MECH 215

Lecture 7

File I/O

File I/O

- It is of great benefit to be able to store and access information from different processes in an efficient manner.
- One such method is storing data in files.
- Data files may have different formats and hold different types and forms of information.
- In C++ we will look at storing and retrieving program data using file input output (File I/O) using the C++ file stream library.

File I/O

- FILE INPUT AND OUTPUT
- The C++ library <fstream> contains definitions of three STREAM
- class types:
  - ifstream - this class is used to create streams which are used to input values from
  - ofstream - this class is used to create streams which are used for output
  - fstream - used to create streams capable of both input and output
- We will study the ifstream and ofstream classes only.

File I/O

- //
- // Program that will compute Fib. Sequence and store results
- //
- #include <iostream>
- #include <string>
- #include <fstream> // Activate file I/O library
- #include <cmath>
- using namespace std;
- int fib_seq(int);

File I/O

- int main ()
- {
- int num_terms,fib_num_1,fib_num_2,index1;
- const int array_size=10;
- int fib_seq[array_size];
- //
- // Define an output file to save the numbers
- //
- ofstream outputfile;
- outputfile.open("fib_num.txt");
- fib_seq[1]=0;
- fib_seq[2]=1;

File I/O

- //
- // Write the numbers to the file
- //
- outputfile <<fib_seq[1] <<endl;
- outputfile <<fib_seq[2] <<endl;
- for (index1=3;index1<=array_size;index1++)
- {
- fib_seq[index1]=fib_seq[index1-1]+fib_seq[index1-2];
- //
- // Write the new numbers to the file
- //
- outputfile << fib_seq[index1] << endl;
- }

File I/O

- //
- // close the file
- //
- outputfile.close ();

- return 0;
- }

File I/O

- We can also read from one file and write to another file.
- In the following program we will read data (radian angle measures) and then process the data (use the data to compute the output using a sine function).
- The output data will then be sent to another file for storage.
- The program follows:

File I/O

- //
- // file: file_io_1.cp
- //
- #include<iostream>
- #include<cmath> // Used for math functions like sine
- #include <fstream> // fstream library defines stream classes ifstream ofstream and fstream
- using namespace std;
- int main()
- {
- ifstream infile("angle.dat"); // associate the file angle.dat with infile object of class ifstream

File I/O

- ofstream outfile("response.dat"); // associate the file response.dat with the outfile object of class ofstream
- float number;
- // a simple while loop which reads numbers from the infile
- // and writes them to the outfile object. Loop will terminate
- // when the end of the infile object is reached
- while (infile >> number)
- outfile << sin(number) << endl;
- return 0;
- }

File I/O

- Input from file is:
- 0
- 0.1
- 0.2
- 0.3
- 0.4
- 0.5
- 0.6
- 0.7
- 0.8
- 0.9
- 1.0

File I/O

- The output from response.dat is:
- 0
- 0.0998334
- 0.198669
- 0.29552
- 0.389418
- 0.479426
- 0.564642
- 0.644218
- 0.717356
- 0.783327
- 0.841471

File I/O

- You can format the output using the iomanip library which is a very useful tool.
- In the following program the previous code is updated to include 9 decimal places of precision and 9 decimal places in width.
- The program and output follow.

File I/O

- //
- // file: file_io_2.cp
- //
- #include<iostream>
- #include<cmath> // Math functions
- #include<iomanip> //Formating input or output
- #include <fstream> // fstream library defines stream classes ifstream ofstream and fstream
- using namespace std;
- int main()
- {
- ifstream infile("angle.dat"); // associate the file angle.dat with infile object of class ifstream

File I/O

- ofstream outfile("response.dat"); // associate the file response.dat with the outfile object of class ofstream
- float number;
- // a simple while loop which reads numbers from the infile
- // and writes them to the outfile object. Loop will terminate
- // when the end of the infile object is reached
- while (infile >> number)
- outfile <<setprecision(9)<<sin(number)<<setw(9)<< endl;
- return 0;
- }

File I/O

- The input is the same but response.dat now holds:
- 0
- 0.0998334214
- 0.198669329
- 0.295520216
- 0.389418334
- 0.47942555
- 0.564642489
- 0.64421767
- 0.717356086
- 0.783326864
- 0.841471016

