[Skip to main content](https://courses.iitbombayx.in/courses/course-v1:IITBombayX+CS101.2x+2022_T1/courseware/a83378a1759347a8a8eb84e3dfcdefcb/56e137a8d36d4fc1a48c6072273b3e13/?child=first" \l "main)

**[IITBombayX Home Page](http://iitbombayx.in/)**

**IITBombayX: CS101.2xObject-Oriented Programming**

[Courses](http://iitbombayx.in/courses)

[Help](https://iitbombayx.in/contact_us)

[[https://courses.iitbombayx.in/static/images/profiles/default_50.3455a6581573.png](https://courses.iitbombayx.in/dashboard) Dashboard for:Paramhans\_Singh](https://courses.iitbombayx.in/dashboard)

1. [Course, current location](https://courses.iitbombayx.in/courses/course-v1:IITBombayX+CS101.2x+2022_T1/course/)

1. [Discussion](https://courses.iitbombayx.in/courses/course-v1:IITBombayX+CS101.2x+2022_T1/discussion/forum/)

1. [Progress](https://courses.iitbombayx.in/courses/course-v1:IITBombayX+CS101.2x+2022_T1/progress)

1. [Grading Policy](https://courses.iitbombayx.in/courses/course-v1:IITBombayX+CS101.2x+2022_T1/487375f5363a49a492641b63d5e814ec/)

1. [Course Slides](https://courses.iitbombayx.in/courses/course-v1:IITBombayX+CS101.2x+2022_T1/pdfbook/0/)

1. [Practice Programming](https://courses.iitbombayx.in/courses/course-v1:IITBombayX+CS101.2x+2022_T1/b5c2b3240ef9416ebb0ea5b2087cb4ee/)

1. [FAQs](https://courses.iitbombayx.in/courses/course-v1:IITBombayX+CS101.2x+2022_T1/f2e9c32b35ae4ef083ed716b4981ff0d/)

[Course](https://courses.iitbombayx.in/courses/course-v1:IITBombayX+CS101.2x+2022_T1/course/)  [Topic 3: Review of File Processing](https://courses.iitbombayx.in/courses/course-v1:IITBombayX+CS101.2x+2022_T1/course/#block-v1:IITBombayX+CS101.2x+2022_T1+type@chapter+block@a83378a1759347a8a8eb84e3dfcdefcb)  [Session 17: Handling Data in Text Files](https://courses.iitbombayx.in/courses/course-v1:IITBombayX+CS101.2x+2022_T1/course/#block-v1:IITBombayX+CS101.2x+2022_T1+type@sequential+block@56e137a8d36d4fc1a48c6072273b3e13)  Walkthrough

 PreviousNext

1. video Handling Data in Text Files
2. other Slides: Handling Data in Text Files
3. other Walkthrough
4. other Walkthrough

**Walkthrough**

 Bookmark this page

**Highlights of File Processing in C++**

**introduction**

This walkthrough will provide a summary of file handling, provide some more methods to iterate over a file and discuss some corner cases in this process. We will discuss all procedures in reference to file reading operations instead of writing to make things easy. The write operations are similar to read. So let's get going...

**GIST of file operations**

By now you must have been familiar with how to open and close files with the help of fopen and fclose operations. These functions return a FILE pointer which can be used to perform read operations. This pointer is used in a while loop reading/writing bytes ( via fread) till you reach the end of the file. *End of File(EOF)  is*a tricky concept which we will see in the next section.

**FILE Pointer and EOF**

* **File Pointer**

Many people would assume that since we are reading from FILE\* the pointer advances to a next byte to be read. Well, this is not true. This pointer never changes till the file is closed. FILE is a complicated class in the standard library **cstdio**. It contains many variables like   \_IO\_read\_ptr, \_IO\_read\_end, \_IO\_read\_base (and similar variables for write part) that help in implementing the functionality of fread() and fwrite() function calls. After a read call, it is not the FILE point that gets updated but the  \_IO\_read\_ptr.

* **EOF**

This read call mentioned above advances the  \_IO\_read\_ptr to a position just after the point it encounters either a **\n** or **EOF**.  *Just after EOF*  means the pointer returns back to start position of the file. The read function sets the ***EOF Indicator*** to TRUE (non zero) whenever it is stopped by EOF and not (*\n*). It is the status of this variable that we check as a condition of while loop.

The function feof(FILE\* fp) checks if while reading, the file has the end of file indicator set.

**BASIC EXAmple**

    FILE \*fp\_input = fopen(filename, "r" ); // filename -> char\* of name of file

    if (fp\_input ==  NULL) {

        printf("Error opening file %s\n", filename);

        return false;

    }

    fgets(string\_name, number\_of\_characters, fip\_input);

    while (!feof(fp\_input){

        // Do read/write

        fgets(string\_name, number\_of\_characters, fip\_input);

    }

**alternatives**

Well, this EOF checking is not the only way to read the files. Some read functions like scanf return integer EOF if EOF is reached. This can also be used as a while loop condition. For example, the function fscanf(FILE \*stream, const char \*format, ...) can be used instead of fread to read formatted input from a file.

 while(fscanf (fp\_input, "%s", string\_name) != EOF){

   // Use string\_name read from above fucntion

}

There are much more ways to read a file in C++. Check [this](https://www.uow.edu.au/~lukes/TEXTBOOK/notes-cpp/io/readtextfile.html) link for more details. It discusses a variant of std::cin to read from a file.

**Corner cases**

One has to be careful in reading files especially the last line of the file. This fact can be illustrated by the case of iterating the file using feof() function in while loop condition (as described above). The function described above takes input for next line at end of while loop. There is a corner case missed in such format.

Suppose the file is such that at the end of the last line there is no *\n* character but directly EOF is reached, then by this mechanism that line will not be processed because fgets() will raise EOF flag as soon as the last line is read. This means loop will break in the next statement before processing this line. If that file's last line contains *\n* at the end then fgets() will be stopped by *\n* and process last line, then get feof() to return true after processing this last line and thus exit the loop safely.

Such cases can be easily solved by paying attention to such subtle details.

**CONCLUSION**

From this discussion, we have gained a better insight into the understanding of file processing in C++. We explored alternate and better ways to iterate over files and saw an example of corner case scenario and how to avoid such mistakes.

 PreviousNext

© All Rights Reserved

[IITBombayX Home Page](http://iitbombayx.in/)

1. [About](http://iitbombayx.in/about)
2. [Research](http://iitbombayx.in/research)
3. [FAQs](http://iitbombayx.in/faq)
4. [Contact Us](http://iitbombayx.in/contact_us)

* [MOOCs Copyright](http://iitbombayx.in/mooc-copyright-guidelines)
* [Terms of Service and Honour Code](http://iitbombayx.in/honor)
* [Authorization Document](http://iitbombayx.in/authorization)
* [Privacy Policy](http://iitbombayx.in/privacy)

[Powered by Drupal](https://www.drupal.org/)

[Creative Common License](https://creativecommons.org/)

[](http://open.edx.org/)