File Handling in C

Course: Introduction to Programming and Data Structures

Laltu Sardar

Institute for Advancing Intelligence (IAI),
TCG Centres for Research and Education in Science and Technology (TCG Crest)

1. Basics of File Handling in C
   - Sample input/output from/to a file
   - Reading from a file
   - Writing to a file
   - `fseek` in C
2. File opening modes
   - Comparison between different Modes
     - Mode: `r` (Read)
     - Mode: `w` (Write)
     - Mode: `a` (Append)
     - Mode: `r+` (Read-Write Mode)
     - Mode: `w+` (Write-Read Mode)
     - Mode: `a+` (Append-Read Mode)
3. Closing a file

tcg crest
Inventing Harmonious Future

Basics of File Handling in C

fscanf and fprintf

- fscanf and fprintf works almost same as scanf and printf

```c
float average(float a, float b){
    return ((a+b)/2.0);
}
int main(){
    float a, b, avg;
    FILE * inp_file_ptr, * out_file_ptr;//File type pointer
    inp_file_ptr = fopen("input_file.txt","r");//Open to reading
    fscanf(inp_file_ptr, "%f %f", &a, &b);//taking input from file
    fclose(inp_file_ptr);    // closing the input file

    avg = average(a, b);     //Computing average

    out_file_ptr = fopen("output_file.txt","w");
    fprintf(out_file_ptr, "%f",avg); //writing on output file
    fclose(out_file_ptr); //closing the output file

    return 0;
}
```

Reading from a file

Function	Description
`fscanf()`	Use formatted string and variable arguments list to take input from a file. `int fscanf(FILE *ptr, const char *format, ...)`
`fgets()`	Input the whole line from the file. `char *fgets(char *str, int n, FILE *stream)`
`fgetc()`	Reads a single character from the file. `int fgetc(FILE *pointer)`
`fread()`	Reads the specified bytes of data from a binary file. `size_t fread(void *ptr, size_t size, size_t nmemb, FILE *stream)`

Table: Some functions to Read from a file

tcg crest
Inventing Harmonious Future

Writing to a file

Function	Description
`fprintf()`	Similar to `printf()`, this function print output to the file. `int fprintf(FILE *fptr, const char *str, ...);`
`fputs()`	Prints the whole line in the file and a newline at the end. `int fputs(const char *str, FILE *stream)`
`fputc()`	Prints a single character into the file. `int fputc(int char, FILE *pointer)`
`fwrite()`	This function writes the specified amount of bytes to the binary file. `size_t fwrite(const void *ptr, size_t size, size_t nmemb, FILE *stream)`

Table: Some functions to Write from a file

tcg crest
Inventing Harmonious Future

textttfseek in C

The `fseek` function in C is used to move the file pointer to a specific location in a file. It is commonly used for random access in files.

Syntax:

- `int fseek(FILE *stream, long offset, int whence);`

Parameters:

- `stream` - Pointer to the file object.
- `offset` - Number of bytes to offset from `whence`.
- `whence` - Position from where `offset` is added.
  - `SEEK_SET` - Beginning of file.
  - `SEEK_CUR` - Current position of the file pointer.
  - `SEEK_END` - End of file.

tcg crest
Inventing Harmonious Future

Example Code Using fseek()

```c
int main() {
  FILE *fp;
  char c;

  // Open file in read mode
  fp = fopen("example.txt", "r");

  if (fp == NULL) {
    perror("Error opening file");
    return -1;
  }
  // Move the file pointer to the 10th byte from the beginning
  fseek(fp, 10, SEEK_SET);

  // Read and print the character at this position
  c = fgetc(fp);
  printf("Character at position 10: %c\n", c);


  fclose(fp); // Close the file
  return 0;
}
```

File opening modes

• When you open a file, you need to specify the mode in which you want to open it. The following are the different file modes:

Mode	Meaning of Mode	During Inexistence of File
`r`	Reading.	If the file does not exist, `fopen()` returns `NULL`.
`w`	Writing.	If the file exists, its contents are overwritten. If the file does not exist, it will be created.
`a`	Append.	Data is added to the end of the file. If the file does not exist, it will be created.
`r+`	Reading and Writing	If the file does not exist, `fopen()` returns `NULL`.
`w+`	Reading and Writing	If the file exists, its contents are overwritten. If the file does not exist, it will be created.
`a+`	Reading and Appending.	If the file does not exist, it will be created.

