

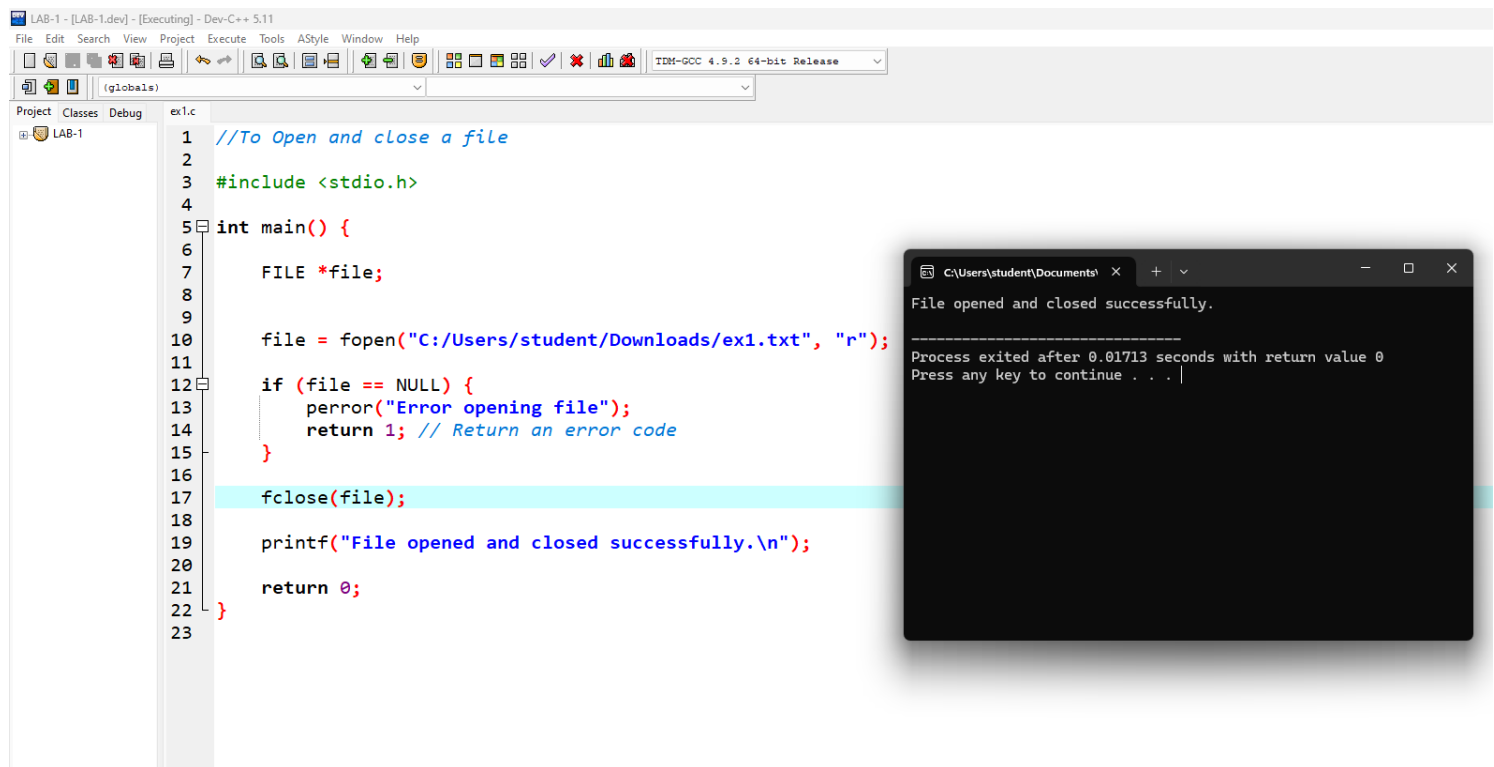
SWE-4001
System Programming

LAB-1

Working on Files and creation of Symbol table

Open and Close a File:

- Write a program to open a file named "test.txt" in read mode. Check if the file is successfully opened. If it is, close the file.



