* Shell.py: It will read the raw input from the terminal and display it.
* Libra.py: Holds all the code of language.
  + Error class: It handles all the error and display it to the user.
    - It returns a string/error which tells path of file in which error occurred, type of error and from which it occurs and line no. in which an error occurred.
  + Position class: It tells index, column, line etc
    - Advance(): It moves the execution/Cursor to next character.
  + Lexer class: It will go through input character by character and break them into list of tokens.
    - Make\_tokens():It checks that the character is of which token type(=,-,\*,/) and append it in tokens list(use else-if ladders).
    - Make\_number(): It takes raw input and specify int or float and make it in integer format.
    - Ex:- Libra > 2 + 2

[INT:2, PLUS, INT:2]

Libra > 2.5\*2.5

[FLOAT:2.5, MUL, FLOAT:2.5]

Libra > 77\* d

Illegal Character: 'd'

* Nodes: It tells us about factor, binary ops and unary ops
  + - NumberNode: It works on factor.
    - BinOpNode: It works on binary operators
    - UnaryOpNode: It woks on Unary operators.
* ParseResult class: It returns the AST(Abstract Syntax Tree) which is created by parse the expression.
* Parser class: It recursively checks for expression, term and factor and parse it.
* Interpreter class: It runs and helps to get the desired result.
* grammar.txt: It specifies grammar which we used to parse the expressions or to write code.
* Strings\_with\_arrows.py: It helps to obtain where the error is came from.
* Power operator-> ‘^’
* [new]Modulo operator-> ‘%’
* Variable syntax: KEYWORD IDENTIFIER = <expr>

Var a = 5

* Conditional Statements : if,elsif,else
* From loop(for): KEYWORD:from IDENTIFIER EQ expr KEYWORD:to expr

(KEYWORD:step expr)? KEYWORD:then expr

* Until loop(while) : KEYWORD:until expr KEYWORD:then expr
* Continue: cont
* Break: brk
* Comments : !! !!
* len(): To get the length
* exec(): to run the program file