File Input and Output

黃世強 (Sai-Keung Wong)

Writing Data to a File

```
#include <iostream>
#include <fstream>
using namespace std;
int main() {
      ofstream output;                  // Create a file object
      output.open("scores.txt");    // Open or create a file for writing
      output << "Mary" << " " << "L" << " " << "Smith" << " " << 99 << endl;
      output << "John" << " " << "P" << " " << "Jones" << " " << 59 << endl;
      output.close();
      cout << "Finished..." << endl;
      return 0;
}
```

Writing Data to a File

We can use it to write the following items to a text file:

- primitive data type values,
- arrays
- strings
- objects

```
#include <iostream>
#include <fstream>
using namespace std;
int main() {
      ofstream output;                  // Create a file object
      output.open("scores.txt");        // Open or create a file for writing
      output << "Mary" << " " << "L" << " " << "Smith" << " " << 99 << endl;
      output << "John" << " " << "P" << " " << "Jones" << " " << 59 << endl;
      output.close();
      cout << "Finished..." << endl;
      return 0;
}
```

Writing Data to a File

```
#include <iostream>
#include <fstream>
using namespace std;
int main() {
      ofstream output;                    // Create a file object
      output.open("scores.txt");          // Open or create a file for writing
      output << "Mary" << " " << "L" << " " << "Smith" << " " << 99 << endl;
      output << "John" << " " << "P" << " " << "Jones" << " " << 59 << endl;
      output.close();
      cout << "Finished..." << endl;
      return 0;
}
```

Writing Data to a File

```
ofstream output;                          // Create a file object
output.open("scores.txt");                // Open or create a file for writing
output << "Mary" << " " << "L" << " " << "Smith" << " " << 99 << endl;
output << "John" << " " << "P" << " " << "Jones" << " " << 59 << endl;
output.close();
cout << "Finished..." << endl;
return 0;
```

Writing Data to a File

```
ofstream output;                    // Create a file object
output.open("scores.txt");          // Open or create a file for writing
output << "Mary" << " " << "L" << " " << "Smith" << " " << 99 << endl;
output << "John" << " " << "P" << " " << "Jones" << " " << 59 << endl;
output.close();
```

ofstream class

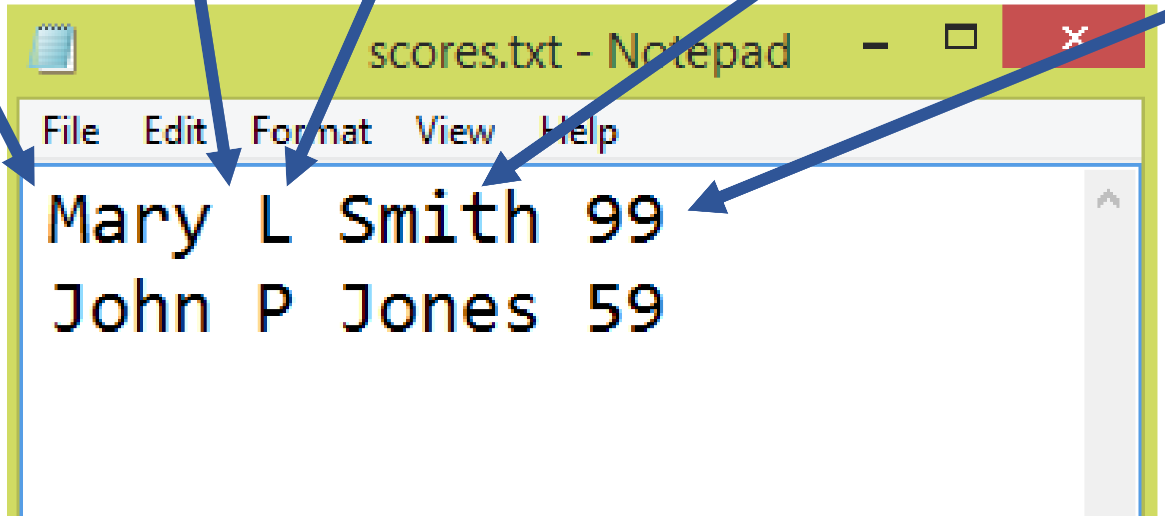
```
ofstream output;                     // Create a file object
output.open("scores.txt");           // Open or create a file for writing
output << "Mary" << " " << "L" << " " << "Smith" << " " << 99 << endl;









output << "John" << " " << "P" << " " << "Jones" << " " << 59 << endl;
output.close();
```

ofstream class

```
ofstream output;                       // Create a file object
output.open("scores.txt");             // Open or create a file for writing
output << "Mary" << " " << "L" << " " << "Smith" << " " << 99 << endl;












output << "John" << " " << "P" << " " << "Jones" << " " << 59 << endl;
output.close();
```

ofstream class

```
ofstream output;                    // Create a file object
output.open("scores.txt");          // Open or create a file for writing
output << "Mary" << " " << "L" << " " << "Smith" << " " << 99 << endl;
```

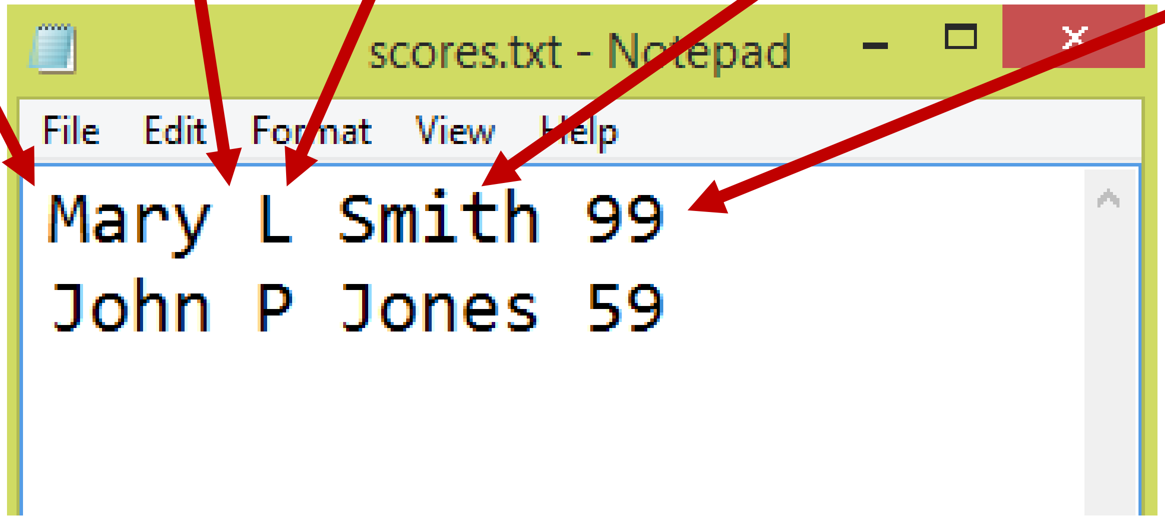


```
Mary L Smith 99
John P Jones 59
```

```
output << "John" << " " << "P" << " " << "Jones" << " " << 59 << endl;
output.close();
```

ofstream class

```
ofstream output;                    // Create a file object
output.open("scores.txt");          // Open or create a file for writing
output << "Mary" << " " << "L" << " " << "Smith" << " " << 99 << endl;
```

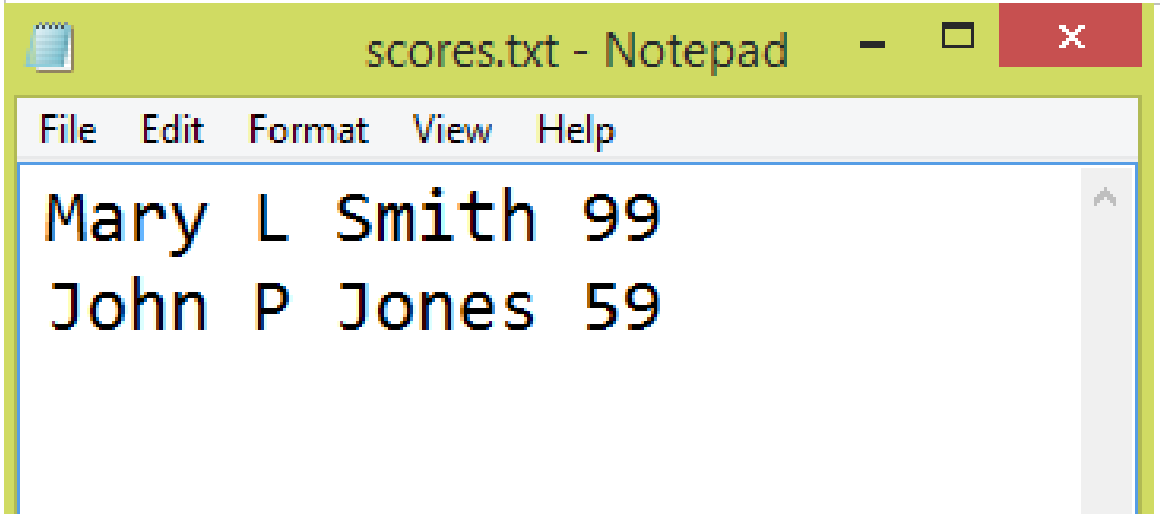


```
Mary L Smith 99
John P Jones 59
```

```
output << "John" << " " << "P" << " " << "Jones" << " " << 59 << endl;
output.close();
```

ofstream class

```
ofstream output;                   // Create a file object
output.open("scores.txt");         // Open or create a file for writing
output << "Mary" << " " << "L" << " " << "Smith" << " " << 99 << endl;
output << "John" << " " << "P" << " " << "Jones" << " " << 59 << endl;
output.close();
```



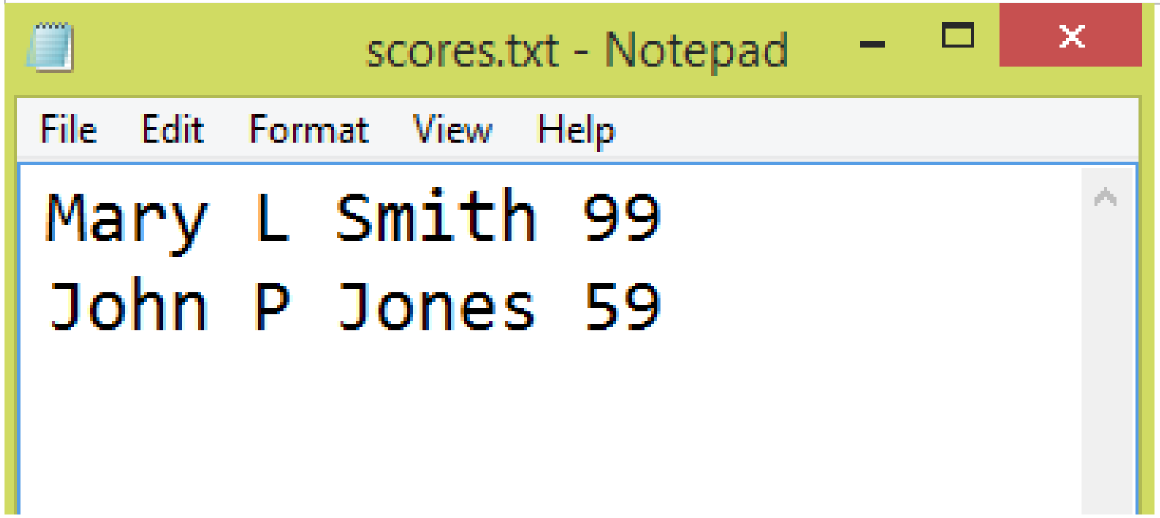
```
Mary L Smith 99
John P Jones 59
```

Could you figure out how the bytes of data are ordered in the file?

A datum or data are represented as a series of bytes.

ofstream class

```
ofstream output;                // Create a file object
output.open("scores.txt");     // Open or create a file for writing
output << "Mary" << " " << "L" << " " << "Smith" << " " << 99 << endl;
output << "John" << " " << "P" << " " << "Jones" << " " << 59 << endl;
output.close();
```



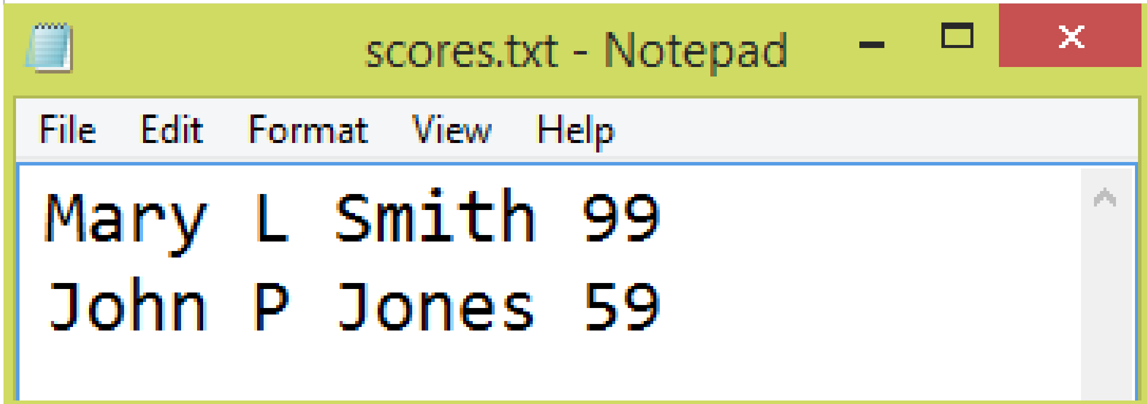
```
Mary L Smith 99
John P Jones 59
```

Could you figure out how the bytes of data are ordered in the file?

A datum or data are represented as a series of bytes.

ofstream class

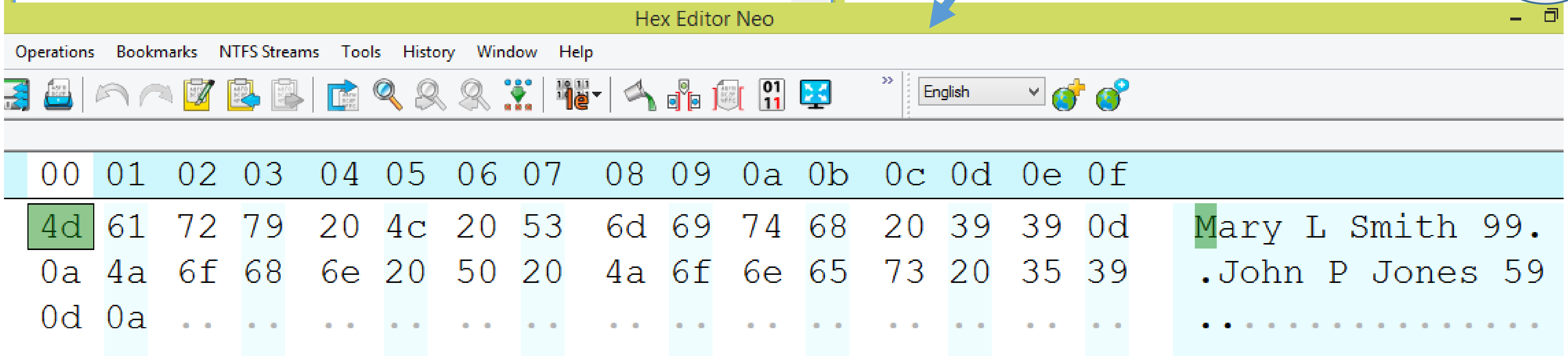
```
ofstream output;                    // Create a file object
output.open("scores.txt");          // Open or create a file for writing
output << "Mary" << " " << "L" << " " << "Smith" << " " << 99 << endl;
output << "John" << " " << "P" << " " << "Jones" << " " << 59 << endl;
output.close();
```



```
Mary L Smith 99
John P Jones 59
```

View the file content in Hex Editor Neo

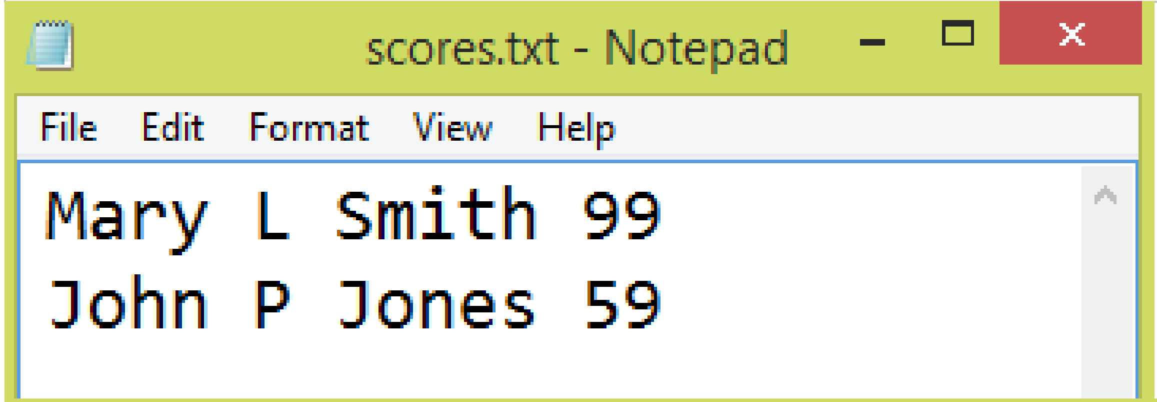
Could you figure out how the bytes of data are ordered?



```
00 01 02 03  04 05 06 07  08 09 0a 0b  0c 0d 0e 0f
4d 61 72 79  20 4c 20 53  6d 69 74 68  20 39 39 0d   Mary L Smith 99.
0a 4a 6f 68  6e 20 50 20  4a 6f 6e 65  73 20 35 39   .John P Jones 59
0d 0a .. ..  .. .. .. ..  .. .. .. ..  .. .. .. ..   ................
```

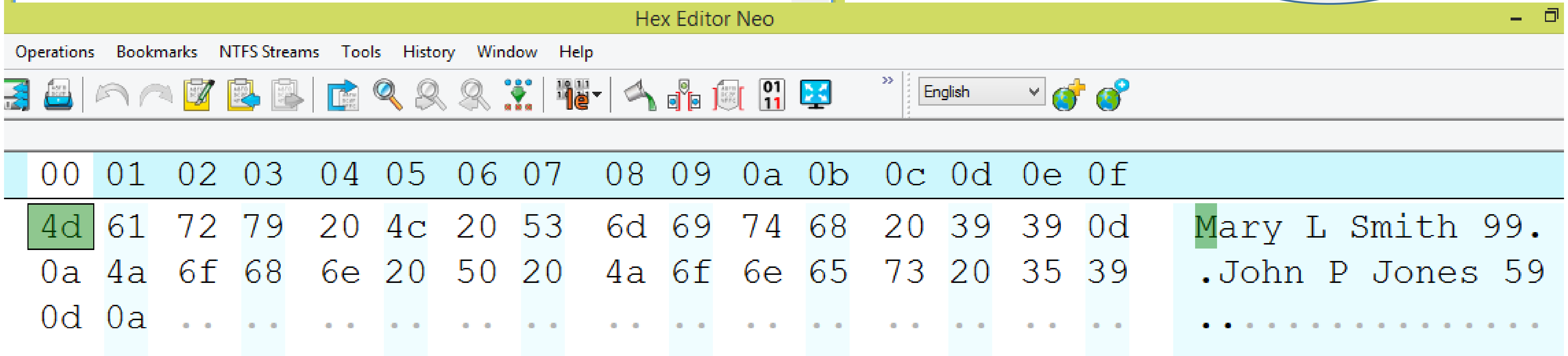
ofstream class

```
ofstream output;                // Create a file object
output.open("scores.txt");      // Open or create a file for writing
output << "Mary" << " " << "L" << " " << "Smith" << " " << 99 << endl;
output << "John" << " " << "P" << " " << "Jones" << " " << 59 << endl;
output.close();
```



```
Mary L Smith 99
John P Jones 59
```

We can read the content of the file directly because the data are stored as text (ASCII code).



```
00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f
4d 61 72 79 20 4c 20 53 6d 69 74 68 20 39 39 0d   Mary L Smith 99.
0a 4a 6f 68 6e 20 50 20 4a 6f 6e 65 73 20 35 39   .John P Jones 59
0d 0a .. .. .. .. .. .. .. .. .. .. .. .. .. ..   ................
```

Close file

We must use the close() function to close the stream for the object.

