

C++程序设计(拾叁)

徐东/计算数学

内容

- 文本文件的读写

任务伍拾

- 将89位学生的个人信息永久保存
 - 姓名
 - 性别
 - 年龄
 - 身高
 - 体重
- 永久保存：将数据保存至外部文件中

数据写入文本文件的步骤

1. `#include<fstream>`
2. 打开输出文件
3. 向目标文件写入数据
4. 关闭目标文件

打开输出文件

- **语法**

- **ofstream 变量名(“文件路径” [, 文件打开模式]);**

- **变量名**

- **符合C++标识符命名规范**
 - **在后续代码中用于指代目标文件**

打开输出文件

- 语法

- `ofstream` 变量名(“文件路径” [, 文件打开模式]);

- 文件路径

- `char[]`

- 文件路径中的斜杠要双写 “\\”

- 注意区分绝对路径与相对路径

打开输出文件

- 语法

- `ofstream 变量名(“文件路径” [, 文件打开模式]);`

- 文件打开模式

- `ios::out`

- 覆盖模式(默认模式)

- `ios::app`

- 追加模式

打开输出文件

- 语法

- `ofstream 变量名(“文件路径” [, 文件打开模式]);`

- `ofstream out_file(“result.txt”);`

- 相对路径

- 覆盖模式

打开输出文件

- //通过string变量保存目标文件的文件名
- `string filename;`
- `cin>>filename;`
- `ofstream out_file(filename.c_str());`
 - `string.c_str()` 将string转换成相应的char[]形式
 - `ofstream/ifstream` 只接受以char[]方式出现的文件路径

向目标文件写入数据

- 语法

- 输出文件变量名 << 数据项;

- 使用 “输出文件变量名” 代替 cout

- `out_file << "this is a text" << 6 + 9 << endl;`

向目标文件写入数据

- 语法
 - 输出文件变量名 << 数据项;
- 文件数据写入行为类似于标准输出
 - <<
 - put()
 - 格式化输出

关闭目标文件

- 语法
 - 输出文件变量名.close();
- 目标文件关闭后不能再向其写入数据

任务伍拾

```
#include <iostream>
#include <fstream>
using namespace std;
.....
int main(){
    const int NUMBER = 89;
    Student one[NUMBER] ;
    ..... // 处理学生信息

    ofstream out_file("d:\\students_information.txt");

    for(int i = 0; i < NUMBER; ++i){
        out_file << one[i].name << "\\t" << one[i].gender << "\\t" << one[i].age
            << "\\t" << one[i].height << "\\t" << one[i].weight << endl;
    }

    out_file.close();

    return 0;
}
```

任务伍拾

```
#include <iostream>
#include <fstream>
using namespace std;
.....

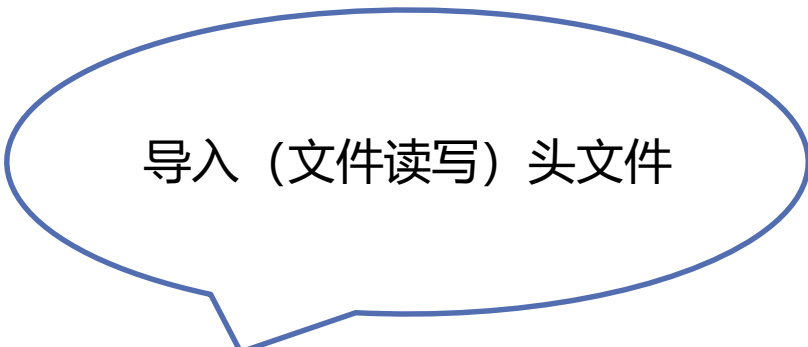
int main(){
    const int NUMBER = 89;
    Student one[NUMBER] ;
    ..... // 处理学生信息

    ofstream out_file("d:\\students_information.txt");

    for(int i = 0; i < NUMBER; ++i){
        out_file << one[i].name << "\\t" << one[i].gender << "\\t" << one[i].age
            << "\\t" << one[i].height << "\\t" << one[i].weight << endl;
    }

    out_file.close();

    return 0;
}
```



导入（文件读写）头文件

任务伍拾

```
#include <iostream>
#include <fstream>
using namespace std;
.....
```

```
int main(){
    const int NUMBER = 89;
    Student one[NUMBER] ;
    ..... // 处理学生信息
```

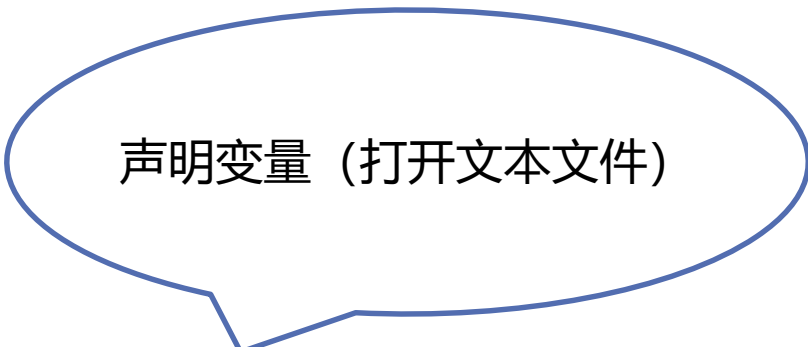
```
    ofstream out_file("d:\\students_information.txt");
```

```
    for(int i = 0; i < NUMBER; ++i){
        out_file << one[i].name << "\\t" << one[i].gender << "\\t" << one[i].age
            << "\\t" << one[i].height << "\\t" << one[i].weight << endl;
    }
```

```
    out_file.close();
```

```
    return 0;
```

```
}
```



声明变量 (打开文本文件)

