File Input/Output

Streams

- Streams
  - Special objects
  - Deliver program input and output to/from your program
  - A stream can be thought of as a flow of characters
- An **input stream** flows into your program
  - Can come from keyboard (***cin***)
  - Can come from file
- An **output stream** flows out of your program
  - Can go to screen (***cout***, or ***cerr***)
  - Can go to file

Streams

- Think of `<<` as an operator to move data from your program to an output stream

- Think of `>>` as an operator to move data from an input stream into your program

Other Streams (than *cin*, *cout*)

- The programmer can create other streams
  - Stream can flow into a file
  - Can have multiple streams each flowing into their own file
  - Streams can flow out of files
  - Can have multiple streams each flowing out of its own file

- Programmer-created streams can be used like C++ pre-defined streams such as *cin* or *cout*.

File I/O Libraries

- To allow both file input and output in your program:

```
#include <fstream>
using namespace std;
```

OR

```
#include <fstream>
using std::ifstream;
using std::ofstream;
```

Create a stream and connect to your files

- Stream must be created/declared like any other class variable:

```
ofstream myOutputStream;// a stream to write to
ifstream myInputStream;// a stream to read from
```

- Must then "connect" to file (also called opening the file):

```
myOutputStream.open("myOutfile.txt");
myInputStream.open("myInfile.txt");
```

File Names

- The filenames “myOutfile.txt”, “myInfile.txt” refer to files that exists (for read or write) or will be created (for write) on your computer disk.
  - If input or output files are kept in the same folder as the executable then **myOutfile.txt** and **myInfile.txt** will be the actual names of the files
  - Otherwise, **myOutfile.txt** or **myInfile.txt** must be replaced by the full path of the file, e.g.

    ```
    C:\Users\Me\Desktop\Try\myInfile.txt
    ```

File Names

- Programs and files
- Files have two names to our programs
  - ***External file name***
    - Also called "physical file name"
    - Like “myInfile.txt"
    - Sometimes considered "real file name"
    - Used only once in program (to open)
  - ***Stream name***
    - Also called "logical file name"
    - Program uses this name for all file activity

Reading from / writing to your files

- To write data to your output stream *(eventually the data will go to "myOutfile.txt")*, use `myOutputStream` just like you used `cout`:

```
myOutputStream << variableName;
```

- To read data from your input stream *(originated from the file "myInfile.txt")*, use `myInputStream` just like you used `cin`:

```
myInputStream >> variableName;
```

Breaking the connection

- When the program has completed reading from or writing to the file, the connection to the file is no longer needed:

- The programmer needs to break the connection between the program and the file :

```
myOutputStream.close();

myInputStream.close();
```

- Files automatically close when program ends

File I/O Example 1

- Run the following program using Visual C++. Then go to your project folder to find the file “myfile.txt”, what’s in it?

```
#include <fstream>
using namespace std;

int main( )
{
    ofstream outStream;
    outStream.open("myfile.txt");
    for (int i = 1; i <= 10; ++i)
        outStream << "Hello\t" << i << endl;
    outStream.close();
    return 0;
}
```

File I/O Example 2

```
1   //Reads three numbers from the file infile.txt, sums the numbers,
2   //and writes the sum to the file outfile.txt.
3   #include <fstream>
4   using std::ifstream;
5   using std::ofstream;
6   using std::endl;

7   int main( )
8   {
9       ifstream inStream;
10      ofstream outStream;

11      inStream.open("infile.txt");
12      outStream.open("outfile.txt");

13      int first, second, third;
14      inStream >> first >> second >> third;
15      outStream << "The sum of the first 3\n"
16                << "numbers in infile.txt\n"
17                << "is " << (first + second + third)
18                << endl;
```

File I/O Example 2 (cont)

```
19        inStream.close( );
20        outStream.close( );

21        return 0;
22    }
```

SAMPLE DIALOGUE

There is no output to the screen
and no input from the keyboard.

infile.txt
(Not changed by program)

```
1
2
3
4
```

outfile.txt
(After program is run)

```
The sum of the first 3
numbers in infile.txt
is 6
```

Checking file open success

- File open could fail
  - File doesn't exist, cannot be found
  - No write permissions to output file
  - Disk full, cannot create the new file
  - …
- Need to check the file was actually opened

```
inStream.open("stuff.txt");
if (inStream.fail())
{
      cout << "File open failed.\n";
      exit(1);
}
```

Opening output files

- When you open an output file using the syntax we have discussed
  - You are able to write information to the file
  - The information begin at the beginning of the file
  - If the output file does not exist it will be created
  - If the output file already exists when you open it, then the information you write will overwrite any information already in the file
- What if you want to add information to the end of an existing file? – **appending** information to the file.

Appending to a File

```
outStream.open("myfile.txt", ios::app);
```

- If file doesn't exist it will be created
- If file already exists information will be added to the end of the existing file

- Add the above line to the end of the program on slide #11 to re-open "myfile.txt", then write to the file anything you want to test this "appending" process.