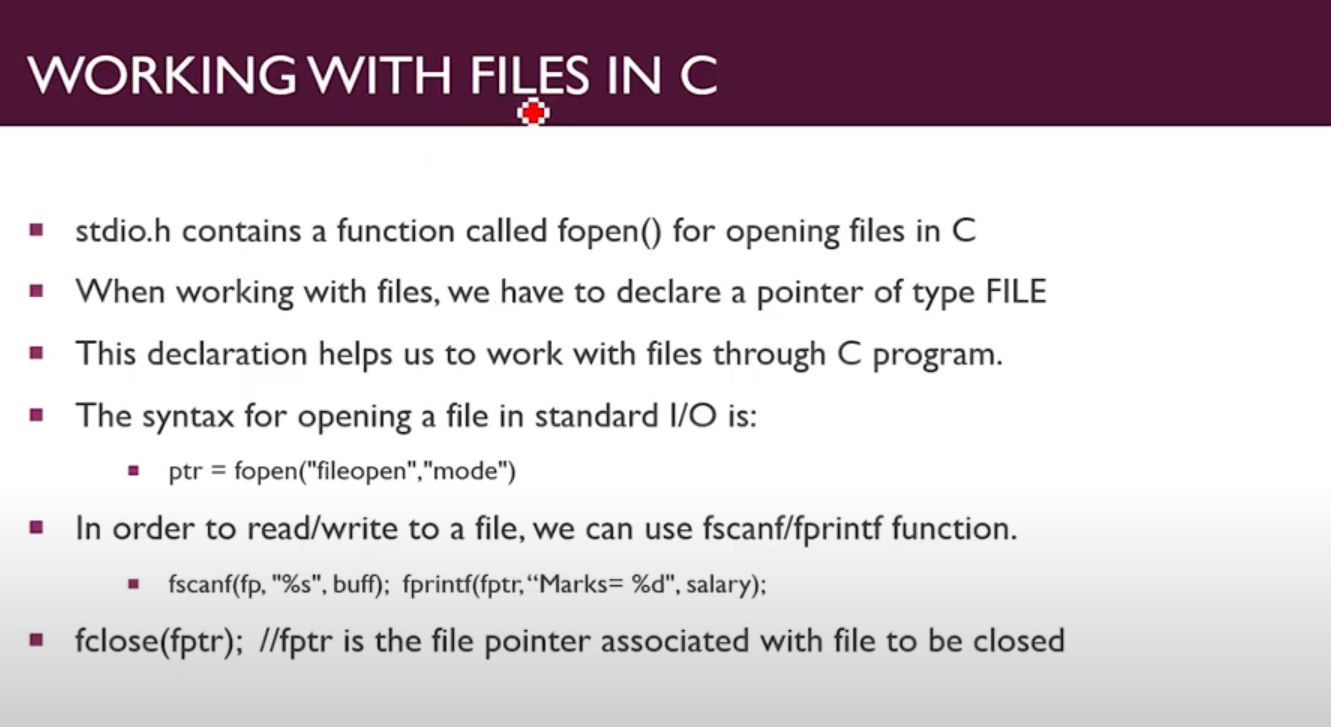
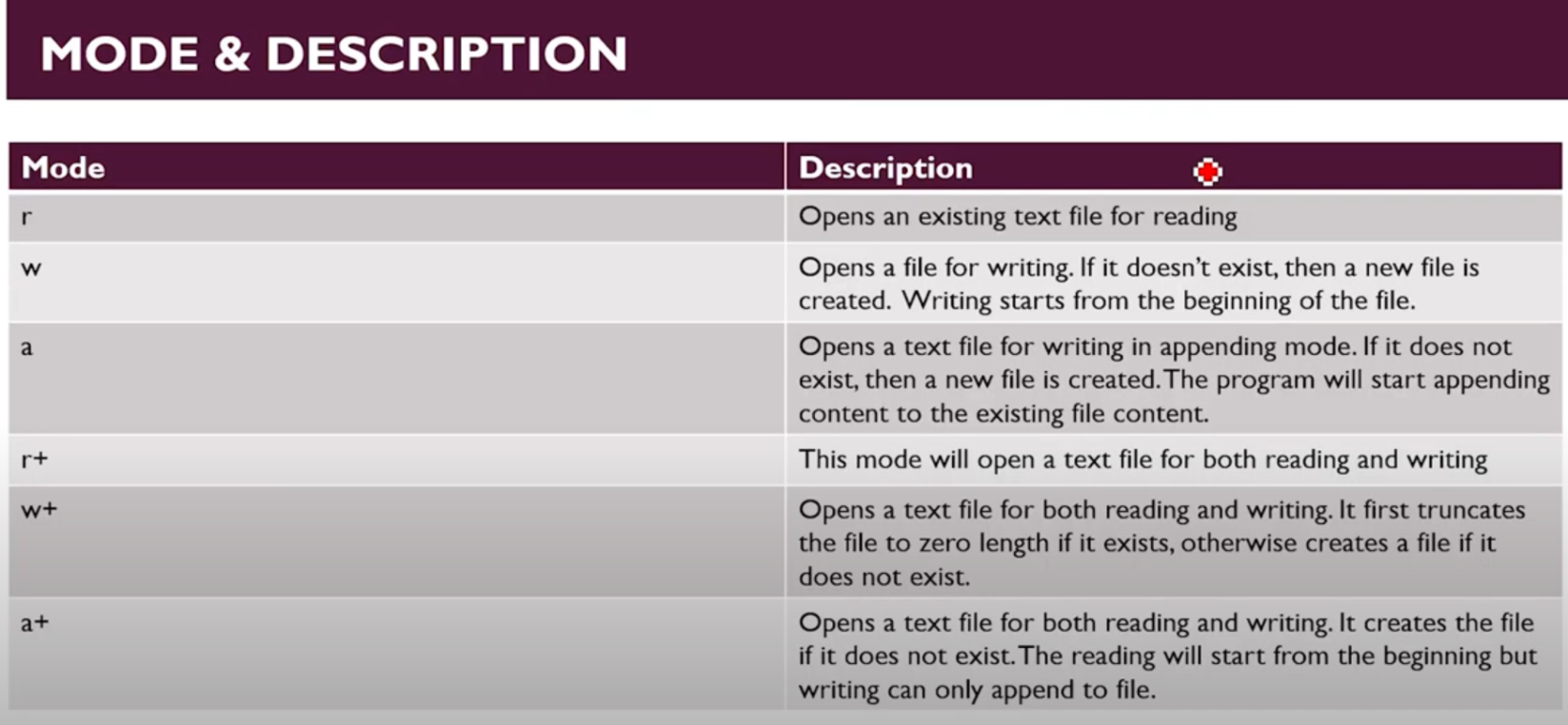
