

# Object Oriented Programming File Input / Output

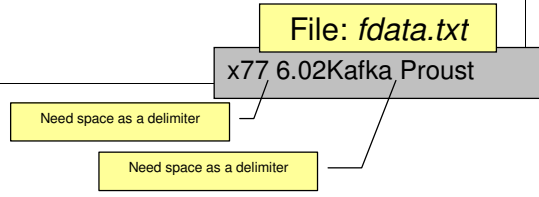
## Writing a Formatted File

```
// formato.cpp -- writes formatted output to a file, using <<
#include <iostream>
#include <fstream> // for file I/O
using namespace std;

#include <string>

void main()
{
  ofstream ofsOut("fdata.txt");// create ofstream object
  if (ofsOut.fail ())
    cout <<"Error opening output file"<<endl;
  else {
    char c = 'x';
    int j = 77;
    double d = 6.02;
    string sString1 = "Kafka"; // strings -- no embedded spaces
    string sString2 = "Proust";

    ofsOut<< c // write data
      << j
      << ' ' // needs space between numbers
      << d
      << sString1
      << ' ' // needs spaces between strings
      << sString2;
    ofsOut.close();
    cout << "File written\n";
  }
}
```



## Re-Reading the Formatted File Written Previously

```
// formati.cpp -- reads formatted output from a file, using >>
#include <iostream>
#include <fstream> // for file I/O
using namespace std;

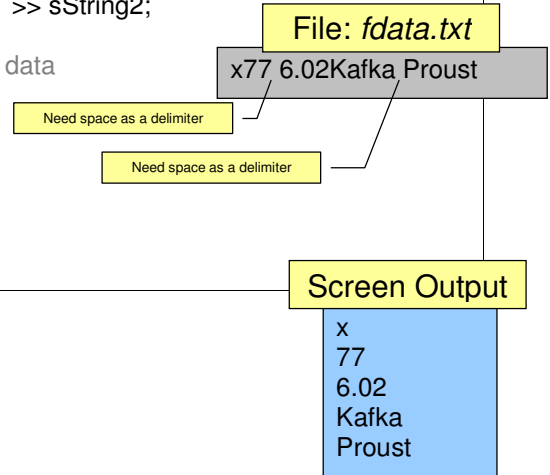
#include <string>

void main()
{
  ifstream ifsIn("fdata.txt"); // create ifstream object

  if (ifsIn.fail ())
    cout <<"Error opening input file"<<endl;
  else {
    char c;
    int j;
    double d;
    string sString1;
    string sString2;

    // fields are delimited by intervening spaces
    ifsIn >> c >> j >> d >> sString1 >> sString2;

    cout << c << endl // display the data
      << j << endl
      << d << endl
      << sString1 << endl
      << sString2 << endl;
    ifsIn.close();
  }
}
```



### String Input Using C++ *string* Object

```

void main ()
{
ifstream ifsPay("Payin1.txt"); // create ifstream object
if (ifsPay.fail()) // check success of open
    cout << "Error opening input file" << endl;
else { // file opened successfully
ofstream ofsPay; // create ofstream object to output
ofsPay.open("Payout.txt"); // open explicitly
if (ofsPay.fail() ) // check success of open
    cout << "Error opening output file" << endl;
else { // all files are opened properly, we can process
while ( ! ifsPay.eof() ) { // watch for the end-of-file
float fHours, fRate, fGrossPay;
string sName;
ifsPay >> sName >> fHours >> fRate;
if ( ! ifsPay.bad() ) {
fGrossPay = fHours * fRate;
ofsPay << sName << " " << fGrossPay << endl;
}
else
break; // terminate processing due to error
}
ofsPay.close();
}
ifsPay.close();
}
}

```

```

// stringin.cpp
#include <iostream>
#include <iomanip>
#include <fstream>
using namespace std;

#include <string>
#include <stdlib.h>

```

File: *Payin1.txt*

```

LindaCrane
20.5 100.00
JoeBlow
35.5 10.00
AYashin
5.0 50000.00

