Deepak Singla, PIET

FILE HANDLING IN C

→ A file is a collection of data stored on a secondary storage device like hard disk.

→ file handling is storing the data in a file using program.

→ In C language the program store result & other data of program using file handling.

Type of file

1) ASCII text file :- A text file is a stream of characters that can be sequentially processed by a computer in forward direction.

→ on text file one kind of operation (reading, writing or appending) is possible at a time.

→ In text file, each line of data ends with a newline character known as end-of-file (EOF).

2) Binary file :- A binary file may contain any

type of data that can be encoded in binary form for computer storage & processing.

→ In binary file, byte & a character are equivalent.

→ Unlike a text file, the binary file can be either processed sequentially or randomly depending upon the need of application.

OPERATION PERFORM ON FILE

1) Declare a file pointer variable
2) open the file
3) process the file. by
   3.1) • Reading data
   3.2) • writing data
4) closing file.

1) Declare a file pointer :- There can be number of files on Hard disk. so in order to access a particular file, we must specify the name of file used.

→ This is done by using the file pointer variable

that points to a particular file.

→ The pointer variable is of type FILE, whose definition is define in <stdio.h> header file.

→ Syntax:- FILE * file_pointer_name;

Example:- FILE *fp;

2. Open the file :- A file must be opened before data can be read from it or written to it.

→ Inorder to open a file fopen() function is used.

→ Syntax of fopen() :-

fp = fopen("filename", "mode of operation");

→ "filename" Specify the name of the file that is to be opened like "data.txt".

→ "mode of operation" Specify the mode in which the file is to be opened.

→ There are various mode of operation like :-

a) "r" → open a text file for reading. if file does not exist, then an error will generate.

b) "w" → open a text file for writing. if file does not exist, then it created. & if file already exist then its content would be deleted.

c) "a" → Append to a text file. if file doesnot exist, then it is created.

d) "r+" → open a text file for both reading & writing. "r+" means file must be read before write to it. & file must exist.

e) "w+" → open a text file for both reading & writing The file created if not exist & it will be truncated if file exist

f) "a+" → open a text file for both reading & writing the content will be added at the end of file content.

Example:-

```
FILE *fp;
fp = fopen("student.txt", "r");
if (fp == NULL)
{ printf("file could not be opened");
  exit(1);
}
```

3.1 Read Data from file :-

→ C provide following function to read data from file.

3.1.1 • fscanf()
3.1.2 • fgets()
3.1.3 • fgetc()
3.1.4 • fread()

3.1.1 fscanf() :- It is used to read formatted data from the file.

Syntax :- fscanf(filepointer, "format Specifier", "variable name");

Example :-

```
main()
{
int num;
FILE *fp;
fp = fopen("student.txt", "r");
if(fp==NULL)
{
printf("Error");
exit(1);
}
fscanf(fp, "%d", &num);
printf("%d", num);
fclose(fp);
}
```

8.1.2 fgets():- fgets() stand for "file get string".

→ The fgets() function is used to get a string from a stream.

→ Syntax:- fgets(char *str, int n, FILE *fp);

→ fgets() function read atmost one less than the number of character specified by size (n) from the given file & store them on string str.

→ fgets() terminate as it encounter with newline character, EOF or error.

→ When all the characters read without any error then a '\0' character is append at the end of string.

Example:-

```
main()
{
FILE *fp;
char str[60];
fp= fopen("file.txt", "r");
if (fp==NULL)
{
printf ("error");
exit(1);
}
if (fgets(str, 59, fp)!=NULL)
  {  puts(str);
  }
fclose(fp);
}
```

3.3.3 fgetc():- This function return the next character from file till Eof is reached.

Syntax:- fgetc(FILE *fp);

→ fgetc() reads a single character from the current position of file & after reading character, the function increments the associated file pointer to point to next character.

Example:-

```
main()
{
char ch;
FILE *fp;
fp = fopen("text.txt", "r");
if(fp==NULL)
{
printf("error");
exit(1);
}
ch = fgetc(fp);
while(!feof(fp))
{
printf("%c", ch);
ch= fgetc(fp);
}
fclose(fp);
}
```

3.4 fread() :- This function is used to read data from a file.

Syntax :- fread (*str, size, num, FILE *fp);

→ fread() reads "num" number of objects & place them into the array pointed by str.

Example :-

```
main()
{
int a[10];i;
FILE *fp;
fp= fopen("text.txt", "r");
if(fp==NULL)
{
Printf("Error");
exit(1);
}
fread(a, sizeof(int), 10, fp);
for(i=0; i<10 ;i++)
{
printf("%d", a[i]);
}
fclose(fp);
}
```

3.2 Writing data to file:

→ C provide the following set of functions to write data on file.