The screenshot displays the Dev-C++ IDE interface. The main window shows a C program named `ex1.c` with the following code:

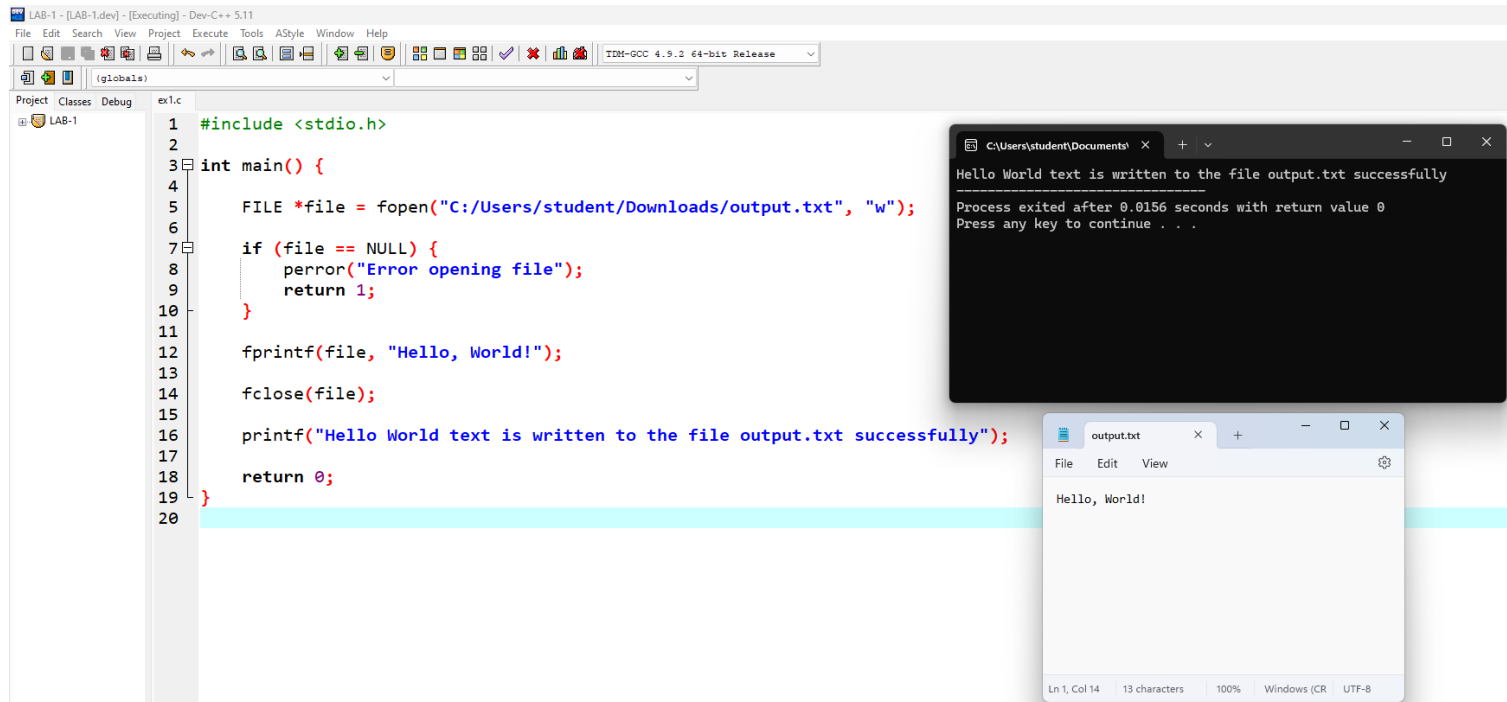
```
1 //To Open and close a file
2
3 #include <stdio.h>
4
5 int main() {
6     FILE *file;
7
8     file = fopen("C:/Users/student/Downloads/ex1.txt", "r");
9
10    if (file == NULL) {
11        perror("Error opening file");
12        return 1; // Return an error code
13    }
14
15    fclose(file);
16
17    printf("File opened and closed successfully.\n");
18
19    return 0;
20 }
```

The line `fclose(file);` is highlighted in light blue. To the right of the code editor, a terminal window is open, showing the output of the program:

```
C:\Users\student\Documents\ X + -
File opened and closed successfully.
-----
Process exited after 0.01713 seconds with return value 0
Press any key to continue . . .
```