任务伍拾

```
#include <iostream>
#include <fstream>
using namespace std;
.....

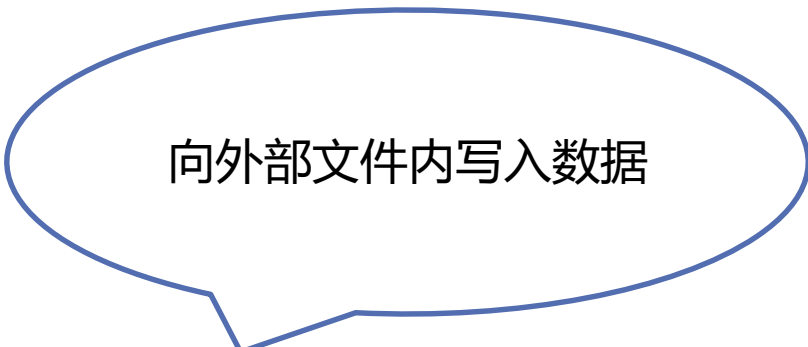
int main(){
    const int NUMBER = 89;
    Student one[NUMBER] ;
    ..... // 处理学生信息

    ofstream out_file("d:\\students_information.txt");

    for(int i = 0; i < NUMBER; ++i){
        out_file << one[i].name << "\\t" << one[i].gender << "\\t" << one[i].age
            << "\\t" << one[i].height << "\\t" << one[i].weight << endl;
    }

    out_file.close();

    return 0;
}
```



向外部文件内写入数据

任务伍拾

```
#include <iostream>
#include <fstream>
using namespace std;
.....

int main(){
    const int NUMBER = 89;
    Student one[NUMBER] ;
    ..... // 处理学生信息

    ofstream out_file("d:\\students_information.txt");

    for(int i = 0; i < NUMBER; ++i){
        out_file << one[i].name << "\\t" << one[i].gender << "\\t" << one[i].age
            << "\\t" << one[i].height << "\\t" << one[i].weight << endl;
    }

    out_file.close();

    return 0;
}
```



关闭外部文件

任务伍拾: cout 和 文件读写

```
#include <iostream>
#include <fstream>
using namespace std;
.....

int main(){
    const int NUMBER = 89;
    Student one[NUMBER] ;
    ..... // 处理学生信息

    ofstream out_file("d:\\students_information.txt");

    for(int i = 0; i < NUMBER; ++i){
        cout << one[i].name << "\\t" << one[i].gender << "\\t" << one[i].age
            << "\\t" << one[i].height << "\\t" << one[i].weight << endl;
    }

    out_file.close();

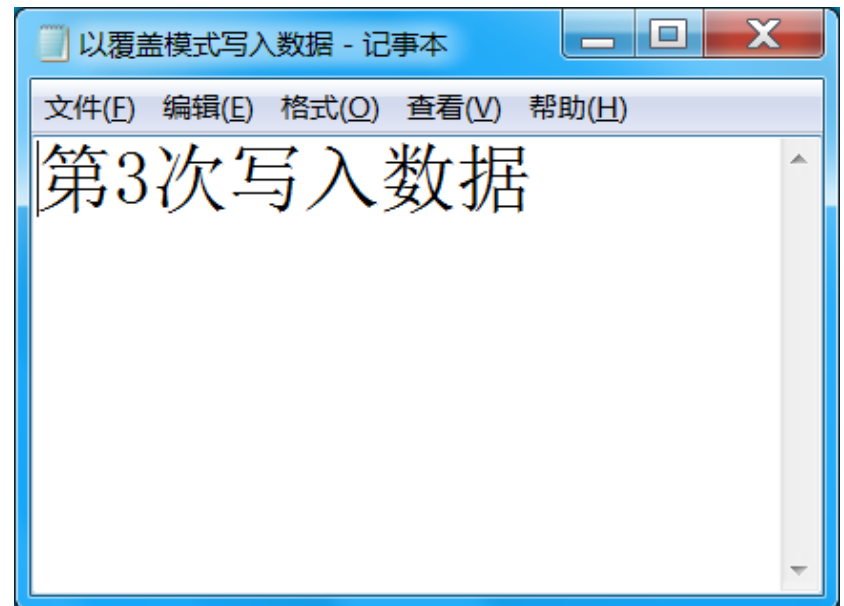
    return 0;
}
```



控制台输出

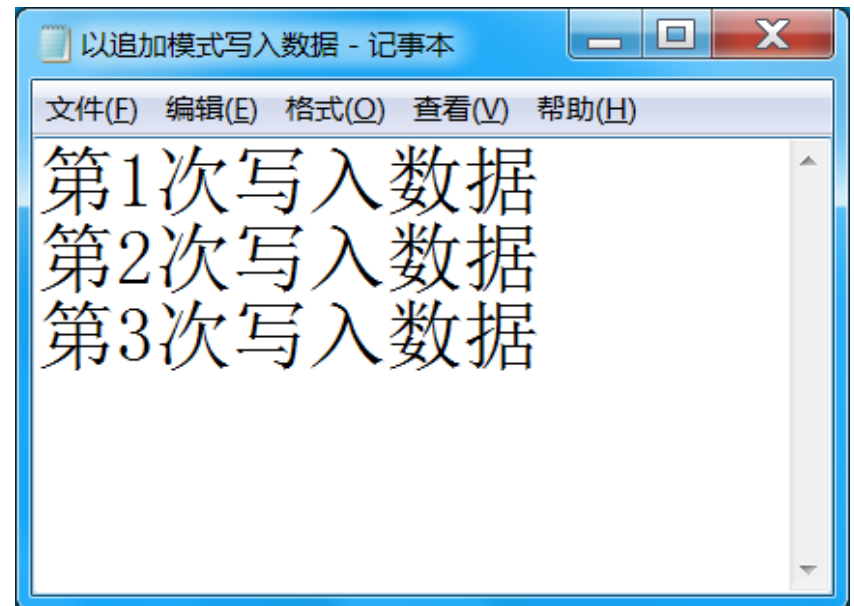
将数据保存至外部的文本文件

- `for(int i = 0; i < 3; ++i)`
- `{`
- `ofstream out_file("以覆盖模式写入数据.txt", ios::out);`
- `out_file << "第" << i + 1 << "次写入数据" << endl;`
- `out_file.close();`
- `}`