If we do not invoke the close function, the **data** may **not be saved** in the file.

```
ofstream output;                  // Create a file object
output.open("scores.txt");        // Open or create a file for writing
output << "Mary" << " " << "L" << " " << "Smith" << " " << 99 << endl;
output << "John" << " " << "P" << " " << "Jones" << " " << 59 << endl;
output.close();
```

Close file

We must use the close() function to close the stream for the object.

If we do not invoke the A1 function, the **data** may **not** A2 in the file.

```
ofstream output;                  // Create a file object
output.open("scores.txt");        // Open or create a file for writing
output << "Mary" << " " << "L" << " " << "Smith" << " " << 99 << endl;
output << "John" << " " << "P" << " " << "Jones" << " " << 59 << endl;
output.close();
```

`output.open("scores.txt");`

- When we use *open* to open a file, we must be careful.
- If the file already exists, its contents will be destroyed without warning.

```
ofstream output;                     // Create a file object
output.open("scores.txt");           // Open or create a file for writing
output << "Mary" << " " << "L" << " " << "Smith" << " " << 99 << endl;
output << "John" << " " << "P" << " " << "Jones" << " " << 59 << endl;
output.close();
```

output.open("scores.txt");

- When we use *open* to open a file, we must be careful.

- If the file A1, its contents will be A2 without warning.

```
ofstream output;                  // Create a file object
output.open("scores.txt");        // Open or create a file for writing
output << "Mary" << " " << "L" << " " << "Smith" << " " << 99 << endl;
output << "John" << " " << "P" << " " << "Jones" << " " << 59 << endl;
output.close();
```

Reading Data from a File

- We use the ifstream class to read data from a text file.

```
void printf_info( ) {
       cout       << firstName << " " << mi << " "
                  << lastName << " " << score << endl;
}
ifstream input("scores.txt");  // open a file to read data
char mi;  string firstName, lastName; int score;

int flg = input.fail( );
if ( flg ) { return; //cannot open }

input >> firstName >> mi >> lastName >> score;
printf_info( );

input >> firstName >> mi >> lastName >> score;
printf_info( );

input.close();
```

File content:
Mary L Smith 99
John P Jones 59

Testing File Existence

```
ifstream input("myscores.txt"); // Read data
char mi;  string lastName; int score;
try {
    int flg = input.fail( );
     if ( flg ) { throw flg; }
     ……

   input.close();
} catch( int flg ) {
   cout << "Error. Cannot open file. Flag:" << flg << endl;
}
```

Known data format

To read data correctly, you need to know exactly how data is stored. For example, the program would not work if the score is a double value with a decimal point.

```
ifstream input("scores.txt"); // Read data
char mi;  string lastName; int score;
try {
    int flg = input.fail( );
   if ( flg ) { return; //cannot open}
   input >> firstName >> mi >> lastName >> score;
    ......
    input.close();
}
```

File content:
John T Smith 90
Eric K Jones 14

eof

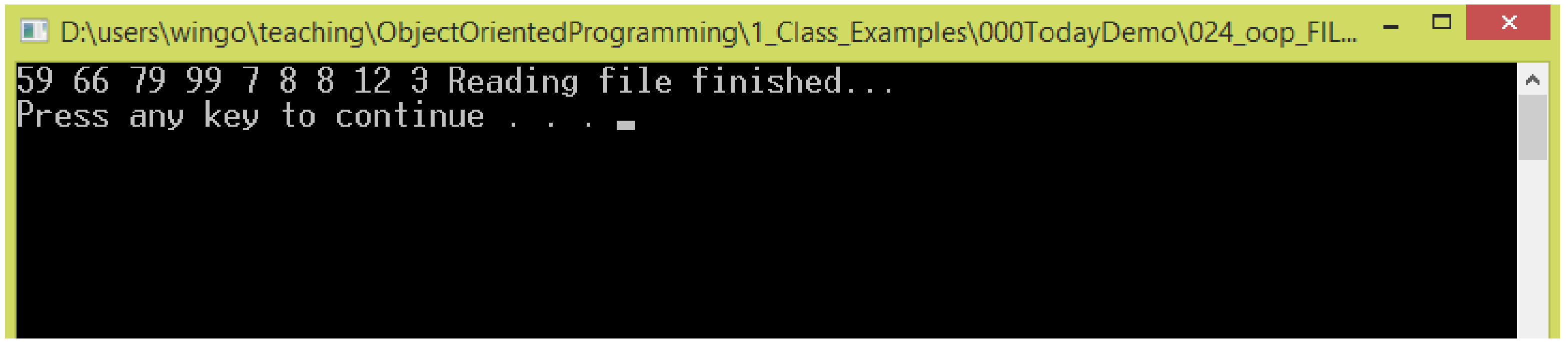
- We invoke the eof() function on the input object to detect the end of file.

```
double sum = 0; double number;
while (input >> number)        // read to the end of file
{
     cout << number << " ";
     sum += number;
}

Note: bool flg = input >> number
flg is true if input reads a number successfully
```

eof

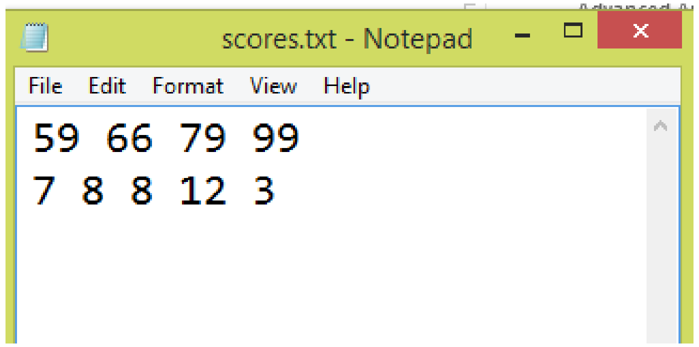
- We invoke the eof() function on the input object to detect the end of file.

```
double sum = 0; double number;
while (input >> number)        // read to the end of file
{
     cout << number << " ";
     sum += number;
}
```



D:\users\wingo\teaching\ObjectOrientedProgramming\1_Class_Examples\000TodayDemo\024_oop_FIL...

```
59 66 79 99 7 8 8 12 3 Reading file finished...
Press any key to continue . . .
```



scores.txt - Notepad

```
59 66 79 99
7 8 8 12 3
```

Functions for reading and writing data

- The stream extraction operator (>>) can read a text that does not contain a whitespace character.

- To read a text that has whitespace characters, we can use getline.

getline(ifstream& input, string &s, **char** delimiter)

getline with delimiter: ' '

File content:
Mary L Smith 99
John P Jones 59

```
ifstream input;
input.open("scores.txt");
while(!input.eof()) {
    string str;
    getline(input, str, ' ');
    cout << "str:" << str << endl;
}
```

```
str:Mary
str:L
str:Smith
str:99
John
str:P
str:Jones
str:59
```
;a newline

getline with delimiter: `'\n'`

File content:
Mary L Smith 99
John P Jones 59

```
ifstream input;
char delimiter = '\n';
input.open("scores.txt");
while(true) {
    string str;
    getline(input, str, delimiter);

    cout << "str:" << str << endl;
}
```

```
str:Mary L Smith 99
str:John P Jones 59
str:
```

problem!

getline with delimiter: `'\n'`

File content:
Mary L Smith 99
John P Jones 59

```
ifstream input;
char delimiter = '\n';
input.open("scores.txt");
while(true) {
    string str;
    getline(input, str, delimiter);
    if ( input.eof() ) break;
    cout << "str:" << str << endl;
}
```

```
str:Mary L Smith 99
str:John P Jones 59
```

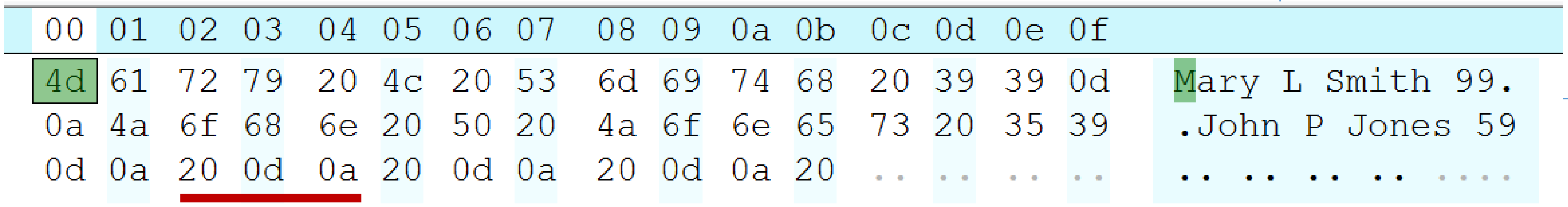
getline with delimiter: '\n'

```
ifstream input;
char delimiter = '\n';
input.open("scores.txt");
while(true) {
     string str;
    getline(input, str, delimiter);
    if ( input.eof() ) break;
    cout << "str:" << str << endl;
}
```

Three lines with a whitespace character

File content:
Mary L Smith 99
John P Jones 59

```
str:Mary L Smith 99
str:John P Jones 59
str:
str:
str:
```



```
00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f
4d 61 72 79 20 4c 20 53 6d 69 74 68 20 39 39 0d   Mary L Smith 99.
0a 4a 6f 68 6e 20 50 20 4a 6f 6e 65 73 20 35 39   .John P Jones 59
0d 0a 20 0d 0a 20 0d 0a 20 0d 0a 20 .. .. .. ..   .. .. .. .. ....
```

0x20: whitespace character in ASCII code

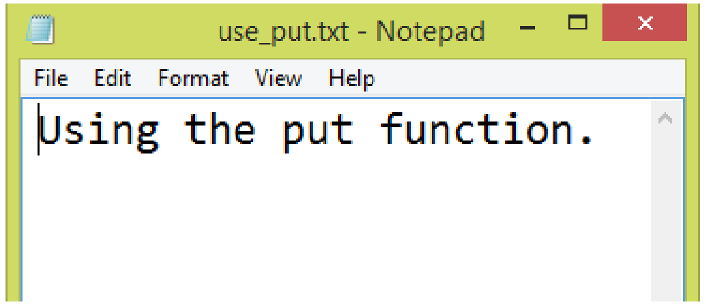
get and put

Invoke the **get** function on **an input object** to read a character.

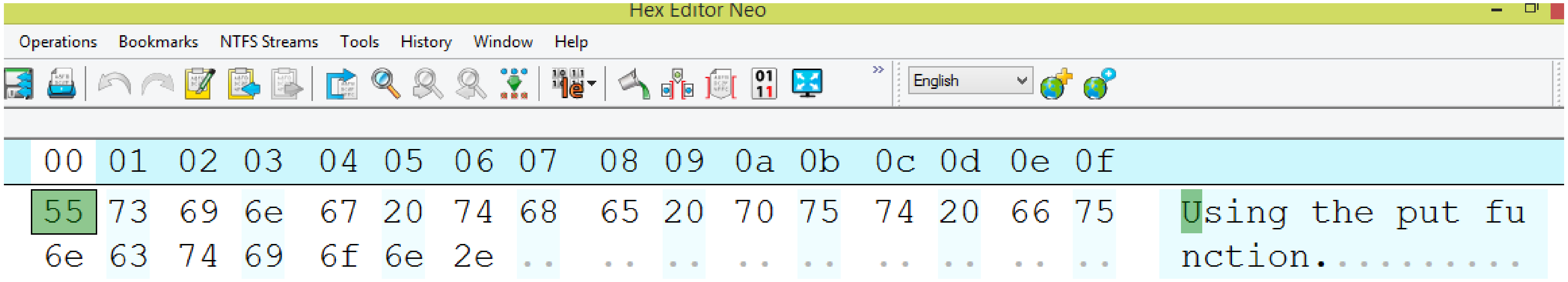
Invoke the **put** function on **an output object** to write a character.

The put function

```
void saveFile_UsingPut( ) {
    ofstream output;
    output.open("use_put.txt");
    string msg("Using the put function.");

    int c = 0;
    char ch;
    while ( ch = msg[ c++ ] ) {
       output.put(ch);     // write one character
    }

    output.close();        // close the file
    cout << "file saved..." << endl;
}
```



```
Using the put function.
```

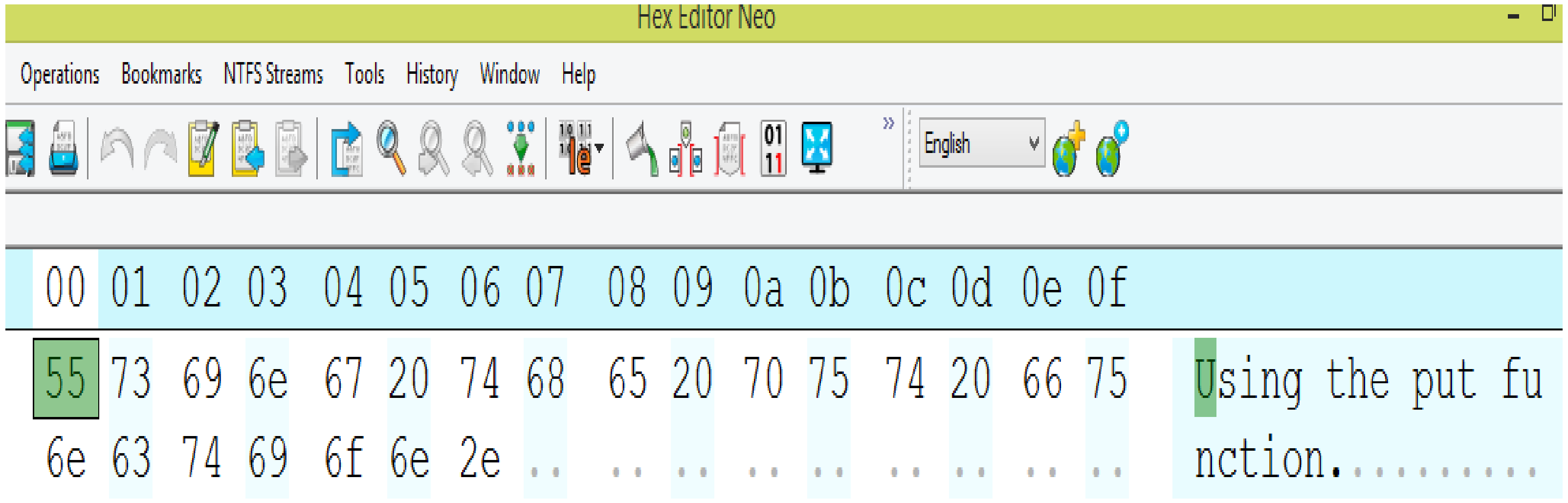


```
 00 01 02 03  04 05 06 07  08 09 0a 0b  0c 0d 0e 0f
 55 73 69 6e  67 20 74 68  65 20 70 75  74 20 66 75   Using the put fu
 6e 63 74 69  6f 6e 2e ..  .. .. .. ..  .. .. .. ..   nction.........
```

The get function

```
void readFile_UsingGet( ) {
    ifstream input;                 // Create a file object
    input.open("use_put.txt");      // Open or create a file for writing

    string msg;

    int c = 0;
    char ch;
    while (input.get(ch)) {
       //input.get(ch);
       //if (input.eof()) break;
       msg += ch;    // append ch to the end of the string in msg
    }
    ......

    input.close();
}
```

```
str size:23
str:Using the put function.
```



```
00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f
55 73 69 6e 67 20 74 68 65 20 70 75 74 20 66 75   Using the put fu
6e 63 74 69 6f 6e 2e ..  ..  ..  ..  ..  ..  ..  ..  ..   nction.........
```

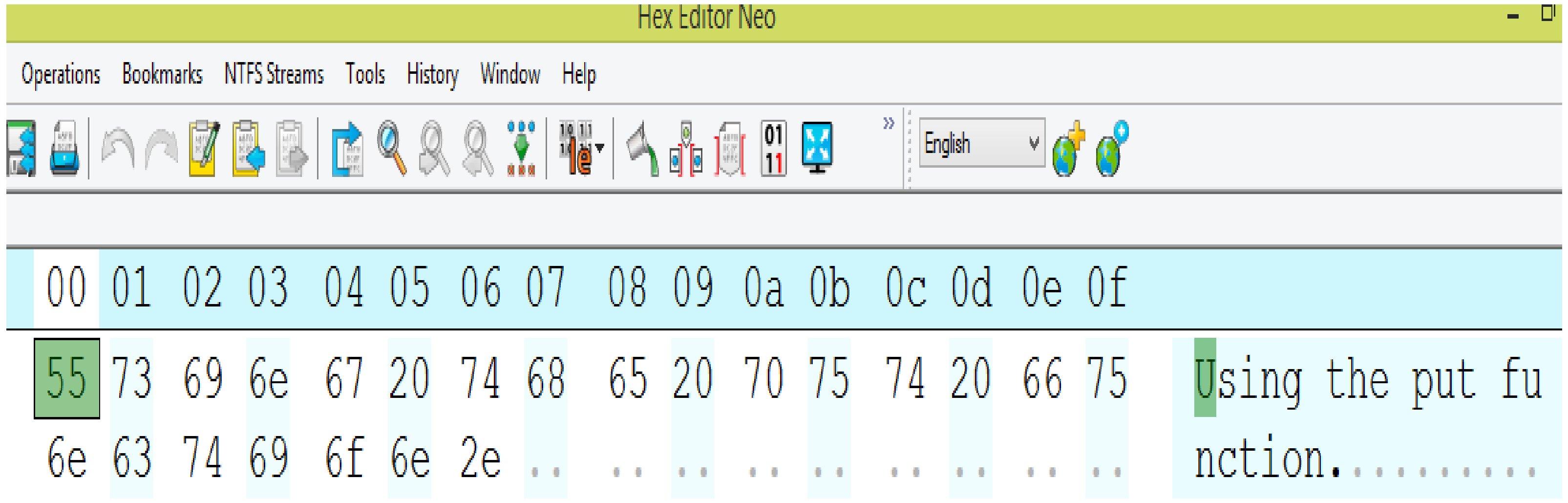
The get function

```
void readFile_UsingGet( ) {
    ifstream input;              // Create a file object
    input.open("use_put.txt");   // Open or create a file for writing

    string msg;

    int c = 0;
    char ch;
    while (input.get(ch)) {
       //input.get(ch);
       //if (input.eof()) break;
       msg += ch;   // append ch to the end of the string in msg
    }
    ......

    input.close();
}
```

str size:23

str:Using the put function.

```
while (true) {
   input.get(ch);
   if (input.eof()) break;
   msg += ch;
}
```



```
00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f
55 73 69 6e 67 20 74 68 65 20 70 75 74 20 66 75   Using the put fu
6e 63 74 69 6f 6e 2e ..  ..  ..  ..  ..  ..  ..  ..  ..   nction..........
```

get and put

Read a sequence of characters from a file and save them to another file.

```
char ch = input.get();            // Read a character
while ( !input.eof( ) )           // Check if end of file
{
        output.put( ch );         // Save the character
        ch = input.get( );        // Read next character
}
```

get and put

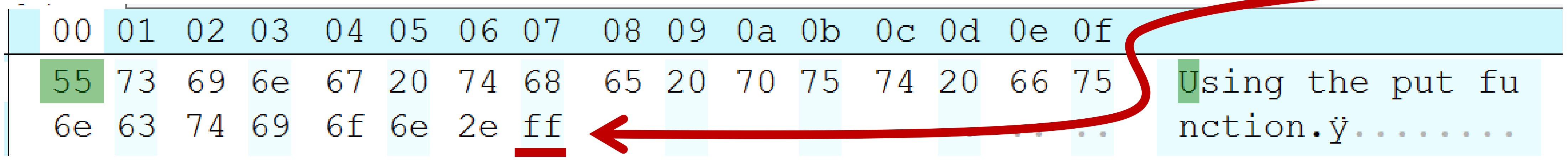
Read a sequence of characters from a file and save them to another file.