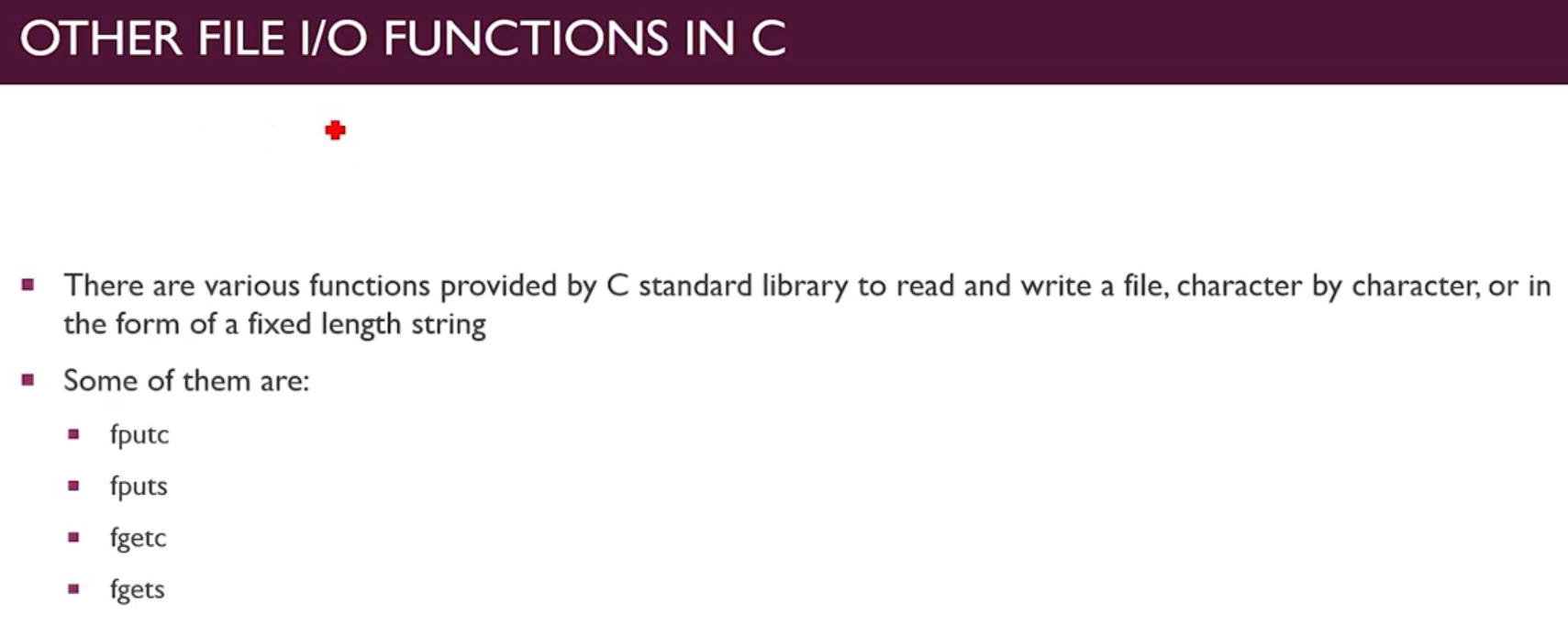
Content 49

