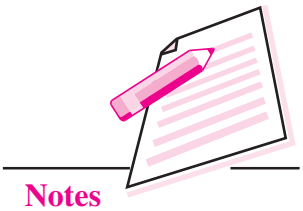


21

FILES

In the previous lesson you have learnt about pointers. Now, you will learn about files. In some application program, sometimes it is required to store data on hard disk or any other storage devices. The data is stored in these devices using the concept of file. In this lesson you will learn to store data in a file, access data from a file, move pointer within the file.

OBJECTIVES

After reading this lesson, you will be able to:

- store data in a file;
- access data record by record from the file;
- move pointer within the file;
- open or close file.

21.1 FILE

A file is a collection of logically related records. A program usually requires two types of data communication, (i) writing data on datafile, (ii) reading data from datafile.

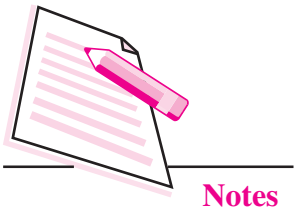
Let us learn about them.

(i) Writing data on the datafile:

The data flows from keyboard to memory and from memory to storage device.

i.e., keyboard → memory → hard disk/ storage device

This is called output stream where stream is the flow of data and requires an **ofstream.h** header file.



Notes

(ii) **Reading data from datafile:**

The data flows from storage device to memory and from memory to output device, particularly monitor.

i.e., datafile → memory → output device (screen) or external storage device (hard disk/storage device)

This is called input stream and requires **ifstream**.h header file. If both input stream and output stream are used in the same program then header file **fstream.h** is required. If header file fstream.h is included in the program, there is no need to include iostream.h explicitly.

File has to be opened, before reading data from datafile or writing data on the datafile. Let us learn how to open a file.

21.1.1 Opening a file

A file can be opened in two ways:

(i) using constructor function of a class.

(ii) using the member function open ( ) of the class.

Opening a file using constructor function

The following statement opens the file STU.DAT in output mode, i.e., for writing data on the file.

```
ofstream outfile ("STU.DAT");
```

ofstream is a class available in the compiler file. outfile is any user defined object.

The statements

```
outfile < < "TOTAL MARKS" << "\n";
outfile < < total << "\n";
```

are used for writing data on the file. The newline character is used for moving the pointer to the next line.

Similarly, the following statement

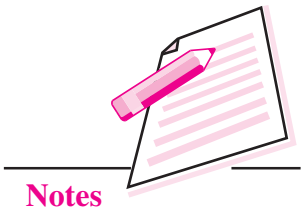
```
ifstream infile ("STU.DAT");
```

opens the file "STU.DAT" in input mode, i.e., for reading purpose

The statements

```
infile >> string;
infile >> number;
```

read the data from the data file.



The following program uses a single file for both writing and reading purposes. First, it takes the data from the keyboard and writes it to the file. After the writing is completed, the file is closed. The program again opens the same file, reads the information already written to it and displays it on the screen.

```
# include < fstream.h >
void main ( )
    {
    char name [30];
    int rn, marks;
    ofstream outfile ( "INF.DAT", ios : : binary);
    cout << "Enter student name";
    cin >> name;
    cout << "Enter student roll number";
    cin >> rn;
    cout << "Enter student marks";
    cin >> marks;
    outfile << name << "\n";
    outfile << rn << "\n";
    outfile << marks << "\n";
    outfile . close ( );
    ifstream infile ( "INF" );
    infile >> name;
    infile >> rn; infile >> marks;
    cout << "Name" << name << "\n";
    cout << "Roll no" << rn << "\n";
    cout << "Marks" << marks << "\n";
    infile.close ( );
    }
```

The output of the program would be:

```
Enter student Name                PARAM
Enter student roll number         20
Enter student marks               90

Name                              PARAM
Roll no                           20
Marks                             90
```

Opening a file using open ( ) function

The function open ( ) can be used to open multiple files that use the same stream object.

First a stream object is assigned to and then it is used to open the file in turn.

```
filestream_class stream_object;
stream_object . open ("filename");
```

For example :

```
ofstream outfile;
outfile . open ("ABC");
_ _ _ _ _
outfile . close ( );
outfile . open ("XYZ");
_ _ _ _
outfile.close ( );
```

Another form of open() is also available.

