Programming with C - FILES

UNIVERSITY
OF PETROLEUM
& ENERGY STUDIES

Files

- Storage of data in variables and arrays is temporary,such data is lost when a program terminates.

- Files are used for permanent retention of large amounts of data.

- Computers store files on secondary devices,especially disk storage devices.

Files

Data Hierarchy:

- *Fields* are composed of characters.
- *Record* is composed of several fields.
- A *file* is a group of related Records.
- A group of related files is called a *database*.
- A collection of programs designed to create and manage database is called *database management system.(DBMS).*

Text File Functions

fopen() → Opens a text file

fclose() → Closes a text file

feof() → Detects end-of-file marker in a file

fprintf() → Prints formatted output to a file

fputs() → Prints a string to a file

fgets() → Reads a string from a file

fputc() → Prints a character to a file

fgetc() → Reads a character from a file

Opening a file

fopen()

- It opens a file for a specified mode(Read/Write). It then returns a file pointer that you use to access the file.

- If the file cannot be opened for any reason then the value NULL will be returned.

Opening a file

The syntax is,

FILE **filepointer;*

***filepointer*=fopen(*“Filename”, “mode”);**

- If the file cannot be opened for any reason then the value NULL will be returned.
- Where *‘filename’* is a pointer to a string of characters that make up a valid file name and may include a path specifications.

- The string pointed to by *‘mode’* determines how the file will be opened.

For example, FILE *fp;

fp=fopen(“test”,”w”);

Opening a file

The legal values for modes:

Mode	Meaning
r	Open a text file for Reading, sets pointer to beginning
w	Create a text file for Writing, sets pointer to beginning
a	Append to a text File, sets pointer to end of the file
r+	Open a text file for read/write, file does not exist, it will not create
w+	Create a text file for read/write, sets pointer to beginning
a+	Open a file in read/append mode
rb	Open a binary file in read mode, sets pointer to beginning
wb	Open a binary file in write mode, sets pointer to beginning
ab	Open a binary file in write mode, sets pointer to end of the file
r+b	Open a binary file in read/write mode, file does not exist, it will not create

Closing a file

fclose():

- This command can be used to disconnect a file pointer from a file.
- This is usually done so that the pointer can be used to access a different file.
- The Syntax is, **fclose**(file_pointer);

For Example :-

```
fclose(fp);
```

Ex

```
main()
{
    FILE *fp;
    char c;
    fp = fopen(“prog.c”, “r”);
    c = getc(fp);
    while (c != EOF)
    {
            putchar(c);
            c = getc(fp);
    }
    fclose(fp);
}
```

This program will read the data from file.

EOF,End of File Marker

- EOF is a character which indicates the end of a file

- It is returned by read commands of the getc and scanf families when they try to read beyond the end of a file. Normally -1 is used for EOF()

For Example :-

```
ch=getc(fp);
while(ch !=EOF)
{
    putchar(ch);
  ch=getc(fp);
 }
```

Writing the contents into the file

- For Example, suppose you want to open a file and write the numbers 1 to 10 in it.

Program:

```
main()
{   FILE *f;
    int x,MAX=10;
    f=fopen("outfile","w");
    if (!f) return 1; //abnorma
   for(x=1;x<=MAX;x++)
     fprintf(f,"%d\n",x);
     fclose(f);
}
```

Output of the file "outfile" is

1
2
3
4
5
6
7
8
9
10

Reading the contents from the file

The following code demonstrates the process of reading a file and dumping its contents to the screen.