File modes, fgets, fputs, fgetc, fputc & more on C file handling: C

We have already covered **file pointer**, opening/closing, reading/writing to a file, and the different modes of opening a file. In this tutorial, we will see a few more built-in functions related to files in C.  By using these functions, we can read or write data from or to the files, character by character, or in the form of a full string.







#### fputc():

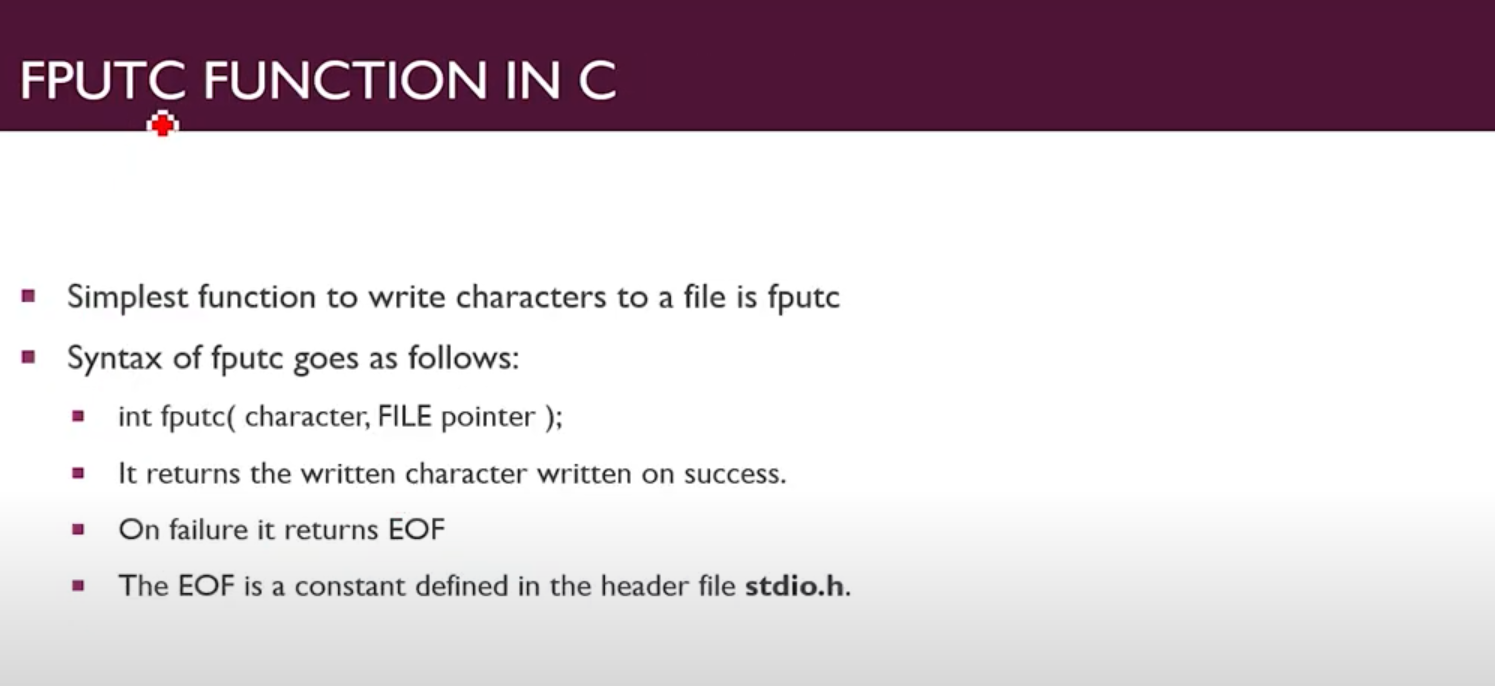
fputc() is used to write**characters** to the file. The C in the name of the function stands for character. The function takes two parameters as input. The first one is the single character that we want to input to the file. The second parameter is the pointer to the file. On successful implementation, it returns the character on to the screen. If it couldn't do so in case of any other issue, it would display an EOF exception. EOF stands for End of File. You will see a lot of this exception while working with files.