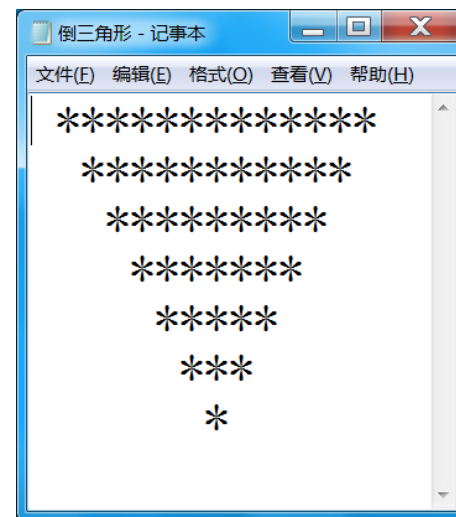
将数据保存至外部的文本文件

- `for(int i = 0; i < 3; ++i)`
- `{`
- `ofstream out_file("以追加模式写入数据.txt", ios::app);`
- `out_file << "第" << i + 1 << "次写入数据" << endl;`
- `out_file.close();`
- `}`



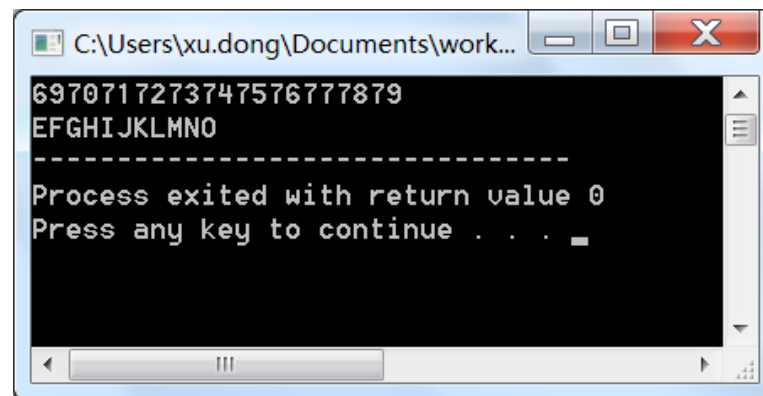
格式化输出

- `#include<fstream>`
- `#include<iomanip>`
- `using namespace std;`
- `int main()`
- `{`
- `ofstream out_file("c:\\倒三角形.txt");`
- `for(int n = 1; n < 8; ++n)`
- `{`
- `out_file << setfill(' ') << setw(n) << " "`
- `<< setfill('*') << setw(15-2*n) << "*" << endl;`
- `}`
- `out_file.close();`
- `return 0;`
- `}`



put()

```
for(int i = 69; i < 80; ++i){  
    cout << i ;  
}  
cout << endl;  
for(int j = 69; j < 80; ++j){  
    cout.put(j);  
}
```

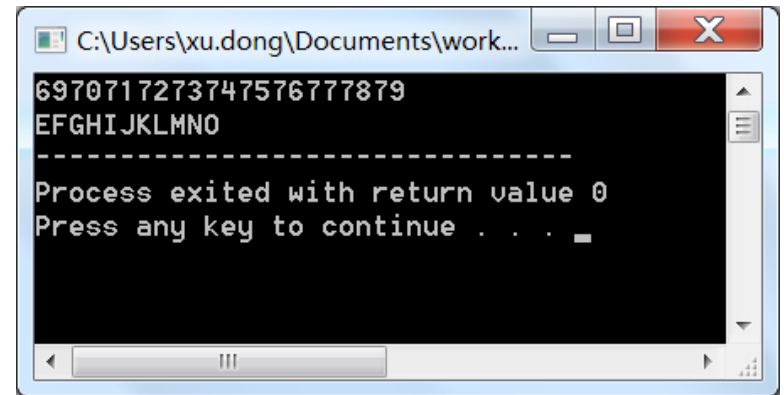


```
C:\Users\xu.dong\Documents\work...  
6970717273747576777879  
EFGHIJKLMNO  
-----  
Process exited with return value 0  
Press any key to continue . . .
```

- `cout << letter;`
 - 以`letter`的数据类型所表示的形式显示

put()

```
for(int i = 69; i < 80; ++i){  
    cout << i ;  
}  
cout << endl;  
for(int j = 69; j < 80; ++j){  
    cout.put(j);  
}
```



```
C:\Users\xu.dong\Documents\work...  
6970717273747576777879  
EFGHIJKLMNO  
-----  
Process exited with return value 0  
Press any key to continue . . .
```

- `cout.put(letter);`
 - 将参数值`letter`以字符方式显示

put()

- `#include<fstream>`
- `using namespace std;`
- `int main()`
- `{`
- `ofstream out_file("letters.txt");`
- `for(int j = 65; j <= 90; ++j)`
- `{`
- `out_file.put(j);`
- `}`
- `out_file.close();`
- `return 0;`
- `}`



读写外部文件时必须判断文件是否正常打开

- 检查文件是否正确打开的方式
 1. 判断文件变量名是否等于零(false)
 2. 判断fail()成员函数是否等于1(true)
- 文件所处目录必须存在
 - 否则无法新建文件

读写外部文件时必须判断文件是否正常打开

- 检查文件是否正确打开的方式

1. 判断文件变量名是否等于零(false)

- `ofstream out_file("x:\\M007.txt");`
- `if(out_file == 0)`
- `{`
- `cout<<"文件无法正常打开"<<endl;`
- `return 0;`
- `}`

读写外部文件时必须判断文件是否正常打开

- 检查文件是否正确打开的方式

- 2. 判断fail()成员函数是否等于1(true)