Program:

```
main()
{    FILE *f;
     char s1[100];
     f=fopen("infile","r");
     if (!f)  return 1;
      while (fgets(s1,100,f) != NULL)
      printf("%s",s1);
      fclose(f);
}
```

Output on the Screen:

Welcome To Malaysia

Use of getchar() & putc()

Program to read a character from the keyboard and write into the file.

```
main()
{   FILE *fp;
    char ch;
    fp=fopen("push1.dat","w");
    if (fp == NULL)
    {
      printf("\nCannot open a file\n");
      exit(1);
     }
```

Use of getchar() & putc()

```
else
 printf("\nEnter the characters until 'q' exists……\n");
 do
 {
   ch=getchar();
   putc(ch,fp);
  }while(ch !='q');
  fclose(fp);
}
```

Use of getchar() & putc()

Output:

Enter the characters until ‘q’ exists……
Hello good morning q

The push1.dat file contains,
Hello good morning q

Reading the file contents and displays on the screen

```
main()
{
  FILE *fp;
  char ch;
  fp=fopen(“push1.dat”,”r”);
  if (fp == NULL)
  { printf(“Cannot open a file”);
    exit(1);
  }
```

Reading the file contents and displays on the screen

```
ch=getc(fp);
while(ch!= EOF)
{
  putchar(ch);
  ch=getc(fp);
}
fclose(fp);
}
```

Note:if the file push1.dat is not available means,"Cannot open a file" will be executed.

Copying a file

```
main()
{
   FILE *in,*out;
   char ch;
   in=fopen("source.dat","r");
   out=fopen("target.dat","w");
   if (in == NULL)
   { printf("\nCannot Open the Source file");
     exit(1);
   }
```

Copying a file

```
if(out == NULL)
{ printf("\nCannot open Target File\n");
  exit(1); }
while(!feof(in))
{   ch=getc(in);
    if (!feof(in))
       putc(ch,out);
}
fclose(in);fclose(out);
}
```

Use of fgets() & fputs()

Functions *fgets()* and *fputs()* which read and write character strings from and to a disk file.

Example:

```
main()
{  FILE *fp;
   char str[50];
   fp=fopen("test.dat","w");
   if (fp == NULL)
   { printf("\nCannot Open a File \n");
     exit(1);  }
```

Use of fgets() & fputs()

```
do
{
 printf("Enter a String(newline to quit)\n");
 gets(str);
 strcat(str,"\n");
 fputs(str,fp);
 }while(*str !='\n');
}
```

SUMMARY

- A **field** is a group of characters that conveys meaning.
- A **record** is a group of related fields.
- The most popular type of organization for records in a file is called a sequential access file in which records are accessed consecutively until the desired data are located.

SUMMARY

- A group of related files is sometimes called a **database**.
- C views each file simply as a sequential stream of bytes.
- Function **fgetc** reads a character from a specified files.
- Function **fputc** writes a character to a specified file.

SUMMARY

- Function **fgets** reads a line from a specified file.
- Function **fputs** writes a line to a specified file.
- **FILE** is a structure type defined in the **stdio.h** header file.The programmer need not know the specifics of this structure to use files.As a file is opened, a pointer to the file's **FILE** structure is returned.

SUMMARY

- Function **fopen** takes two arguments—a file name and a file open mode– and opens the file.
- Function **feof** determines whether the end-of-file indicator for a file has been set.
- Function **fprintf** receives as an argument a pointer to the file to which the data will be written.

SUMMARY

- Function **fscanf** receives as an argument a pointer to the file from which the data will be read.
- Function **fclose** closes the file pointed to by its argument.
- Function **fwrite** writes a block of data to a file and **fread** reads a block of data from a file.

Bit Fields

For example,

```
struct student
{
    unsigned sex :1;
    unsigned age :7;
    unsigned m_status:1;
}result;
```

Summary

- **Structures** are collections of related variables,sometimes referred to as aggregates,under one name.
- Structures cab contain variables of different data types.
- The keyword **struct** begins every structure definition.Within the braces of the structure definition are the structure member declarations.

Summary

- Structures and individual members of structures are passed to functions call by value.
- To pass a structure call by reference, pass the address of the structure variable.
- An array of structures is automatically passed call by reference.

Summary

- A **union** is a derived data type whose members share the same storage space.The members can be any type.
- A union is declared with the ***union*** keyword in the same format as a structure.
- Bit fields reduce storage use by storing data in the minimum number of bits required.
- Bit field members must be declared as **int** or **unsigned.**

End