File Management in C
(Topic 7: Files GTU Syllabus – 2009)

Chaudhari Technical Institute MCA (506)
Gandhinagar

Created By: Vinod Pillai.

vinodthebest@gmail.com

http://vinodthebest.wordpress.com

File Management

- We have been using the functions such as printf and scanf to read and write data. This works fine as long as data is small.
- But it has two major problems:
  - It becomes cumbersome and time consuming to handle large volume of data through terminals.
  - The entire data is lost when either the program is terminated or the computer is turned off.

File Management

- So it is necessary to have more flexible approach where data can be stored on the disks and read whenever necessary, without destroying data.
- This concept is called files.
- Files: A file is a place on the disk where a group of related data is stored.
- FILE is a structure declared in stdio.h . We have to use file pointer, a pointer variable that points to a structure FILE.

C File Management

- C supports a number o functions that have the ability to perform basic file operations:
  - Creating a file
  - Opening a file
  - Reading data from a file
  - Writing data to a file
  - Closing a file
  - Renaming a file
  - Deleting a file

Using Files in C

Declaration of File Pointer

```
FILE *fp;
FILE *fp1;
```

Opening a File

```
fp=fopen("one.txt","r");
```

Checking the result of fopen()

```
if(fp == NULL)
{
    printf("Can't open");
    exit(1);
}
```

Using Files in C

Closing Files

```
fclose(fp);
```

File Opening Modes

r =	Open file for Reading
w=	Create file for Writing
a =	Append text file
rb =	Open binary file for Reading
wb=	Create binary file for Writing
ab =	Append binary file

r+ =	Open file for read/write
w+ =	Create file for read/write
a+ =	Append text file for read/write
r+b =	Open binary file for read/write
w+b=	Create binary file for read/write
a+b=	Append binary file for read/write

Working with Text files

Four functions for read files from the disk:

fscanf()

fgets()

fgetc()

fread()

Four functions for write files into the disk:

fprintf()

fputs()

fputc()

fwrite()

Write Character to a file

```
Char name[30];
int i=0;

FILE *fp;
fp=fopen("one.txt","w");

Printf("Enter Name:");
Scanf("%[^\n]",name);
```

```
While(name[i]!='\0')
{
        putc(name[i],fp);
        i++;
}

fclose(fp);
```

Reading Character from file

```
int ch;                          //IMP

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

ch=getc(fp);
```

```
While(ch!=EOF)
{
        putchar(ch);  //IMP
        ch=getc(fp);
}

fclose(fp);
```

Programs

Example 4 : Count the number of characters and number of lines.

Example 6: Compare two files. ➔

```
int ch1,ch2;
FILE *fp1,*fp2;

fp1=fopen(“one.txt”,”r”);
fp2=fopen(“two.txt”,”r”);

ch1=getc(fp1);
ch2=getc(fp2);
```

Programs

```
while(ch1!=EOF &&
        ch2!=EOF &&
        ch1==ch2)
{
        ch1=getc(fp1);
        ch2=getc(fp2);
}
```

```
if(ch1 == ch2)
{
    printf("Same");
}
else if(ch1 != ch2)
{
    printf("Not Same");
}
fclose(fp1);
fclose(fp2);
```

Programs

Example 7: Copy one file to another.

fprintf() : Writing to a file

fscanf() : Reading content from file

```
FILE *fp1;

fp1=fopen(“one.txt”,”w”);

fprintf(fp1,”%s”, “Hello”);
fprintf(fp1,”%d”,10);

fclose(fp1);
```

Programs

```
FILE *fp1;
char name[30];
int value;

fp1=fopen("one.txt","r");

fscanf(fp1,"%s", name);
fscanf(fp1,"%d",value);

fclose(fp1);
```

Example 13: feof().

```
char value[30];

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

while(!feof(fp))
{
    fgets(value,30,fp);
}
fclose(fp);
```

Binary Files

```
FILE *fp1,*fp2;
fp1=fopen(“one.txt”,”rb”);
fp2=fopen(“two.txt”,”wb”);

int ch;

While(1)
{
    ch=fgetc(fp1);
    if(!feof(fp1))
    {
        fputc(ch,fp2);
    }
    else
    {
        break;
    }
}
    fclose(fp1);
    fclose(fp2);
```

Direct File Input & Output

fread()

fwrite()

feof() – End of file

ferror() - Error has occurred.

fread() & fwrite() = To read and write block of data.

