File Handling in C

What is a File?

- A file is a collection of related data that a computer treats as a single unit.
- Computers store files to secondary storage so that the contents of files remain intact when a computer turns off.
- When a computer reads a file, it copies the file from the storage device to memory; when it writes to a file, it transfers data from memory to the storage device.
- C uses a structure called FILE (defined in stdio.h) to store the attributes of a file.

Steps in Processing a File

1. Create the stream via a pointer variable using the FILE structure: FILE *p;
2. Open the file, associating the stream name with the file name.
3. Read or write the data.
4. Close the file.

Five major operations can be performed on file are:

1. Creation of a new file.
2. Opening an existing file.
3. Reading data from a file.
4. Writing data in a file.
5. Closing a file

To handling files in C, file *input/output* functions available in the *stdio* library are:

Function	Uses/Purpose
fopen	Opens a file.
fclose	Closes a file.
getc	Reads a character from a file
putc	Writes a character to a file

getw	Read integer
putw	Write an integer
fprintf	Prints formatted output to a file
fscanf	Reads formatted input from a file
fgets	Read string of characters from a file
fputs	Write string of characters to file
feof	Detects end-of-file marker in a file

The basic format of fopen is:

Syntax:

```
FILE *fopen( const char * filePath, const char * mode );
```

Parameters

- filePath: The first argument is a pointer to a string containing the name of the file to be opened.
- mode: The second argument is an access mode.

C *fopen()* access mode can be one of the following values:

Mode	Description
r	Opens an existing text file.
w	Opens a text file for writing if the file doesn't exist then a new file is created.
a	Opens a text file for appending(writing at the end of existing file) and create the file if it does not exist.

r+ Opens a text file for reading and writing.

w+ Open for reading and writing and create the file if it does not exist. If the file exists then make it blank.

a+ Open for reading and appending and create the file if it does not exist. The reading will start from the beginning writing can only be appended.

Return Value

C fopen function returns *NULL* in case of a failure and returns a *FILE stream pointer* on success.

Example:

```
#include<stdio.h>


int main()

{

   FILE *fp;

   fp = fopen("fileName.txt","w");

   return 0;

}
```

- The above example will create a file called *fileName.txt*.
- The *w* means that the file is being opened for writing, and if the file does not exist then the new file will be created.

The basic format of fclose is:

Syntax:

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

Return Value

C fclose returns *EOF* in case of failure and returns *0* on success.
Example:

```
#include<stdio.h>


int main()

{

   FILE *fp;

   fp = fopen("fileName.txt","w");

   fprintf(fp, "%s", "Sample Texts");

   fclose(fp);

   return 0;

}
```

- The above example will create a file called *fileName.txt*.
- The *w* means that the file is being opened for writing, and if the file does not exist then the new file will be created.
- The *fprintf* function writes *Sample Texts* text to the file.
- The *fclose* function closes the file and releases the memory stream.

getc() function is C library function, and it's used to read a character from a file that has been opened in read mode by ***fopen()*** function.

```
int getc( FILE * stream );
```

Return Value

- getc() function returns next requested object from the stream on success.
- Character values are returned as an unsigned char cast to an int or EOF on end of file or error.
- The function feof() and ferror() to distinguish between end-of-file and error must be used.

Example:

```
#include<stdio.h>


int main()

{

  FILE *fp = fopen("fileName.txt", "r");

  int ch = getc(fp);

  while (ch != EOF)

  {

    /* To display the contents of the file on the screen */    putchar(ch);


    ch = getc(fp);

  }


  if (feof(fp))

     printf("\n Reached the end of file.");

  else

     printf("\n Something gone wrong.");

  fclose(fp);


  getchar();

  return 0;

}
```

putc() **function is C library function, and it's used to write a character to the file. This function is used for writing a single character in a stream along with that it moves forward the indicator's position.**

```
int putc( int c, FILE * stream );
```

Example:

```
int main (void)

{

 FILE * fileName;

 char ch;

 fileName = fopen("anything.txt","wt");

 for (ch = 'D' ; ch <= 'S' ; ch++) {

  putc (ch , fileName);

  }

 fclose (fileName);

 return 0;

}
```

C getw function is used to read an integer from a file that has been opened in read mode. It is a file handling function, which is used for reading integer values.

```
int getw( FILE * stream );
```

C putw function is used to write an integer to the file.

