



Reading from a File



Revision: Writing in File

Storing output in the file is a **five-step** process:

1. Include the header file `fstream` in the program.
2. Declare file stream variables.
3. Associate the file stream variables with the text file and define the opening mode.
4. Use the file stream variables insertion operator `<<` (to store output in the file i.e., write in the file)
5. Close the file.

Revision

Write a C++ Program to store "Welcome to UET" in a text file.

```
#include <fstream>
using namespace std;
main()
{
    string line = "Welcome to UET";
    fstream newFile;
    newFile.open("TextFile.txt", ios::out);
    newFile << line;
    newFile.close();
}
```

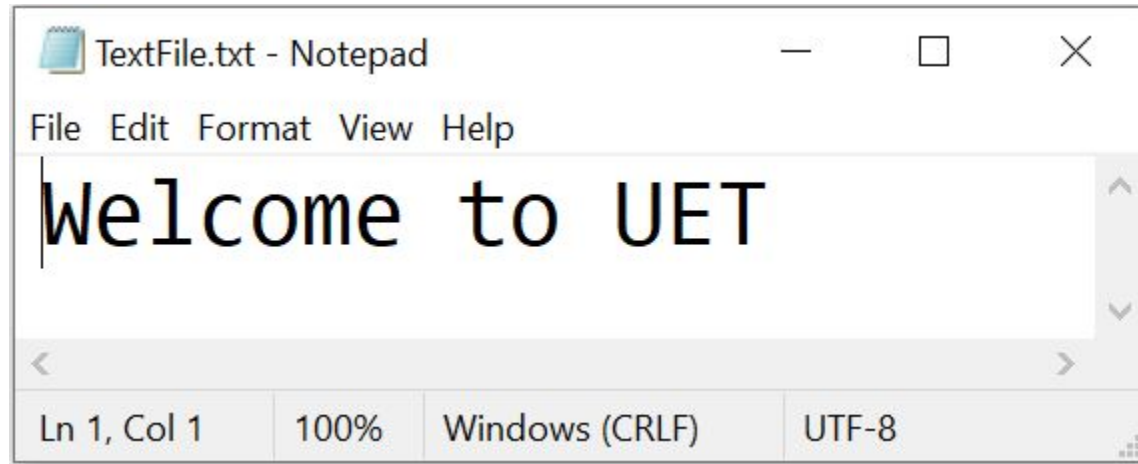
File Stream: Reading from a File

Taking input from the file is also a five-step process:

1. Include the header file `fstream` in the program.
2. Declare file stream variables.
3. Associate the file stream variables with the text file and define the opening mode.
4. Use the file stream variables with extraction operator `>>` (to take input from the file i.e., read the file).
5. Close the file.

Working Example

Write a C++ Program to read "Welcome to UET" from a text file and show it on the console.



Working Example

Step 1: Include the header file `fstream` in the program.




```
#include<iostream>
#include <fstream>
using namespace std;
main()
{

}
}
```

Working Example


Step 2: Declare file stream variable to open the file.



```
#include<iostream>
#include <fstream>
using namespace std;
main()
{
    string line;
    fstream newFile;
}
```

Working Example

Step 3: Associate the file stream variables with the text file and define the opening mode.




```
#include<iostream>
#include <fstream>
using namespace std;
main()
{
    string line;
    fstream newFile;
    newFile.open("TextFile.txt", ios::in);
}
```


Working Example

Step 4: Use the file stream variables extraction operator >> (to read from the file).


```
#include<iostream>
#include <fstream>
using namespace std;
main()
{
    string line;
    fstream newFile;
    newFile.open("TextFile.txt", ios::in);
    newFile >> line;

}
```



Working Example

Step 5: Close the file.



```
#include<iostream>
#include <fstream>
using namespace std;
main()
{
    string line;
    fstream newFile;
    newFile.open("TextFile.txt", ios::in);
    newFile >> line;
    newFile.close();
}
```

Working Example

Display output on the Console.

```
#include<iostream>
#include <fstream>
using namespace std;
main()
{
    string line;
    fstream newFile;
    newFile.open("TextFile.txt", ios::in);
    newFile >> line;
    newFile.close();
    cout << line;
}
```

Working Example: Output

Output on the Console is:

```
C:\C++>c++ program.cpp -o program.exe
```

```
C:\C++>program.exe
```

```
Welcome
```

```
C:\C++>
```

Working Example: Output

Can anyone tell why we only got "Welcome" instead of "Welcome to UET"?

```
C:\C++>c++ program.cpp -o program.exe
```

```
C:\C++>program.exe
```