3.2.1) fprintf()
3.2.2) fputs()
3.2.3) fputc()
3.2.4) fwrite()

3.2.1 fprintf(): This is used to write formatted output to file.

Syntax: fprintf("FILE *fp, "Accessspecifier", variable_name);

Example:

```
main()
{
int num;
FILE *fp;
fp = fopen("student.txt", "w");
if (fp==NULL)
{
printf("Error");
exit(1);
} scanf("%d", &num);
fprintf(fp, "%d", num);
fclose(fp);
}
```

3.2.2 fputs() :- It is just opposite to fgets() function.

→ fputs() is used to write a line to a file.

→ syntax:- fputs(char *str, FILE *fp);

→ fputs() write the string pointed by str to the file

→ on success fputs() returns 0 or produce EOF in case of error.

Example:

```
main()
{
char str[10];
FILE *fp;
fp= fopen("text.txt", "w");
if (fp==NULL)
{
printf("Error");
exit(1);
}
printf("Enter a string");
gets(str);
fputs(str, fp);
fclose(fp);
}
```

3.2.3). fputc(); It is just opposite of fgetc() function.

→ fputc() is used to write character by character in file.

Syntax:- fputc(char *str, FILE *fp);

Example:-

```
main()
{
FILE *fp; int i;
char str[10] = {"Hello"};
fp = fopen("text.txt", "w");
if(fp==NULL)
 {
 printf("Error");
 exit(1);
 }
for(i=0; i< strlen(str); i++)
 {
 printf(str[i], fp);
 }
fclose(fp);
}
```

3.2.4 :→ fwrite():- It is used to write the data to a file.

Syntax:- fwrite(char *str, size, num, FILE *fp);

Example:-

```
main()
{
FILE *fp;
int a[10], i;
fp = fopen("text.txt", "w");
if(fp==NULL)
{
printf("error");
exit(1);
}
printf("Enter the element of array");
for(i=0; i<10; i++)
{
scanf("%d", &a[i]);
}
fprintf(a, sizeof(int), 10, fp);
for(i=0; i<10; i++)
fclose(fb);
}
```

4 Closing a file:-

→ fclose(fp) function is used to close the file..

→ If user not close file by writing fclose() then system will automatically close the file.

FUNCTION TO SELECT DATA RANDOMLY

→ There are some function that are used to randomly access a record stored in binary file.

→ The function are :-

1) fseek()
2) ftell()
3) rewind()

! fseek() :- It is used to reposition a cursor in file.

Syntax :- fseek (FILE *fp , offset , int origin) ;

Parameters :-
- *fp → This is pointer variable of type FILE.
- offset → This is number of byte to move cursor from current position.
- Origin → This is the position from where offset is added. It can be any of following :- 1) SEEK_SET (0)
  2) SEEK_CUR (1)
  3) SEEK_END (2)

→ This function return '0' if successfull, else it return non-zero value.

Example:-

```
main()
{
FILE *fp; ch str[20]="Hello";
fp= fopen("file.txt", "w");
fputs(fp);
fseek(fp, 3, SEEK_SET);
```

Both equal to write

```
fseek(fp, 3, 0);
fputs(fp);
fclose(fp);
};
```

2) ftell():- This function is used to know the current position of file pointer.

→ It is a position where the next input or output operation will be performed.

→ Syntax:- ftell(FILE *fp);

→ on success ftell() returns the current file position in bytes

→ on failure or error ftell() returns -1.

Example:-

```
main()
{
FILE *fp;
int len;
fp = fopen("one.txt", "r");
if (fp == NULL)
{
printf("error");
return(-1);
}
fseek(fp, 0, SEEK_END);
len = ftell(fp);
fclose(fp);
printf("%d", len);
}
```

3 rewind() :- This is used to adjust the position of file pointer so that the next input output operation will taken place at the Beginning of the file.

→ It is defined in <stdio.h> header file.

→ Syntax :- rewind(*fp);

→ rewind() is equivalent to fseek() with following Syntax :- fseek(fp, 0, SEEK_SET(0));

Example:-

```
main()
{
FILE *fp; char ch[20] = {"Hello"};
fp= fopen("file.txt", "w+");
fputs(fp);
fseek(fp, 3, SEEK_SET);
fputs(fp); fclose(fp);
fp= fopen("file.txt", "r");
        while(!EOF(fp))
        {
    ch = fgetc(fb)
    Printf("%c", ch);
        }
    rewind(fp);
    while(!EOF(fp))
      {
    ch= fgetc(fp);
    printf("%c", ch);
      }
}
```