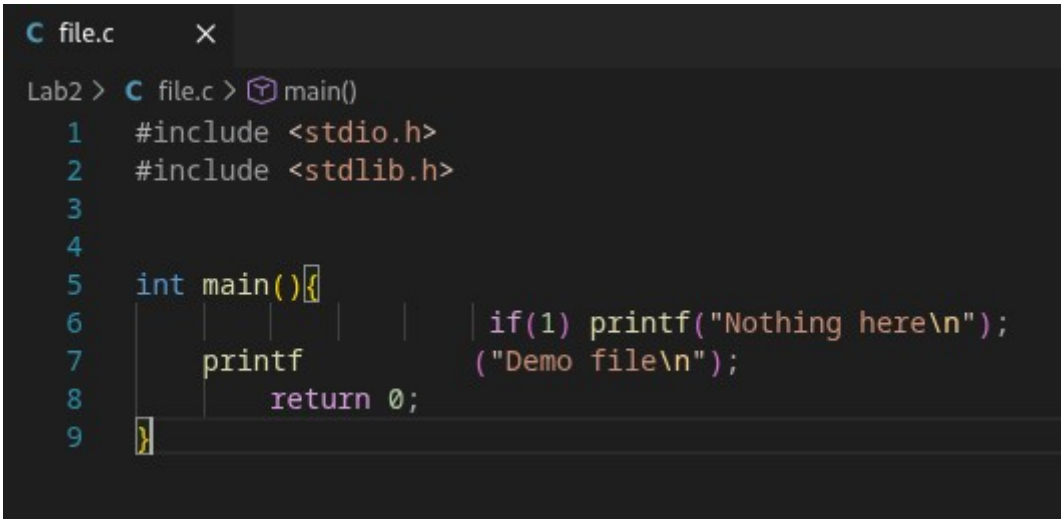
Lab – 2 PRELIMINARY SCANNING APPLICATIONS

Name : Aditya Sinha
Reg no : 230905218
CSE-A-27

Original Test File :



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>
#include <string.h>

int main(){
FILE *src,*dst;
char file[50];
printf("Enter file to remove spaces from : ");
fgets(file, 50, stdin);
file[strcspn(file, "\n")] = '\0';
src = fopen(file, "r");
if(src == NULL){
perror("error ");
return 1;
}

printf("Enter destination file : ");
fgets(file, 50, stdin);
file[strcspn(file, "\n")] = '\0';
dst = fopen(file, "w");
if(dst == NULL){
perror("error ");
fclose(src);
```

```
return 1;
}

int ch;
int state = 0;
while ((ch = fgetc(src)) != EOF) {
if(ch == ' '){
if(state == 0) fputc(ch, dst);
state = 1;
}
else state = 0;
if(state == 0)fputc(ch, dst);
}

fclose(src);
fclose(dst);

return 0;
}
```

Output :

file_after_q1.c

Lab2 > file_after_q1.c > main()

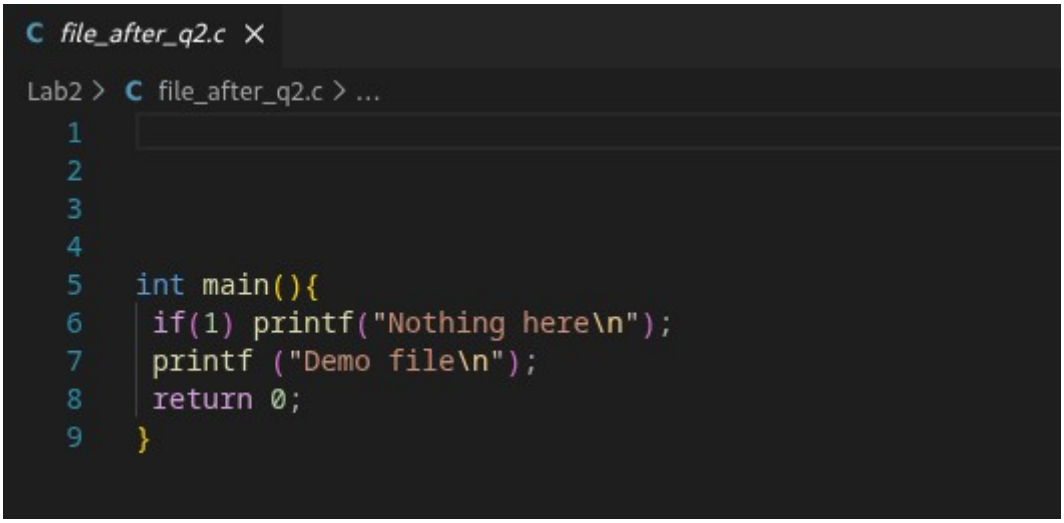
```c
#include <stdio.h>
#include <stdlib.h>