~~char ch = input.get();   // Read a character~~

```
while ( !input.eof( ) )            // Check if end of file
{
        char ch = input.get( ); // Read next character
        output.put( ch );          // Save the character

}
```

Wrong Approach

The last character is an extra character.



```
 00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f
 55 73 69 6e 67 20 74 68 65 20 70 75 74 20 66 75  Using the put fu
 6e 63 74 69 6f 6e 2e ff                          nction.ÿ........
```

The fstream class

- We can use the fstream class to create an input stream or output stream.
- To **open an fstream file**, we **specify a file mode** to tell how the file will be used.
- Please read the file modes.
- | :This is a bitwise inclusive OR operator.

For example,

```
        stream.open("city.txt", ios::out | ios::app);
```

File Open Modes

Mode	Description
ios::in	
ios::out	
ios::app	
ios:ate	
ios::truct	
ios:binary	

Combining Modes

We use the | operator to obtain a combination of modes.

For example, to open an output file named city.txt for appending data, you can use the following statement:

```
  stream.open("city.txt", ios::out | ios::app);
```

Stream States & Stream State Bit Values

Each stream object contains a set of bits that are treated as flags. Each bit value (0 or 1) indicates the state of a stream.

Bit	Description
ios::eofbit	Set when the end of an input stream is reached
ios::failbit	Set when an operation failed
ios::hardfail	Set when an unrecoverable error occurred
ios::badbit	Set when an invalid operation has been attempted
ios::goodbit	Set when an operation is successful

Stream State Functions

Function	Description
eof( )	true if the eofbit flag is set
fail( )	true if the failbit or hardfail flags is set
bad( )	true if the badbit is set
good( )	true if the goodbit is set
clear( )	clears all flags

Binary I/O

- A text file consists of a sequence of characters (ASCII code).
- A binary file as consisting of a sequence of bytes.

Example.

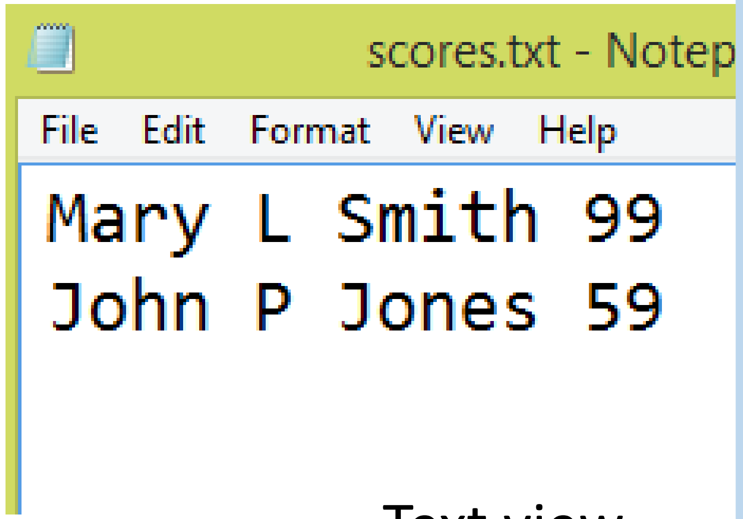
199 (decimal integer) is stored as the sequence of three characters, **'1', '9', ‘7', in a text file.**

A **byte-type value C5 is stored** in a binary file.

Decimal 197 = Hex C5 (199 = 12 * 16 + 5).

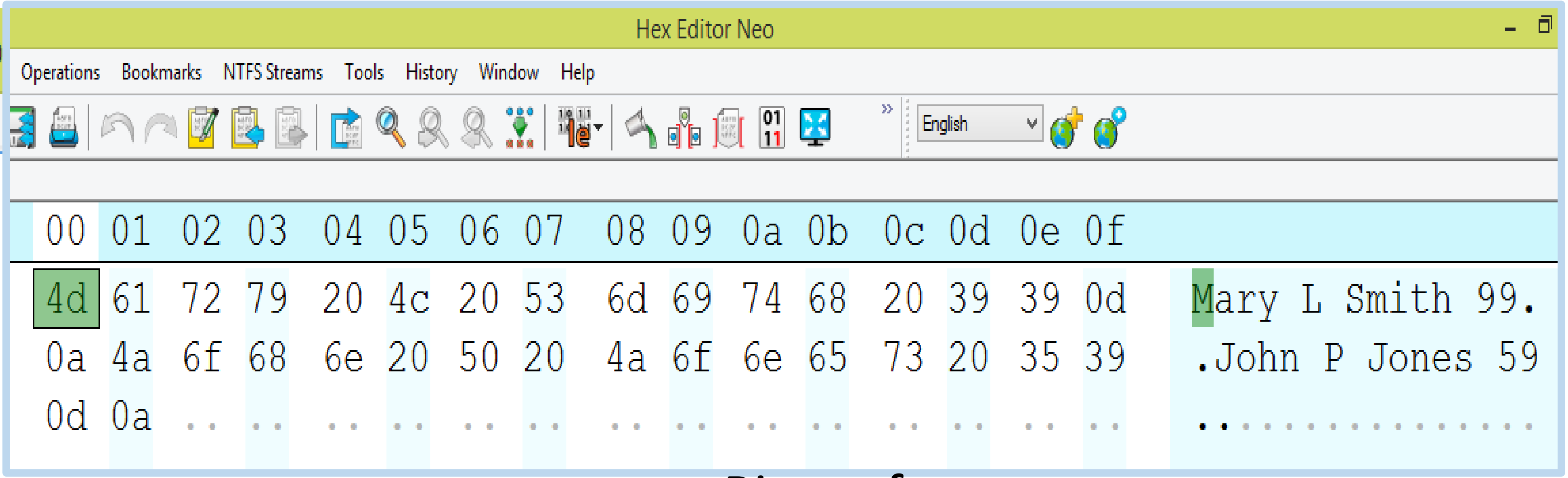
Text vs. Binary I/O

- All files are stored as a sequence of bytes.
- They are stored in binary format.
- We build the text I/O upon binary I/O to provide a level of abstraction for character encoding and decoding.



```
Mary L Smith 99
John P Jones 59
```

Text view



```
 00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f
 4d 61 72 79 20 4c 20 53 6d 69 74 68 20 39 39 0d   Mary L Smith 99.
 0a 4a 6f 68 6e 20 50 20 4a 6f 6e 65 73 20 35 39   .John P Jones 59
 0d 0a .. .. .. .. .. .. .. .. .. .. .. .. .. ..   ................
```

Binary format

Using the ios::binary mode

- **Binary I/O does not require conversions**.
- If we write a numeric value to a file, the exact value is saved to the file.

```
fstream output;                 // Create a file object
output.open("scores.txt", ios::out|ios::binary );
```

```
Syntax for the write/read function of the stream objec
streamObject.write(char* bytes, int size);
streamObject.read(char* bytes, int size);
```

reinterpret_cast

We can use it to cast **the address of a primitive type value or an object** to **a character array pointer for binary I/O**.

Syntax:

reinterpret_cast<dataType>(address)

address: The starting address of the data (primitive, array, or object).

```
class A { public: A() { x = 0x1234; z = 1.234; }
          int x;  double z;
};
A a;
```

```
void write_file( ) { fstream output; // Create a file object
    output.open("scores.txt", ios::out|ios::binary );
    output.write(reinterpret_cast<char*>(&a.x), sizeof(a.x));
    output.write(reinterpret_cast<char*>(&a.z), sizeof(a.z));
    output.close(); }
```

```
void read_file( ) { fstream input; // Create a file object
    input.open("scores.txt", ios::in|ios::binary );
    input.read(reinterpret_cast<char*>(&a.x), sizeof(a.x));
    input.read(reinterpret_cast<char*>(&a.z), sizeof(a.z));
    cout << "a.x:" << a.x << endl;
    cout << "a.z:" << a.z << endl;
    input.close( ); }
```

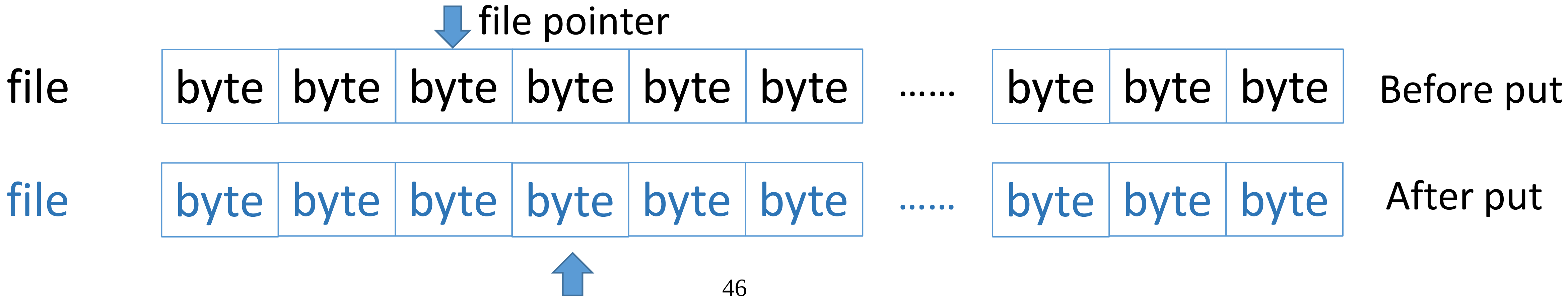
```
class A { public: A() { x = 0x1234; z = 1.234; }
          int x;  double z;
};
A a;
```

```
void write_file( ) { fstream output; // Create a file object
    output.open("scores.txt", ios::out|ios::binary );
    output.write(reinterpret_cast<char*>(&a.x), sizeof(a.x));
    output.write(reinterpret_cast<char*>(&a.z), sizeof(a.z));
    output.close(); }
```

```
void read_file( ) { fstream input; // Create a file object
    input.open("scores.txt", ios::in|ios::binary );
    input.read(reinterpret_cast<char*>(&a.x), sizeof(a.x));
    input.read(reinterpret_cast<char*>(&a.z), sizeof(a.z));
    cout << "a.x:" << a.x << endl;
    cout << "a.z:" << a.z << endl;
    input.close( ); }
```

Random Access

- A file consists of a sequence of bytes.
- A file pointer points to a byte that is to be read.
- It points to the beginning of a file when the file is opened.
- When data are read/write to the file, the file pointer moves forward to the next data item.
- Example: When we put a character using the put( ) function, the file pointer moves one byte ahead of the previous location.



seekp, seekg (note: get and put pointers are the same???)

seekg: for controlling the get pointer

seekp: for controlling the put pointer

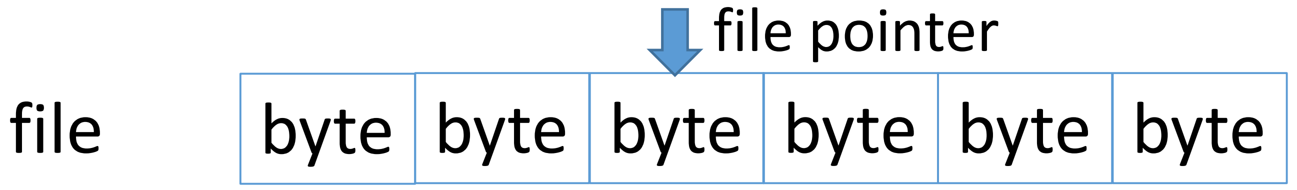
tellg: return the position of the getpointer

tellp: return the position of the put pointer

Statement	Purpose
seekg(128, ios:beg);	Move the get file pointer to 128th byte from the beginning of the file
seekg(-32, ios:cur);	Move the get file pointer to the 32th byte backward from the current get pointer position
seekp(-128, ios:end);	Move the put file pointer to the 128th byte backward from the end of the file
seekp(32);	Move the get file pointer to 128th byte in the file

seekp, seekg, tellp, tellg

ios::beg	Offset from the beginning of the file
ios::end	Offset from the end of the file
ios::cur	Offset from the current file pointer



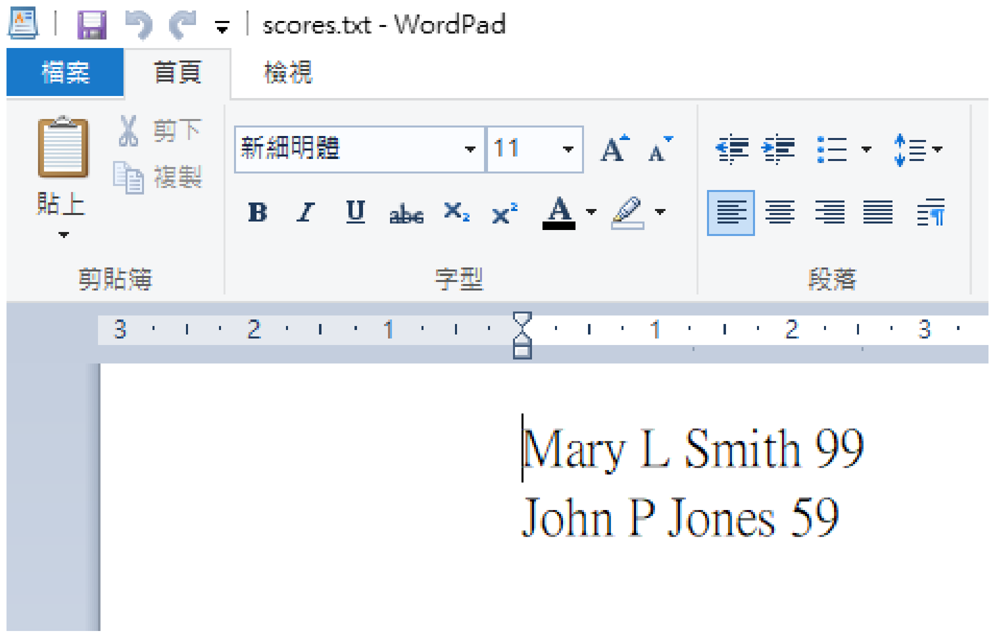
How to output in hex format

```
fstream input;                                  // Create a file  object
input.open("scores.txt", ios::in|ios::binary );
input.seekg(0, ios::beg);                       //get file pointer
while (true) {
        char c;
        c = input.get();
        if (input.eof()) break;
        cout <<  c << endl;                     // in ASCII code
}
```

