**Chapter 12**

**Files**

**Introduction**

There are 3 possible classes we can use for file processing:

ifstream - input (read) files only

ofstream - output (write) files only

fstream - most general, can do everything. This is the class most use and we will too.

**Text Files**

We’ll only be processing text files in our class. These are files that are human-readable and only contain those characters that you can type in using the keyboard. A .txt file is an example of a text file. The range of possible characters for these files are shown below:

0-31 32 - 127 128-255

Control Characters Printable Characters Special Characters

(Binary Files Only) (Text Files) (Binary Files Only)

**Required Header File**

**#include <fstream>**

Most general include - Allows reading/writing/appending

**#include <ifstream>**

Defines ifstream only (for reading only)

**#include <ofstream>**

Defines ofstream only (for writing/appending only)

**Initializing File Stream - Constructor**

**Reading from a file**

fstream my\_stream(file\_name, ios::in);

constructor takes 2 arguments

string file\_name -

ios::in is the mode of the file

The Successful Construction creates an input connection (i.e., stream) to the desired text file. **For our programs, the file name should reside in the same folder as our C++ project.**

**Writing to a file**

fstream my\_stream(file\_name, ios::out);

constructor takes 2 arguments

string file\_name

ios::out is the mode of the file

creates a text file output stream

The Successful Construction creates an input connection (i.e., stream) to the desired text file. If the file does not exist, a new, empty file will be created. If the file does exist, all existing contents will be destroyed.

**Opening file using Default Constructor and Setter (i.e., open)**

fstream my\_file;

my\_file.open(file\_name, ios::in); //Input strm

fstream my\_file

my\_file.open(file\_name,ios::out); //Output str

**Testing if file opened successfully for reading**

string file\_name = "my\_file.lis";

fstream my\_file(file\_name,ios::in);

if ( !my\_file.is\_open() )

{

cout << "Error opening file" << endl;

exit(-1);

}

if ( my\_file.fail() )

{

cout << "Error opening file" << endl;

exit(-1);

}

if (!my\_file) **//This is the one we’ll use**

{

cout << "Error opening file" << endl;

exit(-1);

}

**Reading info from text file**

//Read one word from a file

string word;

my\_file >> word; //default-whitespace delimited

//Read one line (nl delimited) from a file

string line;

getline(my\_file,line);

**Writing info to text file (use <<)**

//Write one character to a file

char c = ‘a’;

my\_file << c;

//Write one word to a file (Terminated by \0)

string word = “Hello”;

my\_file << word ;

//Write one line (\0 delimited) to a file

string line = “Hello World”;

my\_file << line << endl;

**Detecting the end of file**

**eof()** is a member function used for detecting the end of file

bool eof()

Returns a **true** when end-of-file marker (^Z) is read (hidden character)

end-of-file marker is reached when the last byte of the file has been successfully read and another read is attempted

Example of reading all words from a file:

//Read words

while(true)

{

//Read one word at a time (based on white space)

in\_file >> word;

//Test for eof

if (in\_file.eof())

break;

//OW, display the word

cout << word << endl;

}