Computer Science Department
San Francisco State University
CSC 413

Assignment 2 - Modify the Lexer

Note that the due date applies last commit timestamp into the main branch of your repository.

Overview

The purpose of this assignment is to extend the Lexer component of our x language compiler to be able to handle additional tokens, to supplement our understanding of Compilers and Lexical Analysis.

You are provided with the Lexer code, which will be automatically cloned into your GitHub repository when you begin the assignment via the GitHub link posted on ilearn.

Submission

You are required to submit your source code via the GitHub assignment repository cloned for you.

Requirements

You will be extending the Lexer to be able to process four new tokens, as well as to improve the output of the Lexer.

1. The current implementation of Lexer reads a hardcoded file. Lexer must be updated to allow input via a filename provided as a command line argument:
   java lexer/Lexer simple.x (Please note that you do not need to run the project via command line, instead run it through NetBeans, adding the commands line arguments to the project as explained in class).
2. Our compiler must be updated to accommodate 8 additional tokens: >, >=, ||, &&, void, float, double, and **. The tokens file must be updated, and TokenSetup must be ran to re-generate the Tokens and TokenTypes classes.
3. The Token class must be updated to include the line number that a token was found (for subsequent error reporting, etc.).
4. Lexer output must be updated for readability, and to include the line number from the Token (Note that the initial debug text that shows the file information has been removed!) (Note that the initial debug text that shows the file information has been removed!).

```
READLINE:   program {boolean j int i
program   left: 0       right: 6       line: 1
{         left: 8       right: 8       line: 1
boolean   left: 9       right: 15      line: 1
j         left: 17      right: 17      line: 1
int       left: 19      right: 21      line: 1
i         left: 23      right: 23      line: 1
READLINE:     int factorial(int n) {
```

5. Lexer output must be updated to include a printout, with line number, of each of the lines read in from the source file. Note that when an error is encountered, the error should be reported as usual, and the lines of the source file should be printed as well, with line numbers, up to and including the error line.

Normal Execution

```
1. program { int i int j
2.   i = 2
3.   j = 3
4.   i = write( j + 4 )
5. }
```

Error Execution:

```
1. program { int i int j
2.   i = 2;
```

Note the above outputs are missing the output from the Lexer which prints the information for each token. These are just the samples for outputting the source code.

Appendix

The unmodified Lexer output appears below, for comparison:

```
Source file: simple.x
user.dir: F:\CSC413\csc413-p2
READLINE:   program { int i int j
L: 0 R: 6 program : 1
L: 8 R: 8 { : 1
L: 10 R: 12 int : 1
L: 14 R: 14 i : 1
L: 16 R: 18 int : 1
L: 20 R: 20 j : 1
READLINE:      i = i + j + 7
L: 3 R: 3 i : 2
L: 5 R: 5 = : 2
L: 7 R: 7 i : 2
L: 9 R: 9 + : 2
L: 11 R: 11 j : 2
L: 13 R: 13 + : 2
L: 15 R: 15 7 : 2
READLINE:      j = write(i)
L: 3 R: 3 j : 3
L: 5 R: 5 = : 3
L: 7 R: 11 write : 3
L: 12 R: 12 ( : 3
L: 13 R: 13 i : 3
L: 14 R: 14 ) : 3
READLINE:   }
L: 0 R: 0 } : 4
```