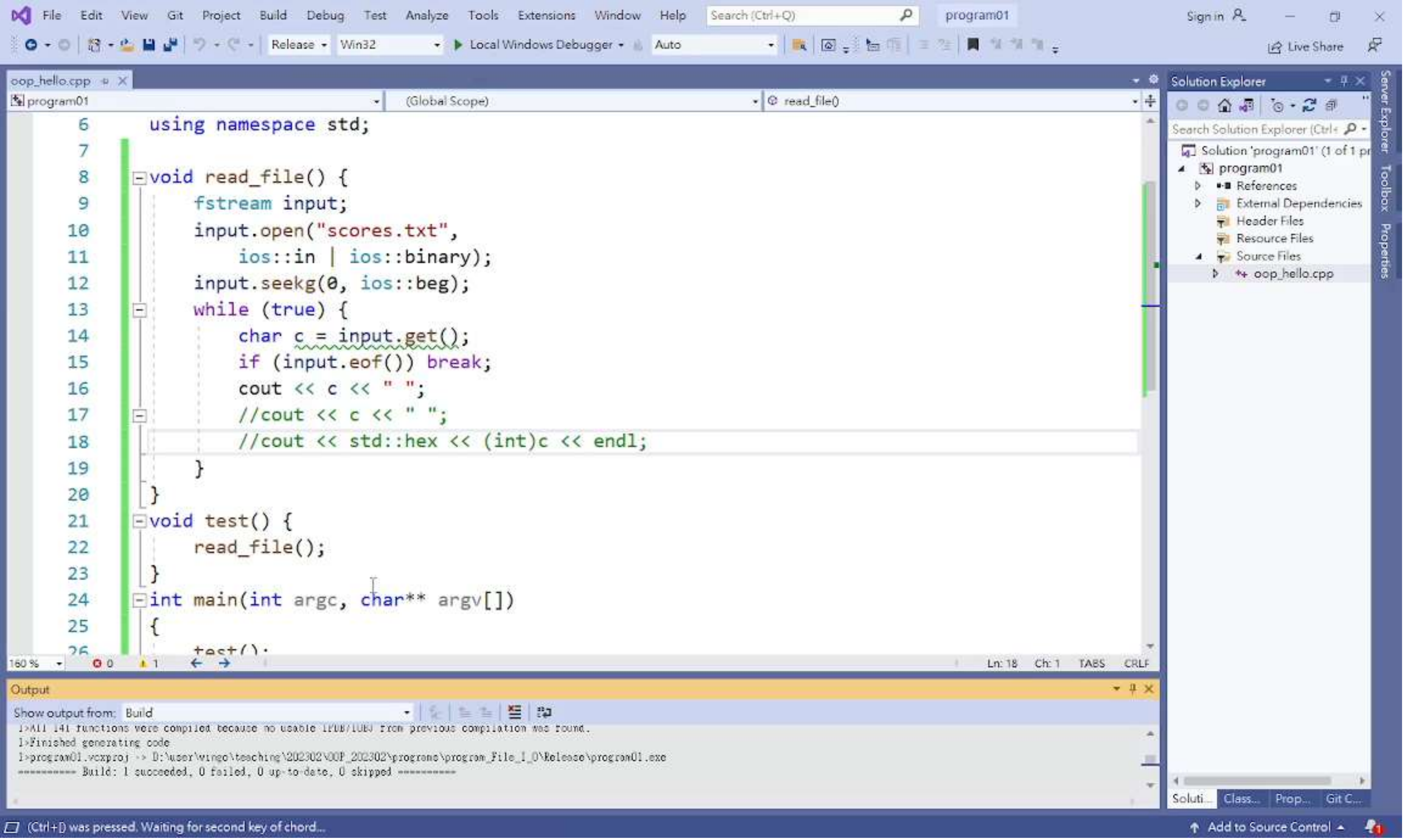
An Example

An Example

```
void read_file() {
    fstream input;
    input.open("scores.txt",
    ios::in | ios::binary);
    input.seekg(0, ios::beg);
    while (true) {
        char c = input.get();
        if (input.eof()) break;
        cout << c << " ";
        //cout << c << " ";
        //cout << std::hex << (int)c << endl;
    }
}
```

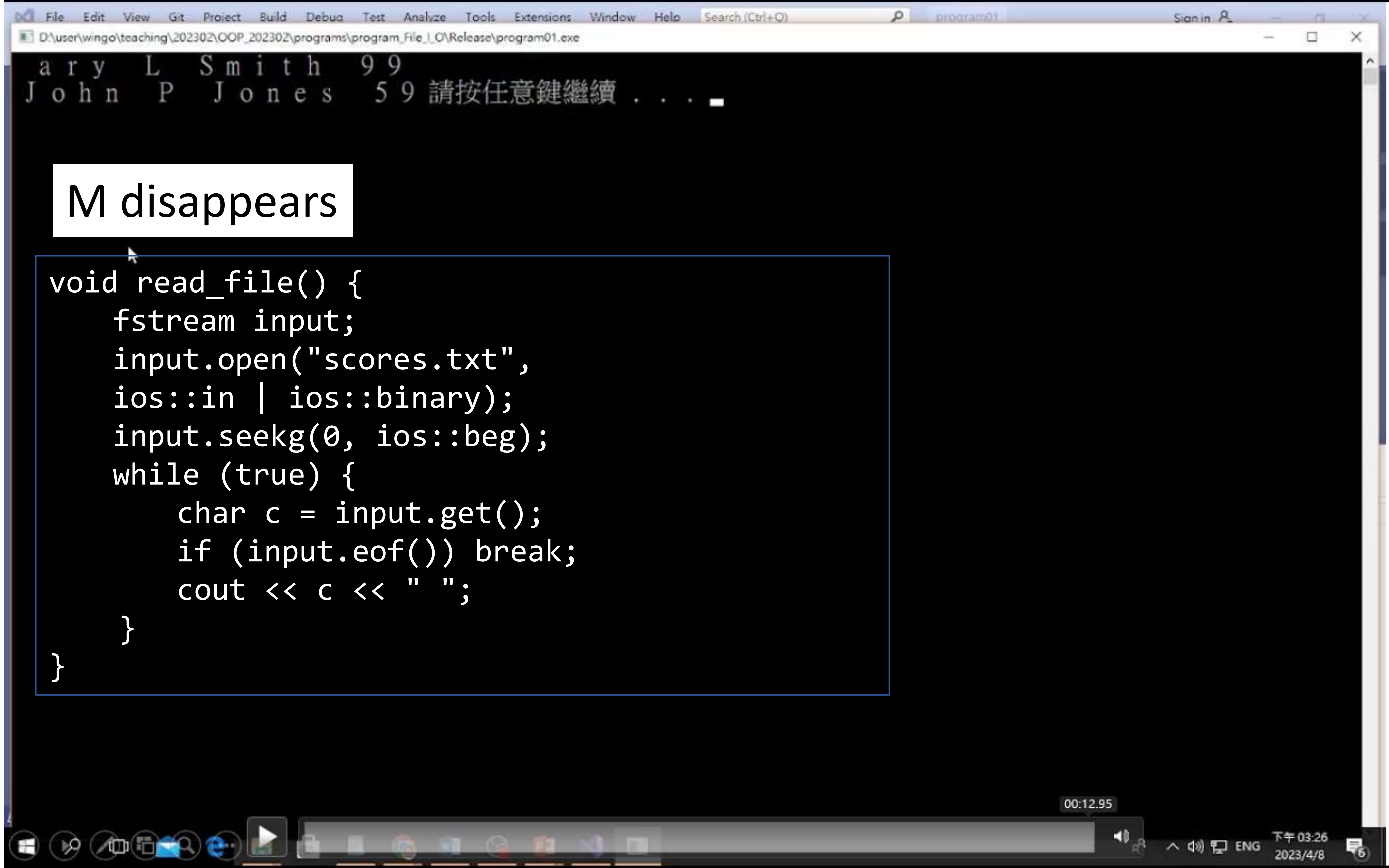


Mary L Smith 99
John P Jones 59



```cpp
using namespace std;

void read_file() {
    fstream input;
    input.open("scores.txt",
        ios::in | ios::binary);
    input.seekg(0, ios::beg);
    while (true) {
        char c = input.get();
        if (input.eof()) break;
        cout << c << " ";
        //cout << c << " ";
        //cout << std::hex << (int)c << endl;
    }
}
void test() {
    read_file();
}
int main(int argc, char** argv[])
{
    test();
```

```
1>All 141 functions were compiled because no usable IPDB/IOBJ from previous compilation was found.
1>Finished generating code
1>program01.vcxproj -> D:\user\wingo\teaching\202302\OOP_202302\programs\program_File_I_O\Release\program01.exe
========== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped ==========
```


File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help
D:\user\wingo\teaching\202302\OOP_202302\programs\program_File_I_O\Release\program01.exe
00:12.95
ENG
下午 03:26
2023/4/8

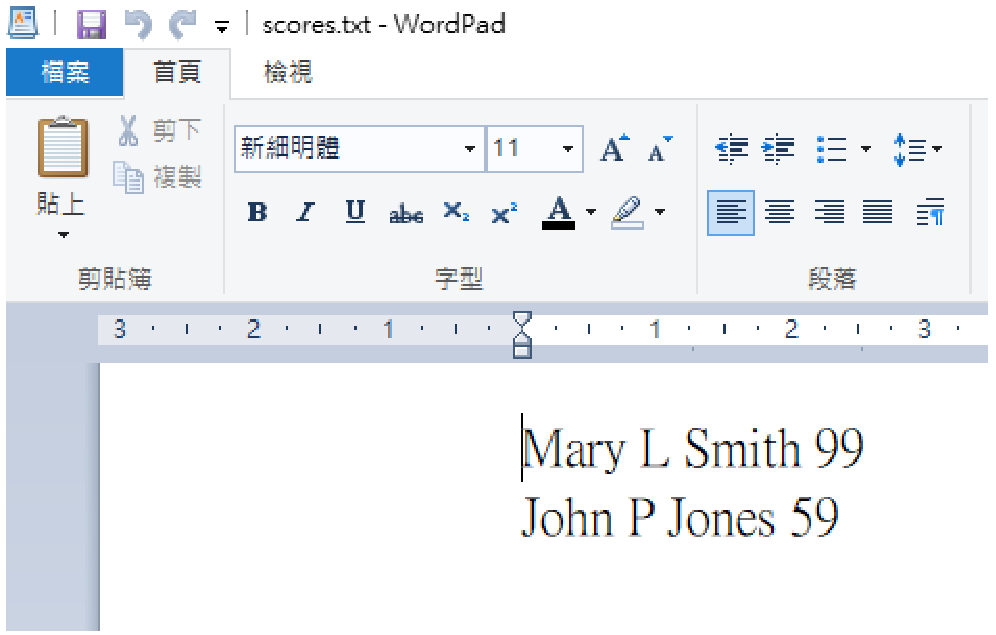
```
a r y   L   S m i t h   9 9
J o h n   P   J o n e s   5 9 請按任意鍵繼續 . . .
```

M disappears

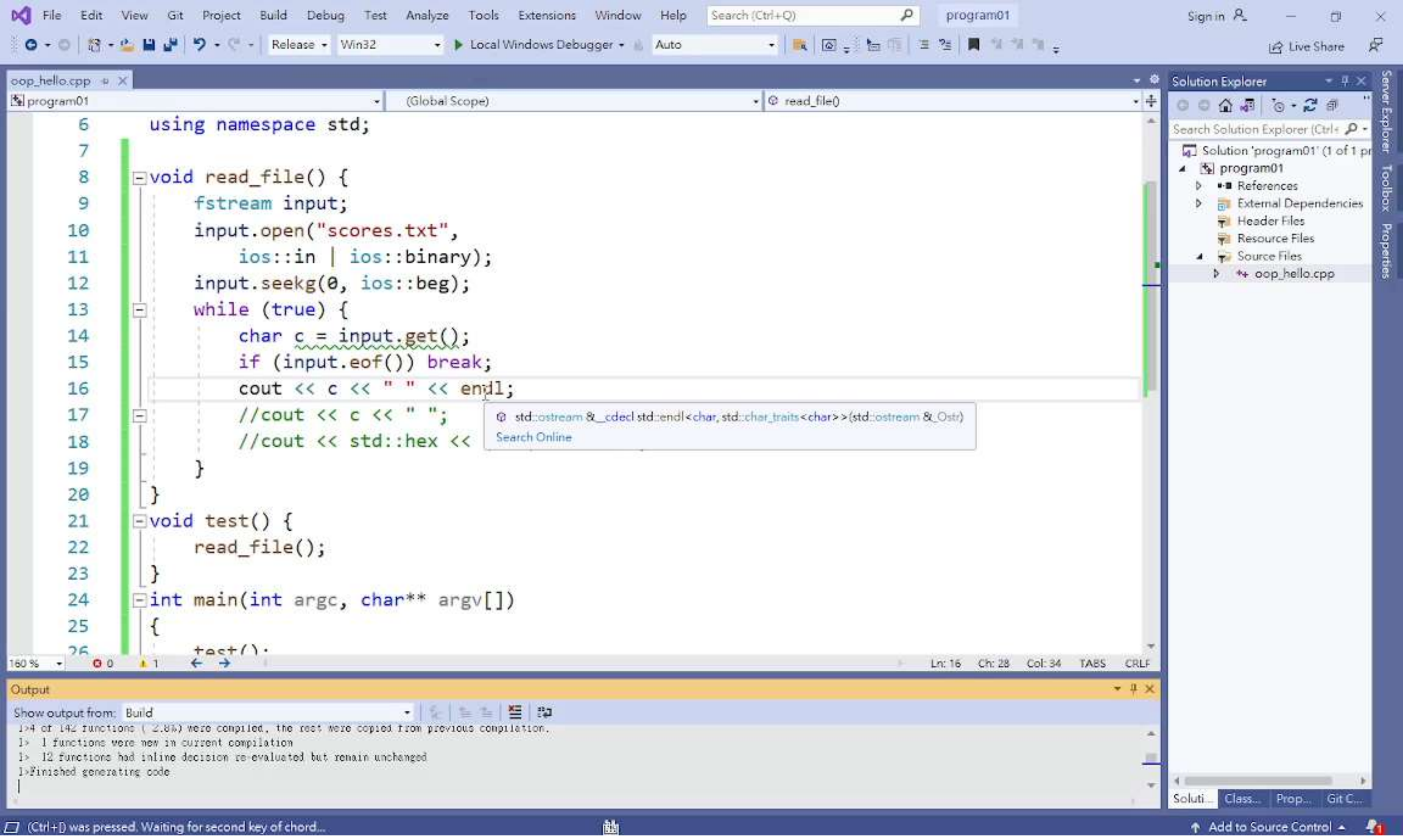
```
void read_file() {
    fstream input;
    input.open("scores.txt",
    ios::in | ios::binary);
    input.seekg(0, ios::beg);
    while (true) {
        char c = input.get();
        if (input.eof()) break;
        cout << c << " ";
     }
}
```

An Example

```cpp
void read_file() {
    fstream input;
    input.open("scores.txt",
    ios::in | ios::binary);
    input.seekg(0, ios::beg);
    while (true) {
        char c = input.get();
        if (input.eof()) break;
        cout << c << " " << endl;
        //cout << c << " ";
        //cout << std::hex << (int)c << endl;
    }
}
```



```
Mary L Smith 99
John P Jones 59
```



```cpp
using namespace std;

void read_file() {
    fstream input;
    input.open("scores.txt",
        ios::in | ios::binary);
    input.seekg(0, ios::beg);
    while (true) {
        char c = input.get();
        if (input.eof()) break;
        cout << c << " " << endl;
        //cout << c << " ";
        //cout << std::hex <<
    }
}
void test() {
    read_file();
}
int main(int argc, char** argv[])
{
    test();
```

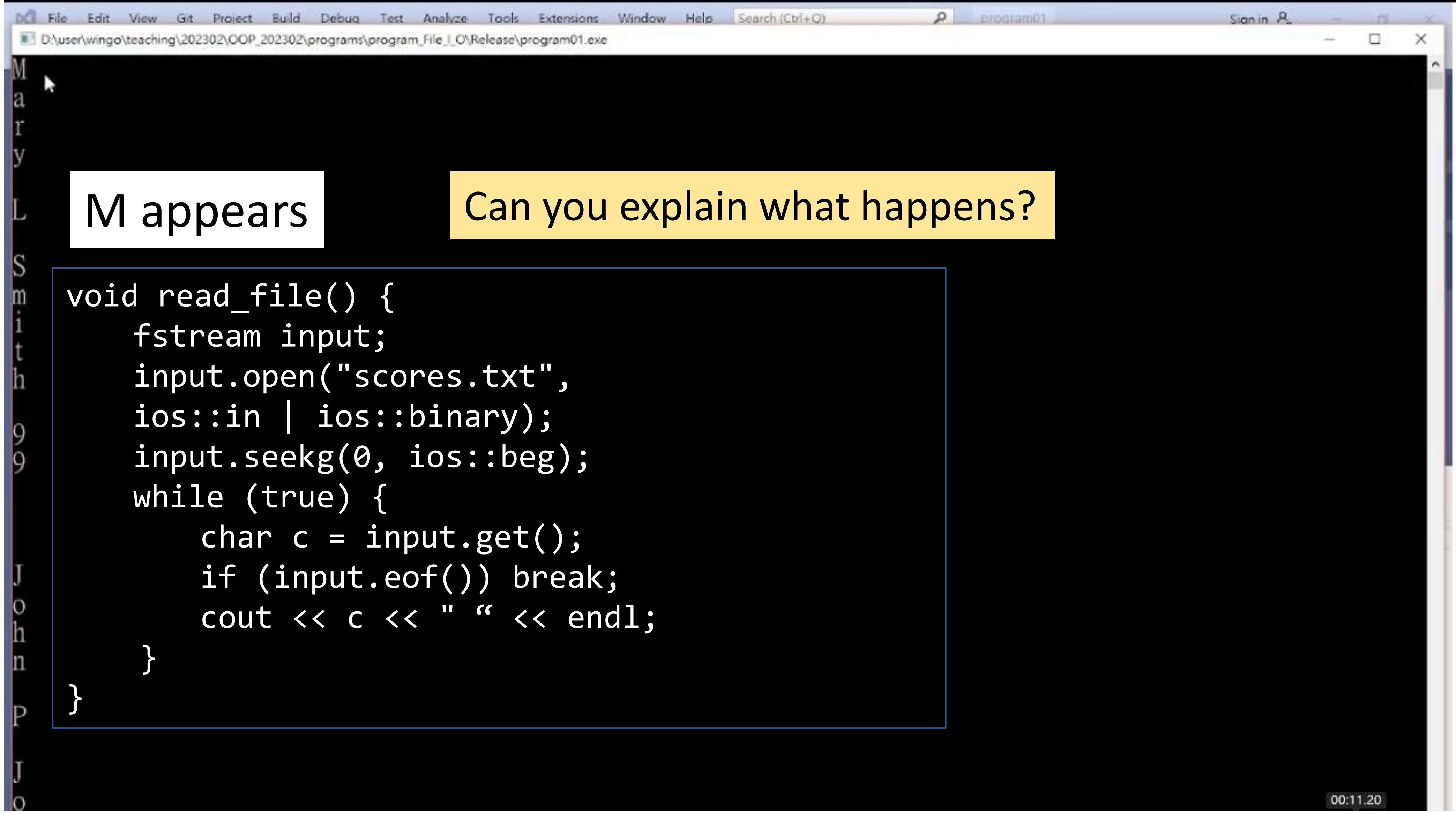
```
std::ostream &__cdecl std::endl<char, std::char_traits<char>>(std::ostream &_Ostr)
Search Online
```

Output

Show output from: Build

```
1>4 of 142 functions ( 2.8%) were compiled, the rest were copied from previous compilation.
1>  1 functions were new in current compilation
1>  12 functions had inline decision re-evaluated but remain unchanged
1>Finished generating code
```

(Ctrl+D) was pressed. Waiting for second key of chord...


