**Lab 2: PRELIMINARY SCANNING APPLICATIONS**

Sample Program

//Program to remove single and multiline comments from a given ‘C’ file.

#include <stdio.h>

#include <stdlib.h>

int main()

{

FILE \*fa, \*fb;

int ca, cb;

fa = fopen("sample\_file.c", "r");

if (fa == NULL)

{

printf("Cannot open file \n");

exit(0);

}

fb = fopen("sample\_file\_out.c", "w");

ca = getc(fa);

while (ca != EOF)

{

if(ca==' ')

{

putc(ca,fb);

while(ca==' ')

ca = getc(fa);

}

if (ca=='/')

{

cb = getc(fa);

if (cb == '/')

{

while(ca != '\n')

ca = getc(fa);

}

else if (cb == '\*')

{

do

{

while(ca != '\*')

ca = getc(fa);

ca = getc(fa);

} while (ca != '/');

}

else

{

putc(ca,fb);

putc(cb,fb);

}

}

else

{

putc(ca,fb);

}

ca = getc(fa);

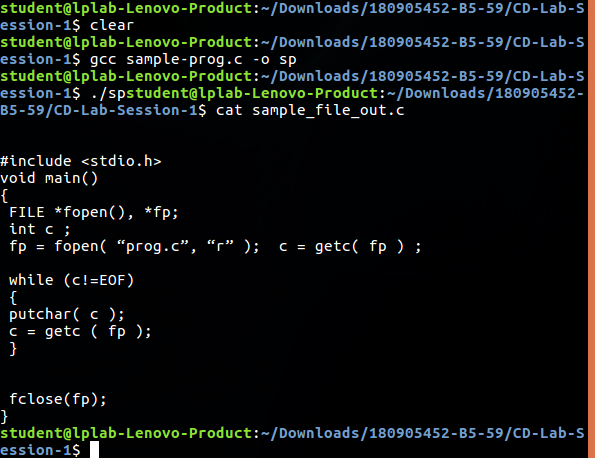
}

fclose(fa);

fclose(fb);

return 0;

}



Write a C Program

1. That takes a file as input and replaces blank spaces and tabs by single space and

writes the output to a file.

#include <stdio.h>

#include <stdlib.h>

int main()

{

int flag=0;

char c;

FILE \*f1,\*f2;

f1 = fopen("sample\_file.c", "r");

f2 = fopen("outputl2q2.c", "w");

if(f1 == NULL || f2 == NULL)

{

perror("Either the input or the output file doesn't exist\n");

return 1;

}

while(1)

{

c = fgetc(f1);

if(c==EOF)

{

break;

}

else if(!flag && (c==' '||c=='\t'))

{

fputc(' ', f2);

flag = 1;

}

else if(!(c==' '||c=='\t'))

{

flag = 0;

fputc(c, f2);

}

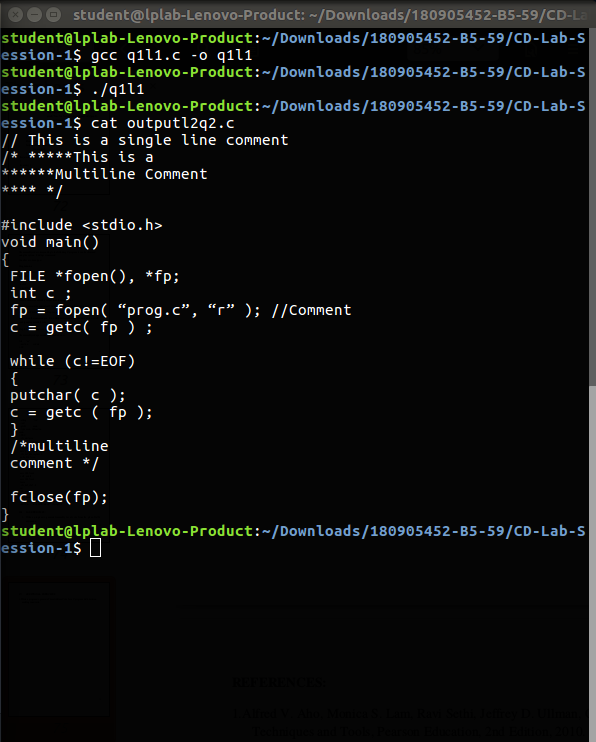
}

fclose(f1);

fclose(f2);

}

Output



2. To discard preprocessor directives from the given input ‘C’ file.

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#define FILEINPUT "q1l1.c"

#define FILEOUTPUT "Outputl1q2.c"

const char \*direct[] = {"#include","#define","#if"};

int is\_directive(const char \*str)

{

for(int i = 0; i < sizeof(direct)/sizeof(char \*); i++)

