Q1

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

#include <stdio.h>

int main()

{

FILE \*fa, \*fb;

int ca, cb;

fa = fopen("input.c", "r");

if (fa == NULL)

{

printf("Cannot open file \n");

exit(0);

}

fb = fopen("output.c", "w");

ca = getc(fa);

while (ca != EOF)

{

if (ca == ' ')

{

putc(ca, fb);

while (ca == ' ')

ca = getc(fa);

}

if (ca == '\t')

{

int temp\_space = ' ';

putc(temp\_space, fb);

while (ca == ' ' || ca == '\t')

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);

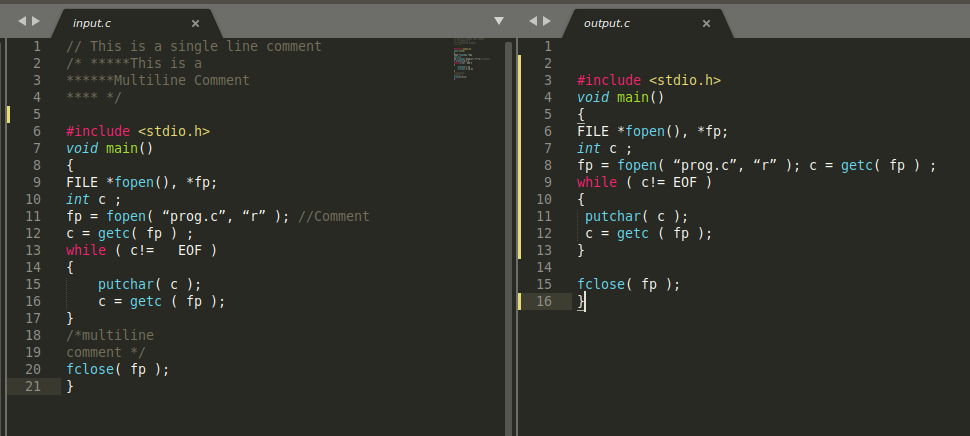
}

fclose(fa);

fclose(fb);

return 0;

}



Q2

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

#include <stdio.h>

#include <stdbool.h>

#include <stdlib.h>

#include <string.h>

int main()

{

FILE \*fa, \*fb;

int ca, cb;

fa = fopen("tempoutput.c", "r");

if (fa == NULL)

{

printf("Cannot open file \n");

exit(0);

}

fb = fopen("output.c", "w");

ca = getc(fa);

while (ca != EOF)

{

if (ca=='#')

{

cb = getc(fa);

if (cb == 'd' || cb == 'i' || cb == 'e' || cb == 'u' || cb == 'p' )

{

while(ca != '\n')

ca = getc(fa);

int new\_line = '\n';

putc(new\_line,fb);

}

else

{

putc(ca,fb);

putc(cb,fb);

}

}

else

putc(ca,fb);

ca = getc(fa);

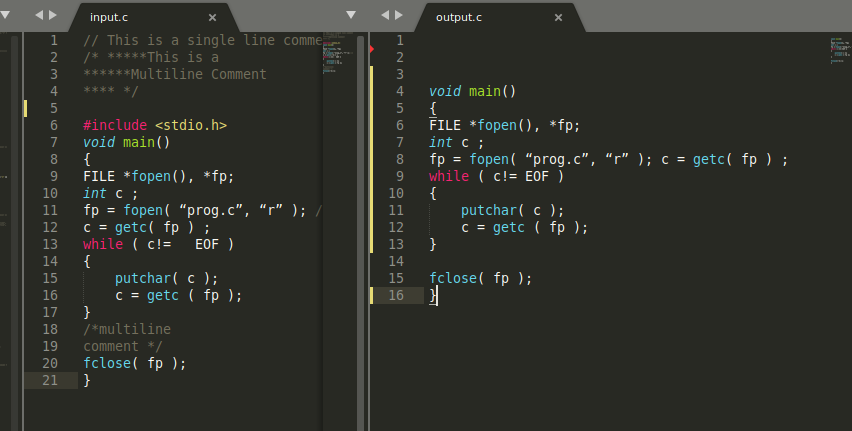
}

fclose(fa);

fclose(fb);

return 0;

}



Q3

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

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <ctype.h>

int main()

{

// File Handling operations

FILE \*fa, \*fb;

int ca, cb;

fa = fopen("input.c", "r");

if (fa == NULL)

{

printf("Cannot open file \n");

exit(0);

}

fb = fopen("tempoutput1.c", "w");

// For the keywords

char buffer[100];

char keywords\_table[32][10] = {

"auto", "break", "case", "char", "const", "continue", "default", "do",

"double", "else", "enum", "extern", "float", "for", "goto", "if", "int", "long", "register",

"return", "short", "signed", "sizeof", "static", "struct",

"switch", "typedef", "union", "unsigned", "void", "volatile", "while"};

// Counters

int index = 0;

int row\_no = 1;

// int column\_no = 1;

// Actual Scanning

ca = getc(fa);

while (ca != EOF)

{

// Single Line Comments

if (ca == ' ')

{

putc(ca, fb);

while (ca == ' ')

ca = getc(fa);

}

// Multiline Comments

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);

// Scanning for keywords

buffer[index++] = ca;

if (ca == '\n')

{

row\_no++;

for (int i = 0; i < 32; ++i)

{

if (strstr(buffer, keywords\_table[i]) != NULL)

{

printf("%s row number: %d column nuber: %d location : %s \n", keywords\_table[i], row\_no, index, strstr(buffer, keywords\_table[i]));

// puts(keywords\_table[i]);

}

}

memset(buffer, 0, sizeof(buffer));

// buffer[0] = 0;

index = 0;

}

// Getting the next chracter from the file descriptor

ca = getc(fa);

}

// Closing the file descriptors

fclose(fa);

fclose(fb);

return 0;

}

