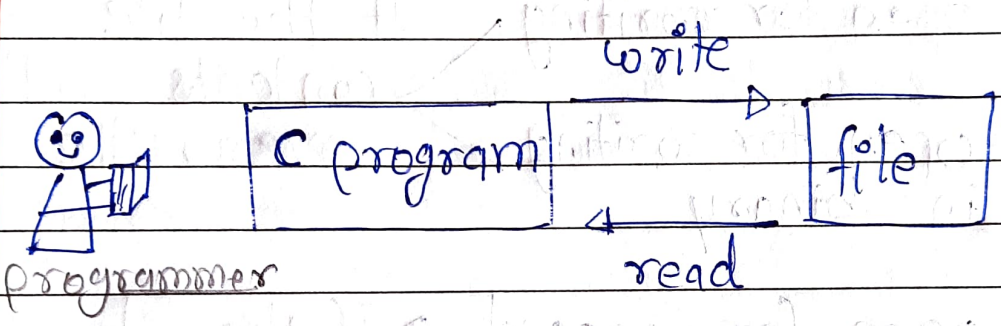
# CHAPTER 10
# File I/O

The Random Access Memory is volatile and its content is lost once the program terminates. In order to persist the data forever we use files.

A file is data stored in a storage device. A C program can talk to the file by reading content from it and writing content to it.




## FILE Pointer

The "FILE" is a structure which needs to be created for opening the file.

A file pointer is a pointer to this structure of the file.

# FILE pointer is needed for communication between the file and the program.

A FILE pointer can be created as follows:

```
FILE *ptr;
ptr = fopen("filename.txt", "mode");
```

DATE:

## File opening modes in C

C offers the programmers to select a mode for opening a files.

following modes are primarily used in C File I/O

"r" → open for reading.
"rb" → open for reading in binary

> If the file does not exist, fopen returns NULL

"w" → open for writing.
"wb" → open for writing in binary

> If the file exists, the contents will be overwritten.

"a" → open for append → If the file does not exists it will be created.

## Types of files

There are two types of files:

1. Text files (.txt, .c)
2. Binary files (.jpg, .dat)

## Reading a file

A file can be opened for reading as follows:

```
FILE *ptr;
ptr = fopen("Abhi.txt", "r");
int num;
```

Let us assume that "Abhi.txt" contains an integer. We can read that integer using:

```
fscanf(ptr, "%d", &num);
// fscanf is file counterpart of scanf.
```

This will read an integer from file in num variable.

Quick Quiz: Modify the program above to check whether the file exists or not before opening the file.

## Closing the files

It is very important to close the file after read or write. This is achieved using fclose as follows:

```
fclose(ptr);
```

This will tell the compiler that we are done working with this file and the associated resources could be freed.

## Writing to a files

We can write to a file in a very similar manner like we read the file.

```
FILE *fptr;
fptr = fopen("Abhi.txt", "w");
int num = 342;
fprintf(fptr, "%d", num);

fclose(fptr);
```

## fgetc() and fputc()

fgetc and fputc are used to read and write a character from to a file.

```
fgetc(ptr)
// used to read a character from file.

fputc('c', ptr);
// used to write character 'c' to the file
```

## EOF : End of file

fgetc returns EOF when all the characters from a file have been read. So we can write a check like below to detect end of file.

DATE :

```
while (1){
    ch = fgetc(ptr);

    if (ch == EOF){
        break;
            }

    //codes
    }
```

// When all the content of a file has been //read, break the loop!

## Chapter 10
## Practice Set

Que1. Writer a program to read three integers from a file.

Que2. Write a program to generate multiplication table of a given number in text format. Make sure that file is readable and wellformatted

Que3. Write a program to read a text file character by character and write its content twice in a separate file.

Que4. Take name and salary of two employees as imput from the user and write them to a text file in the following format.

name1, 3300
name2, 7700

DATE :

Que5. Write a program to modify a file containing an integer to double its value.