**Syntax:**

type fputc(‘character’,file\_pointer);

**Example:**

Int fputc(‘a’,ptr);



#### fputs():

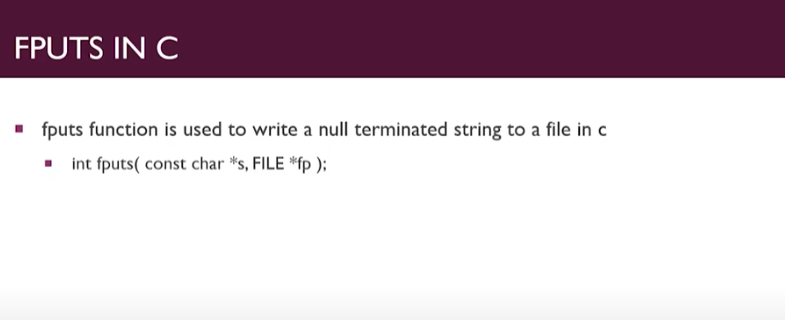
fputs() is used to write a null-terminated string to the file. The S in the name of the function stands for string.it also takes two parameters, the same as fputc(). One is the variable storing the string and the second one is the pointer to the file. A **null-terminated string** is a character string that can be terminated by a null character i.e., \0. You do not have to bother much about null-terminated strings as our computer automatically converts character strings to null-terminated strings.

**Syntax:**

fputs(“string”,file\_pointer);

**Example:**

fputs("code with harry",ptr);



Here Giving about both fputs() and fputc() Together:

**File Before Operation is:**

people can't Judge me.

I am good.

I am Best.

Code: here w mode is used becoz we are writing (editing or modifying).

#include <stdio.h>

int main(int argc, char const \*argv[])

{

     // fgets() and fgetc() Function For String Reading in file

    FILE \*ptr = NULL;

    ptr = fopen("cnt46.txt", "w"); //changing This r with w because I am writing.

    // but first file got Erase because we are opening as w

    char c;

    fputc('O',ptr);

    fputs(“this is me”,ptr);

    fclose(ptr);

    return 0;

}

File After operation: File get all Erase first then perfrom our written operation.

Othis is me

**Here If We Use r+ Mode then information will not erase it would be directly updated.**

**Again Using fputc() and fputs() function:**

My File before.

This file is good.

**Code1:**

#include <stdio.h>

int main(int argc, char const \*argv[])

{

    // fgets() and fgetc() Function For String Reading in file in mode r+

    FILE \*ptr = NULL;

    ptr = fopen("cnt46.txt", "r+"); //changing This r with r+.

    // becoz I it will preserve the information in that file which store before

    fputc('O',ptr);

    fputs("this is me",ptr);

    fclose(ptr);

    return 0;

}

**Now My file becomes:**

Othis is mes good.

It get replaced.

**Now if I talk about w+ it would empty or Erase file as w.**

**Here I replace above program with a+ mode then: Here my files becomes.**

This is meThis file is good.