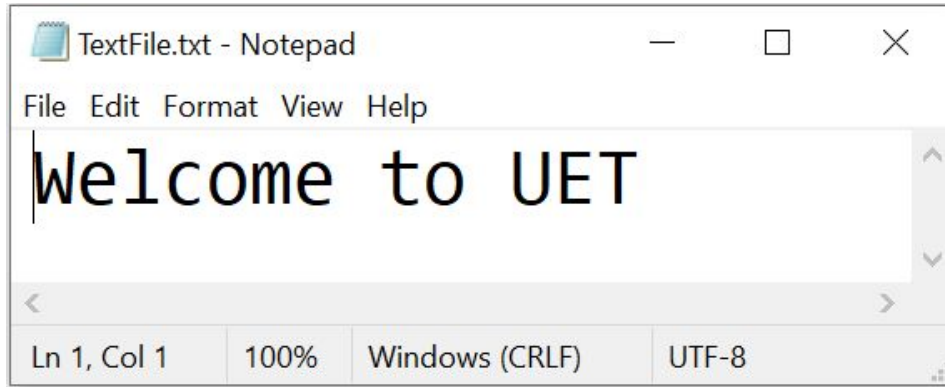

```
Welcome
```

```
C:\C++>
```

Working Example: Output


It reads only **1 word** from the file until cursor reaches a blank space. Following program will only print "**Welcome**"

```
#include<iostream>
#include <fstream>
using namespace std;
main()
{
    string line;
    fstream newFile;
    newFile.open("TextFile.txt", ios::in);
    newFile >> line;
    newFile.close();
    cout << line;
}
```



Working Example: Output


What we have to do to read the complete line?



```
#include<iostream>
#include <fstream>
using namespace std;
main()
{
    string line;
    fstream newFile;
    newFile.open("TextFile.txt", ios::in);
    newFile >> line;
    newFile.close();
    cout << line;
}
```

Working Example: Output

We have to use `getline(file_variable, string_variable)` function.

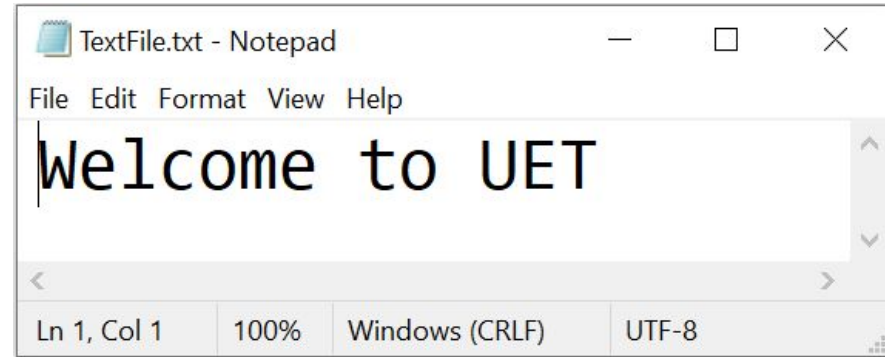



```
#include<iostream>
#include <fstream>
using namespace std;
main()
{
    string line;
    fstream newFile;
    newFile.open("TextFile.txt", ios::in);
    newFile >> line;
    newFile.close();
    cout << line;
}
```


Working Example: Output

Following code will print "Welcome to UET" on the screen

```
#include<iostream>
#include <fstream>
using namespace std;
main()
{
    string line;
    fstream newFile;
    newFile.open("TextFile.txt", ios::in);
    getline(newFile, line);
    newFile.close();
    cout << line;
}
```






Working Example: Output

Output on the Console is:

```
C:\C++>c++ program.cpp -o program.exe  
  
C:\C++>program.exe  
Welcome to UET  
C:\C++>
```

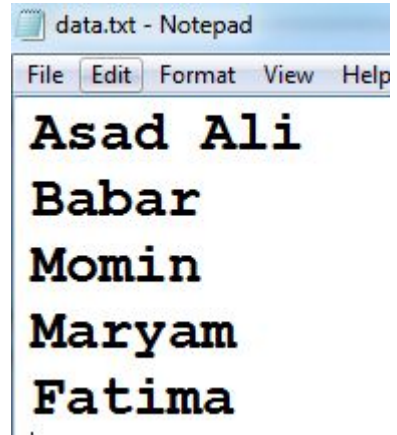
File in the Same Directory

Important thing to note here is that the **.txt** file we are reading is in the **same directory** as that of our **.cpp** file.

 program.cpp	12/9/2021 8:38 PM	CPP File	3 KB
 program.exe	12/9/2021 8:38 PM	Application	50 KB
 TextFile.txt	1/7/2022 11:35 AM	Text Document	1 KB

Reading all the contents from File

How to read all the names from the file and print on the console?



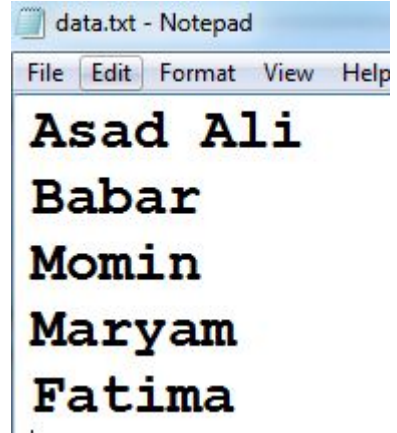
```
data.txt - Notepad
File Edit Format View Help
Asad Ali
Babar
Momin
Maryam
Fatima
```

Reading all the contents from File

Now, we want to read the file until it reaches the end of file.