- ofstream out_file(“x:\\M007.txt”);
- if(out_file.fail()) //fail()==1 表示失败
- {
- cout<<“文件无法正常打开”<<endl;
- return 0;
- }

任务伍拾壹

- 将89位学生的个人信息保存至 “d:\temp.txt” 文件中
 - 姓名
 - 性别
 - 年龄
 - 身高
 - 体重
 - BMI

任务伍拾贰

- 按照 BMI 的升序方式，保存学生的个人信息至指定文件。
- 要求：
 - 外部文件由用户指定
 - 每行保存一位学生的个人信息
 - 各数据项之间以逗号分隔（CSV, Comma-Separated Values）

任务伍拾叁

- 假设文件 “d:\data.txt” 是 “任务伍拾贰” 的结果文件
- 现，要求将结果文件中的学生身高由 单位 m 改为 cm 。
- 解决方法
 - 读取文件中的数据

从文本文件读取数据的步骤

1. `#include<fstream>`
2. 打开输入文件
3. 判断输入文件是否已被正确打开
4. 根据数据的保存格式读取文件中的数据
5. 关闭目标文件

打开输入文件

- 语法

- **ifstream** 输入文件变量名(“文件路径” [,ios::in]);

- `ifstream in_file("x:\\M00008.txt");`

- `ifstream in_file("x:\\M00008.txt", ios::in);`

判断输入文件是否已被正确打开

- 读取文件数据前
 - **必须**检查目标文件是否已被成功打开
- 目标文件未被正确打开
 - 输入文件变量名 == 0
 - 输入文件变量名.fail() == 1

读取保存在文件中的数据

- 语法

- **输入文件变量名** >> 变量名1 >> . . . >> 变量名n;

- 使用 **输入文件变量名** 代替cin

读取保存在文件中的数据

- 语法

- **输入文件变量名** >> 变量名1 >> . . . >> 变量名n;

- 读取文件数据的行为类似于标准输入

- >>

- `getline()`

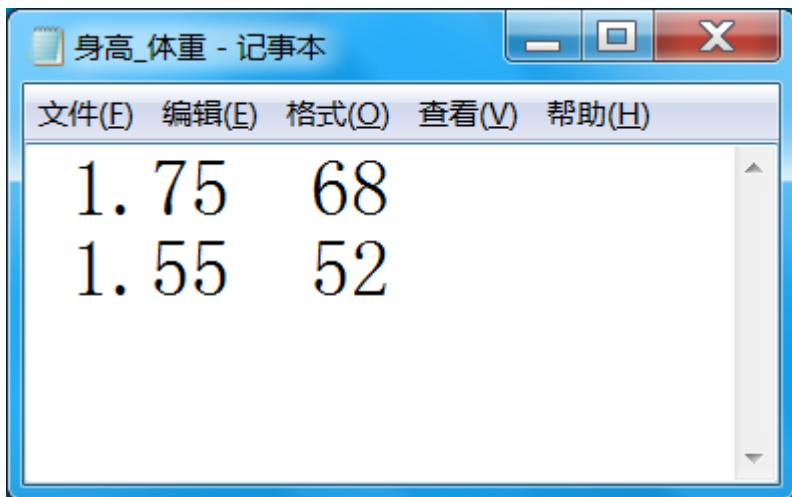
- `get()`

根据数据的保存格式读取文件中的数据

- 以顺序(流)方式读取文件中的数据
- 使用流提取运算符，从文件中读取基本数据类型的数据。
- 根据变量的数据类型（基本数据类型），系统依次从文件中读取数据并保存至目标变量中。
- 空格、回车或换行作为数据项之间的分隔符。
- 根据目标文件的格式，设计读取数据的代码。

根据数据的保存格式读取文件中的数据

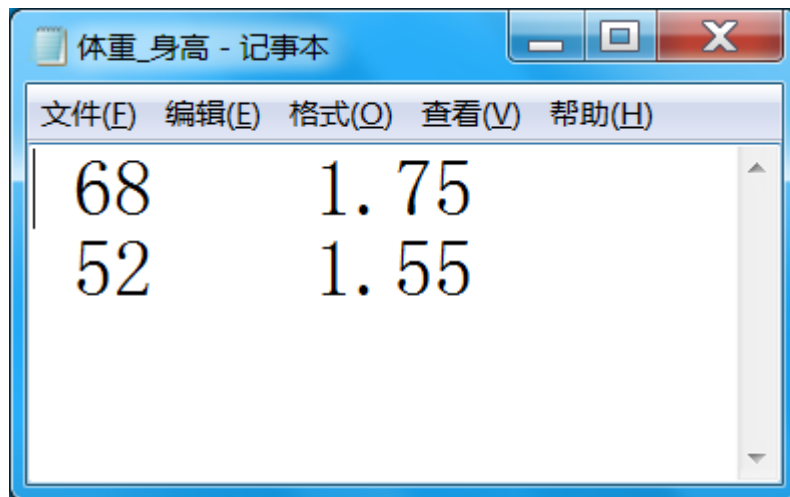
文件1



A Notepad window titled "身高_体重 - 记事本" (Height_Weight - Notepad). The menu bar includes "文件(F)", "编辑(E)", "格式(O)", "查看(V)", and "帮助(H)". The text area contains two lines of data, each with two columns separated by a space.