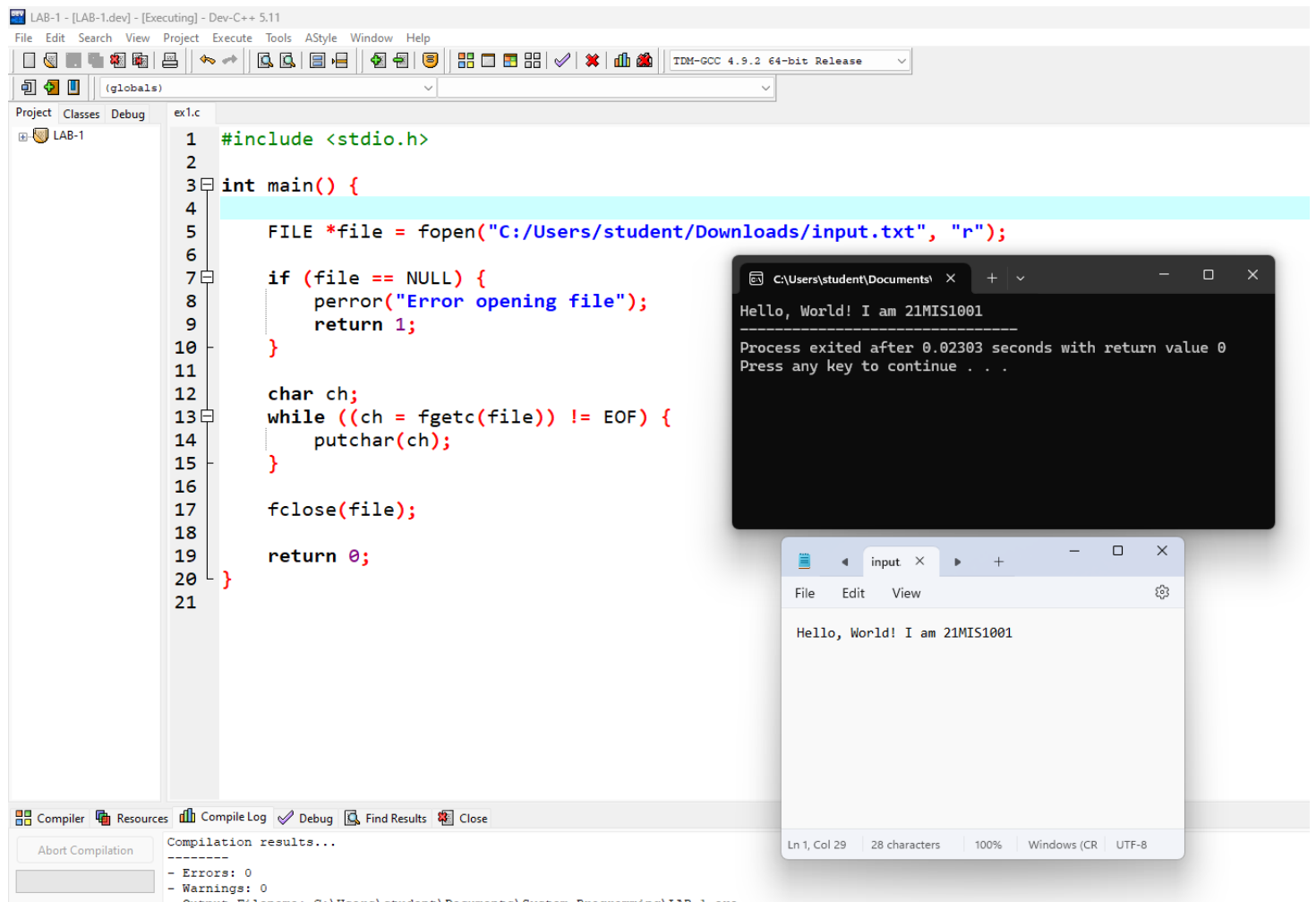
Write to a File:

- Write a program that opens a file named "output.txt" in write mode. Write the string "Hello, World!" to the file. Close the file after writing.



