\*\*Note- if the code does not run, please change path of javaTest.txt in Julia.java.

Here is the break down of all the classes in our code

1.Parser-

•Parser class is responsible for parsing the source code and constructing an abstract syntax tree (AST).

•It contains methods for parsing different types of statements and expressions.

•Methods like varDeclaration, expressionStatement, ifStatement, or, and, whileStatement, etc., handle specific grammar rules.

•The parse method returns a list of statements after parsing the entire source code.

2.Interpereter-

•Interpreter class is responsible for interpreting the AST generated by the Parser.

•It implements the Expr.Visitor<Object> and Stmt.Visitor<Void> interfaces, providing methods to visit different types of expressions and statements.

•Methods like visitBlockStmt, visitExpressionStmt, visitPrintStmt, visitVarStmt, etc., handle the interpretation of specific AST nodes.

•The interpret method takes a list of statements and executes them

3.Generate AST-

•GenerateAst class is responsible for generating AST classes for expressions and statements.

•It includes a main method that takes an output directory as a command-line argument and generates Java code for AST classes

4.Environment-

•Environment class manages variable scopes and their values.

•It has an optional enclosing environment to support nested scopes.

•get method retrieves the value of a variable, define method adds or updates a variable, and assign method updates the value of an existing variable.

5.Scanner-

•The Scanner class takes the source code as input and converts it into a list of tokens (List<Token>).

•The scanTokens method iterates through the source code and calls scanToken to process each character.

•The switch statement inside scanToken handles different types of characters, including operators, punctuation, and literals.

Helper methods like match, peek, advance, addToken, string, number, etc., assist in the scanning process.

•The keywords map is used to recognize keywords and map them to their corresponding TokenType.

•The TokenType enum likely contains constants for different types of tokens (e.g., NUMBER, STRING, IDENTIFIER, etc.)

6.Julia-

•Julia class contains the main method for running the Julia interpreter.

•It uses the Scanner class to tokenize the source code and then passes the tokens to the Parser and Interpreter for parsing and interpretation