AOA

Analysis of Algorithms

Calvin, Mileno, Rowin, Michael

Resource

Computer Language

E R I C

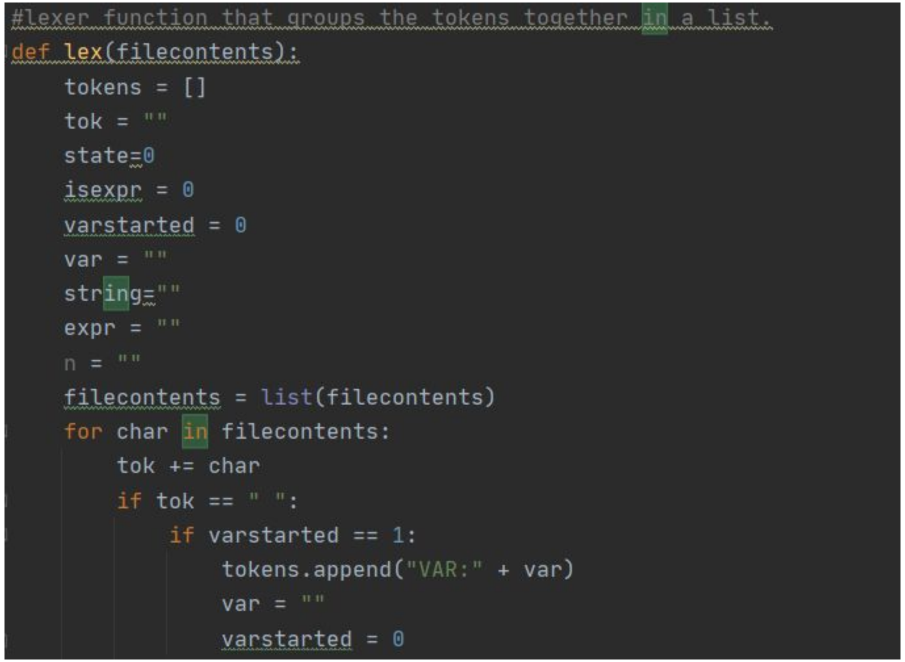
Excessively

Intensive

E.R.I.C Functions

- Print
- Input
- Declaring Variables
- Arithmetic Operations
- For loops
- If statements

Lexer Function

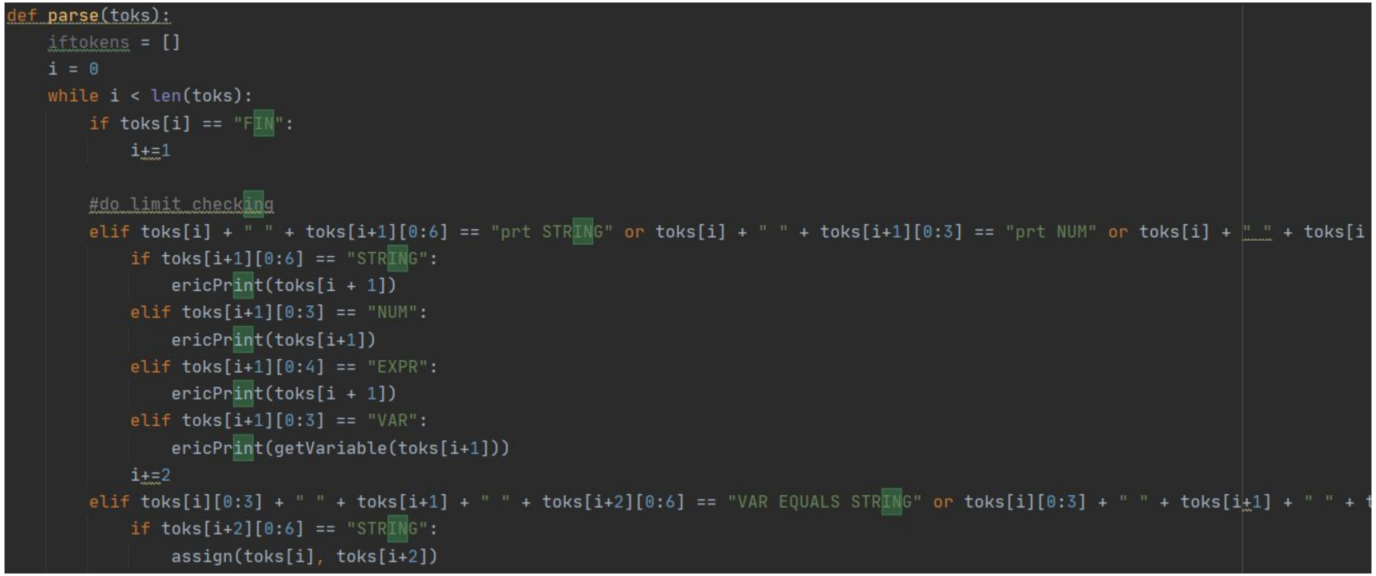


```
#lexer function that groups the tokens together in a list.
def lex(filecontents):
    tokens = []
    tok = ""
    state=0
    isexpr = 0
    varstarted = 0
    var = ""
    string=""
    expr = ""
    n = ""
    filecontents = list(filecontents)
    for char in filecontents:
        tok += char
        if tok == " ":
            if varstarted == 1:
                tokens.append("VAR:" + var)
                var = ""
                varstarted = 0
```

The lexer function groups all the tokens from a certain file and appends them into a list of tokens.

The resulting list will be returned by the function.

Parser Function



```
def parse(toks):
    iftokens = []
    i = 0
    while i < len(toks):
        if toks[i] == "FIN":
            i+=1

        #do limit checking
        elif toks[i] + " " + toks[i+1][0:6] == "prt STRING" or toks[i] + " " + toks[i+1][0:3] == "prt NUM" or toks[i] + " " + toks[i
            if toks[i+1][0:6] == "STRING":
                ericPrint(toks[i + 1])
            elif toks[i+1][0:3] == "NUM":
                ericPrint(toks[i+1])
            elif toks[i+1][0:4] == "EXPR":
                ericPrint(toks[i + 1])
            elif toks[i+1][0:3] == "VAR":
                ericPrint(getVariable(toks[i+1]))
            i+=2
        elif toks[i][0:3] + " " + toks[i+1] + " " + toks[i+2][0:6] == "VAR EQUALS STRING" or toks[i][0:3] + " " + toks[i+1] + " " + t
            if toks[i+2][0:6] == "STRING":
                assign(toks[i], toks[i+2])
```

The parser function takes the resulting list of the lexer function as an argument and iterates through it. It decides what the resulting order of the token list will output.

Difficulties

Obstacles that occur in the making of our project

Limited Knowledge

We are studying how to make our own language in the process of making this project

Online Meetings

Online meetings are harder to follow. Especially when it comes to working on projects

Time

Hard to keep up with other projects and assignments, as well as keeping up with each member's schedule

Time Table

Start of Project

The start of our project, studying ways to implement our own programming language

16 Nov

Coding

Implementing our codes and adding new features to our programming language

20 Nov

Project Milestone

Project Milestone Presentation

30 Nov

Editing

Modifying our codes based on our project milestone feedback

2 Dec

Testing

User testing for bugs and errors

20 Dec

Final Presentation

Project Final Presentation

11 Jan

Project Demo

AOA

Thank you!