Compiler Project 1

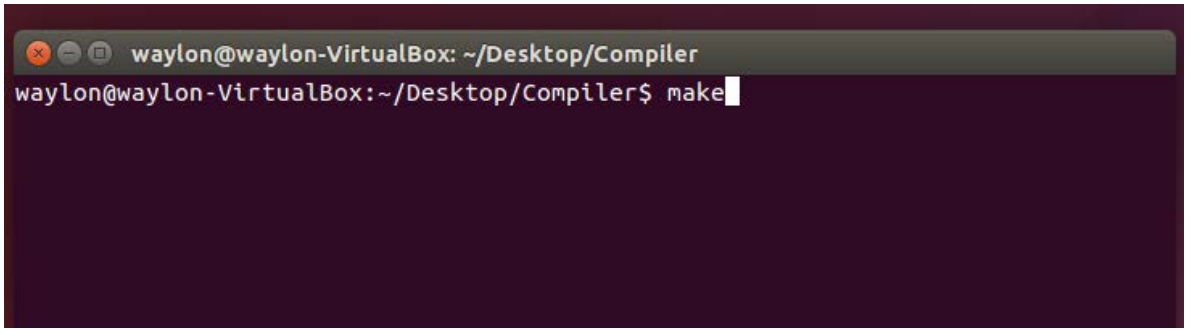
Project: C- compiler lexeme scanner

Date: October 8, 2018

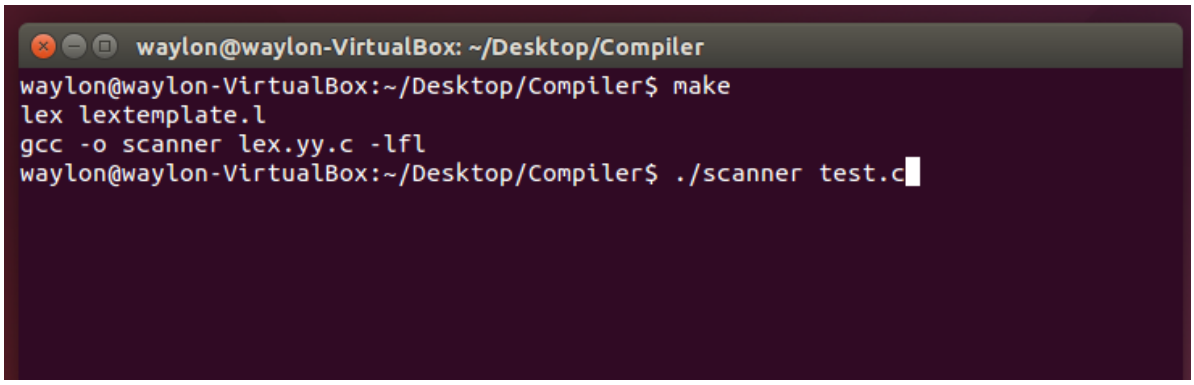
Platform: Ubuntu 14.04

How to run:

Go to the file directory and call the makefile using the command "make" .



Execute scanner and put your c source filename as the second parameter.



Then the scanner will run.

```
waylon@waylon-VirtualBox: ~/Desktop/Compiler

18:

<id:struct>
<id:a>
<delim:;>
19:struct a;

20:

frequencies of identifiers:
main        1
argc        1
char        1
argv        1
a           12
b           4
c           5
d           1
f           1
s           1
printf      1
struct      1
waylon@waylon-VirtualBox:~/Desktop/Compiler$
```

Abilities:

With the help of Lex program, the lexical analyzer (scanner) will break up an input stream into usable elements (tokens). After reading the 1st line of the test file,

```
test.c
1  int main(int argc, char ** argv) {
2      int a, b, c;
3      double d = -0.5654;
4      float f = +6e-6;
5      a = 9;
6      b = 91;
```

the scanner outputs each token name and the lexeme.

```
waylon@waylon-VirtualBox: ~/Desktop/Compiler
<KW:int>
<id:main>
<delim:(>
<KW:int>
<id:argc>
<delim:,>
<id:char>
<"*">
<"*">
<id:argv>
<delim:)>
<delim:{>
1:int main(int argc, char ** argv) {
```

Tokens specification:

. Delimiters:

comma	,
semicolon	;
parentheses	( )
square brackets	[ ]
braces	{ }

. Arithmetic, Relational, and Logical Operators:

addition	+
subtraction	-
multiplication	*
division	/ %
assignment	=
relational	< <= != >= > ==
logical	&& || !

. Keywords:

while do if else true false for int print const read boolean bool void float double string continue break return

. Identifiers

Comments will be ignored by scanner.

```
16:/* this is a

17:c++ style comment */

18:

<id:struct>
<id:a>
```

When the scanner reach the end, it will reveal all the statistic information of the identifiers.

```
20:

frequencies of identifiers:
main          1
argc          1
char          1
argv          1
a             12
b             4
c             5
d             1
f             1
s             1
printf        1
struct        1
waylon@waylon-VirtualBox:~/Desktop/Compiler$
```

When an error occurs, the scanner will print the line number and the unmatched token and exit with code 1.

```
16:/* this is a

17:c++ style comment */

18:

<id:struct>
<id:a>
<delim:{>
19:struct a{

Error at line 20: $
waylon@waylon-VirtualBox:~/Desktop/Compiler$
```

In addition, you can use the #pragma directive to control the actions of scanner without affecting the program as a whole.

A pragma directive starts with "#pragma" followed three options: source, token and statistic. By default, all of the options are on.

```
#pragma source on
#pragma token on
#pragma statistic on

int main(int argc, char ** argv) {
```

(optional, the scanner will run as default)

Source turns source program listing on.

```
#pragma source on
#pragma token off
#pragma statistic off
```

```
waylon@waylon-VirtualBox: ~/Desktop/Compiler

16:    return 0;

17:}

18:

19:

20:/* this is a

21:c++ style comment */

22:

23:struct a{

24:     var;

25:};

26:

waylon@waylon-VirtualBox:~/Desktop/Compiler$
```

Token turns token listing on.

```
#pragma source off
#pragma token on
#pragma statistic off
```

```
waylon@waylon-VirtualBox: ~/Desktop/Compiler
<"+">
<id:a>
<"+">
<id:a>
<"+">
<id:a>
<"+">
<id:a>
<"+">
<id:a>
<delim:)>
<delim:;>
<KW:return>
<integer:0>
<delim:;>
<delim:}>
<id:struct>
<id:a>
<delim:{>
<id:var>
<delim:;>
<delim:}>
<delim:;>
waylon@waylon-VirtualBox:~/Desktop/Compiler$
```

And statistic turns identifier frequencies listing on.

```
1  #pragma source off
2  #pragma token off
3  #pragma statistic on
```

```
waylon@waylon-VirtualBox: ~/Desktop/Compiler
<delim:}>
<id:struct>
<id:a>
<delim:{>
<id:var>
<delim:;>
<delim:}>
<delim:;>
waylon@waylon-VirtualBox:~/Desktop/Compiler$ ./scanner test.c
frequencies of identifiers:
main          1
argc          1
char          1
argv          1
a             12
b             4
c             5
d             1
f             1
s             1
printf        1
struct        1
var           1
waylon@waylon-VirtualBox:~/Desktop/Compiler$
```

Of course, if you turn everything off, scanner will output nothing.