File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help
Search (Ctrl+Q)
program01
Sign in
D:\user\wingo\teaching\202302\OOP_202302\programs\program_File_I_O\Release\program01.exe
00:11.20

```
M
a
r
y

L

S
m
i
t
h

9
9



J
o
h
n

P

J
o
```

M appears

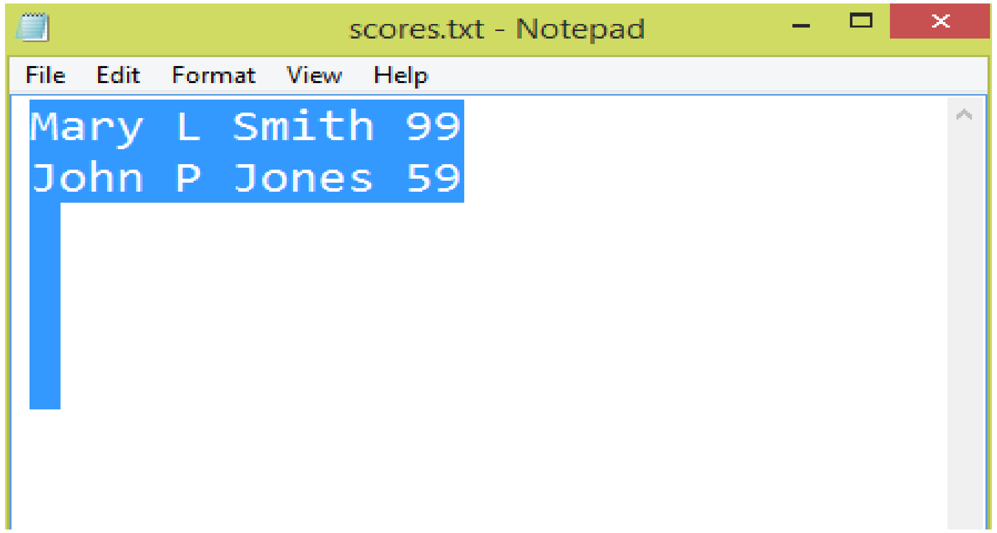
Can you explain what happens?

```
void read_file() {
    fstream input;
    input.open("scores.txt",
    ios::in | ios::binary);
    input.seekg(0, ios::beg);
    while (true) {
        char c = input.get();
        if (input.eof()) break;
        cout << c << " “ << endl;
     }
}
```

An Example

How to output in hex format

```
fstream input;
input.open("scores.txt",
   ios::in|ios::binary );
input.seekg(0, ios::beg);
while (true) {
   char c = input.get();
   if (input.eof()) break;
   cout << std::hex << (int) c << " ";
}
```



```
Mary L Smith 99
John P Jones 59
```

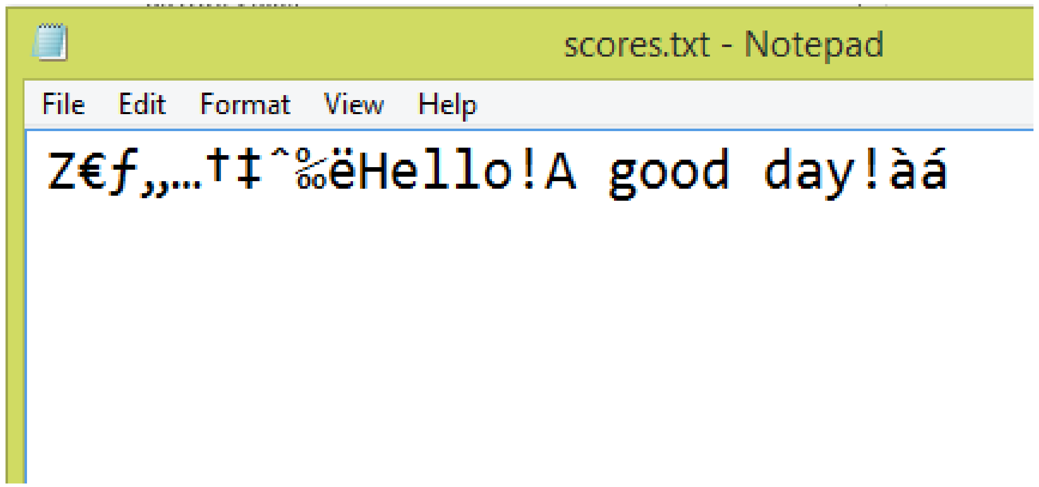
```
4d 61 72 79 20 4c 20 53 6d 69 74 68 20 39 39 d
a 4a 6f 68 6e 20 50 20 4a 6f 6e 65 73 20 35 39
d a 20 d a 20 d a 20 d a 20
```

The ASCII code of M appears!

```
00 01 02 03  04 05 06 07  08 09 0a 0b  0c 0d 0e 0f
4d 61 72 79  20 4c 20 53  6d 69 74 68  20 39 39 0d   Mary L Smith 99.
0a 4a 6f 68  6e 20 50 20  4a 6f 6e 65  73 20 35 39   .John P Jones 59
0d 0a 20 0d  0a 20 0d 0a  20 0d 0a 20  .. .. .. ..   .. .. .. .. ....
```

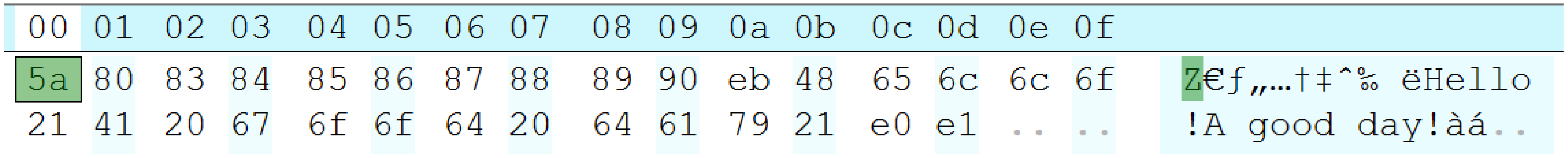
How to output in hex format

```
fstream input;
input.open("scores.txt",
   ios::in|ios::binary );
input.seekg(0, ios::beg);
while (true) {
   char c = input.get();
   if (input.eof()) break;
   cout << std::hex << (int) c << " ";
}
```



```
Z€ƒ„…†‡ˆ‰ëHello!A good day!àá
```

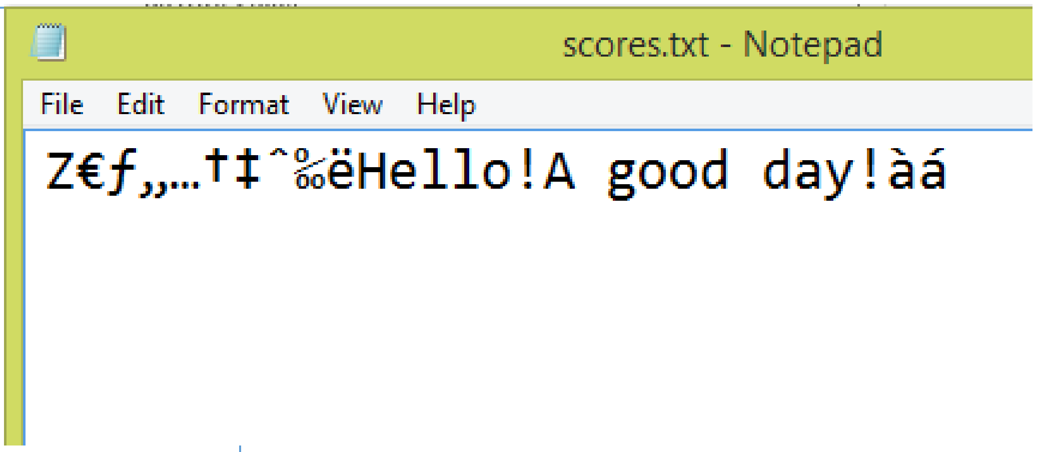
```
5a ffffff80 ffffff83 ffffff84 ffffff85 ffffff86 ffffff87 ffffff88
ffffff89 ffffff90 ffffffeb 48 65 6c 6c 6f 21 41 20 67 6f 6f 64 20
64 61 79 21 ffffffe0 ffffffe1
```



```
00 01 02 03  04 05 06 07  08 09 0a 0b  0c 0d 0e 0f
5a 80 83 84  85 86 87 88  89 90 eb 48  65 6c 6c 6f   Z€ƒ„…†‡ˆ‰ ëHello
21 41 20 67  6f 6f 64 20  64 61 79 21  e0 e1 .. ..   !A good day!àá..
```

How to output in hex format

```
fstream input;
input.open("scores.txt",
   ios::in|ios::binary );
input.seekg(0, ios::beg);
while (true) {
   unsigned char c = input.get();
   if (input.eof()) break;
   cout << std::hex << (~~unsigned~~ int) c << " ";
}
```

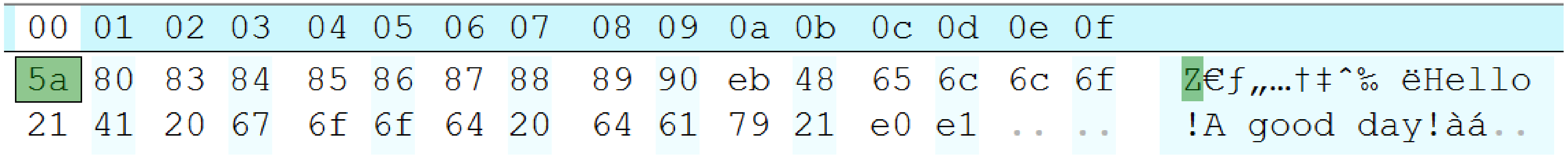


scores.txt - Notepad

File Edit Format View Help

```
Z€ƒ„…†‡ˆ‰ëHello!A good day!àá
```

```
5a 80 83 84 85 86 87 88 89 90 eb 48 65 6c 6c 6f
21 41 20 67 6f 6f 64 20 64 61 79 21 e0 e1
```



```
00 01 02 03  04 05 06 07  08 09 0a 0b  0c 0d 0e 0f
5a 80 83 84  85 86 87 88  89 90 eb 48  65 6c 6c 6f   Z€ƒ„…†‡ˆ‰ ëHello
21 41 20 67  6f 6f 64 20  64 61 79 21  e0 e1 .. ..   !A good day!àá..
```

Report file pointers

```
void reportFilePointers()
{
    cout << "tellp:" << input.tellp() << endl;
    cout << "tellg:" << input.tellg() << endl;
}
```

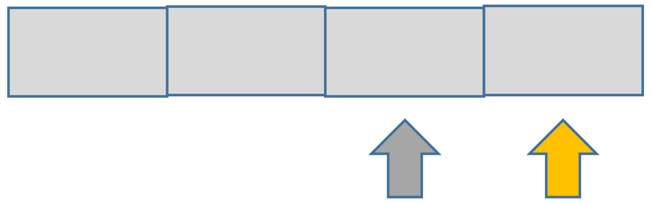
Report Data

```
void reportData(
    unsigned char &a1,
    unsigned char &b1,
    unsigned char &c1,
    unsigned char &d1
    )
{
    cout << "a1:" << hex << (unsigned short) a1 << endl;
    cout << "b1:" << hex << (unsigned short) b1 << endl;
    cout << "c1:" << hex << (unsigned short) c1 << endl;
    cout << "d1:" << hex << (unsigned short) d1 << endl;
}
```

Read a file backward

ios::cur: current position
Ios::end: end position

```
    unsigned char a1, b1, c1, d1;
    fstream input; // Create a file object
    input.open(cn_fileName, ios::in|ios::binary );
    input.seekg(-1, ios::end);
//
    input.read(reinterpret_cast<char*>(&a1), sizeof(a1));
    input.seekp(-2, ios::cur);
    input.read(reinterpret_cast<char*>(&b1), sizeof(b1));
    input.seekp(-2, ios::cur);
    input.read(reinterpret_cast<char*>(&c1), sizeof(c1));
    input.seekp(-2, ios::cur);
    input.read(reinterpret_cast<char*>(&d1), sizeof(d1));
```



Read a file backward

The file content in hexadecimal format:

```
34 12 21 12 58 39 b4 c8 00 be f3 3f
 0  1  2  3  4  5  6  7  8  9 10 11  (address)
```

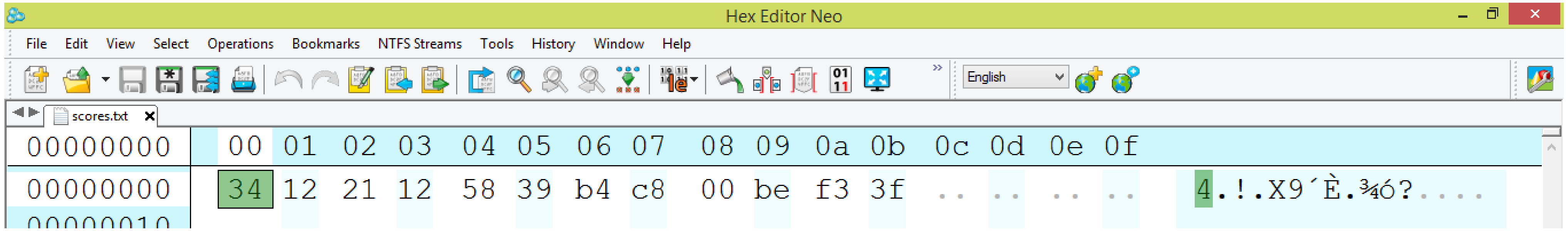
```
unsigned char a1, b1, c1, d1;
fstream input; // Create a file object
input.open(cn_fileName, ios::in|ios::binary );
reportfilePointers(input);
input.seekg(-1, ios::end);
reportfilePointers(input);
input.read(reinterpret_cast<char*>(&a1), sizeof(a1));
reportfilePointers(input);
input.seekg(-2, ios::cur);
reportfilePointers(input);
input.read(reinterpret_cast<char*>(&b1), sizeof(b1));
input.seekg(-2, ios::cur);
input.read(reinterpret_cast<char*>(&c1), sizeof(c1));
input.seekg(-2, ios::cur);
input.read(reinterpret_cast<char*>(&d1), sizeof(d1));
reportData(a1, b1, c1, d1)
```

```
tellp:0
tellg:0
tellp:11
tellg:11
tellp:12
tellg:12
tellp:10
tellg:10
a1:3f
b1:f3
c1:be
d1:0
```

Read a file backward

The file content in hexadecimal format:

34	12	21	12	58	39	b4	c8	00	be	f3	3f	
0	1	2	3	4	5	6	7	8	9	10	11	(address)


Hex Editor Neo
File Edit View Select Operations Bookmarks NTFS Streams Tools History Window Help
English
scores.txt

```
00000000   00 01 02 03  04 05 06 07  08 09 0a 0b  0c 0d 0e 0f
00000000   34 12 21 12  58 39 b4 c8  00 be f3 3f  .. .. .. ..   4.!.X9´È.¾ó?....
00000010
```

$\longrightarrow$ Memory address increasing direction

The file content in hexadecimal format:
34 12 21 12 58 39 b4 c8 00 be f3 3f

tellp:0
tellg:0
tellp:11
tellg:11
tellp:12
tellg:12
tellp:10
tellg:10
a1:3f
b1:f3
c1:be
d1:0

Exercise One
What are the output?

The file content in hexadecimal format:

```
34 12 21 12 58 39 b4 c8 00 be f3 3f
 0  1  2  3  4  5  6  7  8  9 10 11  (address)
```

```
unsigned char a1, b1, c1, d1;
fstream input; // Create a file object
input.open(cn_fileName, ios::in|ios::binary );
reportfilePointers(input);
input.seekg(-5, ios::end);
reportfilePointers(input);
input.read(reinterpret_cast<char*>(&a1), sizeof(a1));
reportfilePointers(input);
input.seekg(-2, ios::cur);
reportfilePointers(input);
input.read(reinterpret_cast<char*>(&b1), sizeof(b1));
input.seekg(-2, ios::cur);
input.read(reinterpret_cast<char*>(&c1), sizeof(c1));
input.seekg(-2, ios::cur);
input.read(reinterpret_cast<char*>(&d1), sizeof(d1));
reportData(a1, b1, c1, d1)
```

```
tellp:0
tellg:0
tellp:?
tellg:?
tellp:?
tellg:?
tellp:?
tellg:?
a1:?
b1:?
c1:?
d1:?
```

Exercise One
What are the output?

The file content in hexadecimal format:

```
34 12 21 12 58 39 b4 c8 00 be f3 3f
 0  1  2  3  4  5  6  7  8  9 10 11  (address)
```

```
unsigned char a1, b1, c1, d1;
fstream input; // Create a file object
input.open(cn_fileName, ios::in|ios::binary );
reportfilePointers(input);
input.seekg(-5, ios::end);
reportfilePointers(input);
input.read(reinterpret_cast<char*>(&a1), sizeof(a1));
reportfilePointers(input);
input.seekg(-4, ios::cur);
reportfilePointers(input);
input.read(reinterpret_cast<char*>(&b1), sizeof(b1));
input.seekg(2, ios::cur);
input.read(reinterpret_cast<char*>(&c1), sizeof(c1));
input.seekg(3, ios::cur);
input.read(reinterpret_cast<char*>(&d1), sizeof(d1));
reportData(a1, b1, c1, d1)
```

```
tellp:0
tellg:0
tellp:7
tellg:7
tellp:8
tellg:8
tellp:4
tellg:4
a1:c8
b1:58
c1:c8
d1:3f
```

Supplemental Materials

Report Data

```
void reportData(
    unsigned char &a1,
    unsigned char &b1,
    unsigned char &c1,
    unsigned char &d1
    )
{
    cout << "a1:" << hex << (unsigned short) a1 << endl;
    cout << "b1:" << hex << (unsigned short) b1 << endl;
    cout << "c1:" << hex << (unsigned short) c1 << endl;
    cout << "d1:" << hex << (unsigned short) d1 << endl;
}
```

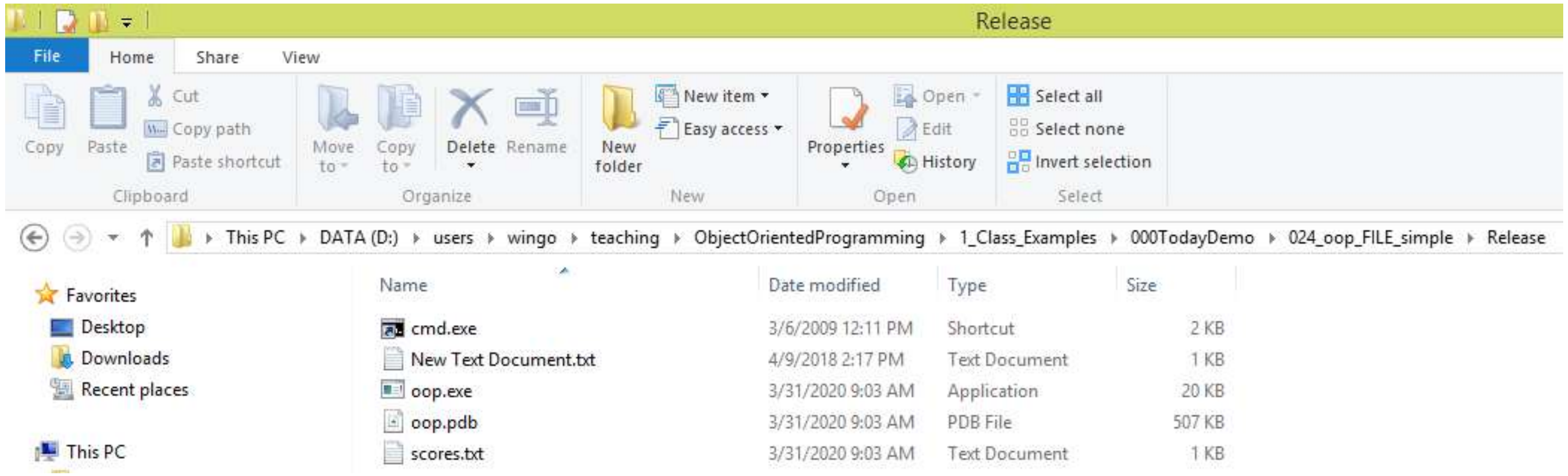
Absolute filename

In a file system, a file is placed in a directory.