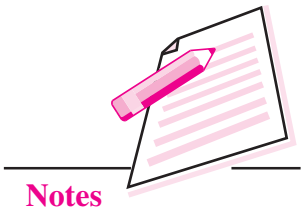
The open ( ) function has two parameters : filename and access mode. The general format is:

```
stream_object . open ("filename", access mode);
```

The second argument specifies the mode in which the file is opened. The default values are taken for ifstream or ofstream functions (the mode is not defined explicitly).

```
ios : : in for ifstream functions
ios : : out for ofstream functions
```

The file mode parameters can take one or more of the constants defined in the class ios.



The following table shows the file mode parameters.

Table 21.1: File mode parameters and its meaning

Parameter	Meaning
ios : : app	It opens the file in output mode. The file pointer is at the end of file and it can add a record.
ios :: ate	The file pointer is at the end of the file and it allows to add data or to modify the existing data anywhere in the file.
ios :: binary	Binary file
ios :: in	It opens the file in input mode. The file pointer is at the top of the file and it is ready for reading.
ios :: nocreate	If file is already present, it opens the file otherwise open fails.
ios :: noreplace	If file is not present, it opens the file otherwise open statement fails.
ios :: out	It opens the file in output mode. The file pointer is at the end of the file. If it already has a data, the output mode erases the content of the file.
ios :: trunc	It deletes the contents of the file if exist.

The mode can combine two or more parameters using bitwise OR operator.

Example:

outfile . open (“ABC.DAT”, ios::in | ios::out | ios :: binary);

21.1.2 Write ( ) and read ( ) functions

The functions write ( ) and read ( ) have two parameters: address of the variable, size of the variable. The address of the variable must be cast to the type char*. The general format is:

infile . read ((char*) & v, sizeof v);

outfile . write ((char*) & v, sizeof v);

where v is the variable.

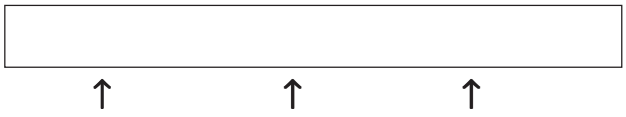
21.1.3 File Pointers

File has two associated pointers called input pointer (or get pointer) and output pointer (or put pointer). Each time an input or output operation takes place, the pointer moves automatically. There are two pointers.

seekg ( ) It moves get pointer to a specified location.

seekp ( ) It moves the put pointer to a specified location.

A file can be viewed as



ios :: beg ios :: cur ios :: end

ios :: beg – means beginning of the file.

ios :: cur – means current position of the pointer

ios :: end – means end of the file

The seekg ( ) and seekp ( ) statement has two parameters.

object . seekg (no. of bytes, refposition);

object . seekp (no. of bytes, refpostion);

The refposition takes one of the above three constants defined in the ios class.

Example 1

infile.seekg (0, ios::beg);

It moves the pointer to the beginning of the file. In this case, the refposition ios :: beg is optional.

infile.seekg (100, ios::cur);

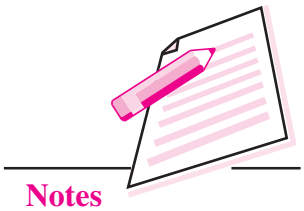
It moves the pointer 100 bytes forward from the current position.

infile.seekg (-200, ios::end);

It moves the pointer 200 bytes backward from the end of the file.

21.1.4 The tellg ( ) and tellp ( ) function

The tellg ( ) function gives the position of get pointer in terms of number of bytes. Similarly, tellp ( ) function gives the position of put pointer in terms of bytes.



Notes

Example 2

ifstream infile;

infile . open ("ABC", ios ::ate);

int B = infile . tellg ( );

On execution of the above statements, the input pointer is moved to the end of the file and B gives the number of bytes in the file.

