

**面向对象程序设计与实践(C++)2**

“by银”论坛

程序说明书

班级：2014211302

姓名：周尧棋

学号：2014211168

班内序号：14

日期：2016/9/19

# 题目一

## 一题题目以及需求分析

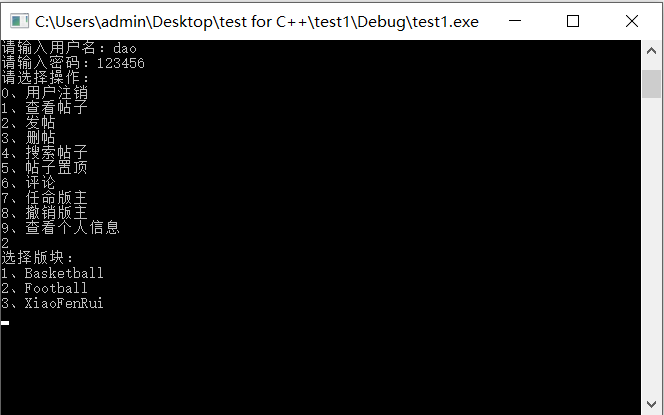
你需要帮学校设计一个论坛—“by银”论坛，作为校内学生和老师们的交流平台。

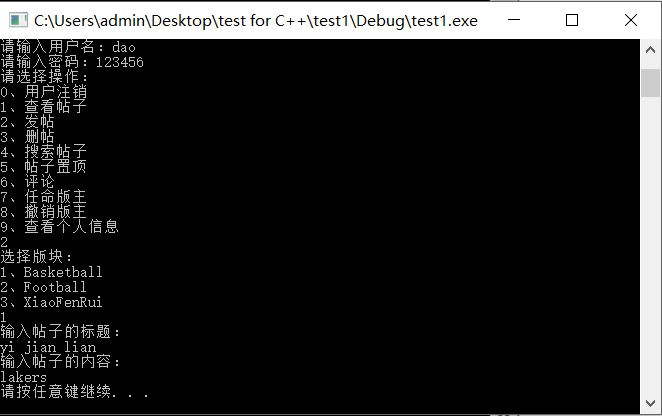
## 各功能实现模块

1. 发帖：用户可以选择在某一板块下发帖
2. 评论：用户可以在某一帖子下评论
3. 查看：查看论坛当前已有的板块及各板块下的帖子
4. 删帖：普通用户可删除自己发布的帖子，版主可删除其管理版块下的某一帖子，管理员可以删除任意帖子。
5. 查看用户个人信息：查看自己的基本信息
6. 系统登录和注销
7. 任命版主功能：管理员可以任命某一版块的版主
8. 撤销版主功能：管理员可以撤销某一版块的版主
9. 注册功能（新加）

### 发帖功能

#### 2.1.1 实现截图





#### 2.1.2 实现思路和代码

void PutPost(vector<forum> &Forums)

{

choose();

int f;

cin >> f;

string title, content;

cout << "输入帖子的标题：" << endl;

cin.ignore();

getline(cin, title);

cout << "输入帖子的内容：" << endl;

getline(cin, content);

post NewPost(Forums[f - 1].f\_post.size(), Forums[f - 1].get\_name(), title, content, get\_name());

NewPost.set\_time();

Forums[f - 1].f\_post.push\_back(NewPost);

}

根据输入的板块号选择板块类容器中的元素，输入帖子的标题和内容后使用帖子的构造函数赋值，set\_time函数设置当前发帖时间，最后在板块类中的帖子容器中加入该帖子。

#### 2.1.3 原版、改进版

无

#### 2.1.4 用到的重点知识简介

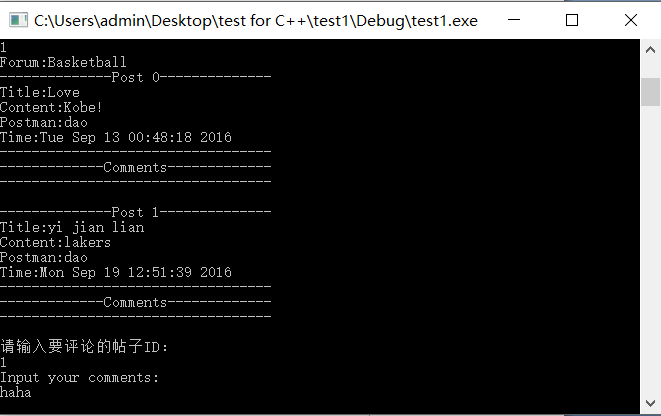
C++容器的使用，构造函数的作用。

#### 2.1.5 心得

在输入帖子的标题后，出现了无法输入帖子内容的情况，后来用输入流ignore()函数解决了问题。

### 2.2 评论功能

#### 2.2.1 实现截图



#### 2.2.2 实现思路和代码

void PutComment(vector<forum> &Forums)

