广州大学学生实验报告

2018年5月10日

**开课学院及实验室：**计算机科学与工程实验室电子楼412A室

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **学院** | 计算机 | **专业班级** | 软件171 | **姓名** | 谢金宏 | | **学号** | 1706300001 |
| **实验课程名称** | | 面向对象程序设计实验 | | | | **成绩** | |  |
| **实验项目名称** | | IO流与文件 | | | | **指导老师** | | 陶文正 |

# 实验目的及要求

1. 掌握流的概念
2. 掌握文本文件和二进制文件的使用方法

# 实验设备及平台

运行Visual Studio 2017的微型计算机。

# 实验内容及步骤

## 文本文件练习

编写一个能显示特定文本文件内容，并且能统计文本文件行数的程序。

程序代码如下，

#include <iostream>

#include <iomanip>

#include <string>

#include <fstream>

using namespace std;

int main()

{

fstream file(\_\_FILE\_\_);

int line\_count = 0, char\_count = 0;

string str;

while (getline(file, str))

{

line\_count += 1;

char\_count += str.size();

cout.width(2);

cout << right << line\_count << " " << str << endl;

}

cout << "行数" << line\_count << endl;

cout << "字符数（不含换行符）" << char\_count << endl;

system("pause");

}

程序产生的输出为：

1 #include <iostream>

2 #include <iomanip>

3 #include <string>

4 #include <fstream>

5 using namespace std;

6

7 int main()

8 {

9 fstream file(\_\_FILE\_\_);

10

11 int line\_count = 0, char\_count = 0;

12

13 string str;

14 while (getline(file, str))

15 {

16 line\_count += 1;

17 char\_count += str.size();

18

19 cout.width(2);

20 cout << right << line\_count << " " << str << endl;

21 }

22 cout << "行数" << line\_count << endl;

23 cout << "字符数（不含换行符）" << char\_count << endl;

24

25 system("pause");

26 }

行数26

字符数（不含换行符）431

请按任意键继续. . .

## 二进制文件练习

建立一个存放图书信息的二进制文件“bookf.dat”，该文件的每条记录包括图书编号、图书名、出版社、作者和价格。编写程序，使得程序能够读取文件中的信息，并能够添加新纪录以及删除指定的记录。

程序采用多文件结构。程序启动时从文本中读取图书信息到内存中，并在退出时将更新的图书信息写入文件。程序实现了题目要求的浏览、插入和删除功能。为了便于调试，程序维护了一个日志模块，该模块将程序运行过程中的关键信息保存到文件中。

具体的代码实现如下，

// book.h

#ifndef BOOK\_H\_

#define BOOK\_H\_

#include <string>

using namespace std;

struct Book

{

string serial;

string title;

string press;

string author;

double price;

};

#endif

// toolchain.h

#ifndef TOOLCHAIN\_H\_

#define TOOLCHAIN\_H\_

#include "book.h"

#include <cstring>

#include <iostream>

#include <fstream>

#include <string>

#include <ctime>

using namespace std;

enum EventLevel

{

Log, Debug, Warning, Error

};

const string event\_name[4] = {

"<日志>", "<调试>", "<警告>", "<故障>",

};

string timestamp();

void error(const string&, EventLevel = Log);

void log(const string&, EventLevel = Log);

string parse\_raw\_string(fstream&);

void eatline();

void show(const Book&, unsigned);

template<typename T>

void input(T &t, const string & comment = "")

{

cout << comment << "> "; cin >> t; eatline();

}

template<>

void input<string>(string &t, const string &comment);

#endif

// main.cpp

#include "book.h"

#include "toolchain.h"

#include <cstring>

#include <iostream>

#include <fstream>

#include <string>

#include <vector>

using namespace std;

const static string dat\_path = "bookf.dat";

const static string log\_path = "log.txt";

ofstream log\_file;

fstream file;

vector<Book> books;

void init();

void menu();

void explore();

void append();

void locate();

void erase();

void dispose();

int main()

{

init();

while (true)

{

menu();

int choice; input(choice);

switch (choice)

{

case 1: explore(); break;

case 2: append(); break;

case 3: locate(); break;

case 4: erase(); break;

case 0: exit(0); break;

default: cout << "Noops" << endl; system("pause");

}

}

system("pause");

}

void init()

{

log\_file.open(log\_path, ios::app);

if (!log\_file.is\_open()) {

error("无法打开日志文件", Error);

exit(-1);

}

clog.rdbuf(log\_file.rdbuf());

file.open(dat\_path, ios::in);

if (file.is\_open()) {

log("图书记录已经存在");

while (true)