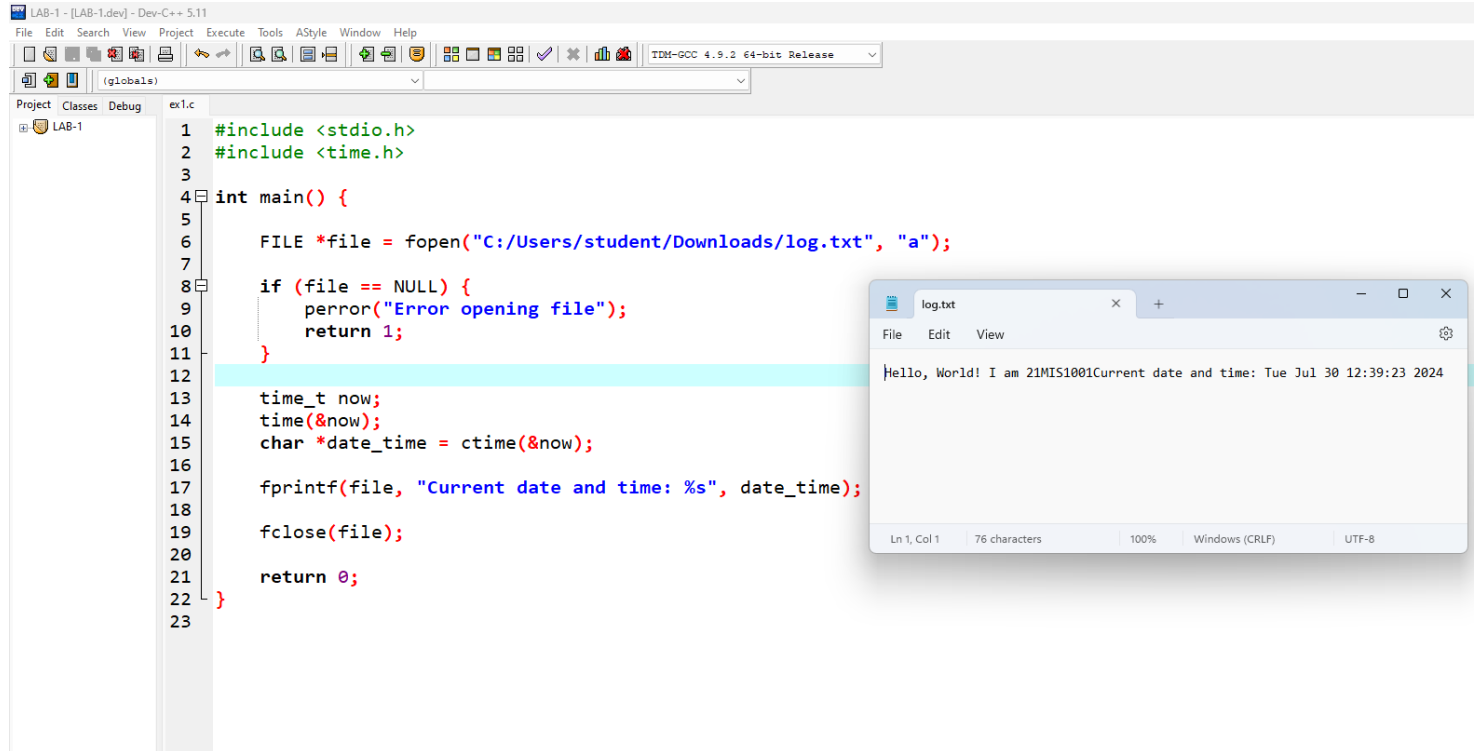
Read from a File:

- Write a program to read the contents of a file named "input.txt" and print the contents to the console.



Append to a File:

- Write a program that opens a file named "log.txt" in append mode. Append the current date and time to the file. Close the file after appending.