{

choose();

int f;

cin >> f;

Forums[f - 1].print\_screen();

cout << "请输入要评论的帖子ID：" << endl;

int id;

cin >> id;

cout << "Input your comments:" << endl;

string content;

cin.ignore();

getline(cin, content);

comment NewCom;

NewCom.set\_content(content);

NewCom.set\_time();

Forums[f - 1].f\_post[id].p\_com.push\_back(NewCom);

}

与发帖功能实现思路相同，先选择评论的板块，打印所有帖子，然后输入要评论帖子的ID号和评论内容，创建新帖后对帖子类中的评论容器进行push\_back操作便增加了评论。

#### 2.2.3 原版、改进版

无

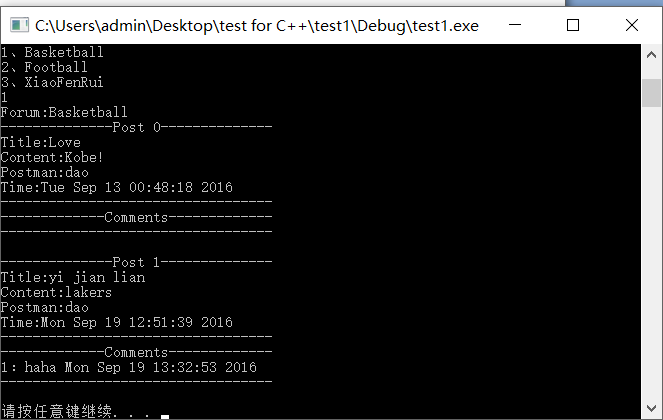
#### 2.2.4 用到的重点知识简介

C++容器的使用，构造函数的作用。

#### 2.2.5 心得

### 2.3 查看功能

#### 2.3.1 实现截图



#### 2.3.2 实现思路和代码

void CheckPost(vector<forum> Forums)

{

choose();

int choice;

cin >> choice;

Forums[choice - 1].print\_screen();

}

调用板块类中的打印函数，打印帖子容器和其元素中的评论容器，使用迭代器对容器的内容进行打印。

#### 2.3.3 原版、改进版

无

#### 2.3.4 用到的重点知识简介

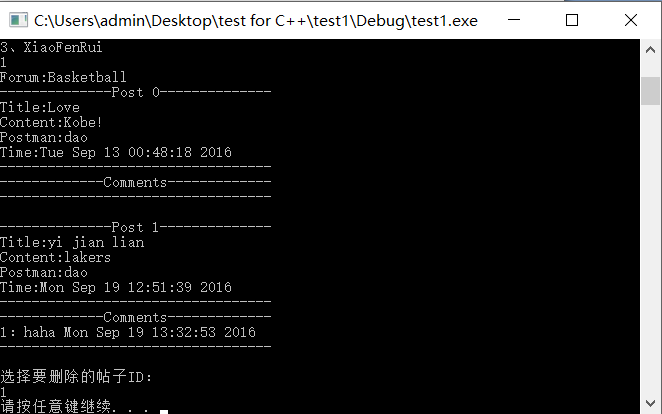
C++容器的使用和迭代器的操作使用

#### 2.3.5 心得

起初将板块类中的帖子容器和帖子类中的评论容器设为了私有成员，操作起来很不方便，后来将其设为共有成员后操作简化了不少。

### 2.4 删帖功能

#### 2.4.1 实现截图

**

#### 2.4.2 实现思路和代码

virtual void Delete(vector<forum> &Forums)

{

choose();

int f, id;

cin >> f;

Forums[f - 1].print\_screen();

cout << "选择要删除的帖子ID：" << endl;

cin >> id;

while (Forums[f - 1].f\_post[id].get\_postman() != get\_name())