File I/O

- There are many choices for how to read and write to a file but it is typical to use an easy to read format.
- There are other factors to consider. Such as file modes and closing files.
- The default mode for output to a stream object is to OVERWRITE the file every time the program is invoked. For example, with the previous two examples, if we run the program with different data in the "input.dat" file, then the contents in the output file will be overwritten with new output every time the program is executed. If we wish to APPEND to the end of the existing output file, then the file must be opened in a slightly different fashion with an extra parameter:

File I/O

- ofstream outfile("myoutput.dat", (ios_base::out | ios_base::app));
- This expression specifies that the file "myoutput.dat" is to be opened for output and that any writing to the file is to be APPENEDED to the current end of the file.
- We can also specify the mode as a parameter to the open() method if we use the second style of specifying a file anme with a stream object:
- ofstream outfile;
- outfile.open("myoutput.dat", (ios_base::out | ios_base::app));
- Some compilers don't support the ios_base::out | ios_base::app (for example the g++ compiler does not support it). Check your compiler documentation to see which modes are supported.

File I/O

- We have already seen examples of opening files. Let us see some examples of closing files. In addition to the member function open(), objects of type ifstream or ofstream also have the member function close() - this is typically used to break a current association of a physical file with a file stream object.
- The program can then make a new association between a new disk file and the given stream object.

- The following program reads from two different disk files and write the output to different disk files using only two file stream objects : infile and outfile:

File I/O

```
// Author: Ted Obuchowicz
//  Nov. 26, 2003
// file:  file_io4.C

#include <fstream> // fstream library defines stream classes ifstream
                   // ofstream and fstream
using namespace std;
int main()
{
ifstream infile;
ofstream outfile;
infile.open("input1.dat") ;
outfile.open("output1.dat");
int number;
```

File I/O

- // a simple while loop which reads numbers from the infile
- // and writes them to the outfile object. Loop will terminate
- // when the end of the infile object is reached
- while (infile >> number)
- outfile << number << endl;
- // now we close the files and associate the file streams infile nd outfile with new disk files:
- infile.close();
- outfile.close();

File I/O

- infile.open("input2.dat") ;
- outfile.open("output2.dat");
- // now simply loop reading numbers from the "input2.dat" file and write them to the "output2.dat" file
- while (infile >> number)
- outfile << number << endl;
- // although not absolutely necessary, it is good programming practice to close any open files before the program terminates...
- infile.close(); // "input2.dat" will be closed
- outfile.close(); // "output2.dat" will be closed

File I/O

- Reading and Writing different data types from and to Text files
- The following program illustrates how different data types may be read from and written to ASCII text files. It makes use of the <string> library as well as the <fstream> library.
- The string library is needed to manipulate string data (a different class from numerical data).

File I/O

- // Author: Ted Obuchowicz
- // Sept.
- // file: file_io_different_data.C
- #include <string>
- #include <fstream> // fstream library defines stream classes ifstream
- // ofstream and fstream

- using namespace std;

File I/O

- int main()
- {
- ifstream infile("myinput_different_data.dat"); // associate the file myinput.dat with infile object of class ifstream

- ofstream outfile("myoutput.dat_different_data.dat"); // associate the file myoutput.dat with the outfile object of class ofstream
- int number;
- string a_string;
- float a_float;

File I/O

- // a simple while loop which reads from the infile
- // and writes them to the outfile object. Loop will terminate
- // when the end of the infile object is reached
- while (infile >> number >> a_string >> a_float)
- outfile << number << a_string << a_float << endl;
- return 0;
- }

File I/O

- The contents of the myinput_different_data.dat file are:
- 123 Hello 3.4
- 2 goodbye 6.456
- -12 keith_richards 999.9
- 99 a 2.3
- When we run the program, the ASCII text file "myoutput.dat_different_data.dat" will
- be created and have as contents :
- 123Hello3.4
- 2goodbye6.456
- -12keith_richards999.9
- 99a2.3

File I/O

- Note how that during the input operation (reading in from the text file and storing the value read into the variables number, a_string, a_float, whitespace characters are skipped over. During the output of the variable, whitespaces are not inserted into the associated output file stream. If they are desired, one would have to explicitly state so as in: outfile << number << " " << a_string << " " << a_float << endl;
- Or use the formatting instructions from the iomanip library.

Next Lecture

- In the next lecture we will continue with the discussion on different file formats.