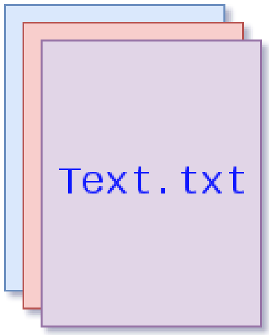
Output Files



Files are able to be written to via a program. Text and data files are what you will encounter in CSI-CSIII. Unlike an input file, an output file does not have to exist before the program begins. The preprocessor directive **#include <fstream>** is required.

Opening a file can be done two ways. The type is **ofstream**. **ofstream variable(“fileName.extension”);** or it can be done on two lines using the member function **open()**:

```
ofstream oFile;
oFile.open(“fileName.extension”);
```

You are able to put the name of the file into a string variable. You may (before C++11) need to use the member function **c_str()** such as

```
string filename = “fileName.ext”;
oFile.open(filename.c_str());
```

It is good practice to immediately check whether or not the file was successfully opened:

```
if(oFile.fail()) {
cout << “File failed to open\n”;
exit(1); }
```

From here, your variable can be used in place of cout to write to the file. For example, if you want to output sentences (strings) to a file, you can **oFile << “This will output into your file!\n”**; The first line of your file will contain what is in quotations and the newline will allow for further input into the file from the beginning of the next line. Other types are able to be written to files, such as **char** and **int** in a similar fashion, such as **int x = 10; oFile << x**;

Files are generally written from top to bottom, left to right. You can append to an existing file by:

```
ofstream oFile;
oFile.open(“fileName.ext”, fstream::out | fstream::app);
```

What this does is specify that the file is for writing (fstream::out) and specify to start writing to the file at the end, or append the file (fstream::app). The | symbol is called the bitwise OR symbol, and it is used to set bit flags (not covered here). If you open an existing file and do not choose to append, whatever you write to the file will overwrite all existing contents.

When finished with the file, invoke the **close()** member function such as **oFile.close();** This ensures the file closed properly.