File Handling

What is a File?

- A *file* is a collection of related data that a computers treats as a single unit.
- Computers store files to secondary storage so that the contents of files remain intact when a computer shuts down.
- 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.

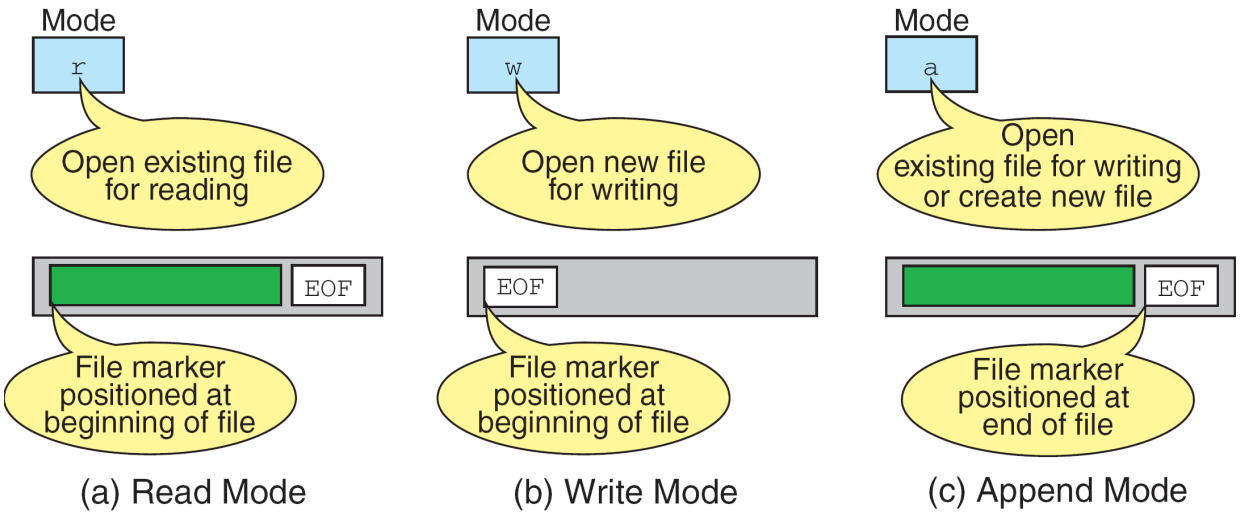
The basic file operations are

- fopen - open a file- specify how its opened (read/write) and type (binary/text)
- fclose - close an opened file
- fread - read from a file
- fwrite - write to a file
- fseek/fsetpos - move a file pointer to somewhere in a file.
- ftell/fgetpos - tell you where the file pointer is located.

File Open Modes

Mode	Meaning
r	Open text file in read mode • If file exists, the marker is positioned at beginning. • If file doesn't exist, error returned.
w	Open text file in write mode • If file exists, it is erased. • If file doesn't exist, it is created.
a	Open text file in append mode • If file exists, the marker is positioned at end. • If file doesn't exist, it is created.

More on File Open Modes



Additionally,

- r+ - open for reading and writing, start at beginning

- w+ - open for reading and writing (overwrite file)
- a+ - open for reading and writing (append if file exists)

File Open

- Syntax:
  filepointer=**fopen("filename", "mode");**
- The file open function (**fopen**) serves two purposes:
  - It makes the connection between the physical file and the stream.
  - It creates "a program file structure to store the information" C needs to process the file.

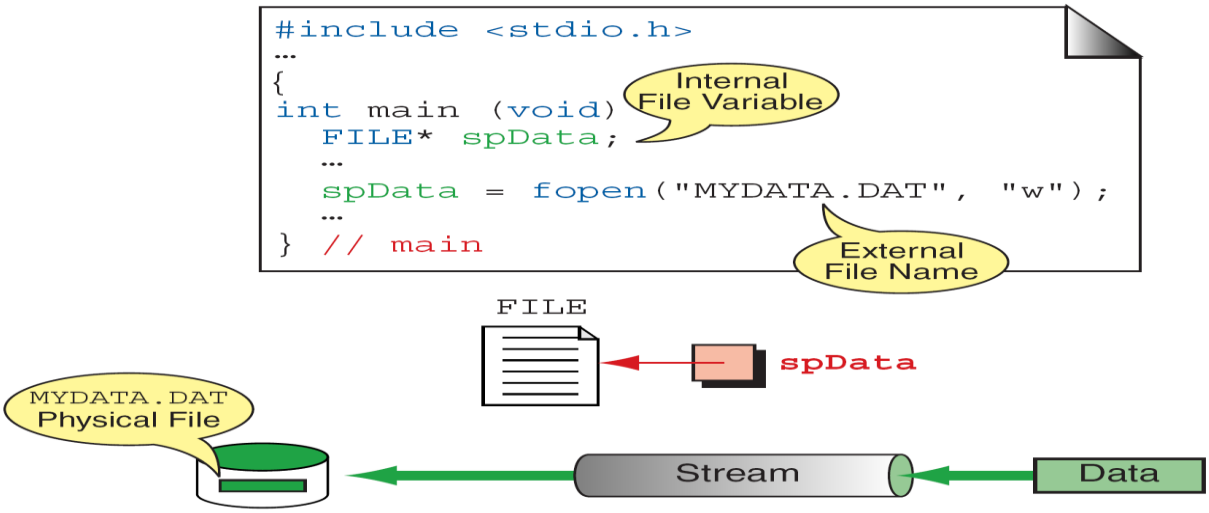
More On fopen

- **The file mode tells C how the program will use the file.**
- **The filename indicates the system name and location for the file.**
- **We assign the return value of fopen to our pointer variable:**

spData = fopen("MYFILE.TXT", "w");

spData = fopen("A:\\MYFILE.TXT", "w");

