**Kennesaw State University**

**College of Computing and Software Engineering**

**DEPARTMENT OF COMPUTER SCIENCE**

**CS 4308 Concepts of Programing Languages Section W01**

Second Deliverable: Parser/Scanner

Stanley Gilstrap

[sgilstr3@students.kennesaw.edu](mailto:sgilstr3@students.kennesaw.edu)

3/20/17

Initial problem:

Develop a complete scanner and parser using a relevant source line as input. The program must show a list of tokens in that line. The input will be a subset of Lua.

Solution:

Using python as a programming language, I developed a scanner that scans the incoming file line by line, then looks at characters on each of those lines and creates lexemes out of the letters, numbers and symbols. Then looking at each lexeme we use the grammar given to assign each lexeme a token type. I used Pycharm as an IDE.

We then pass each lexeme and its token type to the parser class. The parser looks at each lexeme and its token type and parses them into separate statements.

Input:

function  
a = 5  
b = 33  
c = + a b  
d = \* a c  
e = / 9 3  
if >= d 3 then print ( + 3 5 ) end  
if <= e 5 then f = 3 else f = 2 end  
while < f 10 do + f 1 end  
end

Output:

start function

assignment started

a

=

5

assignment ended

assignment started

b

=

33

assignment ended

assignment started

c

=

arithmetic started

add operation start

+

a

b

arithmetic ended

assignment ended

assignment started

d

=

arithmetic started

mul operation start

\*

a

c

arithmetic ended

assignment ended

assignment started

e

=

arithmetic started

div operation start

/

9

3

div operator ended

assignment ended

start if statement

ge comparator start

>=

d

3

ge comparator ended

then started

print started

arithmetic started

add operation start

+

3

5

add operator ended

print ended

if ended

then ended

start if statement

le comparator start

<=

e

5

le comparator ended

then started

assignment started

f

=

3

assignment ended

else started

assignment started

f

=

2

assignment ended

if ended

then ended

else ended

while started

lt comparator start

<

f

10

lt comparator ended

do started

arithmetic started

add operation start

+

f

1

add operator ended

assignment ended

while ended

do ended

function ended