File Handling in C

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 (unchanged) 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 :

   FILE *p;

2. Open the file, associating the pointer with the file name.
3. Read or write the data.
4. Close the file.

Basic file operations

Declaring and opening a file:

syntax:

```
FILE *fp;
fp=fopen("file_name", "mode");
```

File Open Modes:

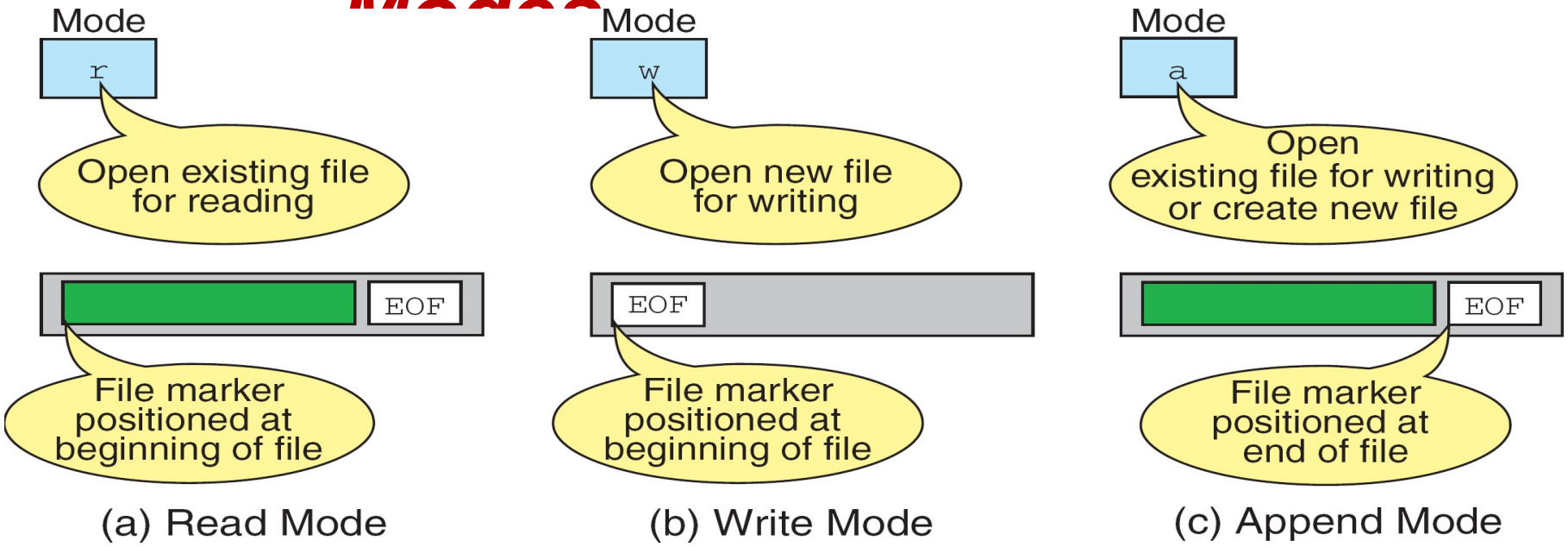
r- read mode *file is opened in read only (file must exist)*

w- write mode *file is opened for write only (content of file deleted if file already exist)*

a- append mode *file is opened for appending(adding new content) (if file does not exist file created).*

r+ w+ a+ files will be opened for both read and write operations.

More on File Open Modes



Closing a File

- When we finish the program, we need to close the file before ending
- To close a file, we use `fclose()` function

  fclose(fp);

read and write operations on file

getc() and putc()

To read a character from file:

```
char ch;
ch = getc (fp);
```

To write a character to the file:

```
char
ch='a';
putc(ch,f
p);
```

getw() and putw()

To read a integer from file:

```
int x;
x = getw(fp);
```

To write a integer to the file:

```
int x=20;
putw(x,fp);
```

Program to write text into a file and then read the same text from file

```
char ch;
FILE *fp1, *fp2;
fp1=fopen("abc.txt",
"w");  ch=getchar();
      while(ch!=EOF)
       {
          putc(ch,fp1);
          ch=getchar();
      }
fclose(fp1);
```

This will open a file “abc.txt” in write mode

Text can be written to file until EOF(^Z) .

read and write operations on file

Writing data to file: **fprintf()**

Syntax:

fprintf (fp, “control string", variables);

Example:

```
int i = 12;
float x =
2.356;  char
ch = 's';  FILE
*fp;
fp=fopen(“abc.txt”, “w”);
fprintf (fp, "%d %f %c", i, x, ch);
```

Reading data from file: **fscanf()**

Syntax:

fscanf (fp, “control string”,variables);

Example:

```
FILE *fp;
fp=fopen(“abc.txt”,
“r”);  int i;
 float j;
fscanf (fp,“%d%f",&i,&j);
printf(“the values are %d
%f”,i,j);
```

fwrite() and fread()

The fwrite() function is used to write records (sequence of bytes) to the file. A record may be an array or a structure.

Syntax:

```
fwrite( ptr, int size, int n, FILE *fp );
```

Parameters:

- **ptr** − ptr is the reference of an array or a structure stored in memory.
- **size** − This is the size in bytes of each element to be written.
- **n** − This is the number of elements, each one with a size of **size** bytes.
- **fp** − FILE* (file) where the records will be written.

fwrite() and fread()

The fwrite() function is used to write records (sequence of bytes) to the file. A record may be an array or a structure.

Syntax:

```
fread( ptr, int size, int n, FILE *fp );
```

Parameters:

- **ptr** − ptr is the reference of an array or a structure stored in memory.
- **size** − This is the size in bytes of each element to be written.
- **n** − This is the number of elements, each one with a size of **size** bytes.
- **fp** − FILE* from where the records will be read.

Random accessing of file

ftell(): it tells the current position of pointer in file.

syntax:

```
int n;
n=ftell(fp
);
```

fseek(): this function moves the pointer to desired position in file.

syntax:

fseek(fp, offset, position);

where-

Position: from where we have to move(initial position) Offset: how many bytes we have to move.

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:

```
FILE * fp;
fp= fopen("myfile.txt", "w");
fputs("Hello World", fp);
fseek(fp, 6, SEEK_SET); //
SEEK_CUR,  fputs(" India", fp);
fclose(f);
```

SEEK_END

Program for copy file

```
#include <stdio.h>
#include <stdlib.h> // For exit()
int main()
{
	FILE *fptr1, *fptr2;
	char c;
	fptr1 = fopen(“file1.txt”, "r");
	if (fptr1 == NULL)
	{
		printf("Cannot open file %s \n", filename);
		exit(0);
	}
	fptr2 = fopen(“file2”, "w");
	if (fptr2 == NULL)
	{
		printf("Cannot open file %s \n", filename);
		exit(0);
	}
```

```
    c = fgetc(fptr1);
    while (c != EOF)
    {
        fputc(c, fptr2);
        c = fgetc(fptr1);
    }

    printf("\nContents copied to %s", filename);
    fclose(fptr1);
    fclose(fptr2);
    return 0;
}
```