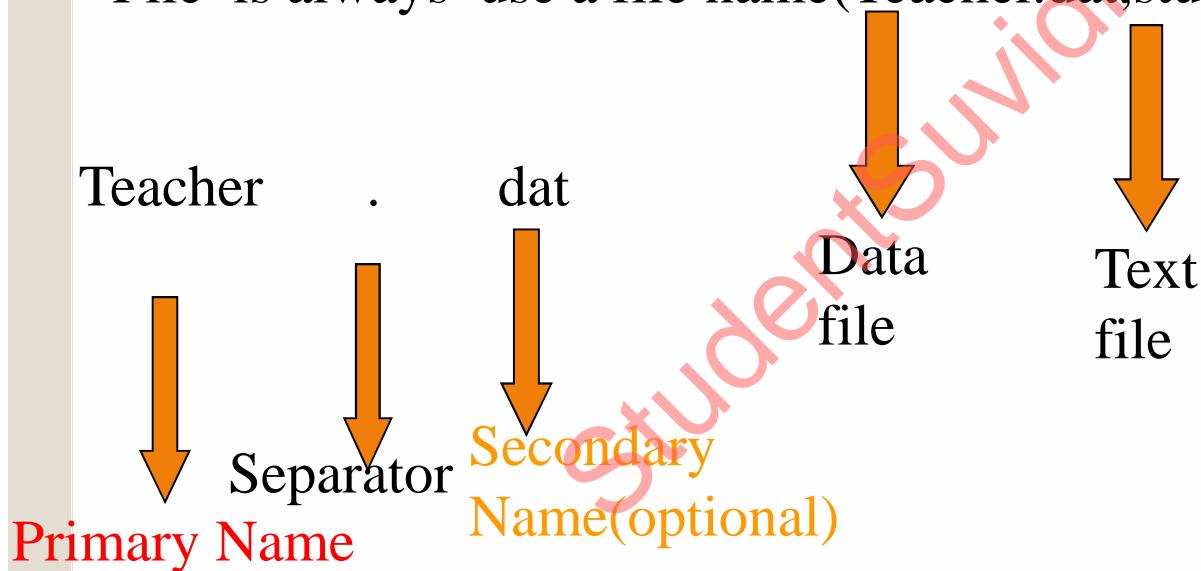


File Handling

INTRODUCTION

- A file contains data and information which is stored permanently in the storage device.
- Large quantity of data is stored and processed, the concept of file is used.
- File is always use a file name(Teacher.dat, student.txt)



- File data type is provided in a stdio.h

Basic File Functions

- ◆ The ANSI file system is composed of several interrelated functions.

Name	Function
<code>fopen()</code>	Opens a file
<code>fclose()</code>	Closes a file
<code>putc()</code>	Writes a character to a file
<code>fputc()</code>	Same as <code>putc()</code>
<code>getc()</code>	Reads a character from a file
<code>fgets()</code>	Same as <code>getc()</code>
<code>fseek()</code>	Seeks to specified byte in a file
<code>fprintf()</code>	Is to a file what <code>printf()</code> is to the console
<code>fscanf()</code>	Is to a file what <code>scanf()</code> is to the console
<code>feof()</code>	Returns true if end-of-file is reached
<code>ferror()</code>	Returns true if an error has occurred
<code>rewind()</code>	Resets the file position locator to the beginning of the file
<code>remove()</code>	Erase a file
<code>fflush()</code>	Flushes a file

TYPE OF FILES

1.Data file(.dat)

2.Text file(.txt)

- Data file contain data and stored in the storage device in the form of record.
- Record is the collection of data related to any person or items.(teacher's name ,address ,subject etc)

Files in C

There are many I/O function in 'c' library.

1. Console I/O Function. (Screen, Keyboard)

- Read from Keyboard and write on the screen.

Console I/O function is divided into two parts:

- Formatted function
 - scanf()
 - printf()
- Unformatted function
 - getch(), getche(), getchar(), putchar(), which is char type.
 - gets(), puts() which is string type.

Formatted console I/O function – input in fixed format and output in the specified form.

Unformatted console I/O function - Deal with single character and string of character.

