What is File ?

Ans. :- A file is a collection of related records. A record is collection of logically related fields.

Example :- student record, Employee record, Department Record, etc.

File handling in C :-

The console oriented input output functions such as scanf and printf always use terminal. This works well as long data is small. But many times we have to handle large volume of data.

File is a permanent storage medium. It can be store on the disk and read whenever necessary. It will contain the data even after program exit. Data structure of a file is defined as FILE in the library std. I/O fun. Definitions.

Various mode of file opening and closing files

Mode	Description
r	Open a text file in read mode. If the file does not exist it returns NULL.
w	Open a text file in write mode. If the file does not exist it returns NULL.
a	Open a text file in append mode. If the file does not exist it returns NULL.
r+	Open a text file in read and write mode. If the file does not exist it returns NULL.
w+	Open a text file in read and write mode. If the file does not exist it returns NULL.
a+	Open a text file in read and write mode. If the file does not exist it returns NULL.
rb	Open a binary file in read mode. If the file does not exist it returns NULL.
wb	Open a binary file in write mode. If the file does not exist it returns NULL.
ab	Open a binary file in append mode. If the file does not exist it returns NULL.
rb+	Open a binary file in read and write mode. If the file does not exist it returns NULL.
wb+	Open a binary file in read and write mode. If the file does not exist it returns NULL.
ab+	Open a binary file in read and write mode. If the file does not exist it returns NULL.

File processing Functions

Function for file handling :-

There are many functions of in c library on file. A list of file functions :-

No.	Function	Description
1.	fopen()	Opens new or existing file.
2.	fprintf()	Write data on file.
3.	fscanf()	Reads data from file.
4	fputc	Writes a character into file.
5.	fgetw()	Reads a character from file.
6.	fclose	Close the file.
7.	fseek	Sets the file pointer to given position.
8.	fputw()	Writes an integer to file.
9.	fgetw()	Reads an integer from file.
10.	ftell()	Returns a current position.
11.	rewind()	Sets the file pointer to the beginning of the file.

See the file function in details refer for the textbook.

File opening error :-

Sometimes when we open a file using fopen, file May not be opened because of various reasons.

1. Read mode may occurs because the file being opened may not be present on disk at all.
2. Write mode may occurs because disc space may be insufficient to open a new file or disc may be write protected and so on.

If the file opening fails due to any reason, the fopen function returns the value NULL which is defined in stdio.h.

remove(filename) Function :-

removes (deletes) a file with specified file name. if your file is opened be sure to close it before removing.

rename(old name, new name) Function :-

rename() function is used to rename a file. Rename changes the name of the file from old name to new name. if driver is specified a new name, it must be same as given in old name.

fEOF() Function :-

the feop() function can be used to test for an end of file condition. EOF stand for " End of File". It takes a file pointer as its only argument and returns a nonzero integer value if alm data from the specified file has been read, and returns zero otherwise.

Syntax :- feof(fp);

Where fp is a file pointer.

Example program to use the all these functions :-

```
#include<stdio.h>

int main()

{

int ch;

FILE *fp;

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

printf("enter characters for the file and press ^z at the end\n");

while ((ch = getch()) != EOF)

{

fputc(ch,fp);

printf("%c",ch);

}

fclose(fp);

fp = fopen(charfile,"r");

fseek(fp,2,0);

printf("Position of character :%d\t character is :%c",ftell(fp),fgetc(fp));

fseek(fp,-6,2);

printf("position of character:%d\t character is :%c",ftell(fp),fgetc(fp));

printf("current position %ld",ftell(p));

rewind(fp);

}
```

The above program uses the random access function to determine the character at various positions by going forward, rewinding etc. follow the program carefully.

Command line arguments :-

Command line argument is a parameter supplied to the program when it is invoked. It is an important concept in c programming. It is mostly used when you need to control your program from outside. These are passed to a c

program through two arguments to the main() function. The parameter are called argc and argv. These parameters are optional and are not used when no command – line arguments are being used.

When command line arguments are to be passed to main(), we have to declare the main() function as follows :-

Syntax :-

main(argc, rgv)

int argc;

char *argv[];

or

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

here argc counts the number of arguments on the command line and argv[] is a pointer array which holds pointer of type char which points to the arguments passed to the program. If argc is greater than zero, the array elements from argv[0] to argv[argc-1] will contain pointers to strings.

Remember that argv[0] holds the name of the program and argv[1] points to the first command line argument and argv[argc-1] gives the last argument. If no argument is supplied, argc will be 1.

e.g. c:>cmdprog Computer Department

in this case,

argv[0] = cmdprog // name of program file.

argv[1] = Computer // first argument.

argv[2] = Department // second argument.

Example for command line argument :-

```
#include<stdio.h>

int main(int argc, char *argv[]);

int cnt;

printf("program Name is :%s",argv[0]);

if (argc == 1){

printf("no extra command line arguments passed other than program name\n");

}

if (argc >=v2){

printf("Number of arguments passed : %d", argc);

printf("----Following are the command line arguments passed-----\n);

for(cnt=0;cnt<=argc;cnt++){

printf("argv[%d] :%s\n",cnt, argv[cnt]);

}

return 0;

}
```