**These are the difference between r+ ,w+ and a+ ;**

#### fgetc():

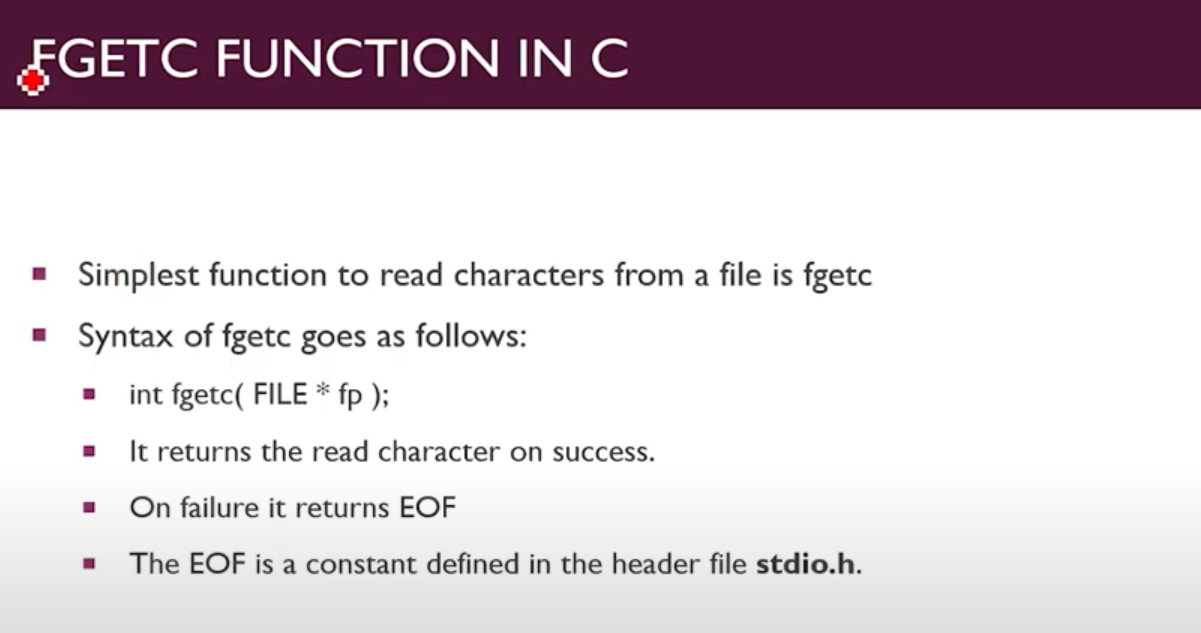
fgetc() works exactly the opposite of fputc(). It reads the character from the file. It reads only **one character at a time**. We can print it as many times as we want to get the next character and so on. Its syntax is straightforward, as we have to send the file pointer as a parameter. We can store the character into another character to display it onto the screen.

**Syntax:**

type = fgetc(file\_pointer);

**Example:**

c = fgetc(ptr);



**File Used in For Operation:**

people can't Judge me.

I am good.

I am Best.

**Code:**

#include <stdio.h>

//fgetc() function for File operations

int main(int argc, char const \*argv[])

{

    FILE \*ptr = NULL;

    ptr = fopen("cnt46.txt", "r");

    char c = getc(ptr);

    printf("The character I read was: %c\n", c);

    c = getc(ptr);

    printf("The 2nd character I read was: %c\n", c);

    fclose(ptr);

    return 0;

}

Output:

The character I read was: p

The 2nd character I read was: e

**fgets():**

fgets() is used to read a string from a file. It takes**three parameters** as input and stores them in a null terminated array. Now talking about the parameters, the first one is the storage array we want our string to store. The third one is the file pointer, and the second one is the count of variables we want to get from the files. For example, we want to take the first four characters from the string, so we will input the second parameter equals to 5. The purpose of doing that is that the null character will hold the fifth place.

Note: Blank space is also considered as a character.

**Syntax:**

int fgets(const char \*s, int n, file\_pointer);

**Example:**

fgets(str, 5, ptr);

**This is my Text file:**

people can't Judge me.

I am good.

I am Best.

**Code:**

#include <stdio.h>

int main(int argc, char const \*argv[])

{

    // fgets() Function For String Reading in file

    FILE \*ptr = NULL;

    ptr = fopen("cnt46.txt", "r");

    char str[10];

    fgets(str,7,ptr);  //here 7 is no of character want to read

    printf("The string is: %s\n", str);

    fclose(ptr);

    return 0;

}

Output:

The string is: people