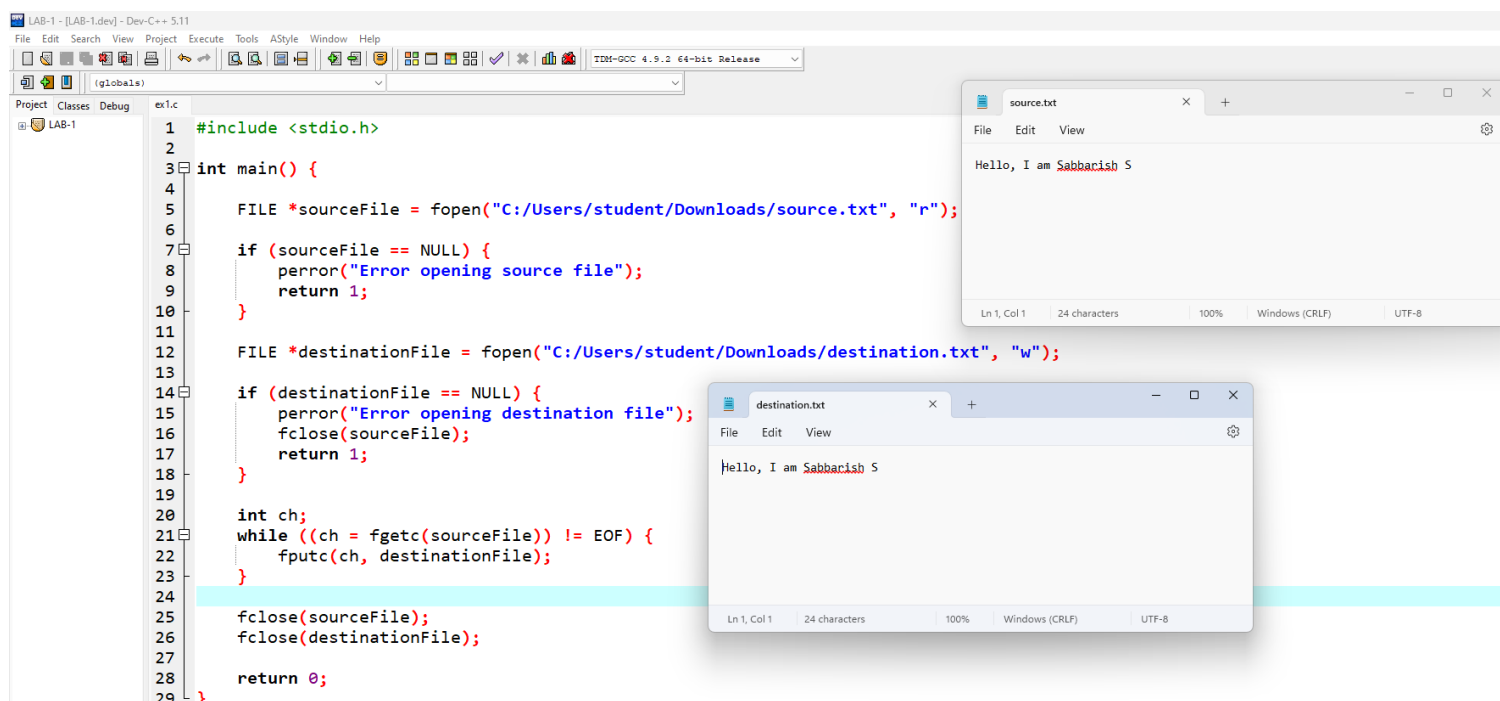
```
1 #include <stdio.h>
2 #include <time.h>
3
4 int main() {
5
6     FILE *file = fopen("C:/Users/student/Downloads/log.txt", "a");
7
8     if (file == NULL) {
9         perror("Error opening file");
10        return 1;
11    }
12
13    time_t now;
14    time(&now);
15    char *date_time = ctime(&now);
16
17    fprintf(file, "Current date and time: %s", date_time);
18
19    fclose(file);
20
21    return 0;
22 }
23
```

log.txt

Hello, World! I am 21MIS1001Current date and time: Tue Jul 30 12:39:23 2024

Copy a File:

- Write a program to copy the contents of a file named "source.txt" to a new file named "destination.txt".



```
1 #include <stdio.h>
2
3 int main() {
4
5     FILE *sourceFile = fopen("C:/Users/student/Downloads/source.txt", "r");
6
7     if (sourceFile == NULL) {
8         perror("Error opening source file");
9         return 1;
10    }
11
12    FILE *destinationFile = fopen("C:/Users/student/Downloads/destination.txt", "w");
13
14    if (destinationFile == NULL) {
15        perror("Error opening destination file");
16        fclose(sourceFile);
17        return 1;
18    }
19
20    int ch;
21    while ((ch = fgetc(sourceFile)) != EOF) {
22        fputc(ch, destinationFile);
23    }
24
25    fclose(sourceFile);
26    fclose(destinationFile);
27
28    return 0;
29 }
```

source.txt

Hello, I am Sabbarish S

destination.txt

Hello, I am Sabbarish S

Count the Number of Lines in a File:

- Write a program that opens a file named "data.txt" and counts the number of lines in the file. Print the total number of lines.

The screenshot shows a C++ IDE with a project named "LAB-1". The source file "ex1.c" contains the following code:

```
1 #include <stdio.h>
2
3 int main() {
4     FILE *file = fopen("C:/Users/student/Downloads/data.txt", "r");
5
6     if (file == NULL) {
7         perror("Error opening file");
8         return 1;
9     }
10
11     int lineCount = 0;
12     int ch;
13
14     while ((ch = fgetc(file)) != EOF) {
15         if (ch == '\n') {
16             lineCount++;
17         }
18     }
19
20     if (ftell(file) > 0 && ch != '\n') {
21         lineCount++;
22     }
23
24     fclose(file);
25
26     printf("Total number of lines: %d\n", lineCount);
27
28     return 0;
29 }
```

The program is executed, and the output window shows:

```
Total number of lines: 5
-----
Process exited after 0.03027 seconds with return value 0
Press any key to continue . . .
```