21.1.5 Close ( ) function

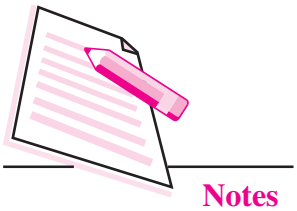
The file should be closed at the end if it is opened either through constructor or open ( ) function. The General format is stream_object.close ( );

The following example works with class object and does the following operations:

- Create a data file
- Display a data file
- Adding a new record
- Modify the existing record

Example 3

```
# include <fstream.h>
class student
{
        char name [30];
        int rn;
public:
        void getdata ( );
        void putdata ( );
};
void student : : getdata ( )
{
        cout <<"Enter student name";
        cin >> name;
        cout << "Enter roll number";
        cin >> rn;
}
```



Notes


```
void student :: putdata ( )
{
        cout << "Student name" << name << "\n";
        cout << "Student roll number" << rn << "\n";
}
void main ( )
{
        fstream file;
        file . open ( "ABC", ios::in | ios::out | ios::binary);
        student st;
        int i, n;
        cout << "How many records to enter";
        cin >> n;
        for (i = 1; i < = n, i ++ )
        {
        st. getdata ( );
        file . write ((char*) & st, sizeof st);
        }
        / / Display a data file
        file . seekg ( 0, ios::beg);
        while (file . read ((char*) & st, sizeof st))
        {
        st. putdata ( );
        }
        file . clear ( ); / / To make the end of file mark false
        // To append record
        st . getdata ( );
        file . write ((char*) & st, sizeof st);
        // To modify a record
        file.clear ( );
        cout << "Enter record number";
        cin >> n;
        file . seekp ((n - 1)* sizeof st, ios::beg);
        st. getdata ( );
        file.write ((char*) & st, sizeof st);
        // To close a file
        file . close ( );
}
```

INTEXT QUESTIONS 21.1

1. Fill in the blanks:

   (a) A ..... is a collection of logically related records.

   (b) The file opened in ofstream is only available for .....

   (c) The file opened in ifstream is only available for .....

   (d) We can open the file using ..... function.

   (e) The mode app opens the file for .....

   (f) The file open in output mode is ..... by default.

2. State whether the following statements are true or false:

   (a) A file is a collection of records.

   (b) The file opened in ofstream is available for writing.

   (c) The output mode of opening a file deletes the contents, if present in the file.

   (d) The close ( ) function is used to close a file.

   (e) The statement

   outfile.write ((char*) & obj, sizeof obj);

   writes only data in obj to outfile.

   (f) The ios::ate mode allows us to write data at the end of the file only.

   (g) We can add data to an existing file by opening in append mode.

   (h) The data written to a file with write ( ) function can be read with the get ( ) function.

WHAT YOU HAVE LEARNT

- File is a collection of logically related records.
- You can write data on the file and read data on the file.
- A file can be opened in two ways: (i) using constructor function of a class, (ii) using the member function open() of the class.

- Open( ) requires two parameters: file name and access mode.
- File has two pointers called input pointer and output pointer.
- Tellg( ) function gives the position of get pointer in terms of number of bytes.
- Tellp( ) function gives the position of put pointer in terms of bytes.

TERMINAL EXERCISE

1. What is an input stream?
2. What is the difference between opening a file with constructor function and opening a file with open () function?
3. What is the file access mode? Describe the various file modes.
4. A file consists of 5 records, each takes 100 bytes of storage:

```
fstream file:
file. seekg ( 0, ios::end);
N = file.tellg ( );
```

   (i) What will be the datatype of N?

   (ii) What will be the value of N?

5. Consider the following statements:

```
fstream file;
file.open ("ABC", ios::in | ios::out);
```

   Write C++ statement(s) for the following:

   (i) To move the pointer at the beginning of file.

   (ii) To move the pointer at the end of file.

   (iii) To find the total number of bytes.

   (iv) To close the file.

6. Explain the functioning of the following:

```
fstream file;
(i)   file.seekg (100, ios::cur);
(ii)  file.seekg (-100, ios::end);
(iii) file.seekg ( 100, ios::beg);
```

7. The record consists of two fields: name and rollno. Write a program that will perform the following:

   (i) create a data file of 5 records

   (ii) display a data file

   (iii) append a record

   (iv) modify one of the records

ANSWERS TO INTEXT QUESTIONS

21.1

1. (a) file (b) output
   (c) input (d) open
   (e) output (f) trunc

2. (a) True (b) True
   (c) True (d) True
   (e) True (f) False
   (g) True (h) False