{

cout << "不能删除别人发布的帖子！" << endl;

cout << "重新输入要删除的帖子的ID号：" << endl;

cout << "退出操作输入-1" << endl;

cin >> id;

if (id == -1)

return;

}

Forums[f - 1].f\_post.erase(Forums[f - 1].f\_post.begin() + id);

}

在基类User中，删除函数是虚函数。在普通用户类中，Delete函数中加有的限制是普通用户只能删除自己发布的帖子，通过判断被选帖子的发帖人postman属性与实行该操作的普通用户的用户名是否相等来提示信心，最后通过容器的erase操作来进行删帖功能。

void Delete(vector<forum> &Forums)

{

choose();

int f, id;

cin >> f;

Forums[f - 1].print\_screen();

int err = 1;

int num = -1;

//异常捕获、处理

try

{

cout << "选择要删除的帖子ID：" << endl;

cin >> id;

vector<post>::iterator iter;

for (iter = Forums[f - 1].f\_post.begin(); iter != Forums[f - 1].f\_post.end(); iter++)

{

if (id == (\*iter).get\_id())

{

num = iter - Forums[f - 1].f\_post.begin();

}

}

if (num == -1)

{

err = 0;

throw err;

}

}

catch(int)

{

cout << "输入错误！" << endl;

return;

}

Forums[f - 1].f\_post.erase(Forums[f - 1].f\_post.begin() + num);

}

而管理员类的Delete函数则无限制，可删除任意一个帖子。

void Delete(vector<forum> &Forums)

{

choose();

int f, id;

cin >> f;

while (Forums[f - 1].get\_name() != get\_sec())

{

cout << "您无权限删除该板块帖子！" << endl;

cout << "重新输入板块号：" << endl;

cout << "退出操作输入-1" << endl;

cin >> f;

if (f == -1)

{

return;

}

}

Forums[f - 1].print\_screen();

int err = 1;

int num = -1;

try

{

cout << "选择要删除的帖子ID：" << endl;

cin >> id;

vector<post>::iterator iter;

for (iter = Forums[f - 1].f\_post.begin(); iter != Forums[f - 1].f\_post.end(); iter++)

{

if (id == (\*iter).get\_id())

{

num = iter - Forums[f - 1].f\_post.begin();

//cout << num;

}

}

if (num == -1)

{

err = 0;

throw err;

}

}

catch (int)

{

cout << "输入错误！" << endl;

return;

}

Forums[f - 1].f\_post.erase(Forums[f - 1].f\_post.begin() + num);

}

版主的Delete函数中先输入板块号，如果输入的板块号与该版主用户管理的板块名不一致则报错提示无权限删除该板块帖子，之后与以上两种用户的操作基本相同。

#### 2.4.3 原版、改进版

原版中没有对输入的异常情况进行处理，在改进版本中加入了try catch的异常处理操作，如果用户输入了不存在的帖子ID号将会结束该操作。这使得程序更加完善，更加健壮。

#### 2.4.4 用到的重点知识简介

1.C++容器的使用和迭代器的操作

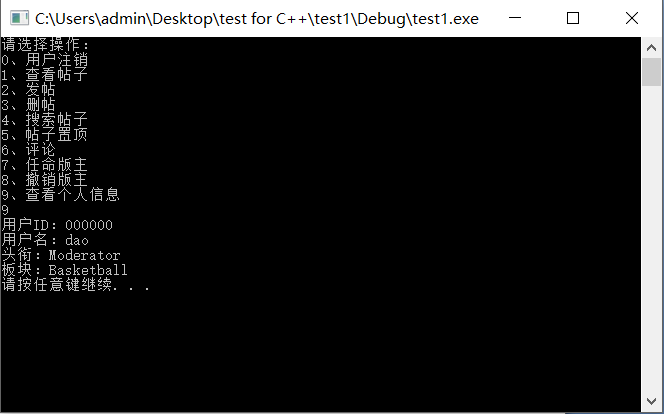
2.try catch异常处理操作

#### 2.4.5 心得

在之前的编程中并没有考虑到异常处理的问题，考虑得不太周全，这告诫我们要站在用户的角度看问题，才能将程序变得更加完美。

### 2.5 查看用户信息功能

#### 2.5.1 实现截图



#### 2.5.2 实现思路和代码

void ShowInfo()