An ***absolute file name*** contains a file name with its **path and drive letter**.

For example, the *exe* file is placed in **D:\teaching\OOP\1_Class_Examples\024_oop_FILE_simple\Release**.

D: is the drive letter. Absolute file names are machine-dependent (Platform dependent).



This PC ▸ DATA (D:) ▸ users ▸ wingo ▸ teaching ▸ ObjectOrientedProgramming ▸ 1_Class_Examples ▸ 000TodayDemo ▸ 024_oop_FILE_simple ▸ Release

Name	Date modified	Type	Size
cmd.exe	3/6/2009 12:11 PM	Shortcut	2 KB
New Text Document.txt	4/9/2018 2:17 PM	Text Document	1 KB
oop.exe	3/31/2020 9:03 AM	Application	20 KB
oop.pdb	3/31/2020 9:03 AM	PDB File	507 KB
scores.txt	3/31/2020 9:03 AM	Text Document	1 KB

Including \ in file names

The backslash (\) is the directory separator for Windows.

To output the backslash character, we must write \\.

For example:

```
output.open("d:\\OOP_test\\student_scores.txt");
```

Relative filename

- ➢We can use **relative file name without drive letters**.

  → platform free

- ➢The directory of the relative filename can be specified as follows:

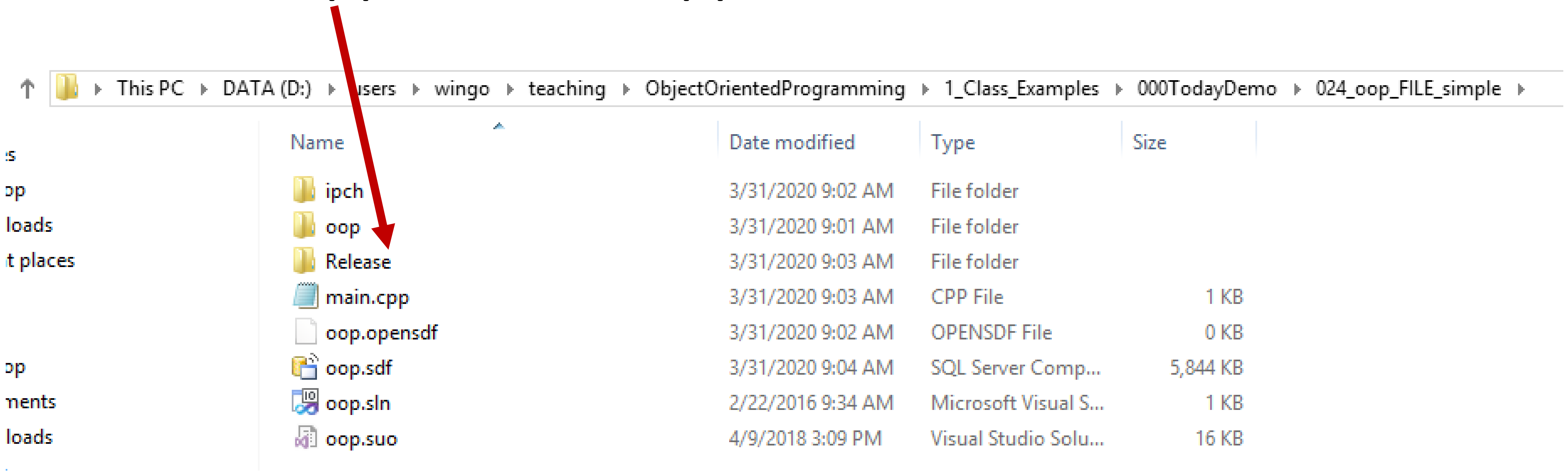
./	; the current directory
../	; the parent directory of the current directory
./../filename	

Relative filename

./ ; the current directory, i.e., 024_oop_FILE_simple

../ ; we will go to the folder: 000TodayDemo

./Release/opp.exe ; the opp.exe file in the Release folder



This PC ▸ DATA (D:) ▸ users ▸ wingo ▸ teaching ▸ ObjectOrientedProgramming ▸ 1_Class_Examples ▸ 000TodayDemo ▸ 024_oop_FILE_simple ▸

Name	Date modified	Type	Size
ipch	3/31/2020 9:02 AM	File folder	
oop	3/31/2020 9:01 AM	File folder	
Release	3/31/2020 9:03 AM	File folder	
main.cpp	3/31/2020 9:03 AM	CPP File	1 KB
oop.opensdf	3/31/2020 9:02 AM	OPENSDF File	0 KB
oop.sdf	3/31/2020 9:04 AM	SQL Server Comp...	5,844 KB
oop.sln	2/22/2016 9:34 AM	Microsoft Visual S...	1 KB
oop.suo	4/9/2018 3:09 PM	Visual Studio Solu...	16 KB

Examples for Relative file paths

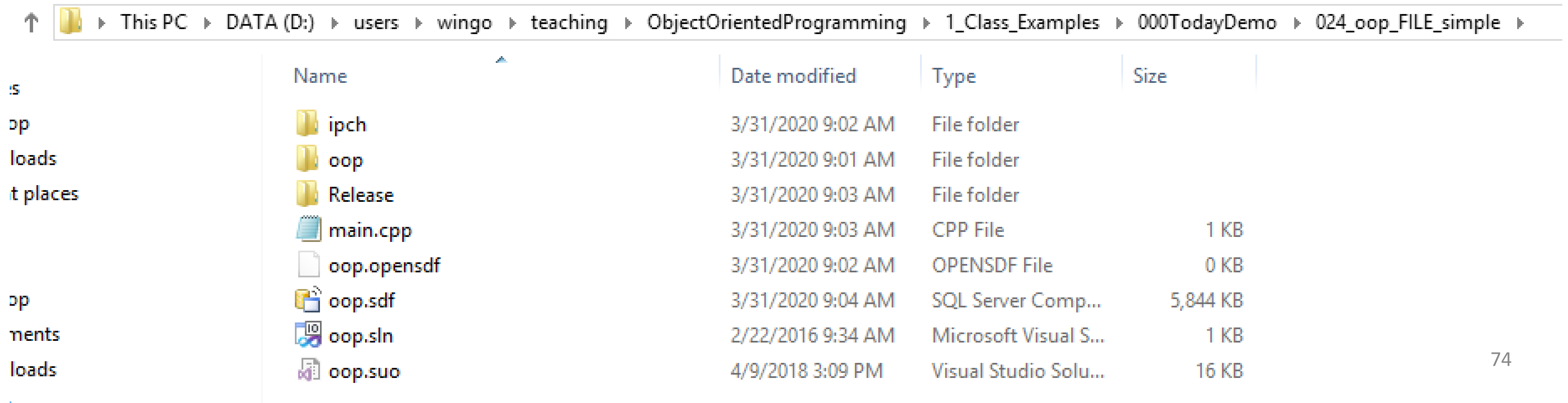
./oop.sln

./main.cpp

./Release/scores.txt

../../Example2/Release/oop2.exe // climb up and down the **directory**

// **hierarchy** to get oop2.exe



This PC ▸ DATA (D:) ▸ users ▸ wingo ▸ teaching ▸ ObjectOrientedProgramming ▸ 1_Class_Examples ▸ 000TodayDemo ▸ 024_oop_FILE_simple ▸

Name	Date modified	Type	Size
ipch	3/31/2020 9:02 AM	File folder	
oop	3/31/2020 9:01 AM	File folder	
Release	3/31/2020 9:03 AM	File folder	
main.cpp	3/31/2020 9:03 AM	CPP File	1 KB
oop.opensdf	3/31/2020 9:02 AM	OPENSDF File	0 KB
oop.sdf	3/31/2020 9:04 AM	SQL Server Comp...	5,844 KB
oop.sln	2/22/2016 9:34 AM	Microsoft Visual S...	1 KB
oop.suo	4/9/2018 3:09 PM	Visual Studio Solu...	16 KB

Testing File Existence

- We should check whether a file exists before performing operations on it.
- We can invoke the fail() function after invoking the open function.
- If the return value of fail() is true, it indicates that the file does not exist.

```
ifstream input("student_scores.txt");
if ( input.fail()) { //cannot open file }
```

The get function

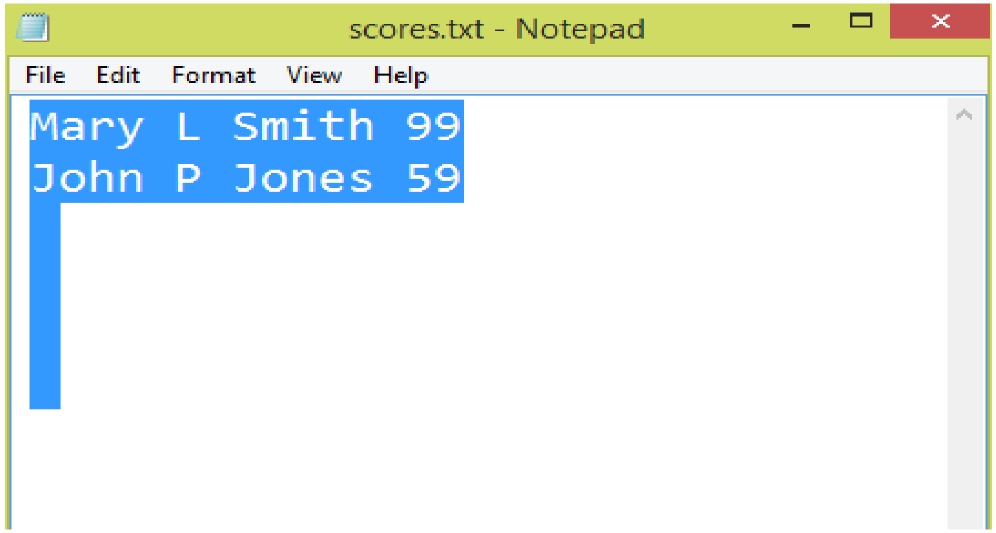
Invoke the **get** function on **an input object** to read a character.

```
void readFile_UsingGet( ) {
    ifstream input;                 // Create a file object
    input.open("use_put.txt");      // Open or create a file for writing

    string msg;

    int c = 0;
    char ch;
    while (true) {
       input.get(ch);
       if (input.eof()) break;
       msg += ch;    // append ch to the end of the string in msg
    }
    cout << "str size:" << msg.size() << endl;
    cout << "str:" << msg << endl;

    input.close();
}
```

```
str size:23
str:Using the put function.
```

How to output in hex format

```
fstream input;
input.open("scores.txt",
   ios::in|ios::binary );
input.seekg(0, ios::beg);
while (true) {
   char c = input.get();
   if (input.eof()) break;
   cout << c << " ";
}
```



```
Mary L Smith 99
John P Jones 59
```

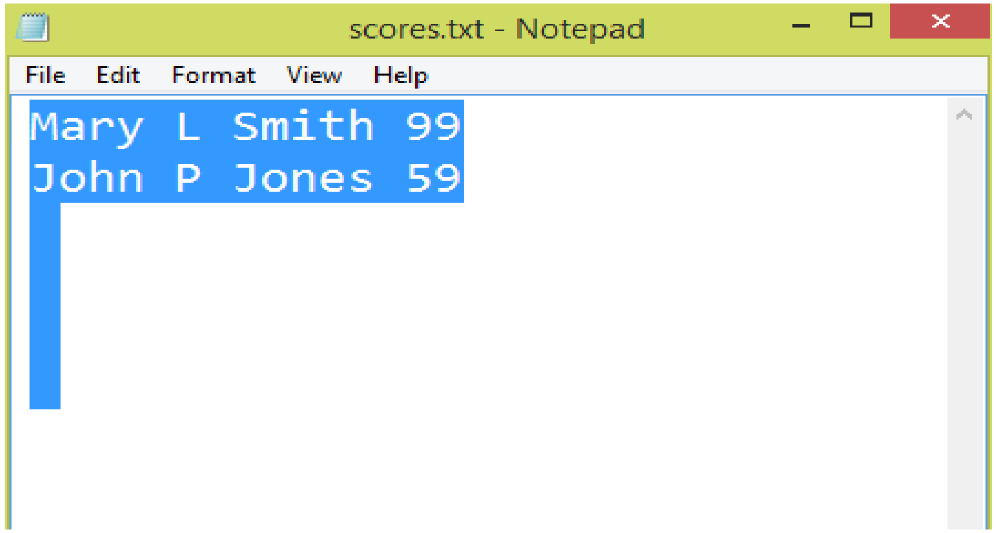
```
 a r y   L   S m i t h   9 9
 J o h n   P   J o n e s   5 9
```

M is missing!

```
00 01 02 03  04 05 06 07  08 09 0a 0b  0c 0d 0e 0f
4d 61 72 79  20 4c 20 53  6d 69 74 68  20 39 39 0d   Mary L Smith 99.
0a 4a 6f 68  6e 20 50 20  4a 6f 6e 65  73 20 35 39   .John P Jones 59
0d 0a 20 0d  0a 20 0d 0a  20 0d 0a 20  .. .. .. ..   .. .. .. .. ....
```

How to output in hex format

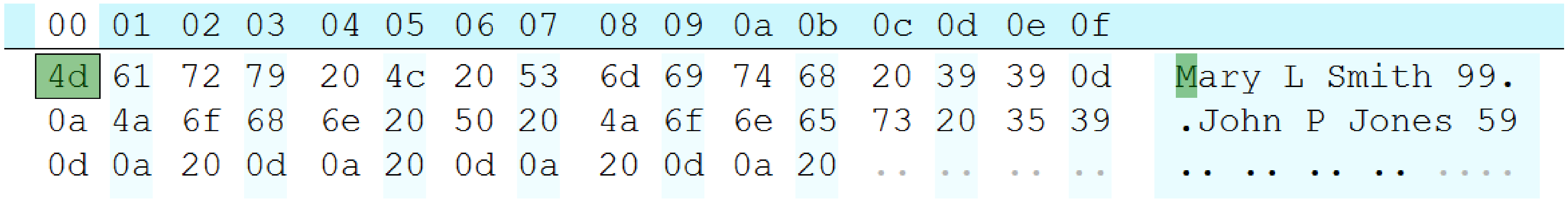
```
fstream input;
input.open("scores.txt",
   ios::in|ios::binary );
input.seekg(0, ios::beg);
while (true) {
   char c = input.get();
   if (input.eof()) break;
   cout << std::hex << c << " ";
}
```



```
Mary L Smith 99
John P Jones 59
```

```
 a r y   L   S m i t h   9 9
 J o h n   P   J o n e s   5 9
```

c is a char!



```
 00 01 02 03  04 05 06 07  08 09 0a 0b  0c 0d 0e 0f
 4d 61 72 79  20 4c 20 53  6d 69 74 68  20 39 39 0d   Mary L Smith 99.
 0a 4a 6f 68  6e 20 50 20  4a 6f 6e 65  73 20 35 39   .John P Jones 59
 0d 0a 20 0d  0a 20 0d 0a  20 0d 0a 20  .. .. .. ..   .. .. .. .. ....
```

Exercise Two
What are the output?

The file content in hexadecimal format:

```
34 12 21 12 58 39 b4 c8 00 be f3 3f
 0  1  2  3  4  5  6  7  8  9 10 11  (address)
```

```
char a1, b1, c1, d1;
fstream input; // Create a file object
input.open(cn_fileName, ios::in|ios::binary );
reportfilePointers(input);
input.seekg(-5, ios::end);
reportfilePointers(input);
input.read(reinterpret_cast<char*>(&a1), sizeof(a1));
reportfilePointers(input);
input.seekg(-2, ios::cur);
reportfilePointers(input);
input.read(reinterpret_cast<char*>(&b1), sizeof(b1));
input.seekg(-2, ios::cur);
input.read(reinterpret_cast<char*>(&c1), sizeof(c1));
input.seekg(-2, ios::cur);
input.read(reinterpret_cast<char*>(&d1), sizeof(d1));
reportData_signed_char(a1, b1, c1, d1)
```

```
tellp:0
tellg:0
tellp:?
tellg:?
tellp:?
tellg:?
tellp:?
tellg:?
a1:?
b1:?
c1:?
d1:?
```

Report Data

```
void reportData_signed_char (
char &a1,
char &b1,
char &c1,
char &d1
    )
{
    cout << "a1:" << hex << (short) a1 << endl;
    cout << "b1:" << hex << (short) b1 << endl;
    cout << "c1:" << hex << (short) c1 << endl;
    cout << "d1:" << hex << (short) d1 << endl;
}
```

Exercise Two
What are the output?

The file content in hexadecimal format:

```
34 12 21 12 58 39 b4 c8 00 be f3 3f
 0  1  2  3  4  5  6  7  8  9 10 11  (address)
```

```
char a1, b1, c1, d1;
fstream input; // Create a file object
input.open(cn_fileName, ios::in|ios::binary );
reportfilePointers(input);
input.seekg(-5, ios::end);
reportfilePointers(input);
input.read(reinterpret_cast<char*>(&a1), sizeof(a1));
reportfilePointers(input);
input.seekg(-4, ios::cur);
reportfilePointers(input);
input.read(reinterpret_cast<char*>(&b1), sizeof(b1));
input.seekg(2, ios::cur);
input.read(reinterpret_cast<char*>(&c1), sizeof(c1));
input.seekg(3, ios::cur);
input.read(reinterpret_cast<char*>(&d1), sizeof(d1));
reportData_signed_char(a1, b1, c1, d1)
```

```
tellp:0
tellg:0
tellp:7
tellg:7
tellp:8
tellg:8
tellp:4
tellg:4
a1:ffc8
b1:58
c1:ffc8
d1:3f
```

What happens if we make a mistake?

The file content in hexadecimal format:

```
34 12 21 12 58 39 b4 c8 00 be f3 3f
 0  1  2  3  4  5  6  7  8  9 10 11 (address)
```

```
unsigned char a1, b1, c1, d1;
fstream input; // Create a file object
input.open(cn_fileName, ios::in|ios::binary );
reportfilePointers(input);
input.seekg(1, ios::end);          // wrong, out of bound
reportfilePointers(input);
input.read(reinterpret_cast<char*>(&a1), sizeof(a1));
reportfilePointers(input);
input.seekg(-2, ios::cur);
reportfilePointers(input);
input.read(reinterpret_cast<char*>(&b1), sizeof(b1));
input.seekg(-2, ios::cur);
input.read(reinterpret_cast<char*>(&c1), sizeof(c1));
input.seekg(-2, ios::cur);
input.read(reinterpret_cast<char*>(&d1), sizeof(d1));
reportData(a1, b1, c1, d1)
```

```
a1:bf
b1:0
c1:2d
d1:6
```

These numbers cannot be found in the file, except for 0.