1.75	68
1.55	52

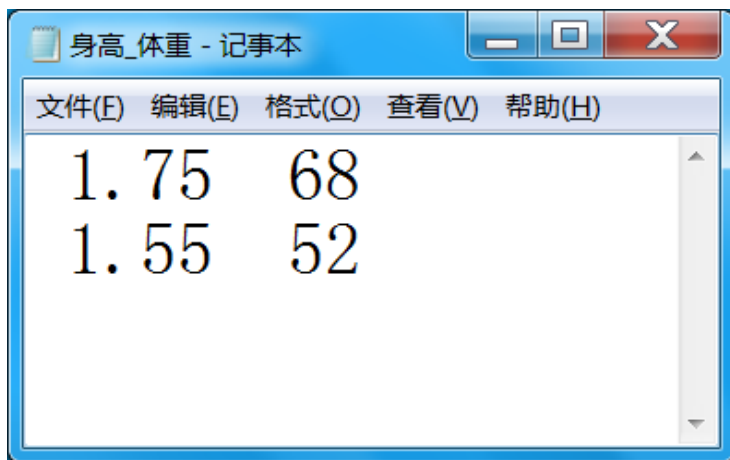
文件2



A Notepad window titled "体重_身高 - 记事本" (Weight_Height - Notepad). The menu bar includes "文件(F)", "编辑(E)", "格式(O)", "查看(V)", and "帮助(H)". The text area contains two lines of data, each with two columns separated by a space. The columns are swapped compared to File 1.

68	1.75
52	1.55

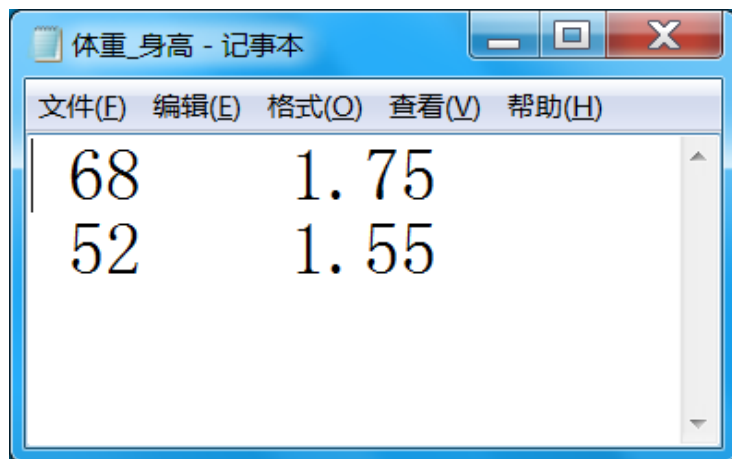
根据数据的保存格式读取文件中的数据



```
ifstream infile("身高_体重.txt");  
infile >> height >> weight;
```

- 文件中的数据保存格式（统一）
 - 每行按统一格式保存一组记录
 - 身高 体重
 - 数据项之间以空格分隔

根据数据的保存格式读取文件中的数据



```
ifstream infile("体重_身高.txt");  
infile >> weight >> height;
```

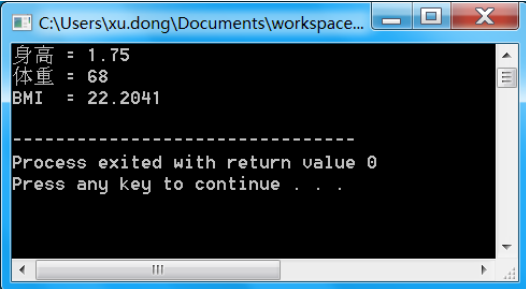
- 文件中的每行按统一格式保存数据
 - 体重 身高
 - 数据项之间以空格分隔

关闭目标文件

- 语法
 - 输入文件变量名.close();
- 目标文件关闭后不能再从文件中读取数据

计算并输出BMI指数

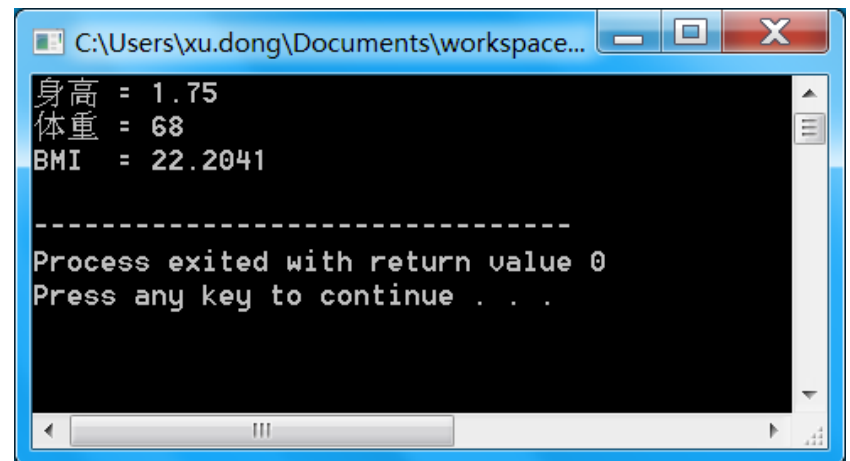
```
• int main()
• {
•     ifstream in_file("c:\\Users\\Documents\\workspace\\身高_体重.txt");
•     if(in_file==0){
•         cerr<<"无法打开文件"<<endl;
•         return 0;
•     }
•     double height = 0.0 , weight = 0.0;
•     in_file >> height >> weight;
•     cout<<"身高 = " << height <<endl
•         <<"体重 = " << weight <<endl
•         <<"BMI  = " << weight/(height*height) <<endl;
•     in_file.close();
•     return 0;
• }
```



```
C:\Users\xu.dong\Documents\workspace...
身高 = 1.75
体重 = 68
BMI = 22.2041
-----
Process exited with return value 0
Press any key to continue . . .
```

