Chpapter 7 : Working With Files

Opening & Closing a File

If programmers want to use a disk file for storing data, they need to decide about the following things about the file and its intended use.

These points that are to be noted are :

- A name for the file
- Data type and structure of the file
- Purpose (reading, writing data)
- Opening method
- Closing the file (after use)

Files can be opened in two ways. They are :
1.Using constructor function of the class
2.Using member function open of the class

A Opening File : The first operation generally performed on an object of one of these classes to use a file is the procedure known as to opening a file. An open file is represented within a program by a stream and any input or output task performed on this stream will be applied to the physical file associated with it.

There are some mode flags used for file opening. These are:

ios::app: append mode

ios::ate: open a file in this mode for output and read/write controlling to the end of the file

ios::in: open file in this mode for reading ios::out: open file in this mode for writing

ios::trunk: when any file already exists, its contents will be truncated before file opening

A Closing File : When any C++ program terminates, it automatically flushes out all the streams releases all the allocated memory and closes all the opened files. But it is good to use the close() function to close the file related streams and it is a member of ifsream, ofstream and fstream objects.
The structure of using this function is :
void close();

File Pointer & their Manipulation : The header file iomanip provides a set of functions called manipulators which can be used to the manipulate the output formats. They provide the same features as that of the ios(Input output system) member functions and flags.

Some manipulators are more convenient to use than their counterparts in the class(Input output system) ios. such that, two or more manipulators can be used as a chain in one statement as shown below :

```
cout << manip1 << manip2 << manip3 << item;

cout << manip1 << item1 << manip2 << item2;
```

This kind of concatenation is useful when we want to display several columns of output.The most commonly used manipulators are shown in Table. The table also gives their meaning and equivalents. To access these manipulators, we must include the file iomanip in the program.

This kind of concatenation is useful when we want to display several columns of output.The most commonly used manipulators are shown in Table. The table also gives their meaning and equivalents. To access these manipulators, we must include the file iomanip in the program.

Manipulators and their meanings

Manipulator	Meaning	Equivalent
setw(int w)setprecision(int d)	Set the field width ton w. Set the following point precision to d.	width()precision()
setfill(intc)	Set the fill character to c.	fill()
setiosflags(long f)	Set he format flag f.	setf()
restiosflags(long f)	Clear the flag specified by f.	unsetf()
Endif	Insert new line and flush stream.	“\n”

Program illustrates the following of the output values using both manipulators and ios functions.

```
 #include < iostream >

#include < iomanip >

using namespace std;

int main()

{   cout.setf(ios : : showpoint);

cout << setw(5) << “n”<< setw(15) << “Inverse_of_ n”<< setw(15) << “Sum_
of_terms\n\n”;

double term, sum=0;
for(int n=1; n<=10; n++)

{

term = 1.0 / float(n);

sum = sum + term;

cout << setw(5) << n<< setw(14) << setprecison(4)<< setiosflags(ios : : scientific)
<< term<< setw(13) << resetiosflags(ios : : scientific)<< sum << end1;

}

return 0;

}
```

The output of Program 10.8 would be :

n	Inverse_of_n	Sum_of_terms
1	1.0000e+000	1.0000
2	5.0000e-001	1.5000
3	3.3333e-001	1.8333
4	2.5000e-001	2.0833
5	2.0000e-001	2.2833
6	1.6667e-001	2.4500
7	1.4286e-001	2.5929
8	1.2500e-001	2.7179
9	1.1111e-001	2.8290
10	1.0000e-001	2.9290