More On **fopen**



Closing a File

- When we finish with a mode, we need to close the file before ending the program or beginning another mode with that same file.
- To close a file, we use **fclose** and the pointer variable:
  fclose(spData);

fprintf()

Syntax:

fprintf (fp,"string",variables);

Example:

int i = 12;

float x = 2.356;

char ch = 's';

FILE *fp;

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

fprintf (fp, "%d %f %c", i, x, ch);

fscanf()

Syntax:

fscanf (fp,"string",identifiers);

Example:

FILE *fp;

Fp=fopen("input.txt","r");

int i;

fscanf (fp,"%d",i);

getc()

Syntax:

identifier = getc (file pointer);

Example:

FILE *fp;

fp=fopen("input.txt","r");

char ch;

ch = getc (fp);

putc()

write a single character to the output file, pointed to by fp.

Example:

```
FILE *fp;

char ch;

putc (ch,fp);
```

End of File

- There are a number of ways to test for the end-of-file condition. Another way is to use the value returned by the *fscanf* function:

```
FILE *fptr1;

int istatus ;

istatus = fscanf (fptr1, "%d", &var) ;

if ( istatus == feof(fptr1) )

{

printf ("End-of-file encountered.\n") ;

}
```

//Reading and Writing Files

```
#include <stdio.h>

int main ( )

{

FILE *outfile, *infile ;

int b = 5, f ;

float a = 13.72, c = 6.68, e, g ;
```

```
outfile = fopen ("testdata", "w") ;
fprintf (outfile, " %f %d %f ", a, b, c) ;
fclose (outfile) ;
infile = fopen ("testdata", "r") ;
fscanf (infile,"%f %d %f", &e, &f, &g) ;
printf (" %f %d %f \n ", a, b, c) ;
printf (" %f %d %f \n ", e, f, g) ;
}
```

Example

```
#include <stdio.h>
#include<conio.h>
void main()
{
char ch;
FILE *fp;
fp=fopen("out.txt","r");
while(!feof(fp))
{
ch=getc(fp);
printf("\n%c",ch);
}
```

getch();

}

***fread* ()**

Declaration:

size_t fread(void *ptr, size_t size, size_t n, FILE *stream);

Remarks:

fread reads a specified number of equal-sized

data items from an input stream into a block.

ptr = Points to a block into which data is read

size = Length of each item read, in bytes

n = Number of items read

stream = file pointer

Example:

```
#include <stdio.h>
int main()
{
FILE *f;
char buffer[11];
```

```
if (f = fopen("fred.txt", “r”))
{
fread(buffer, 1, 10, f);
buffer[10] = 0;
fclose(f);
printf("first 10 characters of the file:\n%s\n", buffer);
}
return 0;
}
```

fwrite()

Declaration:

size_t fwrite(const void *ptr, size_t size, size_t n, FILE*stream);

Remarks:

fwrite appends a specified number of equal-sized data items to an output file.

ptr = Pointer to any object; the data written begins at ptr

size = Length of each item of data

n =Number of data items to be appended

stream = file pointer

Example

Example:

```
#include <stdio.h>
int main()
{
char a[10]={'1','2','3','4','5','6','7','8','9','a'};
FILE *fs;
fs=fopen("Project.txt","w");
fwrite(a,1,10,fs);
fclose(fs);
return 0;
}
```

fseek()

This function sets the file position indicator for the stream pointed to by stream or you can say it seeks a specified place within a file and modify it.

SEEK_SET	**Seeks from beginning of file**
SEEK_CUR	**Seeks from current position**
SEEK_END	**Seeks from end of file**

Example:

```
#include<stdio.h>
intmain()
{
    FILE * f;
    f = fopen("myfile.txt", "w");
    fputs("Hello World", f);
    fseek(f, 6, SEEK_SET);       SEEK_CUR,   SEEK_END
    fputs(" India", f);
    fclose(f);
    return 0;
}
```

ftell()

offset = ftell(file pointer);

"ftell" returns the current position for input or output on the file

```
#include <stdio.h>
int main(void)
{
FILE *stream;
stream = fopen("MYFILE.TXT", "w");
fprintf(stream, "This is a test");
printf("The file pointer is at byte %ld\n", ftell(stream));
fclose(stream);
return 0;
```

```
}
```

THANK YOU

BEST OF LUCK…