{

int len = strlen(direct[i]);

if(strncmp(str, direct[i], len) == 0)

{

return 1;

}

}

return 0;

}

int main()

{

char buf[2048];

FILE \*f1,\*f2;

f1 = fopen(FILEINPUT, "r");

f2 = fopen(FILEOUTPUT, "w");

if(f1 == NULL || f2 == NULL)

{

perror("Either the input or the output file doesn't exist\n");

return 1;

}

while(fgets(buf, 2048, f1) != NULL)

{

if(!is\_directive(buf))

{

fputs(buf, f2);

}

}

fclose(f1);

fclose(f2);

f1= fopen(FILEINPUT,"w");

f2=fopen(FILEOUTPUT,"r");

char copy;

copy=getc(f2);

while(copy!=EOF)

{

putc(copy,f1);

copy=getc(f2);

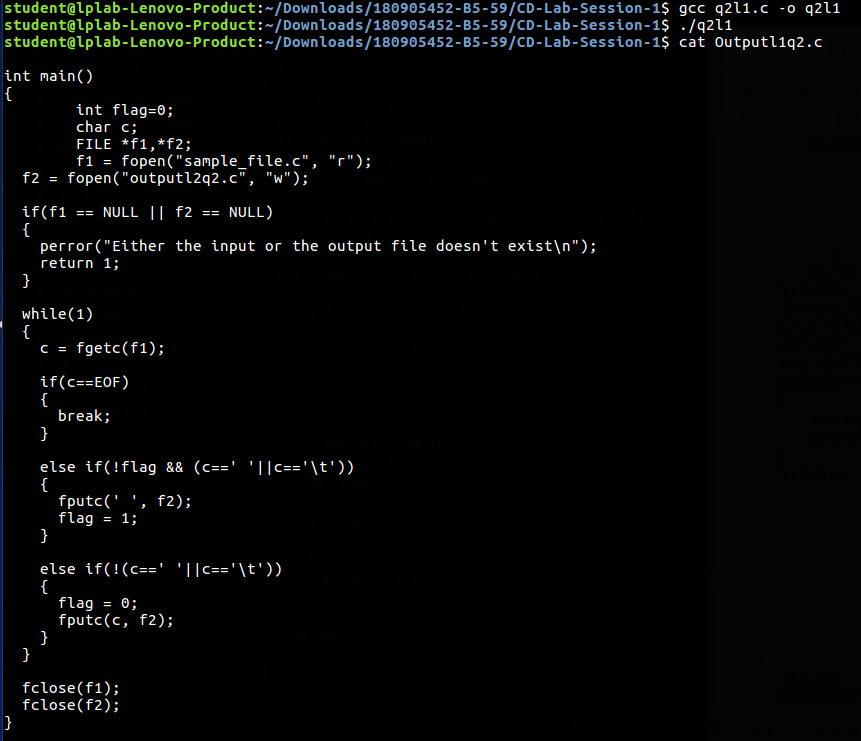
}

fclose(f1);

fclose(f2);

}

Output



3. That takes C program as input, recognizes all the keywords and prints them in

upper case.

#include <stdio.h>

#include <stdlib.h>

#define FILEINPUT "q2l1.c"

const char \*keywords[] = {"const", "char", "int","return", "for", "while", "do", "switch",

"if", "else","case", "break"};

int is\_keyword(const char \*str)

{

for(int i = 0; i < sizeof(keywords)/sizeof(char \*); ++i)

{

if(strcmp(str, keywords[i]) == 0)

{

return 1;

}

}

return 0;

}

void strtoupper(char \*str, const int len)

{

for(int i = 0; i < len; ++i)

{

str[i] = toupper(str[i]);

}

}

enum

{

INSIDE\_WORD,

OUTSIDE\_WORD

};

int main()

{

FILE \*f1,\*f2;

int line=1,col=1,k=0;

char c,buf[512];

f1 = fopen(FILEINPUT, "r");

if(f1 == NULL)

{

perror("The input file doesn't exist\n");

return 1;

}

int state = OUTSIDE\_WORD;

printf("Keywords : \n");

while((c = fgetc(f1)) != EOF)

{

switch(state)

{

case INSIDE\_WORD:

if(isalpha(c))

{

buf[k++]=c;

}

else

{

buf[k]='\0';

if(is\_keyword(buf))

{

strtoupper(buf, k);

printf("%s : at (%d , %d)\n", buf, line, col - k);

}

k=0;

state=OUTSIDE\_WORD;

}

break;

case OUTSIDE\_WORD:

if(isalpha(c))

{

buf[k++]=c;

state=INSIDE\_WORD;

}

break;

}

if(c == '\n')

{

++line;

col = 1;

}

else

{

++col;

}

}

fclose(f1);

}

Output

