**PRINCIPLES OF PROGRAMMING LANGUAGES**

**PROJECT II**

**25.05.2023**

**Muhammed Enes Gündüz - 150120038**

**Mustafa Tolga Akbaba - 15012001**

**Efe Özgen - 150121077**

**Programming Language Used: Java**

**Purpose**

This project wants us to write a java or C code to develop a new programming language called PPLL. This code reads every char one by one and analyzes it for each time. If any error occurs, it prompts an error message. These error messages depend on the type of error (syntactically or lexically). In case everything works well, it prints the output to the console. Output consists tags, i.e. <program>, <statements>, and the characters that the reader reads, i.e. (n), (fibonacci) and tabs. A tag can call an empty string, keywords(identifier, number), and tag(s). So, it basically reads characters, analyzes it and prints them.

**How to Run**

Your directory should be:

-Main.java, Parser.java and input text file should be under src folder

1-Run Main.java

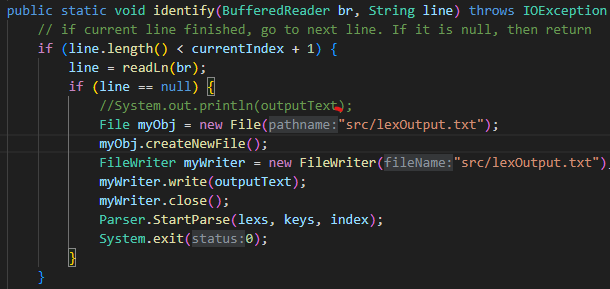
2-Program will ask you to write the name of input file



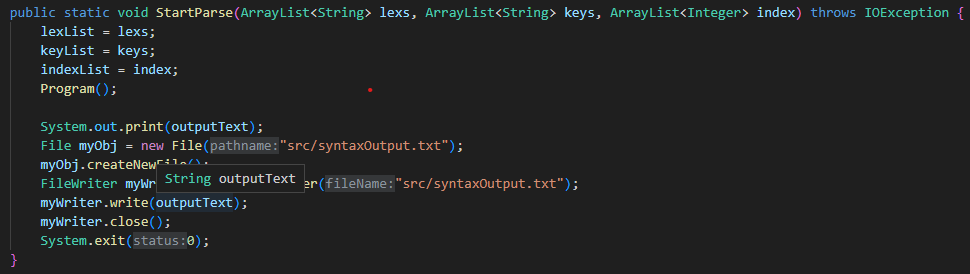
write only the name of text file, without .txt

3-It will read and generate 2 output text files, lexOutput.txt and syntaxOutput.txt. lexOutput will include lexical output and syntaxOutput will include syntactic output. It also outputs the result in the console.

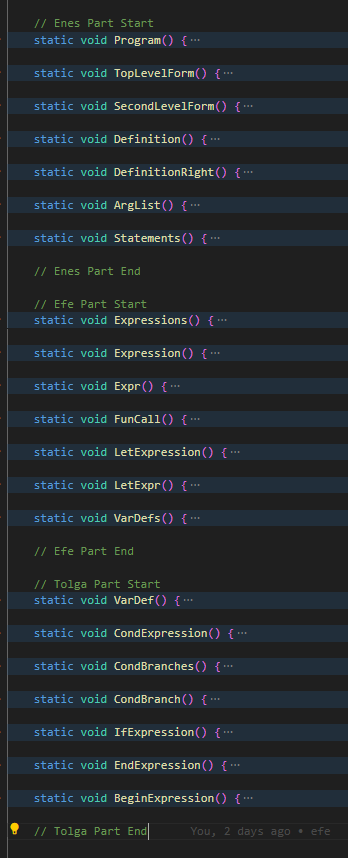
**Code Overview**



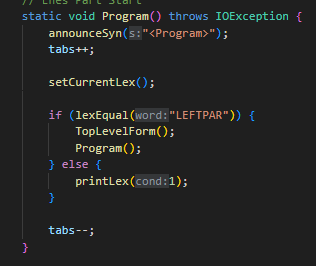
In Main class, under identify function, before the program ends, we call Parser.StartParse function. Parser is the class that parses the output of lexical analyzer. It takes lexs, keys and indexes. lexs includes NUMBER, LEFTPAR, IDENTIFIER exc., keys includes (, fibonacci, n exc., index includes the index of the lexs’.



StartParse function calls the very first function Program. After calling Program, all functions below calls others both recursively and directly.