```

File: *Payout.txt*

```

LindaCrane 2050
JoeBlow 355
AYashin 250000

```

### String Input Using Array of *char*

```

void main ()
{
ifstream ifsPay("Payin2.txt"); // create ifstream object
if (ifsPay.fail()) // check success of open
    cout << "Error opening input file" << endl;
else { // file opened successfully
ofstream ofsPay; // create ofstream object to output
ofsPay.open("Payout.txt"); // open explicitly
if (ofsPay.fail() ) // check success of open
    cout << "Error opening output file" << endl;
else { // all files are opened properly, we can process
char szName[64]; // create temporary char array to hold input
float fHours, fRate, fGrossPay;
while ( ! ifsPay.eof() ) { // watch for the end-of-file
int nStrLen;
ifsPay >> nStrLen; // input the length of the following string
ifsPay.get(szName, nStrLen+1); // get characters up to limit

ifsPay >> fHours >> fRate;
if ( ! ifsPay.bad() ) {
fGrossPay = fHours * fRate;
ofsPay << szName << " " << fGrossPay << endl;
}
else
break; // terminate processing due to error
}
ofsPay.close();
}
ifsPay.close();
}
}

```

```

// chararr.cpp
#include <iostream>
#include <iomanip>
#include <fstream>
using namespace std;

#include <string.h>
#include <stdlib.h>

```

File: *Payin2.txt*

```

11Linda Crane
20.5 100.00
8Joe Blow
35.5 10.00
7AYashin
5.0 50000.00

```

Count of number of characters

File: *Payout.txt*

```

Linda Crane 2050
Joe Blow 355
AYashin 250000

```

Must handle embedded space in string

### Inputing *Student* Objects (containing Array of *char* for name)

```
class Student {
private:
    static const int nMAX_NAME_LEN = 31;
    long lld;
    char szName[nMAX_NAME_LEN+1];
    int nGrade;
public:
    Student() : lld(0), nGrade(0) { szName[0] = '\0'; }
    void InputFromKeyboard();
    void InputFromDisk(ifstream& rifsInFile);
    void OutputToScreen();
    void OutputToDisk(ofstream& rofsOut);
};
```

Fixed size array. Size predefined at compile time.

```
// studentio.cpp
#include <iostream>
#include <iomanip>
#include <fstream>
using namespace std;
```

```
#include <string.h>
#include <stdlib.h>
```

```
void Student::InputFromKeyboard()
{
    cout << "ID: "; cin >> lld;
    cout << "Name: "; cin >> szName;
    cout << "Grade: "; cin >> nGrade;
}
```

```
void Student::OutputToScreen()
{
    cout << "ID:" << setw(5) << lld << " Name:" << szName
    << " Grade:" << setw(3) << nGrade << endl;
}
```

```
void Student::InputFromDisk(ifstream& risInFile)
{
    int nStrLen;
    risInFile >> lld >> nStrLen;
    risInFile.get(szName, nStrLen + 1);
    risInFile >> nGrade;
}
```

```
void Student::OutputToDisk(ofstream& rofsOut)
{
    rofsOut << lld << ' ' << strlen(szName) << szName
    << ' ' << nGrade << endl;
}
```

### Initial Version: Array of Objects

```
void main()
{
    const int nMAX = 100;
    Student asCourse[nMAX];
    int nNumStudents = 0;

    ifstream ifsInputFile("input.txt");

    while (ifsInputFile.peek() != EOF && nNumStudents < nMAX) {
        asCourse[nNumStudents].InputFromDisk(ifsInputFile);
        nNumStudents++;
    }
    ifsInputFile.close();

    ofstream ofsOutputFile("output.txt");
    for (int i = 0; i < nNumStudents; i++) {
        asCourse[i].OutputToScreen();
        asCourse[i].OutputToDisk(ofsOutputFile);
    }
    ofsOutputFile.close();
}
```

File: *input.txt*

```
12345 3Jim 95
23456 4Bill 78
34567 4Jill 97
45678 6Janice 83
```

