DISPLAY THE CONTENTS OF A FILE AND DETECTING END OF FILE

2.4.3 Programs in C++ to Display the contents of a File

The first simple file processing program opens a file for input and reads it, character by character, sending each character to the screen after it is read from the file. This program includes the following steps

1. Display a prompt for the name of the input file.
2. Read the user's response from the keyboard into a variable called filename.
3. Open the file for input.
4. While there are still characters to be read from the input file,
   - Read a character from the file;
   - Write the character to the terminal screen.
5. Close the input file.

Figures 2.2 and 2.3 are C++ implementations of this program using C streams and C++ stream classes, respectively.

```
// listc.cpp
// program using C streams to read characters from a file
// and write them to the terminal screen
#include <stdio.h>
main( ) {
   char ch;
   FILE * file; // pointer to file descriptor
   char filename[20];
   printf("Enter the name of the file: ");     // Step 1
   gets(filename);                             // Step 2
   file =fopen(filename, "r");                 // Step 3
   while (fread(&ch, 1, 1, file) != 0)         // Step 4a
     fwrite(&ch, 1, 1, stdout);                // Step 4b
   fclose(file);                               // Step 5
}
```

Figure 2.2 The file listing program using C streams (`listc.cpp`).

```
// listcpp.cpp
// list contents of file using C++ stream classes
#include <fstream.h>
main () {
  char ch;
  fstream file; // declare unattached fstream
  char filename[20];
  cout <<"Enter the name of the file: " // Step 1
    <<flush; // force output
  cin >> filename;                      // Step 2
  file . open(filename, ios::in);       // Step 3
  file . unsetf(ios::skipws);// include white space in read
  while (1)
  {
    file >> ch;                         // Step 4a
    if (file.fail()) break;
    cout << ch;                         // Step 4b
  }
  file . close();                       // Step 5
}
```

Figure 2.3 The file listing program using C++ stream classes (`listcpp.cpp`).

In the C++ version, the call file.unsetf(ios::skipws) causes operator >> to include white space (blanks, end-of-line,tabs, ans so on).

2.4.4 Detecting End of File

end-of-file

A physical location just beyond the last datum in a file.

- The acronym for end-of-file is EOF.
- When a file reaches EOF, no more data can be read.
- Data can be written at or past EOF.
- Some access methods set the end of file flage after a read reaches the end of file position.

Other access methods set the end of file flag after a read attempts to read beyond the end of file position.

Detecting End of File

The C++ *feof* function is used to detect when the file pointer of an fstream is **past** end of file..

The FILE *feof* function has one argument.

- A pointer to the FILE structure of the logical file

The value returned by the *feof* function is 1 if end of file is true and 0 if end of file is false.

Prototypes:

```
int feof (FILE * Stream);
```

Example:

```
if (feof (Input))
cout << "End of File\n";
```

In some languages, a function end_of_file can be used to test for end-of-file. The OS keeps track of read/write pointer. The end_of_file function queries the system to see whether the read/write pointer has moved past the last element in the file.