Concepts of Programming Languages, CSCI 305, Fall 2014
MiniGo14 Lexical Analyzer in Scheme, Due Friday, Oct. 31 by midnight

Convert your table-driven lexical analyzer for MiniGo14 into Scheme. For maximum credit, use Scheme in a functional, rather than an imperative, style.

The interface to your program must:

- Describe the purpose of the lexical analyzer and how to use it.
- Allow the user to repeatedly enter a path to a file containing a MiniGo14 program. (If the lexical analyzer is in the middle of scanning a program, warn the
- When a legal path is given, allow the user to repeatedly request a token, until the MiniGo14 program has been scanned or the user decides to quit scanning.
- Display the token identified and the lexeme associated with the token.
- Display helpful errors when the lexical analyzer finds an error. Display what characters caused the error.
- Allow the user to exit the program.

You cannot use the Unix tool lex for this assignment.

To submit your program, zip the directory containing the .mz files and a human-readable copy of your scanning table. As before, you do not need to turn in a paper copy of your nfa or dfa.

- For maximum credit: Your program should be well commented, which includes descriptive item names and a block of code before every function definition describing the inputs, the outputs and what the function does.
- Your program should be well designed
- Your program should be in a functional style rather than an imperative style. That is, it should avoid the *set!* built in procedures or its derivatives (the built in procedure *let* is ok), avoid begin procedures, except with I/O, and use recursion rather than iteration
- Whenever your program uses sequencing (two or more statements to be executed in sequence) your program should use the *begin* procedure
- Your program should not include any extra, unused code

Grading:

	Points	
Describe the purpose of the lexical analyzer and how to use it.	5	
Allow the user to repeatedly enter a path to a file containing a MiniGo14 program. (If the lexical analyzer is in the middle of scanning a program, warn the user that the scanner is not done.)	5	
When a legal path is given, allow the user to repeatedly request a token, until the MiniGo14 program has been scanned or the user decides to quit scanning, and displays the token identified and the lexeme associated with the token.	50	
Display helpful errors when the lexical analyzer finds an error. Display what characters caused the error.	10	
Allow the user to exit the program.	5	
Program is well commented, which includes descriptive variable, class and method names	5	
Program is well designed	5	
Program does not include any extra, unused code.	5	
Program is written in a functional, rather than imperative, style	10	

You cannot use the Unix tool lex for this assignment.