{

Book nxt;

nxt.serial = parse\_raw\_string(file);

nxt.title = parse\_raw\_string(file);

nxt.press = parse\_raw\_string(file);

nxt.author = parse\_raw\_string(file);

file.read((char\*)&nxt.price, sizeof(double));

if (!file.eof())

books.push\_back(nxt);

else break;

}

log(string("从文件中读取") + to\_string(books.size()) + string("条记录"));

}

else log("图书记录不存在");

atexit(dispose);

log("程序完成初始化");

}

void menu()

{

system("cls");

cout << "神奇的图书管理系统" << endl;

cout << "1 - 浏览图书" << endl;

cout << "2 - 添加图书" << endl;

cout << "3 - 以编号查询图书" << endl;

cout << "4 - 以编号删除图书" << endl;

cout << "0 - 退出" << endl;

}

void explore()

{

for (unsigned i=0; i<books.size(); ++i) {

show(books[i], i);

}

log("浏览图书记录");

system("pause");

}

void append()

{

Book nxt;

input(nxt.serial, "编号");

input(nxt.title, "书名");

input(nxt.press, "出版社");

input(nxt.author, "作者");

input(nxt.price, "价格");

books.push\_back(nxt);

cout << "已添加图书: " << nxt.title << endl;

system("pause");

log(string("添加图书") + nxt.title);

}

void locate()

{

string serial; input(serial, "图书编号");

for (auto it = books.begin(); it != books.end(); ++it)

{

if (it->serial == serial) {

show(\*it, it-books.begin());

system("pause");

log(string("显示标题为") + serial + string("的图书"));

return;

}

}

cout << "不存在标题为" << serial << "的图书" << endl;

system("pause");

log("不存在标题为" + serial + "的图书");

}

void erase()

{

string serial; input(serial, "图书编号");

for (auto it = books.begin(); it != books.end(); ++it)

{

if (it->serial == serial) {

books.erase(it);

cout << "删除了编号为" << serial << "的图书" << endl;

system("pause");

log(string("删除标题为") + serial + string("的图书"));

return;

}

}

cout << "不存在编号为" << serial << "的图书" << endl;

system("pause");

log(string("找不到编号为") + serial + string("的图书"));

return;

}

void dispose()

{

log("准备将更改保存到文件");

file.close();

file.open(dat\_path, ios::out | ios::trunc);

file.seekp(0, ios::beg);

for (unsigned i=0; i<books.size(); ++i)

{

file.write(books[i].serial.c\_str(), strlen(books[i].serial.c\_str()) + 1);

file.write(books[i].title.c\_str(), strlen(books[i].title.c\_str()) + 1);

file.write(books[i].press.c\_str(), strlen(books[i].press.c\_str()) + 1);

file.write(books[i].author.c\_str(), strlen(books[i].author.c\_str()) + 1);

file.write((char \*)&books[i].price, sizeof(double));

}

log("程序终止");

file.close();

log\_file.close();

}

// toolchain.cpp

#include "toolchain.h"

string timestamp()

{

time\_t current;

time(&current);

string stamp = string(ctime(&current));

stamp = "[" + stamp.substr(0, stamp.size()-1) + "]";

return stamp;

}

void error(const string &comment, EventLevel level)

{

cerr << timestamp() << ' ' << event\_name[level] << ' ' << comment << endl;

}

void log(const string &comment, EventLevel level)

{

clog << timestamp() << ' ' << event\_name[level] << ' ' << comment << endl;

}

string parse\_raw\_string(fstream &file)

{

char buf[2048]; memset(buf, 0, sizeof(buf));

unsigned tail = 0;

while (!file.eof() && file.read(buf+tail, 1))

{

if (buf[tail] == '\0') break;

++tail;

}

return string(buf);

}

void show(const Book &book, unsigned idx)

{

cout << "<-- " << idx << endl;

cout << "编号：" << book.serial << endl;

cout << "书名：" << book.title << endl;

cout << "出版社：" << book.press << endl;

cout << "作者：" << book.author << endl;

cout << "价格：" << book.price << endl;

cout << "-->" << endl;

}

template<>

void input<string>(string &t, const string &comment)

{

cout << comment << "> "; getline(cin, t);

}

void eatline()