3.Disk I/O Function.

- Disk I/O function perform operation on entities(column)called file.

3.File I/O Function

- Read and write in the same file

File I/O is always done in the following sequence

- File Naming(.dat/.txt) and open the file.
- Read and write the file.
- Close the file.

4. Port I/O function

- **Function to perform I/O operation on various ports.**

FILE HANDLING FUNCTION

Feof

- Function : Associate a new file with an open stream.
- Include: `#include<stdio.h>`
- Syntax:
- Return value:
- Description:

fprintf

- Function :Write formatted output to a stream
- Include: `#include<stdio.h>`
- Syntax:
- Return value:
- Description:

Continue.....

Fputc

- Function : put a character on a stream.
- Include: #include<stdio.h>
- Syntax:
- Return value:
- Description:

fclose

- Function : close a stream.
- Include: #include<stdio.h>
- Syntax:
- Return value:
- Description:

Continue.....

fscanf

- Function : scan and format an input from a stream.
- Include: #include<stdio.h>
- Syntax:
- Return value:
- Description:

fgetc

- Function :get a character from stream
- Include: #include<stdio.h>
- Syntax:
- Return value:
- Description:

Continue.....

fgetpos

- Function :get the current file pointer.
- Include: #include<stdio.h>
- Syntax:
- Return value:
- Description:

fwrite

- Function : write to a stream.
- Include: #include<stdio.h>
- Syntax:
- Return value:
- Description:

setbuf

- Function :Assign a buffer to a stream.
- Include: #include<stdio.h>
- Syntax:
- Return value:
- Description:

Continue.....

fread

- Function : read data from a stream.
- Include: #include<stdio.h>
- Syntax:
- Return value:
- Description:

feof

- Function : To find end of file on a stream.
- Include: #include<stdio.h>
- Syntax:
- Return value:
- Description:

ftell

- Function : return the current file pointer
- Include: #include<stdio.h>
- Syntax:
- Return value:
- Description:

fflush

- Function :flushes a stream.
- Include: #include<stdio.h>
- Syntax:
- Return value:
- Description:

Continue.....

remove

- Function : To remove a file.
- Include: #include<stdio.h>
- Syntax:
- Return value:
- Description:

rename

- Function : To rename a file.
- Include: #include<stdio.h>
- Syntax:
- Return value:
- Description:

tmpnam

- Function : To create a unique file name.
- Include: #include<stdio.h>
- Syntax:
- Return value:
- Description:

fseek

- Function : To reposition a file pointer on a stream.
- Include: #include<stdio.h>
- Syntax:
- Return value:
- Description:

File open mode

r –read only mode

w –write only mode

a –append mode

r+ - To open existing file for Reading and Writing.

w+ - To open anew file for Reading and Writing.

a+ - To open an existing file for Reading and Append.

File Pointer

- ◆ A file pointer is needed in order to read or write files.
- ◆ A file pointer is a pointer to a structure, which contains information about the file - its name, its current position, whether it is being read or written, and whether errors or end of the file have occurred.
- ◆ The details are not known to the user because the definitions obtained from `stdio.h` include a structure declaration called `FILE`.
- ◆ The only declaration needed for a file pointer is symbolized by :

*FILE *fp*

Opening A File

- ◆ The **fopen()** function opens a stream for use and links a file with that stream.
- ◆ A file pointer associated with that file is then returned by the **fopen()** function.
- ◆ The prototype for the **fopen()** is -

*FILE *fopen(const char *filename, const char *mode);*

FILE *fp;

fp = fopen("xyz", "w");

Closing A File

- ◆ It is very important to close an opened file after its usage. This frees system resources and reduces the risk of overshooting the set limit.
- ◆ Closing a stream also flushes out any associated buffer, an important operation that prevents loss of data when writing to a disk.
- ◆ The `fclose()` function closes a stream that was opened by a call to `fopen()`. It writes any data still remaining in the disk buffer to the file.

*`int fclose (FILE *fp);`*

- ◆ The `fcloseall()` function closes all the open files.

