COP5621/CIS4930 - Spring 2020
Assignment 2
cxref - lexical analysis for a C cross reference listing

cxref performs lexical analysis to produce a cross reference listing of a C file. After processing the file, all identifiers that are not reserved words will be printed in alphabetical order. Beside each identifier will be the list of line numbers in which the identifiers appear. If an identifier appears in a line more than once, then only list that line number once. Note that identifiers in comments, strings, character literals, or preprocessor commands should be ignored.

You should not make any assumptions about the number of identifiers or the number of lines on which each identifier can appear. You should create a single lex file to generate the cross reference listing. You should at least have lex rules to detect the following cases. You may find additional rules are necessary.

a. start of a C comment
b. end of a C comment
c. start or end of a string
d. start or end of a character literal
e. list of reserved words
f. preprocessor command
g. newline
h. default character

The list of C reserved words are given below.

```
auto        double      int         struct
break       else        long        switch
case        extern      register    typedef
char        float       return      union
continue    for         short       unsigned
default     goto        sizeof      void
do          if          static      while
```

You should comment your program so that others (e.g. the grader) can understand it. You should also have comments at the top of the file indicating your name, this course, and the assignment. You have to implement this assignment in C so it will be compatible with lex. Create a file called *cxref.l*. To process your lex specification you can simply issue the command:

```
lex cxref.l
```

You also need to link this file with the lex library as done with the following command.

```
gcc -g -o cxref lex.yy.c -ll
```

An example is given on the following page. You can check the output of your solution with the output produced by the executable */home/faculty/whalley/cop5621exec/cxref*. Your output should exactly match this executable's output. Note that this executable can only be executed on the machine *program*. Your solution will also be tested on the machine *program*. E-mail your lex specification file to *whalley@cs.fsu.edu* before the beginning of class on January 21. Please put COP5621 somewhere on the subject line of the message.

The following C program given as input:

```
static char *sccsid = "@(#)echo.c   4.1 (Berkeley) 10/1/80";
#include <stdio.h>

main(argc, argv)
int argc;
char *argv[];
{
      register int i, nflg;

      nflg = 0;
      if(argc > 1 && argv[1][0] == '-' && argv[1][1] == 'n') {
            nflg++;
            argc--;
            argv++;
      }
      for(i=1; i<argc; i++) {
            fputs(argv[i], stdout);
            if (i < argc-1)
                  putchar(' ');
      }
      if(nflg == 0)
            putchar('\n');
      exit(0);
}
```

would produce the following output:

```
                    argc: 4, 5, 11, 13, 16, 18
                    argv: 4, 6, 11, 14, 17
                    exit: 23
                   fputs: 17
                       i: 8, 16, 17, 18
                    main: 4
                    nflg: 8, 10, 12, 21
                  putchar: 19, 22
                   sccsid: 1
                   stdout: 17
```