按固定格式从文件中反复读取数据

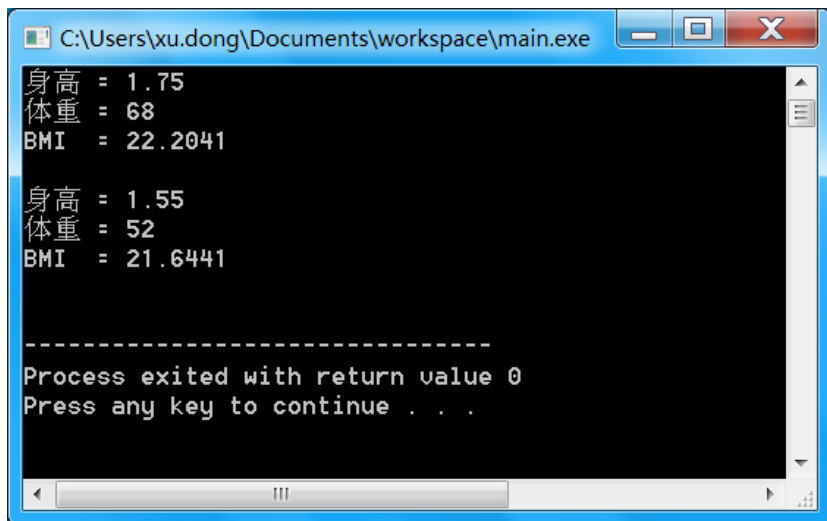
1. `in_file >> height >> weight;`
 2. `cout<<"身高 = " << height <<endl`
 - `<<"体重 = " << weight <<endl`
 - `<<"BMI = " << weight/(height*height) <<endl;`
- `in_file>>height>>weight;`
 - 只执行一次
 - 只从目标文件中读取
 - 一个身高
 - 一个体重



```
C:\Users\xu.dong\Documents\workspace...  
身高 = 1.75  
体重 = 68  
BMI = 22.2041  
  
-----  
Process exited with return value 0  
Press any key to continue . . .
```

按固定格式从文件中反复读取数据

要求



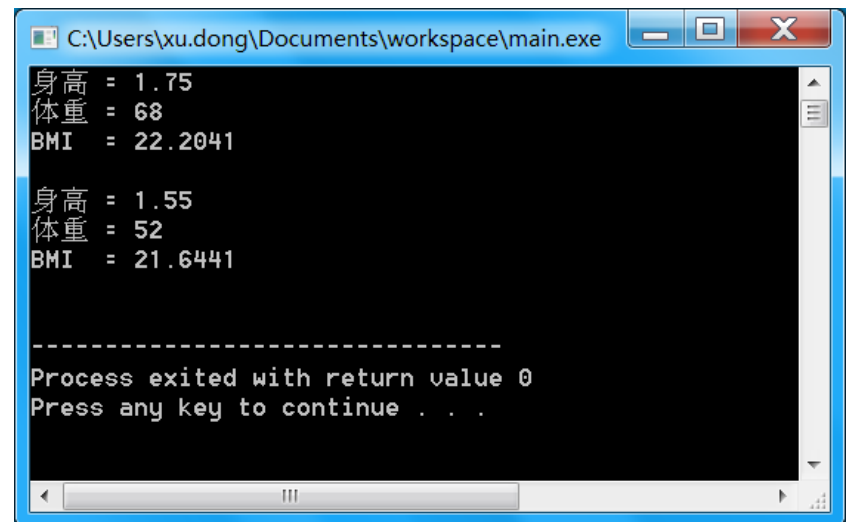
```
C:\Users\xu.dong\Documents\workspace\main.exe  
身高 = 1.75  
体重 = 68  
BMI = 22.2041  
  
身高 = 1.55  
体重 = 52  
BMI = 21.6441  
  
-----  
Process exited with return value 0  
Press any key to continue . . .
```

说明

- 从文件中读取两行记录
- 每行记录的存储格式一致
 - 身高 体重

按固定格式从文件中反复读取数据

- `for(int i=0; i<2; ++i){`
- `in_file >> height >> weight;`
- `cout<< . . . << weight/(height*height) <<endl;`
- `}`
- `in_file>>height>>weight;`
 - 执行两次
 - 每次都从目标文件中读取
 - 一个身高
 - 一个体重



The screenshot shows a Windows command prompt window titled "C:\Users\xu.dong\Documents\workspace\main.exe". The window displays the following output:

```
身高 = 1.75
体重 = 68
BMI = 22.2041

身高 = 1.55
体重 = 52
BMI = 21.6441

-----
Process exited with return value 0
Press any key to continue . . .
```

从文件中读取数据

- 检查是否已经达到目标文件尾部的方式
 - 输入文件变量名.eof()
- if(输入文件变量名.eof() == true){
- cout<<“达到文件的尾部”<<endl;
- cout<<“已经无法再从目标文件中获取任何数据”
- <<endl;
- }

计算并输出BMI指数

```
#include<iostream>
#include<fstream>
using namespace std;
int main(){
    ifstream in_file("身高_体重.txt");
    if(in_file == 0) return 0;
    double height = 0.0, weight = 0.0;
    while(in_file.eof() == false){
        in_file >> height >> weight;
        cout << "身高      = " << height << endl
             << "体重      = " << weight << endl
             << "BMI      = " << weight/(height*height) << endl;
    }
    in_file.close();

    return 0;
}
```