Writing a character

- ◆ Streams can be written to either character by character or as a string
- ◆ The function used for writing characters to a file, that was previously opened using `fopen()`, is `fputc()`.
- ◆ The prototype for this is -

```
int fputc(int ch, FILE *fp);
```

Reading a character

- ◆ The **fgetc()** function is used for reading characters from a file opened in read mode, using **fopen()**.
- ◆ The prototype for **fgetc()** is -

```
int fgetc(FILE *fp);
```

String Input / Output

- ◆ In addition to `fgetc()` and `fputc()`, C supports the related functions `fputs()` and `fgets()`, which write and read character strings to and from disk file.
- ◆ The prototypes for the above functions are -

*int fputs(const char *str, FILE *fp);*

*char *fgets(char *str, int length, FILE *fp);*

Using feof()

- ◆ When a file is opened for binary input, an integer value equal to the EOF (End Of File) may be read.
- ◆ The input routine will indicate an EOF in such a case, even though the end of file has not reached.
- ◆ A function feof() can be used in such cases. The prototype of this function is -

```
int feof(FILE *fp);
```

The `rewind()` function

- ◆ The **`rewind()`** function resets the file position indicator to the beginning of the file.
- ◆ It takes the file pointer to the file, which is to be rewound as its argument.
- ◆ The syntax for **`rewind()`** is :

`rewind(fp);`

- ◆ The prototype for the **`rewind()`** is available in `stdio.h`

Erasing Files

- ◆ The **remove()** function erases the specified file. Its prototype is:

*int remove (char*filename);*

- ◆ It returns **0** if successful else returns a non zero value

fprintf() and fscanf()

- ◆ These functions are similar to printf() and scanf() except that they operate with files.
- ◆ The prototypes of **fprintf()** and **fscanf()** are:

fprintf()

```
int fprintf(FILE *fp, const char *control_string, ...);
```

fscanf()

```
int fscanf(FILE *fp, const char *control_string, ...);
```

The fread() and fwrite() function

- ◆ Referred to as unformatted read and write functions.
- ◆ Used to read and write an entire block of data to and from a file
- ◆ The prototypes for these functions are -

fread()

```
size_t fread(void *buffer, size_t num_bytes, size_t count FILE *fp);
```

fwrite()

```
size_t fwrite(const void *buffer, size_t num_bytes, size_t count FILE *fp);
```


The fread() and fwrite() function - I

- ◆ The **fread()** function returns the number of items read.
- ◆ The **fwrite()** function returns the number of items written.
- ◆ As long as the file is open for binary operations, **fread()** and **fwrite()** can read and write any type of information.
- ◆ One of the most useful applications of **fread()** and **fwrite()** involves reading and writing user-defined data types, especially structures

/* WAP to count character,spaces,tabs and newline in a file*/

#include<stdio.h>

#include<conio.h>

void main()

{ int ch=0,sp=0,line=0,tb=0;

char c;

FILE *fpt;

clrscr();

if((fpt=fopen("sample.dat","r"))==NULL)

printf("cannot open a file");

else

while(1)

{ c=fgetc(fpt);

if(c==EOF)

break;

ch++;

Continue

```
if(c==' ')  
    sp++;  
if(c=='\n')  
    line++;  
if(c=='\t')  
    tb++;  
}  
fclose(fpt);  
printf("No of line%d\n",line);  
printf("No of space%d\n",sp);  
printf("No of Character%d\n",ch);  
printf("No of tab%d",tb);  
getch();  
}
```

COPY A FILE

Example:

`C=fgetc(fpt1),`

`fprintf(fpt1,"%c",c);`

Target file

Source file

StudentSuvidha.com

/*Write a program to copy the contents of one file to another file.*/

```
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
void main()
{
FILE *fpt1,*fpt2;
char ch;
clrscr();
fpt1=fopen("sample.c","r");
fpt2=fopen("hello.c","w");
while(1)
{ ch = getc(fpt1);
if(ch==EOF)
{
break;
}
else
putc(ch,fpt2);
}
printf("File copied succesfully!");
fclose(fpt1);
fclose(fpt2);
}
```

Assignment

- Write some of the function related to file handling.
- Write a program add and delete a record of a student into a file where student is declared as a structure.