For that we have a function `eof()`.

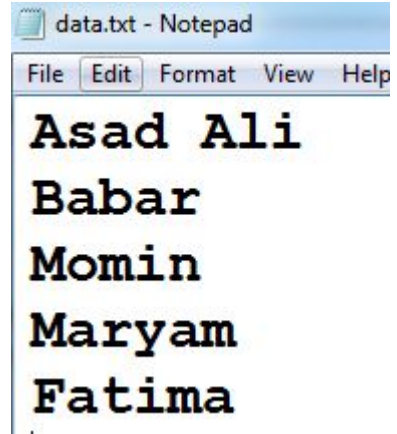
Syntax to use that function is:
`file_variable.eof()`



Reading all the contents from File

How to read all the names from the file and print on the console?

```
fstream file;  
file.open("data.txt", ios::in);  
string line;  
while (!file.eof())  
{  
    getline(file, line);  
    cout << line << endl;  
}  
file.close();
```



Learning Objective

Write **C++ Program** that reads permanently stored data from the text file.



Conclusion

- Taking input from the file is also a **five-step** process:
 1. Include the header file `fstream` in the program.
 2. Declare file stream variables.
 3. Associate the file stream variables with the text file and define the opening mode.
 4. Use the file stream variables with extraction operator `>>` (to take input from the file i.e., read the file).
 5. Close the file.

Conclusion

- `file_variable >>` only reads a single word from the file.
- `getline(file_variable, string_variable)` is used to read a complete line from the user.
- `file_variable.eof()` returns non-zero (true) when the cursor reaches end of the file, otherwise it returns zero.

Self Assessment

1. Develop a **SigIn** and **SignUp** Application using File System.

- As a user, when I signup to the system the username and password stores into the file.
- As a user, when I signin to the system the username and password is verified from the file.



Self Assessment

2. A file "**students.txt**" has data stored like this on separate lines

Admission_number

Name

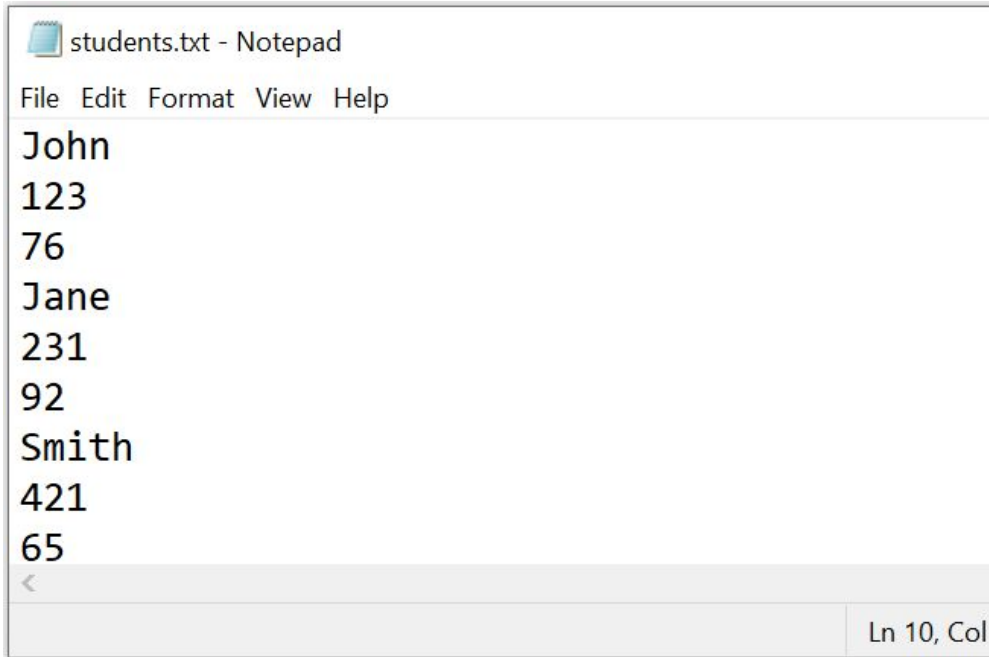
Percentage.

Write a function **read_rec()** in c++ that reads contents of the file "student.txt" and then write a function **write_rec()** that stores the details of those students whose percentage is above 75 in descending order in a separate file named "**topperStudents.txt**".



Self Assessment

Input



A screenshot of a Notepad window titled "students.txt - Notepad". The window has a menu bar with "File", "Edit", "Format", "View", and "Help". The text content is as follows:

```
John  
123  
76  
Jane  
231  
92  
Smith  
421  
65
```

The status bar at the bottom right indicates "Ln 10, Col 1".



Self Assessment

Output

```
topperStudents.txt - Notepad
File Edit Format View Help
Name: Jane
Admission Number: 231
Percentage: 92

Name: John
Admission Number: 123
Percentage: 76

Ln 9, Col 1
```