任务伍拾叁

```
ifstream in_file("d:\\data.txt");

if(in_file == 0){
    cout << "无法打开文件, 程序提前结束。" << endl;
    return 0;
}

Student one;
char ch = ' ';
double bmi = 0.0;
while(in_file.eof()==false){
    in_file >> one.name >> ch >> one.gender >> ch >> one.age >> ch
        >> one.height >> ch >> one.weight >> ch
        >> bmi ;

    cout << one.height * 100 << endl; // 输出改变单位后的身高
}
in_file.close();
```

```
string file_name = "d:\\data.txt", temp_file_name = "temp.txt";

ifstream in_file(file_name);
if(in_file == 0) return 0;

ofstream out_file(temp_file_name);

Student one;
char ch = '.';
double bmi = 0.0;
while(in_file.eof()==false){
    in_file >> one.name >> ch >> one.gender >> ch >> one.age >> ch
        >> one.height >> ch >> one.weight >> ch >> bmi ;

    out_file << one.name << ch << one.gender << ch << one.age << ch
        << one.height * 100 << ch
        << one.weight << ch << bmi << endl;
}
out_file.close();
in_file.close();

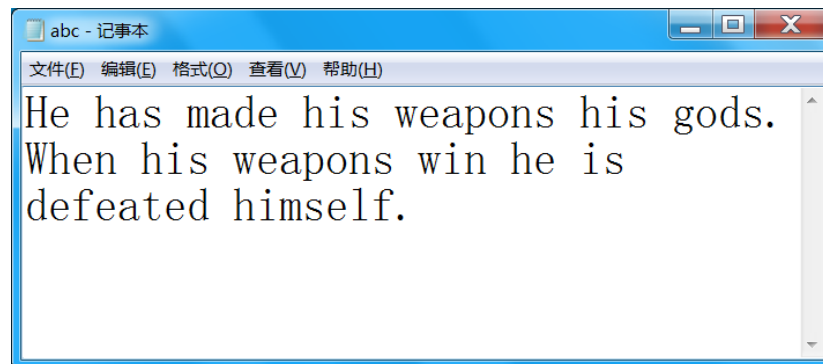
remove(file_name.c_str());
rename(temp_file_name.c_str(),file_name.c_str());
```


文件读写

代码

- `ifstream in_file("abc.txt");`
- `char ch = ' ';`
- `while(in_file.eof()==false)`
- `{`
- `in_file >> ch;`
- `cout << ch;`
- `}`
-
- `in_file.close();`

文件内容

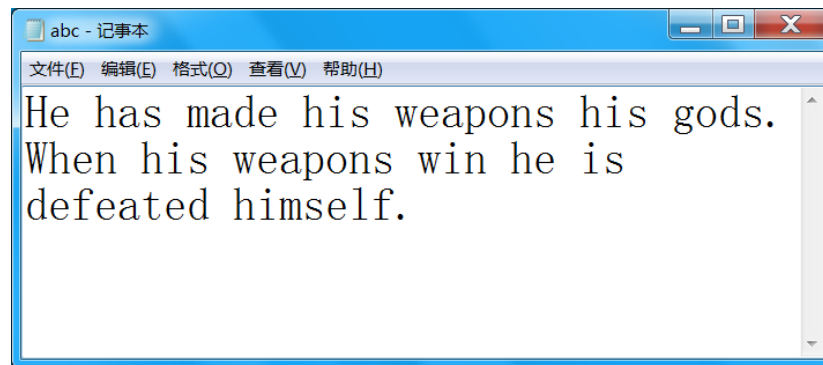
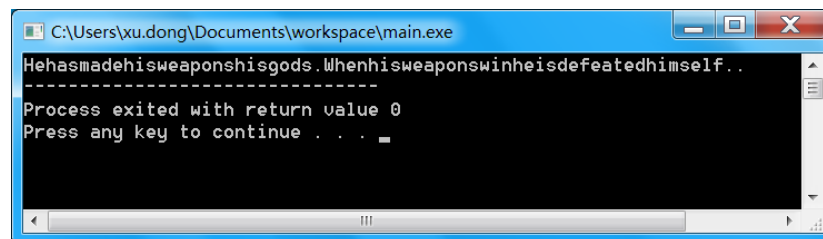


文件读写

代码

- `ifstream in_file("abc.txt");`
- `char ch = ' ';`
- `while(in_file.eof()==false)`
- `{`
- `in_file >> ch;`
- `cout << ch;`
- `}`
-
- `in_file.close();`

输出结果



文件读写

- 空白符作为数据项的分隔标记
- >>
 - 自动跳过空白符
- 读取空白符
 - `getline()`
 - `get()`

get()

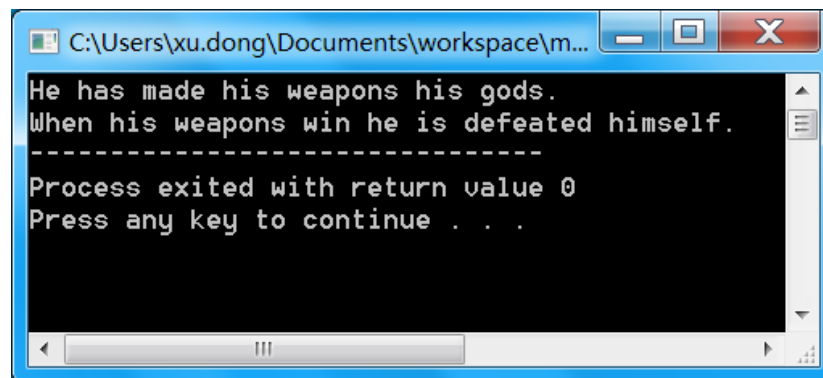
- **语法**
 - **输入文件变量名.get()**
- **正常情况**
 - **得到读入的字符**
- **达到文件末尾时**
 - **得到-1**

文件读写

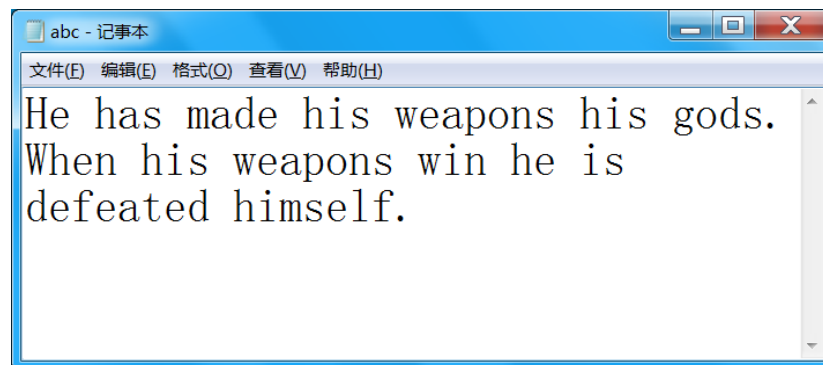
代码

- `ifstream in_file("abc.txt");`
- `char ch = ' ';`
- `while(in_file.eof()==false)`
- `{`
- `ch = in_file.get();`
- `cout << ch;`
- `}`
-
- `in_file.close();`