==========

```
void main()
{
    FILE *fp1;
    int i, arr1[10], arr2[10];
```

Direct File Input & Output

```
for(i=0;i<10;i++)
{
    arr1[i]=i * 2;
}

fp1=fopen(“one.txt”,”wb);

fwrite(arr,sizeof(int),10,fp1)
```

```
fclose(fp1);

fread(arr2,sizeof(int),10,fp)

fclose(fp1);

for(i=0;i<10;i++)
{
    printf(“%d”,arr2[i]);
}
```

IMP Programs

- Example 16. (fprintf & fscanf)
- Example 17. (fread & fwrite)

Storing Structure in File

```
Struct item
{
    int code;
    double price;
};

Void main()
{
```

```
struct item t1,t2;
File *fp;
fp=fopen(“one.txt”,”wb);

t1.code=10;
t1.price=10.56;
fwrite(&t1,sizeof(t1),1,fp);

fclose(fp);
```

Storing Structure in File

```
fp=fopen(“one.txt”,”rb);

Fread(&t2,sizeof(t2),1,fp);

fclose(fp);
```

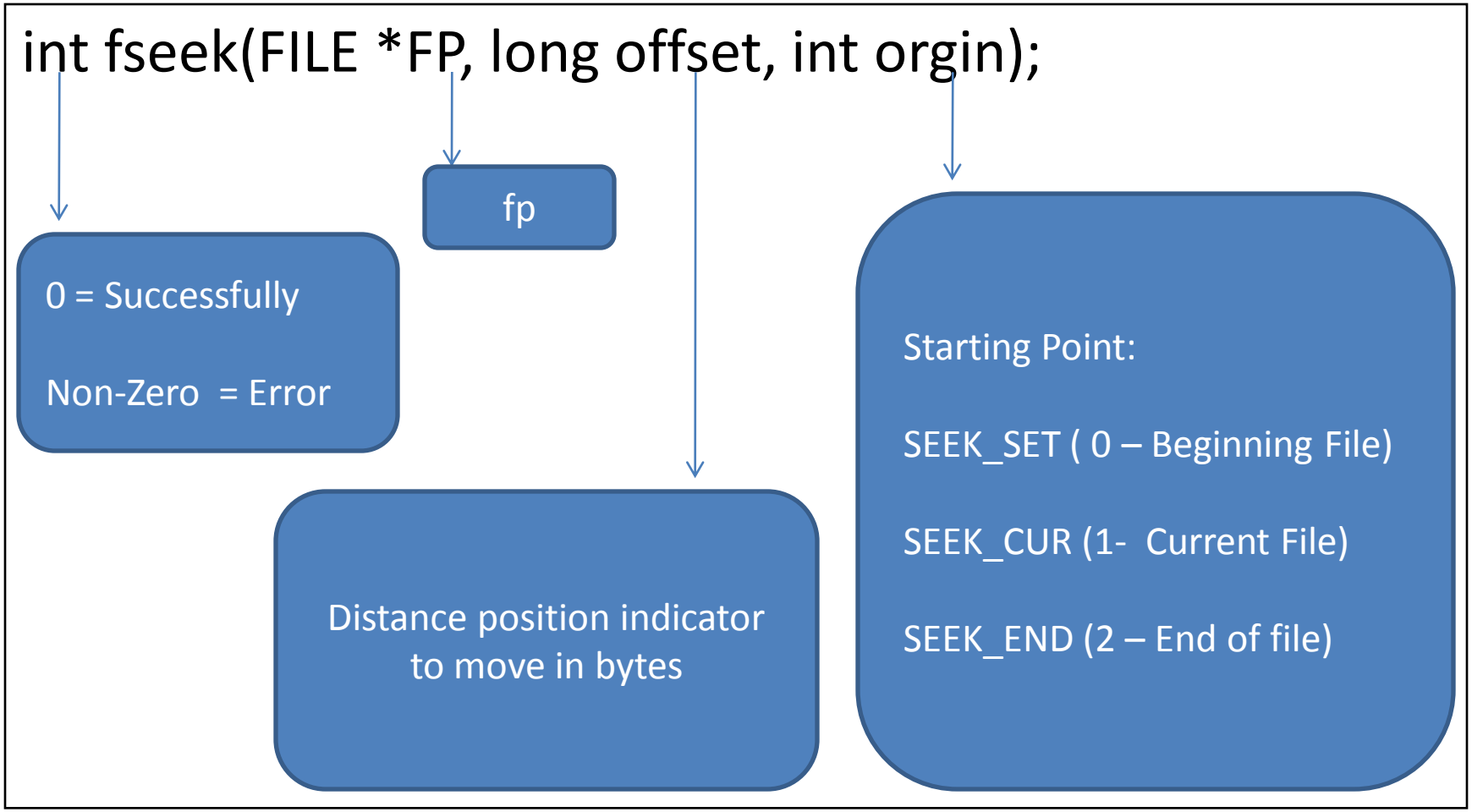
Random Access of Files

fseek() = To Jump to particular Location in a File

ftell() = Current Location in a File

rewind() = Back to First Location in a File

Random Access of Files (fseek())



Rewind() & ftell()

Void rewind(fp);

Long ftell(fp);

value=ftell(fp);

Renaming & Deleting File

Deleting a File:

```
int remove(filename);
```

0 = Success 1 = Failure

Renaming a File:

```
int
    rename(old_name,new
    _name);
```

0 = Success 1 = Failure

References

- Programming in ANSI C Edition 2.1 by E Balagurusamy.
- Programming in C by Paradip Dey & Manas Ghosh.