What happens if we make a mistake?

The file content in hexadecimal format:

```
34 12 21 12 58 39 b4 c8 00 be f3 3f
 0  1  2  3  4  5  6  7  8  9 10 11  (address)
```

```
unsigned char a1, b1, c1, d1;
fstream input; // Create a file object
input.open(cn_fileName, ios::in|ios::binary );
reportfilePointers(input);
input.seekg(1, ios::end);           // wrong, out of bound
reportfilePointers(input);
input.read(reinterpret_cast<char*>(&a1), sizeof(a1));
reportfilePointers(input);
input.seekg(-2, ios::cur);
reportfilePointers(input);
input.read(reinterpret_cast<char*>(&b1), sizeof(b1));
input.seekg(-2, ios::cur);
input.read(reinterpret_cast<char*>(&c1), sizeof(c1));
input.seekg(-2, ios::cur);
input.read(reinterpret_cast<char*>(&d1), sizeof(d1));
reportData(a1, b1, c1, d1)
```

```
tellp:0
tellg:0
tellp:13
tellg:13
tellp:-1
tellg:-1
tellp:-1
tellg:-1
a1:bf
b1:0
c1:2d
d1:6
```

What happens if we make a mistake?

The file content in hexadecimal format:

```
34 12 21 12 58 39 b4 c8 00 be f3 3f
 0  1  2  3  4  5  6  7  8  9 10 11  (address)
```

```
unsigned char a1, b1, c1, d1;
fstream input; // Create a file object
input.open(cn_fileName, ios::in|ios::binary );
reportfilePointers(input);
input.seekg(1, ios::end);             // wrong, out of bound
reportfilePointers(input);
input.read(reinterpret_cast<char*>(&a1), sizeof(a1));
reportfilePointers(input);
input.seekg(-2, ios::cur);
reportfilePointers(input);
input.read(reinterpret_cast<char*>(&b1), sizeof(b1));
input.seekg(-2, ios::cur);
input.read(reinterpret_cast<char*>(&c1), sizeof(c1));
input.seekg(-2, ios::cur);
input.read(reinterpret_cast<char*>(&d1), sizeof(d1));
reportData(a1, b1, c1, d1)
```

What is the value in decimal for the signed char bf? Need two's complement.

```
tellp:0
tellg:0
tellp:13
tellg:13
tellp:-1
tellg:-1
tellp:-1
tellg:-1
a1:bf
b1:0
c1:2d
d1:6
```

Binary numbers

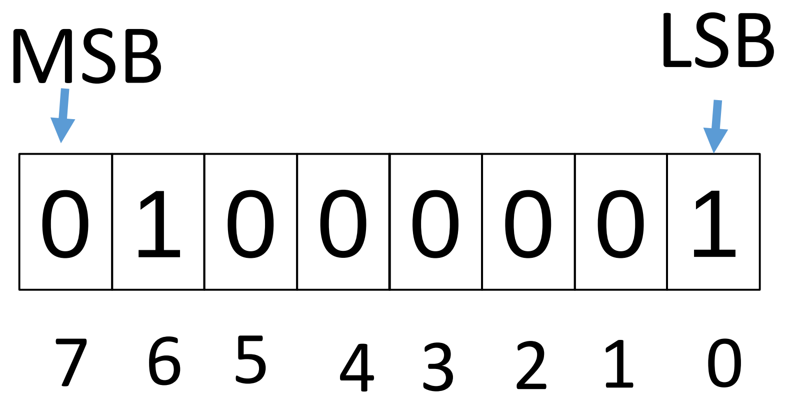
Digits: 1 (true) or 0 (false)

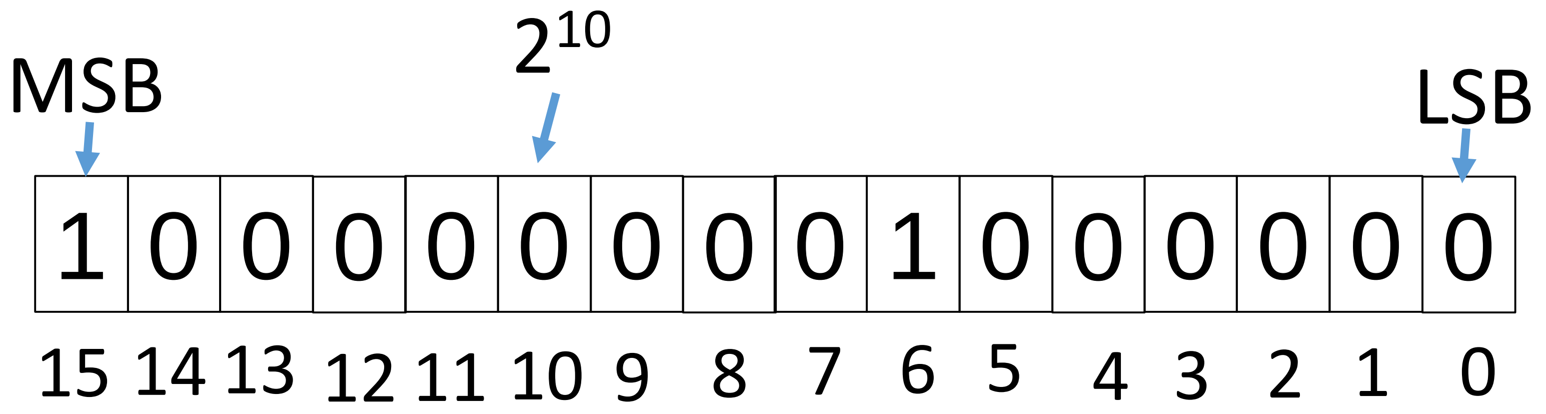
MSB: most significant bit

LSB: least significant bit

Bit numbering:

Position value.





Hexadecimal digits

Decimal	Hex	Decimal	Hex
0		10	A
1		11	B
2		12	C
3		13	D
4		14	E
5		15	F
6			
7			
8			
9			

Examples:

21 (Dec) = 15 (Hex)

16 (Dec) = 10 (Hex)

32 (Dec) = 20 (Hex)

257 (Dec) = 101 (Hex)

Position value: 256

Hexadecimal digits

Decimal	Hex	Decimal	Hex
0		10	A
1		11	B
2		12	C
3		13	D
4		14	E
5		15	F
6			
7			
8			
9			

```
  1001 1010 (Bin)
=    9    A (Hex)

  1111 0011 1101 0001 (Bin)
=    F    3    D    1 (Hex)
```

F3D1 (Hex)

It has two bytes, F3 and D1.

Hexadecimal digits

Decimal	Hex	Decimal	Hex
0		10	A
1		11	B
2		12	C
3		13	D
4		14	E
5		15	F
6			
7			
8			
9			

```
1D2B4A5F  (Hex)

Most significant byte: 1D
Least significant byte:5F
```

Endianness

Endianness: is the sequential order in which bytes are arranged into larger numerical values when stored in memory or when transmitted over digital links.

Big endian: whenever addressing memory, or sending or storing words bytewise, the most significant byte is stored first.

Big endian: whenever addressing memory, or sending or storing words bytewise, the least significant byte is stored first.

Source: wiki

Endianness

Big endian: whenever addressing memory, or sending or storing words bytewise, the most significant byte is stored first (has the lowest address). Example:

0x12345678 (in C++)

Store to memory: 12 34 56 78

$\longrightarrow$ Memory address

Little endian: whenever addressing memory, or sending or storing words bytewise, the least significant byte is stored first (the lowest address). Example:

0x12345678 (in C++)

Store to memory: 78 56 34 12

$\longrightarrow$ Memory address

Source: wiki

Endianness

```
1D2B4A5F  (Hex)     : 1D has the highest position value
                    : 5F has the lowest position value
Most significant byte: 1D
Least significant byte:5F
In memory, memory increasing from left to right
Big Endian:     1D 2B 4A 5F (LSB has the highest address)
Little Endian:  5F 4A 2B 1D (MSB has the highest address)
```

Two's complement for unsigned integers

10 (Dec)

= 0A (Hex)

= 0000 1010 (Bin)

-10 (Dec)

= -0A (Hex) ?

= -0000 1010 (Bin)?

Two’s complement for unsigned integers

10 (Dec)

= 0A (Hex)

= 0000 1010 (Bin)

-10 (Dec)

= -0A (Hex) ?

= -0000 1010 (Bin)?

10 (Dec)

= 0A (Hex)

= 0000 1010 (Bin)

Invert the bits =>

1111 0101

Add 1 to the result =>

1111 0110

Two’s complement for unsigned integers

-10 (Dec)

= -0A (Hex) ?

= -0000 1010 (Bin)?

10 (Dec)

= 0A (Hex)

= 0000 1010 (Bin)

Invert the bits =>

1111 0101

Add 1 to the result =>

1111 0110

MSB:

Sign bit

Two’s complement for unsigned integers

If MSByte >7,

the number is negative.

10 (Dec)

= 0A (Hex)

= 0000 1010 (Bin)

Invert the bits =>

1111 0101

Add 1 to the result =>

1111 0110

MSB:

Sign bit

Example

```
short int a = 7;
short int b = -7;

cout << "a:" << a << endl;
cout << "a(hex):" << hex << a << endl;

cout << "b:" << dec << b << endl;
cout << "b(hex):" << hex << b << endl;
```

```
Output:
a:7
a(hex):7
b:-7
b(hex):fff9
```

7 (Dec)

0000 0111 (Bin)

0000 0000 0000 0111

Invert bits
1111 1111 1111 1000

Add 1
1111 1111 1111 1001

=>
f f f 9
2 bytes

Example

```
int a = 7;
int b = -7;

cout << "a:" << a << endl;
cout << "a(hex):" << hex << a << endl;

cout << "b:" << dec << b << endl;
cout << "b(hex):" << hex << b << endl;

Output:
a:7
a(hex):7
b:-7
b(hex):fffffff9
```

7 (Dec)

0000 0111 (Bin)

0000 0000 0000 0000
0000 0000 0000 0111

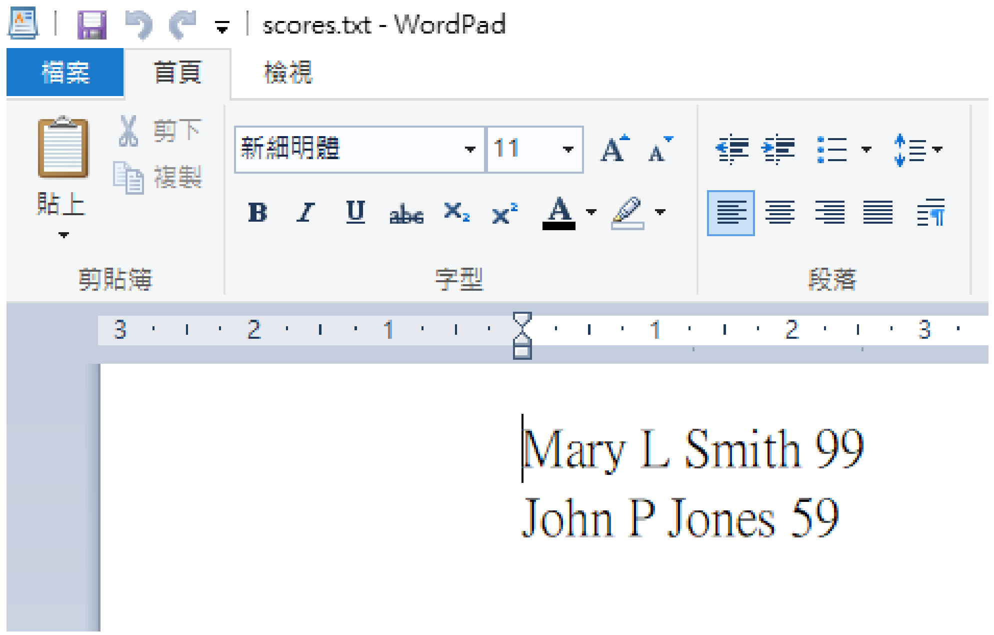
Invert bits
1111 1111 1111 1111
1111 1111 1111 1000

Add 1
1111 1111 1111 1111
1111 1111 1111 1001

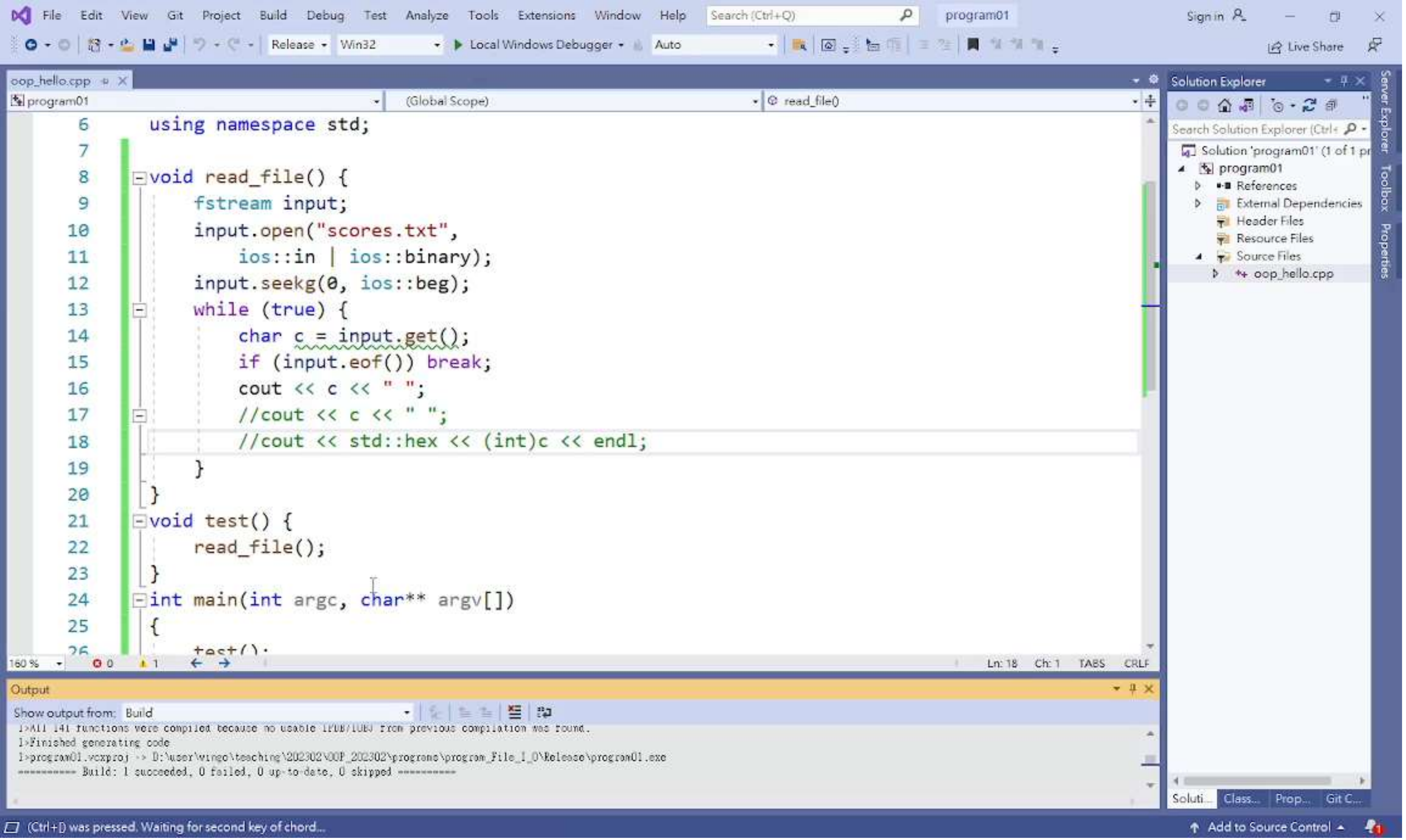
=>
f f f f f f f 9
Four bytes

An Example

```
void read_file() {
    fstream input;
    input.open("scores.txt",
    ios::in | ios::binary);
    input.seekg(0, ios::beg);
    while (true) {
        char c = input.get();
        if (input.eof()) break;
        cout << c << " ";
        //cout << c << " ";
        //cout << std::hex << (int)c << endl;
    }
}
```



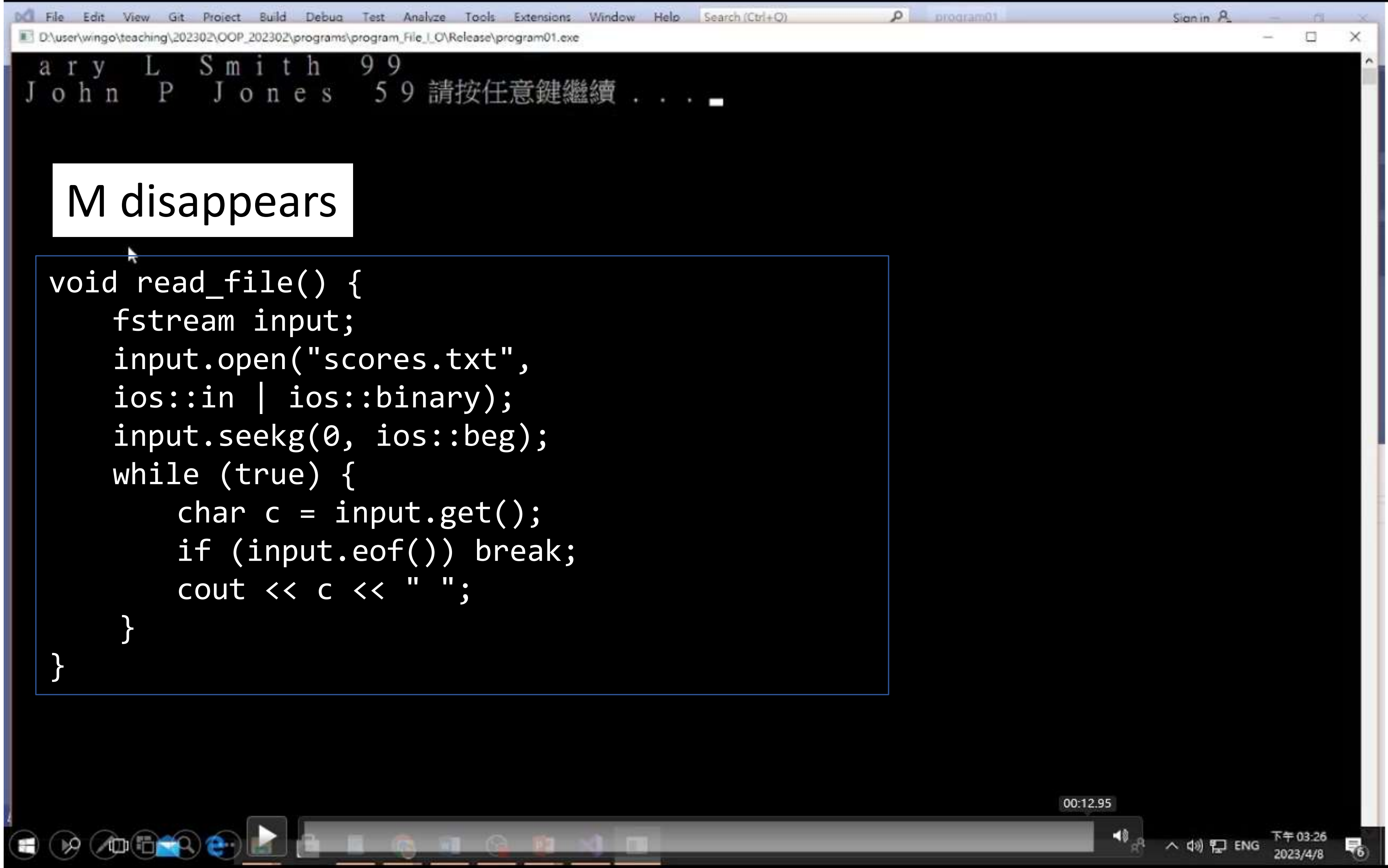
```
Mary L Smith 99
John P Jones 59
```



```cpp
using namespace std;

void read_file() {
    fstream input;
    input.open("scores.txt",
        ios::in | ios::binary);
    input.seekg(0, ios::beg);
    while (true) {
        char c = input.get();
        if (input.eof()) break;
        cout << c << " ";
        //cout << c << " ";
        //cout << std::hex << (int)c << endl;
    }
}
void test() {
    read_file();
}
int main(int argc, char** argv[])
{
    test();
```

```
1>All 141 functions were compiled because no usable IPDB/IOBJ from previous compilation was found.
1>Finished generating code
1>program01.vcxproj -> D:\user\wingo\teaching\202302\OOP_202302\programs\program_File_I_O\Release\program01.exe
========== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped ==========
```

(Ctrl+J) was pressed. Waiting for second key of chord...


