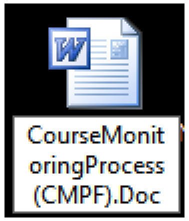
What is a File?

- A file is a collection of information, usually stored on a computer’s disk. Information can be saved to files and then later reused.

File Names

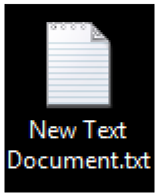
All files are assigned a name that is used for identification purposes by the operating system and the user.

File Name and Extension	File Contents
MYPROG.BAS	BASIC program
MENU.BAT	DOS Batch File
INSTALL.DOC	Documentation File
CRUNCH.EXE	Executable File
SSUET.HTML	HTM L (Hypertext M arkup Language) File
3DMODEL.JAVA	Java program or applet
INVENT.OBJ	Object File
PROG1.PRJ	Borland C++ Project File
ANSI.SYS	System Device Driver
README.TXT	Text File


CourseMonitoringProcess(CMPF).Doc

RM.exe

RM.sys


New Text Document.txt

Focus on Software Engineering: The Process of Using a File

- Using a file in a program is a simple three-step process
  - The file must be opened.

    If the file does not yet exits, opening it means creating it.
  - Information is then saved (write)to the file, read from the file, or both.
  - When the program is finished using the file, the file must be closed.

C++ Files and Streams

Classes for file stream operation

ofstream: Stream class to write on files

ifstream: Stream class to read from files

fstream: Stream class to both read and write from/to files.

These classes are derived directly or indirectly from the classes istream and ostream.

C++ Files and Streams

- C++ views each files as a sequence of bytes.
- Each file ends with an ***end-of-file*** marker.
- When a file is ***opened***, an object is created and a stream is associated with the object.
- To perform file processing in C++, the header files **<iostream>** and **<fstream**> must be included.
- <fstream> includes <ifstream> and <ofstream>

open a file

The first operation generally performed on an object of one of these classes is to associate it to a real file. This procedure is known as to *open a file*.

open ("filename.extension", mode);

File Open Modes

ios::in	Open for input operations.
ios::out	Open for output operations.
ios::binary	Open in binary mode.
ios::ate	Set the initial position at the end of the file.
ios::app	All output operations are performed at the end of the file, appending the content to the current content of the file.
ios::trunc	If the file is opened for output operations and it already existed, its previous content is deleted and replaced by the new one.

Open File

is_open()

- To check if a file stream was successful opening a file
- you can do it by calling to member is_open.
- This member function returns a bool value i.e. true or false.
- True in the case that indeed the stream object is associated with an open file, or false otherwise:

```
if ( myfile.is_open() )
{ /* ok, proceed with output */ }
```

How to close a file in C++?

The file is closed implicitly when a destructor for the corresponding object is called

OR

by using member function *close:*

myfile.close();

Once this member function is called, the stream object can be re-used to open another file, and the file is available again to be opened by other processes.

Text file

These files are designed to store text and thus all values that are input or output from/to them can suffer some formatting transformations, which do not necessarily correspond to their literal binary value.

WRITING ON A TEXT FILE

```
#include <iostream>
#include <fstream>

int main (){
  ofstream myfile ("example.txt");

if (myfile.is_open())
  {

myfile << “HARITH AHMAD\n";
myfile << “BSCS\n";
myfile << “SIR SYED UNIVERSITY.\n";

    myfile.close();
  }
  else cout << "Unable to open file";
                                  getch();
}
```

READING A TEXT FILE

```
#include <iostream>
#include <fstream>
#include <string>

int main () {
  string line;
  ifstream myfile ("example.txt");
  if (myfile.is_open())
  {
    while (! myfile.eof() )
    {
      getline (myfile,line)
      cout << line << '\n';
    }
    myfile.close();
  }
  else
{
cout << "Unable to open file";
}
 getch();
}
```

- This last example reads a text file and prints out its content on the screen.
- We have created a while loop that reads the file line by line, using getline.
- The value returned by getline is a reference to the stream object itself, which when evaluated as a Boolean expression

True

✓if the stream is ready for more operations

False

✓ if either the end of the file has been reached
✓if some other error occurred.

eof() which stands for "end of file".

- The eof() function is a boolean function
- check whether or not the file has reached the end.
- It returns true when the file is at the end and false otherwise.

Checking state flags

Myfile.bad()

Returns true if a reading or writing operation fails. For example, in the case that we try to write to a file that is not open for writing or if the device where we try to write has no space left.

Myfile.fail()

Returns true in the same cases as bad(), but also in the case that a format error happens, like when an alphabetical character is extracted when we are trying to read an integer number.

Myfile.eof()

Returns true if a file open for reading has reached the end.

Myfile.clear()

can be used to reset the state flags.