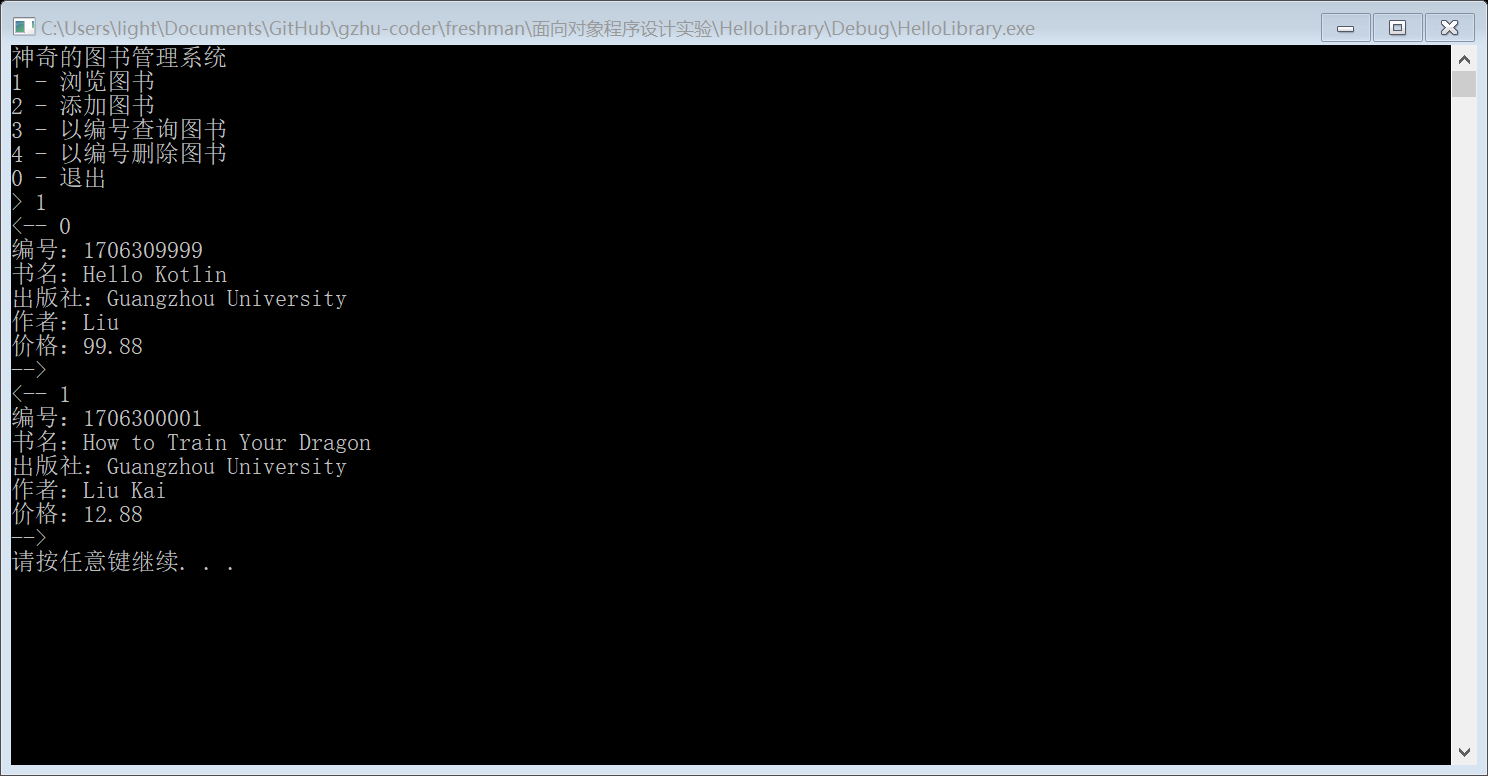
{

cin.clear();

while (cin.get() != '\n') continue;

}

一次可能的交互如下，



部分日志如下，

[Thu May 10 01:15:20 2018] <日志> 图书记录已经存在

[Thu May 10 01:15:20 2018] <日志> 从文件中读取2条记录

[Thu May 10 01:15:20 2018] <日志> 程序完成初始化

[Thu May 10 01:15:51 2018] <日志> 添加图书How to Train Your Dragon

[Thu May 10 01:15:53 2018] <日志> 浏览图书记录

[Thu May 10 01:16:08 2018] <日志> 删除标题为1706300001的图书

[Thu May 10 01:16:10 2018] <日志> 浏览图书记录

[Thu May 10 01:16:14 2018] <日志> 浏览图书记录

……

# 总结

C++流内容比较繁杂，关键的是平时多练习，并且把握住重要的脉络。例如小数精度控制等等内容是较为重要的。文件部分的内容不难，注意掌握不同的文件打开方式带来的效果。掌握了C++的流和文件，就可以借助文件保存程序的运行状态了，这是非常大的进步。