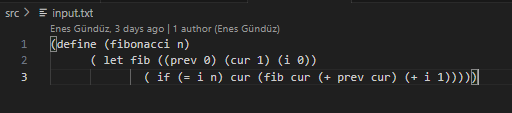
Program function is given below as an example of 21 different function. Because they are mostly similar, we thought that showing one would be enough.

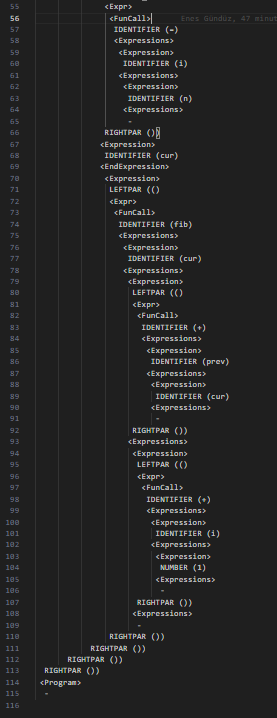
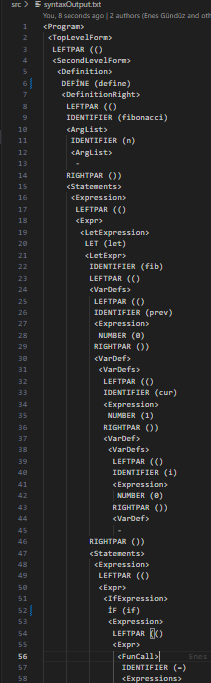


The basic functions used in Program and other functions are explained below.

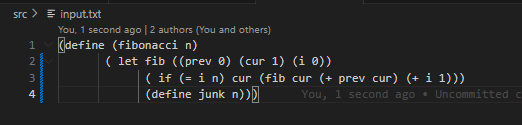
****

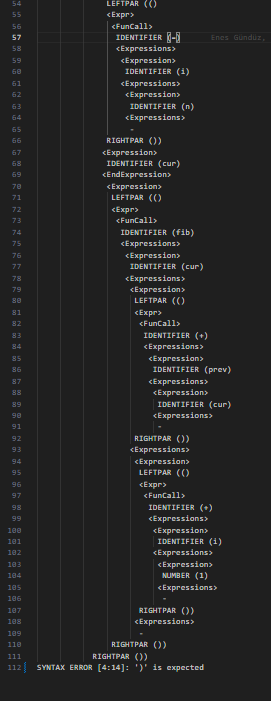
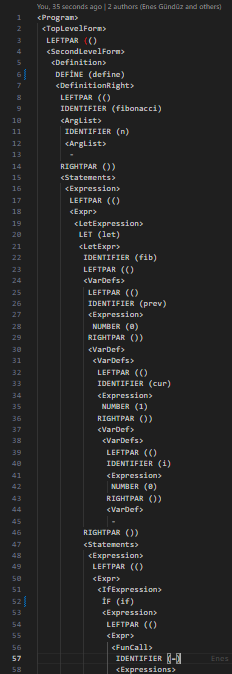
These functions are generally taking care of tokens. lexEqual compares two tokens if they are equal or not. setCurrentLex function updates the current token with incrementing the index. announceSync function updates the output according to the tab spaces. printLex updates the output according to the current token.

**Correct Input & Output**

****

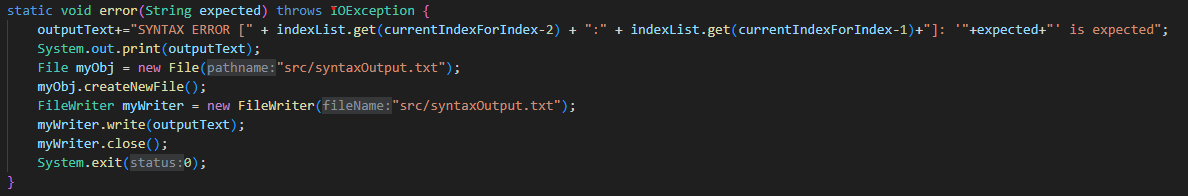
**Incorrect Input & Output**

****

****

**Error Handling**

In lexical errors, it prompts lexical error and invalid token and prints which line and column the error is in. For example, if an invalid character is detected, the program will output "invalid token" with the line and column number of the invalid character. Also it prints the output until the first error occurs.

In syntactic errors, it prompts syntactic error and invalid token and prints which line and columns the error is in. This works the same as lexical error prompt.

**Conclusion**

We managed to do every task that is desired. Every part is working correctly. It reads input, analyzes it, prints it and creates an output file and writes over that file.