Syntax:

```
int putw( int c, FILE * stream );
```

Example:

```
int main (void)

{

  FILE *fileName;
```

```
  int i=2, j=3, k=4, n;

  fileName = fopen ("anything.c","w");

  putw(i, fileName);
```

```
  putw(j, fileName);

  putw(k, fileName);

  fclose(fileName);



  fileName = fopen ("test.c","r");

  while(getw(fileName)! = EOF)

  {

    n= getw(fileName);

    printf("Value is %d \t: ", n);

  }

  fclose(fp);

  return 0;

}
```

C fprintf function pass arguments according to the specified format to the file indicated by the stream. This function is implemented in file related programs for writing formatted data in any file.

Syntax:

```
int fprintf(FILE *stream, const char *format, ...)
```

Example:

```
int main (void)
```

```
{

  FILE *fileName;

  fileName = fopen("anything.txt","r");

  fprintf(fileName, "%s %s %d", "Welcome", "to",  2018);

  fclose(fileName);

  return(0);

}
```

C fscanf function reads formatted input from a file. This function is implemented in file related programs for reading formatted data from any file that is specified in the program.

Syntax:

```
int fscanf(FILE *stream, const char *format, ...)
```

Its return the number of variables that are assigned values, or EOF if no assignments could be made.

Example:

```
int main()

{

   char str1[10], str2[10];

   int yr;

   FILE* fileName;

   fileName = fopen("anything.txt", "w+");

   fputs("Welcome to", fileName);

   rewind(fileName);

   fscanf(fileName, "%s %s %d", str1, str2, &yr);

   printf("---------------------------------------------- \n");

   printf("1st word %s \t", str1);
```

```
    printf("2nd word  %s \t", str2);

    printf("Year-Name  %d \t", yr);

    fclose(fileName);

    return (0);

}
```

C fgets function is implemented in file related programs for reading strings from any particular file. It gets the strings 1 line each time.

Syntax:

```
char *fgets(char *str, int n, FILE *stream)
```

Example:

```
void main(void)

{

    FILE* fileName;

    char ch[100];

    fileName = fopen("anything.txt", "r");

    printf("%s", fgets(ch, 50, fileName));

    fclose(fileName);

}
```

- On success, the function returns the same str parameter
- C fgets function returns a NULL pointer in case of a failure.

C fputs function is implemented in file related programs for writing string to any particular file.

- Syntax:

```
int fputs(const char *str, FILE *stream)
```

- Example:

```
#include<stdio.h>
```

```
int main()
{
FILE *fp;
fp = fopen("fileName.txt","w");

fputs("This is a sample text file.", fp);
fputs("This file contains some sample text data.", fp);

fclose(fp);
return 0;
}
```

- In this function returns non-negative value, otherwise returns EOF on error.

C feof function is used to determine if the end of the file (stream), specified has been reached or not. This function keeps on searching the end of file (eof) in your file program.

Syntax:

```
int feof(FILE *stream)
```

Here is a program showing the use of feof().

Example:

```
#include<stdio.h>

int main()

{

FILE *filee = NULL;

char buf[50];

filee = fopen("infor.txt","r");

if(filee)

  {

     while(!feof(filee))

     {

        fgets(buf, sizeof(buf), filee);

        puts(buf);
```

```
        }

        fclose(filee);

    }

    return 0;

}
```

Output:

F:\example\c-feof.exe

```
This is a sample text file.

This file contains some sample text data.

--------------------------------
Process exited after 0.9847 seconds with return value 0
Press any key to continue . . .
```

C feof function returns true in case end of file is reached, otherwise it's return false.

Explanation:

1. It first tries to open a text file infor.txt as read-only mode.
2. Then as the file gets opened successfully to read, it initiates the while loop.
3. The iteration continues until all the statement/lines of your text file get to read as well as displayed.
   Lastly, you have to close the file.

Exercise:

1. **Write a Program to get/find total number of lines in a file in C.**
2. **Write a Program to copy the content of one file to another.**
3. **Write a Program to count the vowels, constant, spaces, tabs, special symbols and number of lines in a file.**