输出结果



```
C:\Users\xu.dong\Documents\workspace\m...
He has made his weapons his gods.
When his weapons win he is defeated himself.
-----
Process exited with return value 0
Press any key to continue . . .
```



```
abc - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)
He has made his weapons his gods.
When his weapons win he is
defeated himself.
```

文件读写

代码

- `ifstream in_file("abc.txt");`
- `char ch = ' ';`
- `while(in_file.eof()==false)`
- `{`
- `ch = in_file.get();`
- `cout << ch;`
- `}`
-
- `in_file.close();`

等价形式

- `ifstream in_file("abc.txt");`
- `char ch=' ';`
- `while((ch=in_file.get()) != -1)`
- `{`
- `//循环体内只有一条语句`
- `cout<<ch;`
- `}`
-
- `in_file.close();`

getline()

- **语法**

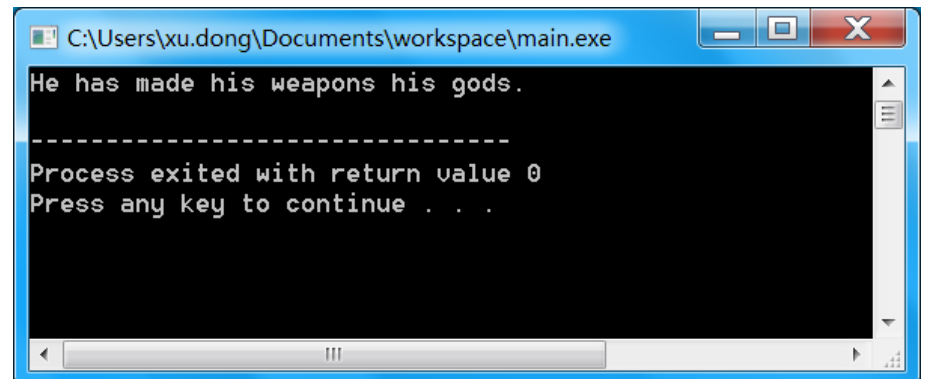
- **输入文件变量名.getline(字符数组名,**
• **字符个数,**
• **结束字符)**

- **注意**

- **字符数组char[]的长度**
- **不包括结束字符**

getline()

- `ifstream in_file("abc.txt");`
- `char text[1000];`
- `in_file.getline(text, 999, '\n');`
- `cout << text << endl;`
-
- `in_file.close();`



A screenshot of a Windows command prompt window titled "C:\Users\xu.dong\Documents\workspace\main.exe". The window has a black background with white text. The first line of output is "He has made his weapons his gods." followed by a line of dashes. Below the dashes, the text reads "Process exited with return value 0" and "Press any key to continue . . .". The window includes standard Windows window controls (minimize, maximize, close) in the top right corner and a scrollbar on the right side.

getline()

- **语法**

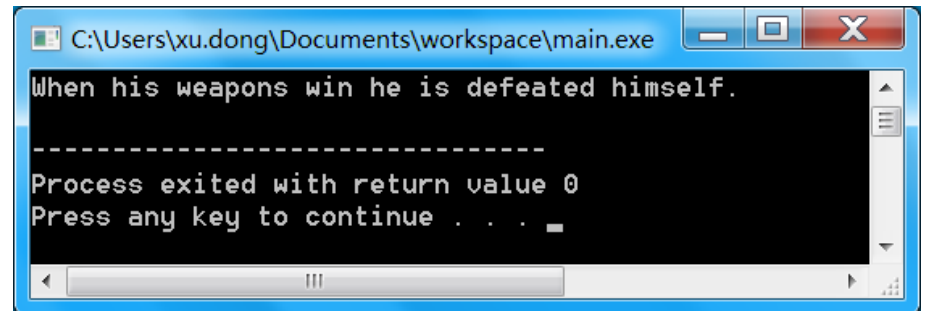
- `getline(输入文件变量名, str_var);`

- **注意**

- `string`
- **从文件中读取一整行字符串**

getline()

- `ifstream in_file("abc.txt");`
- `string line;`
- `getline(in_file, line);`
- `getline(in_file, line);`
- `cout << line << endl;`
- `in_file.close();`



```
C:\Users\xu.dong\Documents\workspace\main.exe
When his weapons win he is defeated himself.
-----
Process exited with return value 0
Press any key to continue . . . _
```

文件读写

```
ifstream in_file("abc.txt");  
if(in_file == 0) return 0;  
char c = ' ';  
while((c = in_file.get()) != -1){  
    c = in_file.get( );  
    cout << c;  
}  
in_file.close();
```

- **不能正确显示文件内容**

- **在每轮迭代中执行两次 `in_file.get()`**

文件读写

- 将目标文件路径常量改为变量从而提高可用性
- 利用流插入运算符<<和流提取运算符>>只能输入输出标准类型的数据
- 只能读写基本数据类型的数据项
 - 运算符重载
- put, get, getline等成员函数进行字符的输入输出

打开输出文件

- `ofstream outfile("d:\\binfile.data" ,`
- `ios::binary|ios::app);`

- `ios::binary|ios::app`

- 打开二进制文件

- 写方式

- 追加模式(输出到文件尾部)

问题

- **统计文本文件中字符的数目**
 - **统计非空白字符的数目**
 - **统计包括空白字符在内的全体字符的数目**
- **复制文件**

问题

- 从外部文件中，读取9527个学生的身高、体重数据，计算BMI指数并将计算结果保存至另一外部文件中。
- 原始数据文件的格式
 - 从第一行起 学号 体重(m) 身高(kg)
 - 每行保存一位学生的信息

待续.....