CEE432/CEE532/MAE541
Developing Software for Engineering Applications

Lecture 9: File Handling
(Chapter 12)

File Types

- Text (or ASCII file)
  - Can be viewed/edited using a command such as **`type`** or program such as Notepad or Word or VS2005 editor.
  - Access to the file is sequential.
- Binary
  - A file containing information that is in (specific) machine-readable form; it can be read only by a specialized application. "Binary_file" usually refers to a file that uses all 8 bits of each byte for information.

Handling Text File

- Opening a file

```
#include <fstream>

std::ifstream object_name;
or
std::ofstream object_name;

Object_name.open (char string, openmode);
```

Handling Text File

- Text file for reading

```
#include <fstream>
….
std::ifstream InputFile;
InputFile.open (″DataSet1.dat″, std::ios::in);
…
InputFile >> nX >> nY >> nZ;
…..
InputFile.close ();
```

Handling Text File

- Text file for writing

```
#include <fstream>
….
std::**ofstream** OutputFile;
OutputFile.open (″DataSet1.out″, std::ios::out);
….
OutputFile << “This is x : “ << fX << ‘\n’;
…..
OutputFile.close ();
```

Special Ops

- Reading a single character
- Reading an entire line

```
char c;
InputFile.get (c);
…
const int MAXCHARS = 256;
char szInput[MAXCHARS];
FileForInput.getline(szInput, MAXCHARS);
```

Stream States

- A state `bit` is used to store the current state of the iostream
- Fail bit is used to test if the last operation was successful or not via the **`fail`** member function.

```
InputFile.open ("DataSet1.dat", std::ios::in);
if (InputFile.fail())
{
….
}
```

Stream States

- End-of-file bit is used to test if the end-of-file condition has been reached via the **`eof`** member function. Cannot read beyond the end-of-file.

```
InputFile << nX << nY << nZ;
if (InputFile.eof())
{
….
}
```

Using In-Memory Files

```
#include <sstream>
void ShowMessage (std::string& szMessage)
{
    std::cout << szMessage << std::flush;
}
….

double PI = 22.0/7.0;
std::ostringstream szM;
szM << "The value of PI is " << PI << ".\n";
std::string szMessage = szM.str();
ShowMessage (szMessage);
….
```

Using In-Memory Files

```
#include <sstream>
std::string szMessage = “1.34 4.5 6.3”;
std::istringstream szM (szMessage);
float fA, fB, fC;
szM >> fA >> fB >> fC;
```