COVER PAGE

CS323 Programming Assignments

Fill out all entries 1 - 6. If not, there will be deductions!

Peer Review (Check one)

1. Names [1. Lambert Liu], (ThumbUP [✓] or ThumbDown [ ])

[2. Kun Fang], (ThumbUP [✓] or ThumbDown [ ])

[**if 3**.], (ThumbUP [ ] or ThumbDown [ ])

2. Assignment Number [1]

3. Turn-In Dates: **Final Iteration with Documentation** [09-28-2018]

4. Executable FileName [run.exe] EC. lexical_analyzier.py
(**A file that can be executed without compilation by the instructor**)

5. LabRoom [CS 101]
(Execute your program in a lab in the CS building before submission)

6. Operating System/Language [Windows/C++] EC. python

To be filled out by the Instructor:

GRADE:

COMMENTS:

N:

Curr. State	Inputs							
	L	D	S	O	!	$	.	sp
1	2	4	10	8	6	1	1	1
2	2	2	3	3	3	2	3	3
3	1	1	1	1	1	1	1	1
4	5	4	5	5	5	5	12	5
5	1	1	1	1	1	1	1	1
6	6	6	6	6	7	6	6	6
7	1	1	1	1	1	1	1	1
8	9	9	9	8	9	9	9	9
9	1	1	1	1	1	1	1	1
10	11	11	10	11	11	11	11	11
11	1	1	1	1	1	1	1	1
12	13	12	13	13	13	13	13	13
13	1	1	1	1	1	1	1	1

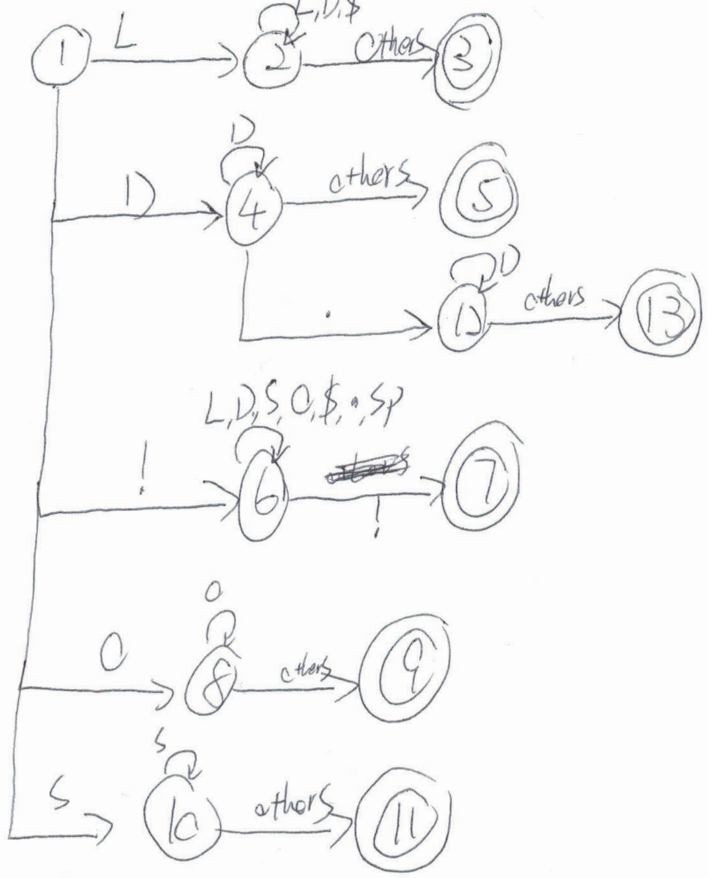
1. Starting State
2. In Identifier
3. End Identifier
4. In Number
5. End Number
6. In !! Comment
7. End !! Comment
8. In Operator
9. End Operator
10. In Separator
11. End Separator
12. In Real
13. End Real

L:	Letter
D:	Digit
S:	Separator
O:	Operator
sp:	space

$F = \{3, 5, 7, 9, 11, 13\}$

R^* — kleene Closure ~~L(L|D)*~~

$l \cdot (l|D|\$)^*$



q1 = 1
Q = {1,2,3,4,5,6,7,8,9,10,11,12,13}
Σ = {A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,P,Q,R,S,T,U,V,W,X,Y,Z,a,b,c,d,e,f,g,h,i,j,k,l,m,n,o,p,q,r,s,t,u,v,w,x,y,z,0,1,2,3,4,5,6,7,8,9,*,+,-,=,/,>,<,%,',(,),{,},[,],,,.,:,;,! }

Problem Statement

The assignment is to build a lexical analyzer, which can read a .txt file and detect each character in the file. Output a file that contains all keywords, reals, numbers, separators, operators, and identifiers.

A simple test case

(Partial) Source code: while (fahr < upper) a = 23.00 whileend

```
Output:   token                        lexeme
          Keyword                      while
          Separator                    (
          Identifier                   fahr
          Operator                         <
          Identifier                   upper
          separator                      )
          Identifier                   a
          Operator                         =
          Real                         23.00
          keyword                      whileend
```

How to use your program

Option1: Windows: Inside the Zip file, there is a file call “run.exe”, run it by double click

Option2: Linux: 1. Open terminal; 2. Change directory to where files located; 3. Type: “g++ -o run lexical_analyzer.cpp”; 4. Type “./run”

Design of your program

Basically, most data structures are in global. We use char array to contain operators, separators, keywords, and buffer for storing identifiers. Using bool to detect each FSM states: endOperator, endSeparator, endNumber, endReal, endIdentifier, etc. Pre-define 3 index for looping using int. Boolean structure for comments detection and a char array(size 2) to store each “!” read from file.

Functions:

bool isKeyword(char buffer[]): To check if buffer contains a keyword

bool isOperator(char ch): To check if current ch(cursor in the reading file) is an operator

bool isSeparator(char ch): To check if current ch is a separator

bool isReal(ch buffer[]): To check if buffer contains float(real number) by comparing first part (before ’.’) is digit and second part (after ‘.’) is digit.

bool isNumber(ch buffer[]): To check if buffer contains a number
void print(): print every end states, and store the output in output file
void lexer(): check comments first, and check each bool function list above, change to each end states, and reset everything
int main(): main function, ask user input filename to analysis, and output filename. Print the table format: (Tokens Lexemes), looping while readfile not reach end of the file, and keep calling lexer() function. Exit when end of read file.

Algorithms (Special functions use):

int isalnum (int c): Checks whether c is either a decimal digit or an uppercase or lowercase letter. Return 0 if not true
int isdigit (int c): Checks whether c is a decimal digit character. Return 0 if not true
int strcmp (const char * str1, const char * str2): Compares the C string str1 to the C string str2. Return 0 if not true
size_t find (string s): Searches the string for the first occurrence of the sequence specified by its argument. Return string::npos if not true.

Limitation

Keywords, operators, separators, should be pre-defined.
'!', ' '(space), '$', '.' are not count as separators.

Shortcomings

Cannot detect compound operators, such as "<=", the lexer() will output it separately but not as one operator.