int main(){
 if(1) printf("Nothing here\n");
 printf ("Demo file\n");
 return 0;
}
```

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

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

int main(){
FILE *src,*dst;
char file[50];
printf("Enter file to remove preprocessor directives from : ");
fgets(file, 50, stdin);
file[strcspn(file, "\n")] = '\0';
src = fopen(file, "r");
if(src == NULL){
perror("error ");
```

```
return 1;
}

printf("Enter destination file : ");
fgets(file, 50, stdin);
file[strcspn(file, "\n")] = '\0';
dst = fopen(file, "w");
if(dst == NULL){
perror("error ");
fclose(src);
return 1;
}

int ch;
int state = 0;
while ((ch = fgetc(src)) != EOF) {
if(ch == '#') state = 1;
if(ch == '\n') state = 0;
if(state == 0)fputc(ch, dst);
}

fclose(src);
fclose(dst);

return 0;
}
```

Output :



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

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>

const char *keywords[] = {
```

```
"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"
};

void checkAndPrintWord(char *word, FILE *dst);
int isKeyword(char *word);
void makeWordUpperAndPrint(char *word, FILE *dst);
char *makeWordUpper(char *word);
void printWord(char *word, FILE *dst);

int main(void)
{
FILE *src, *dst;
char file[50];

printf("Enter source file: ");
fgets(file, sizeof(file), stdin);
file[strcspn(file, "\n")] = '\0';

src = fopen(file, "r");
if (!src) {
perror("Source file error");
return 1;
}

printf("Enter destination file: ");
fgets(file, sizeof(file), stdin);
file[strcspn(file, "\n")] = '\0';

dst = fopen(file, "w");
if (!dst) {
perror("Destination file error");
fclose(src);
return 1;
}

int ch, i = 0;
char word[1024];

while ((ch = fgetc(src)) != EOF) {
if (isalnum(ch) || ch == '_') {
word[i++] = ch;
} else {
if (i > 0) {
word[i] = '\0';
```

```
checkAndPrintWord(word, dst);
i = 0;
}
fputc(ch, dst);
}
}

if (i > 0) {
word[i] = '\0';
checkAndPrintWord(word, dst);
}

fclose(src);
fclose(dst);
return 0;
}

void checkAndPrintWord(char *word, FILE *dst)
{
if (isKeyword(word)) {
makeWordUpperAndPrint(word, dst);
} else {
printWord(word, dst);
}
}

int isKeyword(char *word)
{
int count = sizeof(keywords) / sizeof(keywords[0]);

for (int i = 0; i < count; i++) {
if (strcmp(word, keywords[i]) == 0)
return 1;
}
return 0;
}

void makeWordUpperAndPrint(char *word, FILE *dst)
{
makeWordUpper(word);
printWord(word, dst);
printf("Keyword detected: %s\n", word);
}

char *makeWordUpper(char *word)
{
for (int i = 0; word[i]; i++) {
word[i] = toupper((unsigned char)word[i]);
}
return word;
```

```
}

void printWord(char *word, FILE *dst)
{
fputs(word, dst);
}
```

Output :