{

string ID, name, kind;

ID = get\_id();

name = get\_name();

kind = get\_kind();

cout << "用户ID：" << ID << endl;

cout << "用户名：" << name << endl;

cout << "头衔：" << kind << endl;

cout << "板块：" << sec << endl;

}

查看用户信息打印操作很容易实现，复杂的是之前的从文件中读取用户信息并创建用户的实现。在程序的一开始用户输入自己的用户名与密码，然后对users.txt这个文件进行整行读取，先判断前两个属性用户名和密码是否与输入匹配，如果不匹配则重新登录，若匹配则继续对剩下的信息进行读取，存入一个结构体类型变量中。在执行打印用户个人信息的操作中将这些属性打印出来即可。

#### 2.5.3 原版、改进版

无

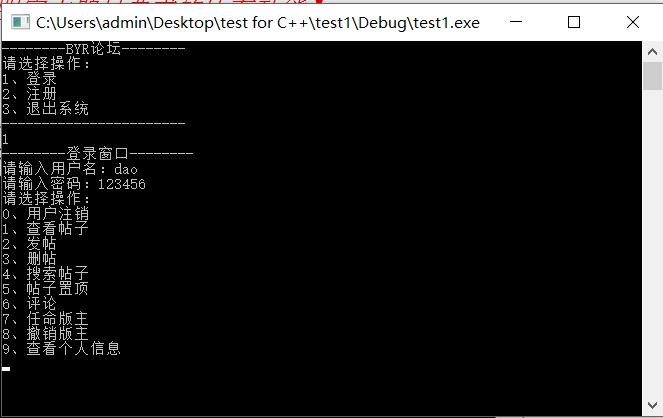
#### 2.5.4 用到的重点知识简介

文件操作和C++IO操作

#### 2.5.5 心得

### 2.6 登录和注销功能

#### 2.6.1 实现截图



#### 2.6.2 实现思路和代码

int Login(struct info &temp)

{

ifstream File;

flag = false;

int error = -1;

while (!flag)

{

cout << "--------登录窗口--------" << endl;

File.open("users.txt");

try

{

if (!File)

{

error = 0;

throw error;

}

}

catch (int)

{

cout << "Can't open the file!" << endl;

return 0;

}

string name, password;

cout << "请输入用户名：";

cin >> name;

cout << "请输入密码：";

cin >> password;

string str1, str2;

//从文件读取信息判断是否为存在用户

while (!File.eof())

{

getline(File, str1, '\t');

getline(File, str2, '\t');

if (str1 == name && str2 == password)

{

flag = true;

string str3, str4;

getline(File, str3, '\t');

getline(File, str4, '\n');

temp.UserName = str1;

temp.password = str2;

temp.id = str3;

if (str4 == "OrdinaryUser")

{

temp.kind = str4;

temp.sec = "";

return 1;

}

else if (str4 == "Admin")

{

temp.kind = str4;

temp.sec = "";

return 2;

}

else

{

string sub = str4;

temp.kind = sub.substr(0, 9);

temp.sec = sub.substr(10, sub.length()-10);

return 3;

}

}

else

{

string skip;

getline(File, skip, '\n');

}

}

cout << "用户名或密码错误！" << endl;

File.close();

}

}

登录函数传引用一个temp型结构体变量，对读取到的用户信息进行保存，以便在创建对象的时候进行赋值操作。在程序的一开始首先对读取文件进行了异常处理，如果文件不存在则提示报错信息并结束函数。用户输入自己的用户名与密码，然后对users.txt这个文件进行整行读取，先判断前两个属性用户名和密码是否与输入匹配，如果不匹配则重新登录，若匹配则继续对剩下的信息进行读取，存入结构体变量中。在读取到第四个属性时出现了不同的情况，普通用户和管理员只有头衔，而版主又有头衔又有所管理的板块名，所以对读取到的字符串还要进行分割操作。

void generate(vector<forum> &Forums)

{

int error = -1;

ofstream output("posts.txt", ios::trunc);

try

{

if (!output)

{

error = 0;

throw error;

}

}

catch (int)

{

cout << "Can't open the file!" << endl;

return;

