trying to solve a line it will order the operation according to their perspective logical order of operations and will actively seek the operation with the highest order

Example of declaration of an arithmetic op line EX: X = X – Y \* m

1. The if statement must follow the syntax rules and all variables written in the statement must be declared like stated previously EX: IF X < Y DO

Following the if statement you can write only one arithmetic operation line in that if statement otherwise it will be flagged as an error, after you write the athematic operations line you must end the if statement with the keyword ENDIF otherwise ERROR!!!

1. In case of for statements it has the same rules of the for statement except its syntax is

EX: FOR I = X TO Y DO , the I variable cannot be changed and it must be written, and you should end the for statement with the keyword ENDFOR , and you can only write one arithmetic operations line in a for statement.

## Language Grammar:

<int> ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |8 | 9

<identifier> ::= <letter > {<letter or digit>}

<letter or digit> ::= <letter> | <digit>

<Keyword> ::= INT | PROGAM | FOR | ENDFOR | DO | IF |ENDIF | BEGIN | END

<vairable declaration> ::= INT <assign statement>

<assign statement> ::= <identifier> = <int> | <identifier> = <identifier>

<expression> ::= <simple expression> | <simple expression> <relational operator> <simple expression> | <if statement> | <for statement>

<simple expression> ::= <term> | <simple expression> <adding operator> <term>

<adding operator> ::= + | -

<term> ::= <factor> | <term> <multiplying operator> <factor>

<factor> ::= <identifier>

<multiplying operator> ::= \* | /

<relational operator> := < | > | ==

<for statement> ::= FOR I = <initial value> TO <final value> DO <statement> ENDFOR

<initial value> ::= <expression>

<final value> ::= <expression>

<if statement> ::= IF <expression> DO <expression> ENDIF

<compound statement> ::= ***begin*** <statement> { <statement> } ***end***;

<program> ::= ***program*** <identifier> {<block>}

<block> ::= <declaration part> <compound statement part>