Screen Output

```
ID:12345 Name:Jim Grade: 95
ID:23456 Name:Bill Grade: 78
ID:34567 Name:Jill Grade: 97
ID:45678 Name:Janice Grade: 83
```

### Inputing Student Objects (containing pointer to Array of char)

```
class Student {
private:
    long lld;
    char* pszName;
    int nGrade;
public:
    Student() : lld(0), pszName(NULL), nGrade(0) { }
    ~Student() { delete [ ] pszName; }
    void InputFromKeyboard();
    void InputFromDisk(ifstream& rifsInFile);
    void OutputToScreen();
    void OutputToDisk(ofstream& rofsOut);
};
```

Pointer to array to be created later.

```
// studentio.cpp
#include <iostream>
#include <iomanip>
#include <fstream>
using namespace std;

#include <string.h>
#include <stdlib.h>
```

```
void Student::InputFromKeyboard()
{
    cout << "ID:";   cin >> lld;
    char szTemp[128]; cout << "Name:"; cin >> szTemp;
    pszName = strcpy(new char[strlen(szTemp) + 1], szTemp);
    cout << "Grade:"; cin >> nGrade;
}
```

```
void Student::OutputToScreen()
{
    cout << "ID:" << setw(5) << lld << " Name:" << pszName
        << " Grade:" << setw(3) << nGrade << endl;
}
```

```
void Student::InputFromDisk(ifstream& risInFile)
{
    int nStrLen;
    risInFile >> lld >> nStrLen;
    pszName = new char[nStrLen+1];
    risInFile.get(pszName, nStrLen + 1);
    risInFile >> nGrade;
}
```

```
void Student::OutputToDisk(ofstream& rofsOut)
{
    rofsOut << lld << ' ' << strlen(pszName) << pszName
        << ' ' << nGrade << endl;
}
```

### Initial Version: Array of Objects

```
void main()
{
    const int nMAX = 100;
    Student asCourse[nMAX];
    int nNumStudents = 0;

    ifstream ifsInputFile("input.txt");

    while (ifsInputFile.peek() != EOF && nNumStudents < nMAX) {
        asCourse[nNumStudents].InputFromDisk(ifsInputFile);
        nNumStudents++;
    }
    ifsInputFile.close();

    ofstream ofsOutputFile("output.txt");
    for (int i = 0; i < nNumStudents; i++) {
        asCourse[i].OutputToScreen();
        asCourse[i].OutputToDisk(ofsOutputFile);
    }
    ofsOutputFile.close();
}
```

File: input.txt

```
12345 3Jim 95
23456 4Bill 78
34567 4Jill 97
45678 6Janice 83
```

Screen Output

```
ID:12345 Name:Jim Grade: 95
ID:23456 Name:Bill Grade: 78
ID:34567 Name:Jill Grade: 97
ID:45678 Name:Janice Grade: 83
```

### Improved Version: Array of Pointers to Objects

```
void main()
{
const int nMAX = 100;
Student* apsCourse[nMAX];
int nNumStudents = 0;

ifstream ifsInputFile("input.txt");

while (ifsInputFile.peek() != EOF && nNumStudents < nMAX) {
apsCourse[nNumStudents] = new Student();
apsCourse[nNumStudents]->InputFromDisk(ifsInputFile);
nNumStudents++;
}
ifsInputFile.close();

ofstream ofsOutputFile("output.txt");
for (int i = 0; i < nNumStudents; i++) {
apsCourse[i]->OutputToScreen();
apsCourse[i]->OutputToDisk(ofsOutputFile);
}
ofsOutputFile.close();

for (i = 0; i < nNumStudents; i++)
delete apsCourse[i];
}
```

File: *input.txt*  
12345 3Jim 95  
23456 4Bill 78  
34567 4Jill 97  
45678 6Janice 83

Screen Output  
ID:12345 Name:Jim Grade: 95  
ID:23456 Name:Bill Grade: 78  
ID:34567 Name:Jill Grade: 97  
ID:45678 Name:Janice Grade: 83