D:\user\wingo\teaching\202302\OOP_202302\programs\program_File_I_O\Release\program01.exe

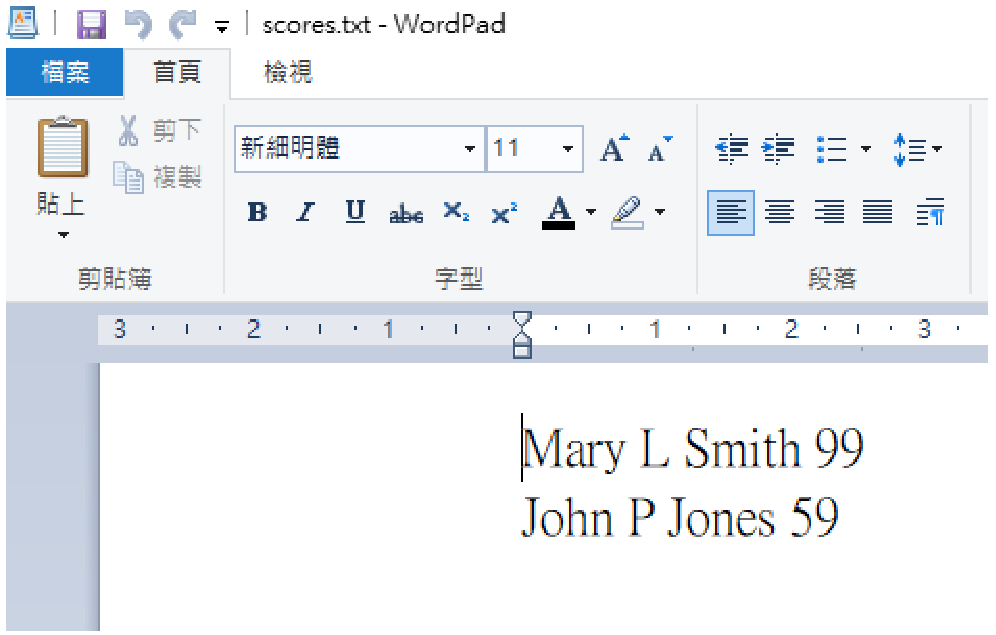
```
 a r y   L   S m i t h   9 9
J o h n   P   J o n e s   5 9 請按任意鍵繼續 . . .
```

M disappears

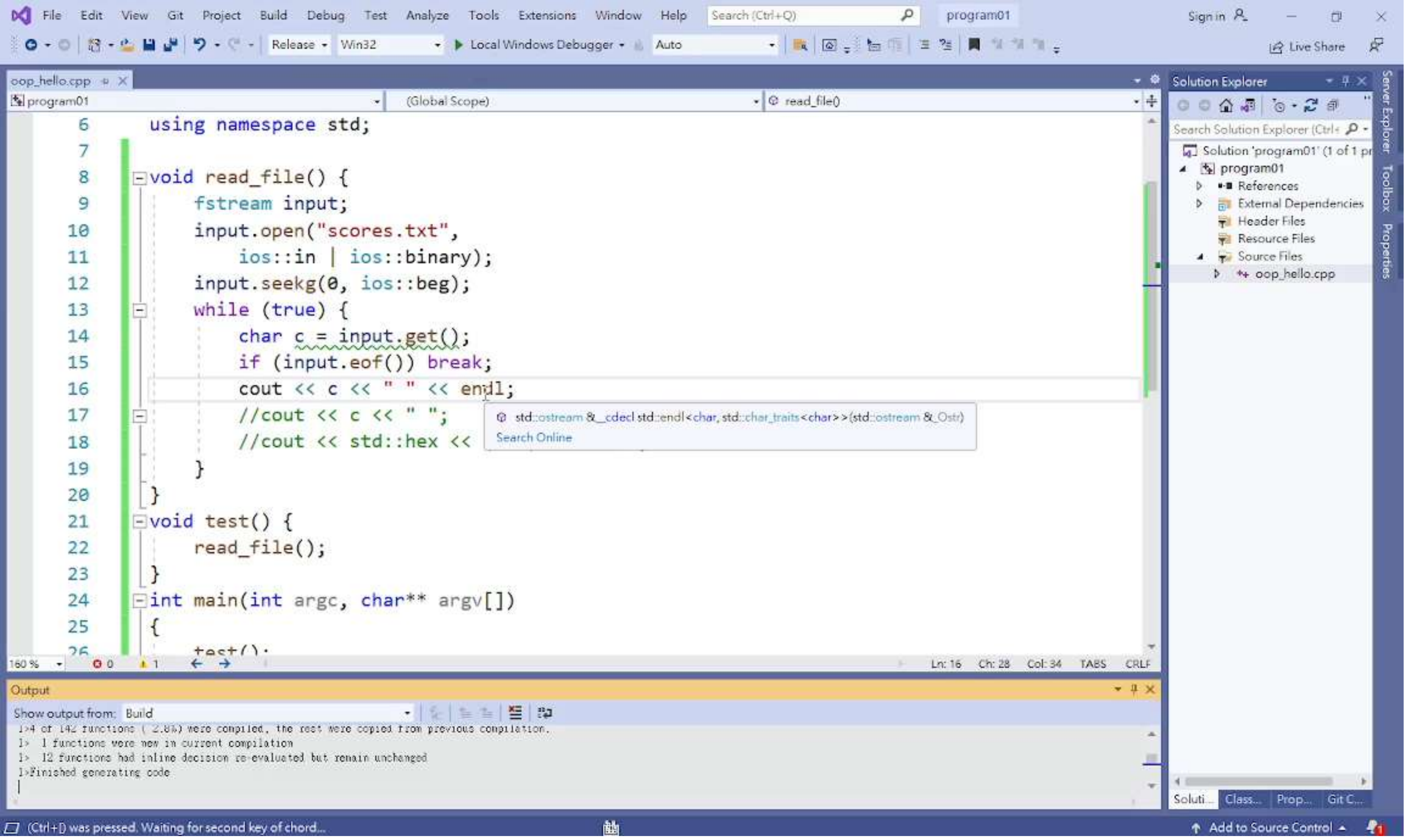
```
void read_file() {
    fstream input;
    input.open("scores.txt",
    ios::in | ios::binary);
    input.seekg(0, ios::beg);
    while (true) {
        char c = input.get();
        if (input.eof()) break;
        cout << c << " ";
    }
}
```

An Example

```
void read_file() {
    fstream input;
    input.open("scores.txt",
    ios::in | ios::binary);
    input.seekg(0, ios::beg);
    while (true) {
        char c = input.get();
        if (input.eof()) break;
        cout << c << " " << endl;
        //cout << c << " ";
        //cout << std::hex << (int)c << endl;
    }
}
```



Mary L Smith 99
John P Jones 59



```cpp
using namespace std;

void read_file() {
    fstream input;
    input.open("scores.txt",
        ios::in | ios::binary);
    input.seekg(0, ios::beg);
    while (true) {
        char c = input.get();
        if (input.eof()) break;
        cout << c << " " << endl;
        //cout << c << " ";
        //cout << std::hex <<
    }
}
void test() {
    read_file();
}
int main(int argc, char** argv[])
{
    test();
```

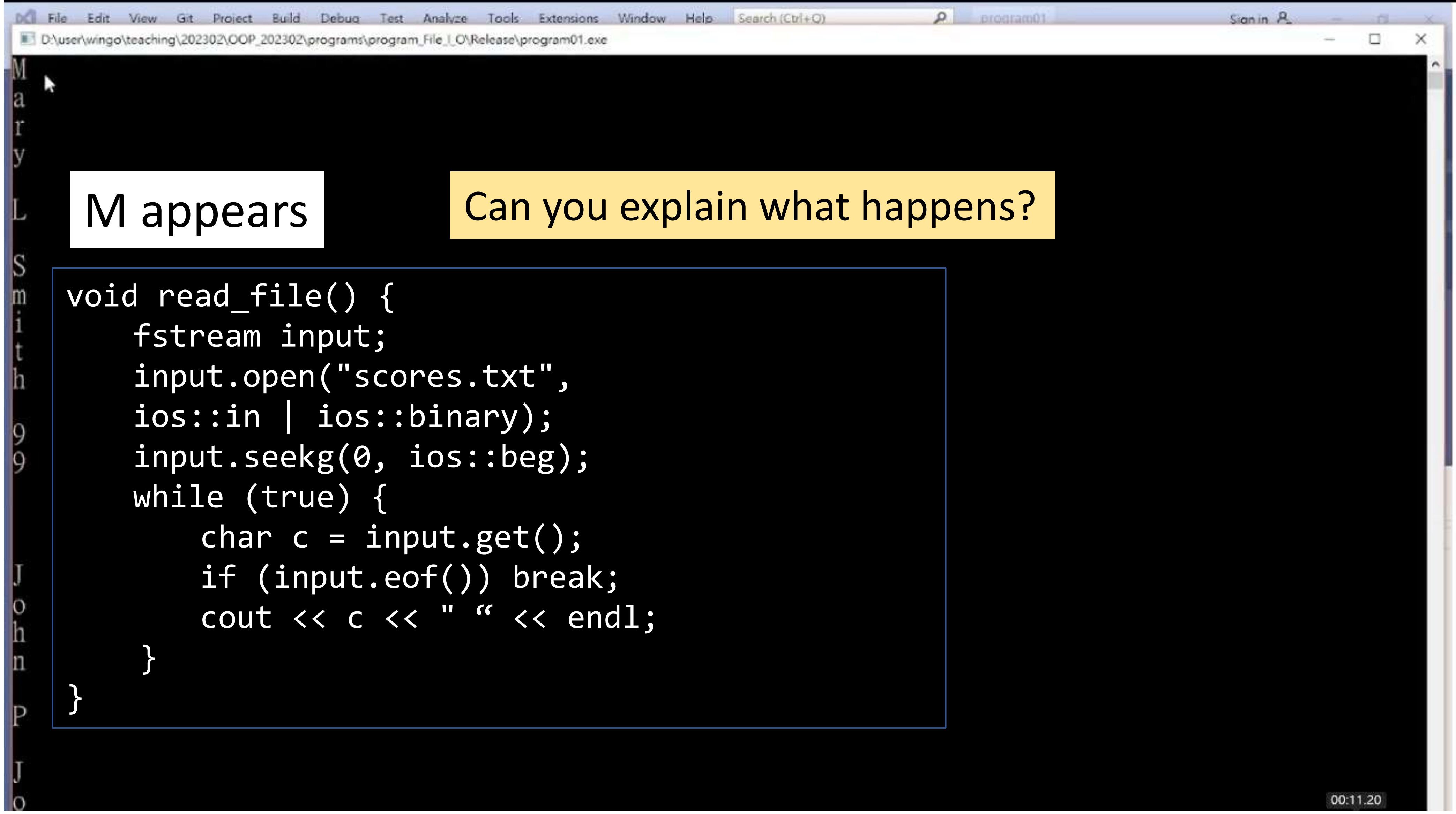
std::ostream &__cdecl std::endl<char, std::char_traits<char>>(std::ostream &_Ostr)
Search Online

Ln: 16 Ch: 28 Col: 34 TABS CRLF

Show output from: Build

```
1>4 of 142 functions ( 2.8%) were compiled, the rest were copied from previous compilation.
1>  1 functions were new in current compilation
1>  12 functions had inline decision re-evaluated but remain unchanged
1>Finished generating code
```

(Ctrl+D) was pressed. Waiting for second key of chord...


File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help Search (Ctrl+Q) program01 Sign in
D:\user\wingo\teaching\202302\OOP_202302\programs\program_File_I_O\Release\program01.exe
00:11.20

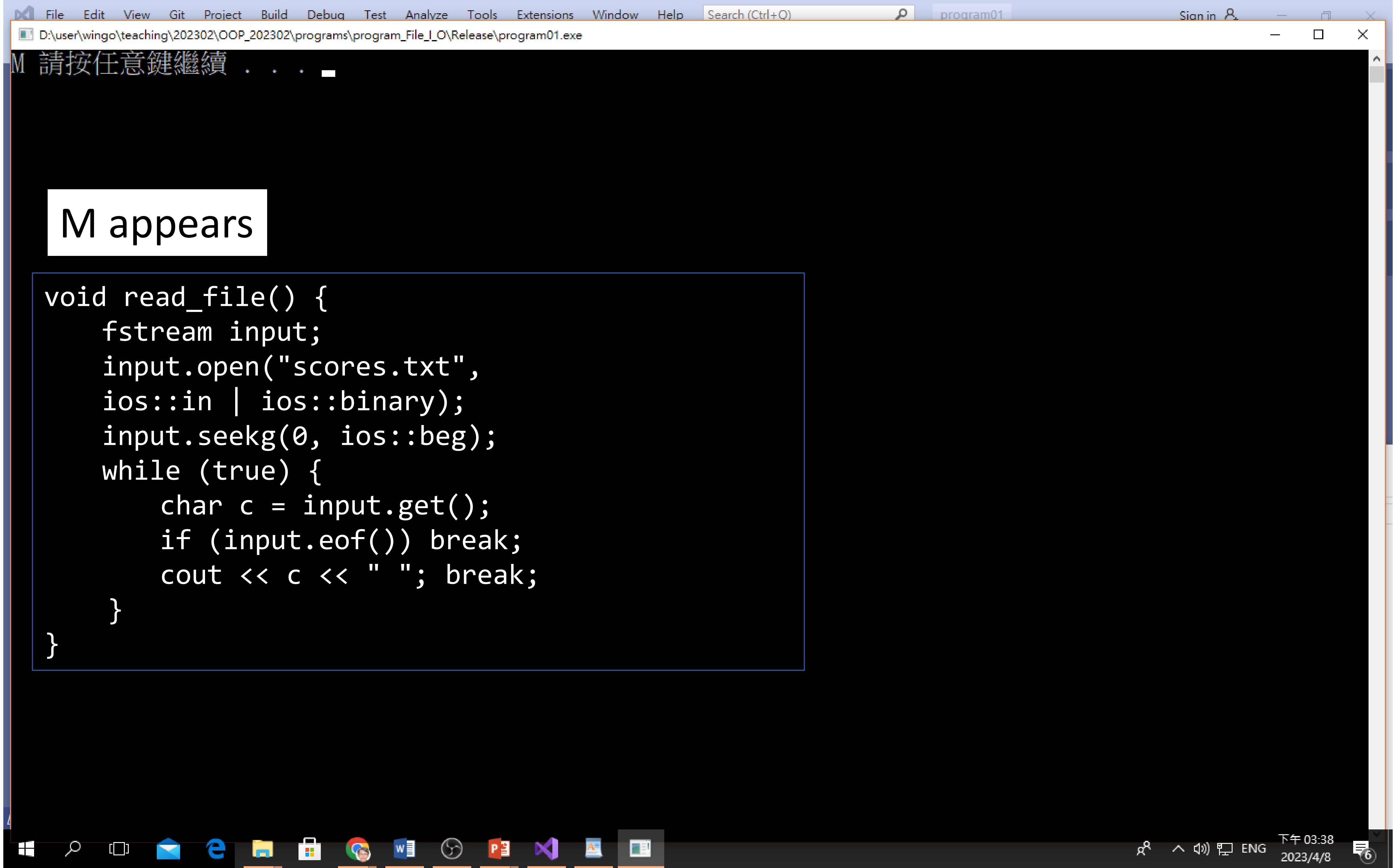
```
M
a
r
y

L

S
m
i
t
h

9
9



J
o
h
n

P

J
o
```

M appears

Can you explain what happens?

```
void read_file() {
    fstream input;
    input.open("scores.txt",
    ios::in | ios::binary);
    input.seekg(0, ios::beg);
    while (true) {
        char c = input.get();
        if (input.eof()) break;
        cout << c << " " << endl;
     }
}
```

As the character c is output, it can be a special character. The cursor is moved to ‘M’ and ‘M’ is cleared.



File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help
Search (Ctrl+Q)
program01
Sign in
D:\user\wingo\teaching\202302\OOP_202302\programs\program_File_I_O\Release\program01.exe
下午 03:38
2023/4/8
ENG

```
M 請按任意鍵繼續 . . .
```

M appears

```
void read_file() {
    fstream input;
    input.open("scores.txt",
    ios::in | ios::binary);
    input.seekg(0, ios::beg);
    while (true) {
        char c = input.get();
        if (input.eof()) break;
        cout << c << " "; break;
    }
}
```