File Handling in C

An Introduction to File Operations in C Programming

Prasad Wilagama
Assistant Lecturer

16 July 2025

Objectives

- Understand the need for file handling
- Differentiate between types of file access
- Learn basic file operations: open, read, write, close
- Use standard file I/O functions in C
- Handle errors and end-of-file conditions

Why File Handling?

- Data stored in variables is temporary
- Files allow data persistence
- Ideal for storing user inputs, logs, configurations
- Used in real-world apps: text editors, databases, etc.

File Operations Overview

1. Opening a file
2. Reading from a file
3. Writing to a file
4. Closing the file

File Pointer and `fopen()`

- Syntax: `FILE *fp;`
- Use `fopen()` to open a file

Example

```
FILE *fp;
fp = fopen("data.txt", "r");
if (fp == NULL) {
    printf("File not found!\n");
}
```

Modes in `fopen()`

Mode	Description
"r"	Open for reading
"w"	Open for writing (creates if not exists)
"a"	Append to end of file
"r+"	Read and write
"w+"	Write and read (truncates)
"a+"	Append and read

Writing to a File

`fprintf()` or `fputs()` is used.

Example

```
FILE *fp;
fp = fopen("output.txt", "w");
fprintf(fp, "Hello UCSC students!");
fclose(fp);
```

Reading from a File

`fscanf()` or `fgets()` is used.

Example

```
FILE *fp;
char str[50];
fp = fopen("output.txt", "r");
fgets(str, 50, fp);
printf("Read: %s", str);
fclose(fp);
```

End-of-File and Error Checking

- Use `feof(fp)` to check EOF
- Use `ferror(fp)` to check error

Example

```
if (feof(fp)) { printf("End of file
n"); }
```

Closing a File

- Always close files with `fclose(fp);`
- Prevents memory leaks and corruption

Example

```
fclose(fp);
```

Mini Example: Copy File Content

```
FILE *src, *dest;
char ch;
src = fopen("input.txt", "r");
dest = fopen("copy.txt", "w");
while ((ch = fgetc(src)) != EOF) {
    fputc(ch, dest);
}
fclose(src);
fclose(dest);
```

Common Mistakes

- Forgetting to close the file
- Not checking if the file opened successfully
- Writing in read mode
- Reading after end-of-file

Practical Activity

- Task 1: Write a program to write student names and marks to a file
- Task 2: Read the file and calculate the average
- Task 3: Append new student details

Real-World Use Cases

- Log files in applications
- Config files for games/software
- Saving form data (e.g., student registration)
- Loading and saving game states

Summary

- Files help persist data
- Use `fopen()`, `fclose()`, `fprintf()`, `fscanf()` etc.
- Always validate file operations
- Practice reading/writing different file formats

Questions?

Contact: pdw@ucsc.cmb.ac.lk