The file "data.txt" is also open in the IDE, showing its contents:

```
include
main
data
integer
lineeee
```

Count the Number of Words in a File:

- Write a program to count the number of words in a file named "text.txt". Assume that words are separated by spaces, tabs, or newlines

The screenshot shows a C++ IDE with a project named "LAB-1". The source file "ex1.c" contains the following code:

```
1 #include <stdio.h>
2 #include <ctype.h>
3
4 int main() {
5     FILE *file = fopen("C:/Users/student/Downloads/text.txt", "r");
6
7     if (file == NULL) {
8         perror("Error opening file");
9         return 1;
10    }
11
12    int wordCount = 0;
13    int inWord = 0;
14    int ch;
15
16    while ((ch = fgetc(file)) != EOF) {
17        if (isspace(ch)) {
18            if (inWord) {
19                wordCount++;
20                inWord = 0;
21            }
22        } else {
23            inWord = 1;
24        }
25    }
26
27    if (inWord) {
28        wordCount++;
29    }
30
31    fclose(file);
32
33    printf("Total number of words: %d\n", wordCount);
34
35    return 0;
36 }
```

The program is executed, and the output window shows:

```
Total number of words: 6
-----
Process exited after 0.01144 seconds with return value 0
Press any key to continue . . .
```

The file "text.txt" is also open in the IDE, showing its contents:

```
include main
main
data
integer
line
```

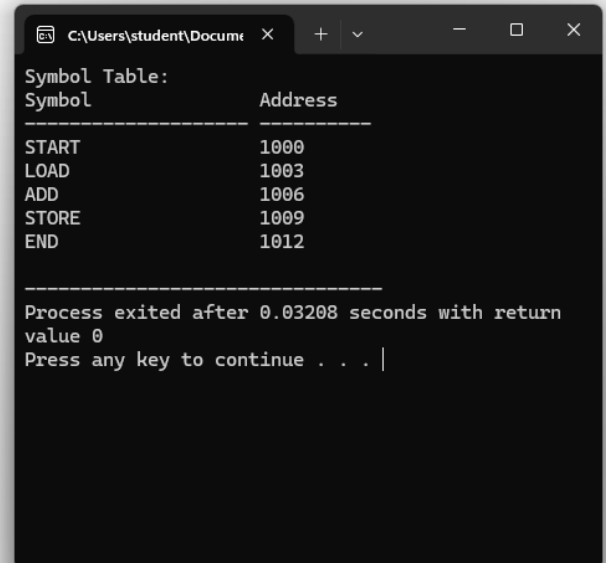
Creation of Symbol table

Write a C program that implements a symbol table with functions to create, insert, modify, search, and display symbols. The program reads symbols from a file named "input.txt", with each symbol assigned a starting address of 1000 and each symbol occupying 3 bytes.

Requirements:

- Read symbols from "input.txt".
- Assign addresses starting from 1000, with each symbol occupying 3 bytes.
- Store each symbol and its corresponding address in a symbol table.
- Print the symbol table in a formatted manner

```
[*] ex1.c
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <string.h>
4
5  #define MAX_SYMBOLS 100
6  #define SYMBOL_SIZE 256
7
8  typedef struct {
9      char symbol[SYMBOL_SIZE];
10     int address;
11 } Symbol;
12
13 Symbol symbolTable[MAX_SYMBOLS];
14 int symbolCount = 0;
15
16 void readSymbolsFromFile(const char *filename);
17 void insertSymbol(const char *symbol, int address);
18 void displaySymbolTable();
19
20 int main() {
21     readSymbolsFromFile("C:/Users/student/Downloads/input.txt");
22
23
24     displaySymbolTable();
25
26     return 0;
27 }
28
29
30 void readSymbolsFromFile(const char *filename) {
31     FILE *file = fopen(filename, "r");
32     if (file == NULL) {
33         perror("Error opening file");
34         exit(EXIT_FAILURE);
35     }
36
37     char symbol[SYMBOL_SIZE];
38     int address = 1000;
39     int i;
40
41     while (fgets(symbol, sizeof(symbol), file) != NULL) {
42         symbol[strcspn(symbol, "\n")] = '\0';
43
44         if (strlen(symbol) > 0) {
```



```
C:\Users\student\Docume X + - □ X
Symbol Table:
Symbol      Address
-----
START      1000
LOAD       1003
ADD        1006
STORE      1009
END        1012

-----
Process exited after 0.03208 seconds with return
value 0
Press any key to continue . . . |
```

