



CMPT125 TUTORIAL 3: FILE I/O

Thanks to Jamal (jrahim@sfu.ca)

IN THIS TUTORIAL

SFU

- We learned how to direct content of files into stdin.
- We learned how to direct stdout to a file.
- We will now learn how to directly read from and write to files.

READING A FILE

SFU

Exercise I:

- Let's read a file and display the contents!
- We will ask the user to input the name of the file to be read.
- We will check if the file was opened successfully.
- Finally, the contents of the file will be printed.

```
T3 > C read.c
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  int main() {
5      FILE *fptr;
6      char fname[20];
7      printf("\n\n Read an existing file :\n");
8      printf("-----\n");
9      printf(" Input the filename to be opened : ");
10     scanf("%s",fname);
11
12     /* opening file for reading */
13     fptr = fopen (fname, "r");
14
15     /* Check if file is open */
16     if (fptr == NULL) {
17         printf(" File does not exist or cannot be opened.\n");
18         exit(0);
19     }
20
21     printf("\n The content of the file %s is :\n",fname);
22     char str[256];
23     while( fgets(str, 256, fptr)!=NULL ) {
24         /* printing content to stdout */
25         printf("%s", str);
26     }
27     printf("\n");
28     fclose(fptr)
29     return 0;
30 }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
jrahim@asb9804u-a01:~/sp125/T3$ gcc read.c -o read
jrahim@asb9804u-a01:~/sp125/T3$ ./read

Read an existing file :
-----
Input the filename to be opened : read.txt

The content of the file read.txt is :
This is line 1
This is line 2
This is the third and last line
jrahim@asb9804u-a01:~/sp125/T3$
```

WRITING TO FILE

SFU

Exercise 2:

- Let's extend the previous exercise and write the read content to a new file!
- We will ask the user to input the name of the new file.
- We will check if the file was opened successfully.
- Finally, the contents of the previous file will be copied to the new file.

```
T3 > C copy.c
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  int main() {
5      FILE *fptr;
6      char fname[20];
7      printf("\n\n Read an existing file :\n");
8      printf("-----\n");
9      printf(" Input the filename to be opened : ");
10     scanf("%s",fname);
11
12     /* opening file for reading */
13     fptr = fopen (fname, "r");
14
15     printf("\n Copying to file :\n");
16     printf("-----\n");
17     printf(" Create then Input the filename to be opened : ");
18     scanf("%s",fname);
19
20     /* opening file for writing */
21     FILE *fptr2;
22     fptr2 = fopen (fname, "w");
23
24     /* Check if files is open */
25     if (fptr == NULL || fptr2 == NULL) {
26         printf(" File(s) do not exist or cannot be opened.\n");
27         exit(0);
28     }
29
30     char str[256];
31     while( fgets(str, 256, fptr)!=NULL ) {
32         /* printing content to writing file */
33         fprintf(fptr2, "%s", str);
34     }
35     printf("\ncontent copied to %s\n", fname);
36     fclose(fptr);
37     fclose(fptr2);
38     return 0;
39 }
```

WRITING TO FILE – TERMINAL OUTPUT

SFU

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL

jrahim@asb9804u-a01:~/sp125/T3$ cat read.txt
This is line 1
This is line 2
This is the third and last line
jrahim@asb9804u-a01:~/sp125/T3$ cat write.txt
jrahim@asb9804u-a01:~/sp125/T3$ gcc copy.c -o copy
jrahim@asb9804u-a01:~/sp125/T3$ ./copy

Read an existing file :
-----
Input the filename to be opened : read.txt

Copying to file :
-----
Create then Input the filename to be opened : write.txt

content copied to write.txt
jrahim@asb9804u-a01:~/sp125/T3$ cat write.txt
This is line 1
This is line 2
This is the third and last line
jrahim@asb9804u-a01:~/sp125/T3$ █
```

WRITING TO FILE – MODIFICATION

SFU

Exercise 3:

- Let's modify the previous exercise.
- We will try the append mode, "a", to open the writing file this time (line 22), and run the code to see what happens.

```
T3 > C copy2.c
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  int main() {
5      FILE *fptr;
6      char fname[20];
7      printf("\n\n Read an existing file :\n");
8      printf("-----\n");
9      printf(" Input the filename to be opened : ");
10     scanf("%s",fname);
11
12     /* opening file for reading */
13     fptr = fopen (fname, "r");
14
15     printf("\n Copying to file :\n");
16     printf("-----\n");
17     printf(" Create then Input the filename to be opened : ");
18     scanf("%s",fname);
19
20     /* opening file for writing */
21     FILE *fptr2;
22     fptr2 = fopen (fname, "a");
23
24     /* Check if files is open */
25     if (fptr == NULL || fptr2 == NULL) {
26         printf(" File(s) do not exist or cannot be opened.\n");
27         exit(0);
28     }
29
30     char str[256];
31     while( fgets(str, 256, fptr)!=NULL ) {
32         /* printing content to writing file */
33         fprintf(fptr2, "%s", str);
34     }
35     printf("\ncontent copied to %s\n", fname);
36     fclose(fptr);
37     fclose(fptr2);
38     return 0;
39 }
```

WRITING TO FILE – MODIFICATION – TERMINAL

SFU

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL

jrahim@asb9804u-a01:~/sp125/T3$ cat read.txt
This is line 1
This is line 2
This is the third and last line
jrahim@asb9804u-a01:~/sp125/T3$ cat write.txt
This is line 1
This is line 2
This is the third and last line
jrahim@asb9804u-a01:~/sp125/T3$ gcc copy2.c -o copy2
jrahim@asb9804u-a01:~/sp125/T3$ ./copy2

Read an existing file :
-----
Input the filename to be opened : read.txt

Copying to file :
-----
Create then Input the filename to be opened : write.txt

content copied to write.txt
jrahim@asb9804u-a01:~/sp125/T3$ cat write.txt
This is line 1
This is line 2
This is the third and last line
This is line 1
This is line 2
This is the third and last line
jrahim@asb9804u-a01:~/sp125/T3$ █
```