Table: File opening modes in C

tcg crest
Inventing Harmonious Future

Comparison between different Modes

Mode	Creates File	Overwrites Existing File	Reading	Writing
r	No	No	Yes	No
w	Yes	Yes	No	Yes
a	Yes	No	No (starts at end)	Yes
r+	No	No	Yes	Yes
w+	Yes	Yes	Yes (after writing)	Yes
a+	Yes	No	Yes (starts at end)	Yes

Table: Comparing the different file modes

We have not discussed opening in binary modes

Mode: r (Read)

- Opens a file for reading.
- The file must exist; otherwise, NULL is returned.

Example

```
1  int main() {//Reads a File and shows content in the terminal
2    FILE *fp;
3    char ch;
4
5    fp = fopen("my_file.txt", "r");
6    if (fp == NULL) {
7      printf("Error opening file.\n");
8      return 1;
9    }
10   while ((ch = fgetc(fp)) != EOF) {
11     printf("%c", ch);
12   }
13   fclose(fp);
14   return 0;
15 }
```

Mode: w (Write)

- Opens a file for writing.
- If the file exists, it is truncated (emptied).
- If the file does not exist, a new file is created.

Example

```c
int main() {
  FILE *fp;

  fp = fopen("new_file.txt", "w");
  if (fp == NULL) {
    printf("Error opening file.\n");
    return 1;
  }

  fprintf(fp, "This is a new file.\n");

  fclose(fp);
  return 0;
}
```

Mode: a (Append)

- Opens a file for appending.
- If the file exists, the data is added to the end of the file.
- If the file does not exist, a new file is created.

Example

```
1  int main() {
2    FILE *fp;
3
4    fp = fopen("my_file.txt", "a");
5    if (fp == NULL) {
6      printf("Error opening file.\n");
7      return 1;
8    }
9
10   fprintf(fp, "This is appended text.\n");
11
12   fclose(fp);
13   return 0;
14 }
```

Mode: r+ (Read-Write Mode)

- Opens a file for both reading and writing.
- The file must exist; otherwise, NULL is returned.

Example

```
1  int main() {
2    FILE *fp;
3    fp = fopen("my_file.txt", "r+");
4    if (fp == NULL) {
5      goto Error;
6    }
7    char ch;
8    while ((ch = fgetc(fp)) != EOF) {    // Read the contents
9      printf("%c", ch); //and print the contents
10   }
11   fseek(fp, 0, SEEK_SET); // Seek to the beginning and write
12   fprintf(fp, "New content at the beginning.\n");
13
14   fclose(fp);
15   return 0;
16 }
```

Mode: w+ (Write-Read Mode)

- Opens a file for both reading and writing.
- If the file exists, it is truncated (emptied).
- If the file does not exist, a new file is created.

Example

```c
int main() {
  FILE *fp;
  fp = fopen("my_file.txt", "w+");
  if (fp == NULL) {
    goto Error;
  }
  fprintf(fp, "This is new content.\n"); // Write some data
  fseek(fp, 0, SEEK_SET);
  char ch;
  while ((ch = fgetc(fp)) != EOF) { //Read the content
    printf("%c", ch);      //Print the content
  }
  fclose(fp);
  return 0;
}
```

Mode: a+ (Append-Read Mode)

- Opens a file for both reading and appending.
- If the file exists, data is added to the end of the file.
- If the file does not exist, a new file is created.

Example

```c
int main() {
  FILE *fp;
  fp = fopen("my_file.txt", "a+");
  if (fp == NULL) {
    goto Error;
  }
  fprintf(fp, "This is appended text.\n"); // Append some data
  fseek(fp, 0, SEEK_SET); // Seek to the beginning
  char ch;
  while ((ch = fgetc(fp)) != EOF) { //Read the content
    printf("%c", ch);      //write the content
  }
  fclose(fp);
  return 0;
}
```

Closing a file

1. The fclose() function is used to close the file
2. After successful file operations, you must always close a file **to remove it from the memory**.
3. Syntax of fclose()
   `fclose(file_pointer);`

Try to open as fewer files as possible at a time; (Max 23)

tcg crest
Inventing Harmonious Future

Thank You

for your attention.

Questions?

tcg crest

Inventing Harmonious Future

Laltu Sardar

laltu {dot} sardar [at]tcgcrest(.)org

https://laltu-sardar.github.io