File streams are the mechanism by which a program variable is associated with a named file listed in a Linux directory or a Windows ® folder.

Line	Statement	Comment
1	*ifstream i*;	Variable *i* has type input file stream. Do this first.
2	*i*.*open*(**"i22.dat"**);	Open file **i22.dat** in the directory or folder that contains this program
3	**if (!***i*) *FileException*(**"i22.dat"**);	Test variable *i* to determine if the prior call to function *i*.*open*() actually opened file **i22.dat**. Variable *i* is a pointer and will have a non-zero value if file **i22.dat** was opened successfully. The value of variable *i* will be zero if file **i22.dat** does not exist. Call function *FileException* if file **i22.dat** could not be opened. The most likely cause of a file exception is file **i22.dat** does not exist in the local directory or folder. Create or copy file **i22.dat** to inhibit the exception.
4	**for (;;) {**	Read all strings for input file stream *i* and display them on separate lines.
5	*string s*;	Declare variable s to provide storage for a single string read from input file stream *i*.
6	*i* >> *s*;	Attempt to read the next string from input file stream *i*. The most likely possibilities are that either a string will be read or the end-of-file marker will be read.
7	**if (***i*.*eof*()) **break;**	Stop when the end-of-file marker is read. This sequence is critical. Read, test, process. Never process an item read from a file prior to testing for the end-of-file marker. The end-of-file marker is ***not***, repeat ***not***, valid data.
8	*cout* << *s* << *endl*;	Print the strings read from the input file stream *i* to the display on separate lines.
9	}	End of the *compound-statement* executed repeatedly under the control of the *for-statement*.
10	*i*.*close*();	Close input file stream *i*.

Table 1. Input file stream operations.

Line	Statement	Comment
1	*ofstream o*;	Variable *o* has type output file stream. Do this first.
2	*o*.*open*(**"o22.dat"**);	Open file **o22.dat** in the directory or folder that contains this program. If file **o22.dat** does not exist, it is created. If file **o22.dat** does exist, it is discarded and the file position marker it set to write at the beginning of the file.
3	**if (!***o*) *FileException*(**"o22.dat"**);	Test variable *o* to determine if the prior call to function *o*.*open*() actually opened file **o22.dat**. Variable *o* is a pointer and will have a non-zero value if file **o22.dat** was opened successfully. The value of variable *o* will be zero if file **o22.dat** does not exist. Call function *FileException* if file **o22.dat** could not be opened.
4	**for (;;) {**	Read all strings for output file stream *o* and display them on separate lines.
5	*string s*;	Declare variable s to provide storage for a single string read from standard input file stream *cin*.
6	*cin* >> *s*;	Attempt to read the next string from input file stream *cin*.

Table 2. Output file stream operations.

Line	Statement	Comment
7	**if (***cin***.***eof***()) break;**	Stop when the end-of-file marker is read. An end-of-file marker can be created from the keyboard by typing a **ctrl-D**.
8	*o* **<<** *s* **<<** *endl***;**	Print the strings read from standard input file stream *cin* to output file stream *o* on separate lines.
9	**}**	End of the *compound-statement* executed repeatedly under the control of the *for-statement*.
10	*o***.***close***();**	Close output file stream *o*. Closing output file stream *o* flushes the memory buffer to disk. Failing to close output file stream o may leave some strings in memory so that not all strings read from *cin* are written to output file stream *o*.

Table 2. Output file stream operations.

```
#include <iostream>
#include <fstream>
using namespace std;
void FileException(char* fn)
{    cout << endl;
     cout << "File " << fn << " does not exist." << endl;
     cout << "Program terminated." << endl;
     exit(EXIT_FAILURE);
}
int main()
{    ifstream i;
     i.open("i22.dat");
     if (!i) FileException("i22.dat");
     ofstream o;
     o.open("o22.dat");
     if (!o) FileException("o22.dat");

     for (;;) {
          string s;
          i >> s;
          if (i.eof()) break;
          o << s << endl;
     }

     o.close();
     i.close();
     return 0;
}
```

Figure 1. Program **p01** illustrates simple input and output file stream operations.

```
Cows are of the bovine ilk.
One end is moo, the other milk.
```

Figure 2. File **i22.dat.**

```
Cows
are
of
the
bovine
ilk.
One
end
is
moo,
the
other
milk.
```

Figure 3. File **o22.dat.**

Standard input object *cin* has type *istream*. It is convenient to treat standard input or any input stream in the same way as an input file stream (*ifstream*). An input file stream (*ifstream*) and an input stream (*istream*) can be passed as reference parameters having type *istream***&**.

Standard output object *cout* has type *ostream*. It is convenient to treat standard output or any output stream in the same way as an output file stream (*ofstream*). An output file stream (*ofstream*) and an output stream (*ostream*) can be passed as reference parameters having type *ostream***&**.

Program **p02** illustrates how standard input *cin* and standard output *cout* can be interchanged to help find errors in a program.

```
#include <iostream>
#include <fstream>
#include <string>
using namespace std;
void FileException(char* fn)
{    cout << endl;
     cout << "File " << fn << " does not exist." << endl;
     cout << "Program terminated." << endl;
     exit(EXIT_FAILURE);
}
void CommandLineException(int max,int actual)
{    cout << endl;
     cout << "Too many arguments on the command line." << endl;
     cout << max << " arguments are permitted." << endl;
     cout << actual << " arguments were entered." << endl;
     cout << "Program terminated." << endl;
     exit(EXIT_FAILURE);
}
```

Figure 4. Program **p02**, part 1.

```
void IOMgr(istream& i,ostream& o)
{
    for (;;) {
        string s;
        i >> s;
        if (i.eof()) break;
        o << s << endl;
    }
}
int main(int argc, char* argv[])
{   char ifn[255];                  //Input File Name
    char ofn[255];                  //Output File Name
    switch (argc) {
        case 1:                     //Prompt for both file names
            cout << "Enter the input file name. ";
            cin >> ifn;
            cout << "Enter the output file name. ";
            cin >> ofn;
            break;
        case 2:                     //Prompt for the output file name.
            strcpy(ifn,argv[1]);    //Copy the input file name from argv[1]
            cout << "Enter the output file name. ";
            cin >> ofn;
            break;
        case 3:                     //Copy both file names from argv
            strcpy(ifn,argv[1]);
            strcpy(ofn,argv[2]);
            break;
        default:                    //Too many arguments on the command line
            CommandLineException(2,argc-1);
            break;
    }
    ifstream i(ifn); if (!i) FileException(ifn);
    ofstream o(ofn); if (!o) FileException(ofn);

    IOMgr(cin,cout);
    IOMgr(i,o);

    o.close();
    i.close();
    return 0;
}
```

Figure 4. Program **p02**, part 2