IMPORTANT CONCEPTS RECAP

SFU

- Reading from and writing to files directly is almost the same as stdin and stdout.
- File opened using `fopen()`. Open mode has to be specified in this function. Some open modes are:
 - “r”: Opens a file for reading. The file must exist.
 - “w”: Creates an empty file for writing. If a file with the same name already exists, its content is erased and the file is considered as a new empty file.
 - “a”: Appends to a file. Writing operations, append data at the end of the file. The file is created if it does not exist.
 - “r+”: Opens a file to update both reading and writing. The file must exist.
 - “w+”: Creates an empty file for both reading and writing.
 - “a+”: Opens a file for reading and appending.

IMPORTANT CONCEPTS RECAP – CONTINUED

SFU

- Reading content can be done by using `fgets()`. We have used this before too! The only difference is that we can pass the open file pointer to `fgets` instead of `stdin`. Example:
 - `fgets(str, 256, fptr)` instead of `fgets(str, 256, stdin)`
- Reading can also be done by using `fscanf()` as such: `fscanf(fptr, "%s", str)`
- We are assuming that the maximum length of each line in the text file is 256. If size is unknown, we can use `fgetc()` to read character by character.
- Writing content can be done by using `fprintf()` and passing the file pointer as the first argument.

EXERCISE – CAESAR CIPHER

SFU

In cryptography, a Caesar cipher, is one of the simplest and most widely known encryption techniques. It is a type of substitution cipher in which each letter in the plaintext is replaced by a letter some fixed number of positions down the alphabet. For example, with a left shift of 3, D would be replaced by A, E would become B, and so on.

Copy the following text to a txt file:

```
F PLIBJKIV PTBXO QEXQ F XJ RM QL KL DLLA
QEB NRFZH YOLTK CLU GRJMP LSBO QEB IXWV ALD
```

Now read this file, decrypt the code using a left shift of 3, and store the decrypted text to a new file. (For decryption, you will be doing the opposite and applying a right shift of 3.)

Note about exception cases for decryption: Z will be right shifted to A, Y to B, and X to C.

PLEASE ATTEMPT IT YOURSELF BEFORE MOVING ON!

CAESAR CIPHER – SOLUTION

SFU

```
T3 > C caesar.c > main()
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  int main() {
5      FILE *fptr;
6      char fname[20];
7      printf("\n\n Read an existing file :\n");
8      printf("-----\n");
9      printf(" Input the filename with encrypted code to be opened : ");
10     scanf("%s",fname);
11
12     /* opening file for reading */
13     fptr = fopen (fname, "r");
14
15     printf("\n Decrypting to file :\n");
16     printf("-----\n");
17     printf(" Input the filename to store the decrypted text : ");
18     scanf("%s",fname);
19
20     /* opening file for writing */
21     FILE *fptr2;
22     fptr2 = fopen (fname, "w");
23
24     /* Check if files is open */
25     if (fptr == NULL || fptr2 == NULL) {
26         printf(" File(s) do not exist or cannot be opened.\n");
27         exit(0);
28     }
29
```

```
30     char str[256];
31
32     while( fgets(str, 256, fptr)!=NULL ) {
33         int i = 0;
34         while(str[i]!='\0') {
35             char new = str[i];
36             if(str[i]=='X') {
37                 new = 'A';
38             } else if(str[i]=='Y') {
39                 new = 'B';
40             } else if(str[i]=='Z') {
41                 new = 'C';
42             } else if(str[i]>='A' && str[i]<='W') {
43                 new += 3;
44             }
45             fprintf(fptr2, "%c", new);
46             i++;
47         }
48     }
49     printf("\ndecrypted text written to %s\n", fname);
50     fclose(fptr);
51     fclose(fptr2);
52     return 0;
53 }
```

CAESAR CIPHER – TERMINAL

SFU

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL

jrahim@asb9804u-a01:~/sp125/T3$ cat encrypted.txt
F PLIBJKIV PTBXO QEXQ F XJ RM QL KL DLLA
QEB NRFZH YOLTK CLU GRJMP LSBO QEB IXWV ALD
jrahim@asb9804u-a01:~/sp125/T3$ gcc caesar.c -o caesar
jrahim@asb9804u-a01:~/sp125/T3$ ./caesar

Read an existing file :
-----
Input the filename with encrypted code to be opened : encrypted.txt

Decrypting to file :
-----
Input the filename to store the decrypted text : decrypted.txt

decrypted text written to decrypted.txt
jrahim@asb9804u-a01:~/sp125/T3$ cat decrypted.txt
I SOLEMNLY SWEAR THAT I AM UP TO NO GOOD
THE QUICK BROWN FOX JUMPS OVER THE LAZY DOG
jrahim@asb9804u-a01:~/sp125/T3$ █
```