}

vector<forum>::iterator iter1;

for (iter1 = Forums.begin(); iter1 != Forums.end(); iter1++)

{

output << (\*iter1).get\_name() << endl;

(\*iter1).print\_file(output);

}

}

注销功能是在用户输入注销选项后将当前的所有板块内容存到posts.txt的操作。在程序初始化时就将posts.txt中的内容存到一个板块Forums容器中，每个Forums容器元素中有post容器，每个post容器元素中又有一个comment容器。在Forum类、post类、comment类中分别对<<输出符进行重载，然后即可通过迭代器在文件中进行所有帖子的格式化输出了。

#### 2.6.3 原版、改进版

原版中没有使用文件读取异常操作的处理，改进版中添加了该处理。且原版是在登录函数中直接定义了对象，后来发现函数结束后无法保存用户的信息，改为根据登录函数的返回值在主函数中定义不同的对象。

#### 2.6.4 用到的重点知识简介

1.C++文件处理操作，getline函数读取整行等。

2.字符串的函数操作，如取字串等。

3.try catch的异常处理操作

4.<<操作符重载

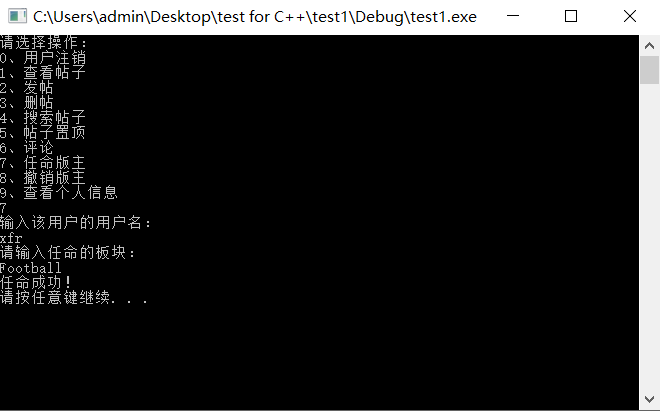
5.C++容器和迭代器的使用

#### 2.6.5 心得

登录函数在一开始有两个构想，一是一开始将文件中的用户信息先存到一个用户的容器中，再根据登录输入进行判断操作，二是一开始只创建一个用户对象，每次登录读取文件。开始觉得将所有的信息读取到内存中很占内存，但是如果每次登录都要读取文件的话操作速度很慢，最后还是选择了每次登录读取文件，这种空间和时间的利弊权衡还是没有做得太好。

### 2.7 任命版主功能

#### 2.7.1 实现截图



#### 2.7.2 实现思路和代码

void SetModerator()

{

cout << "输入该用户的用户名：" << endl;

string name;

cin >> name;

fstream file;

ofstream filetmp("temp.txt");

string getName;

char ch;

file.open("users.txt");

bool flag = false;

while (!file.eof())

{

getline(file, getName, '\t');

filetmp << getName << '\t';

if (name == getName)

{

flag = true;

getline(file, getName, '\t');

filetmp << getName << '\t';

getline(file, getName, '\t');

filetmp << getName << '\t';

getline(file, getName, '\n');

cout << "请输入任命的板块：" << endl;

string F;

cin >> F;

while (!(F == "Basketball" || F == "Football" || F == "XiaoFenRui"))

{

cout << "输入的板块名错误！请重新输入：" << endl;

cin >> F;

}

filetmp << "Moderator" << '\t' << F;

if (!file.eof())

{

filetmp << endl;

}

cout << "任命成功！" << endl;

}

else

{

file.get(ch);

while (ch != '\n' && !file.eof())

{

filetmp << ch;

file.get(ch);

}

if(!file.eof())

filetmp << endl;

}

}

if (!flag)

{

cout << "不存在该用户！" << endl;

}

file.close();

filetmp.close();

ifstream in("temp.txt");

ofstream out("users.txt");

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

while (in.get(ch))

{

out << ch;

}

in.close();

out.close();

remove("temp.txt");

}

在任命完版主后，存放用户信息的users.txt的内容也要相应更改，首先对原文件中第一个属性即用户名与输入任命的用户名进行匹配，如果不相同则整行读取存入另一个temp.txt文件中，如果相同，则输入修改后的信息至temp.txt文件中，最后将temp文件中的内容全部复制到users.txt中。

#### 2.7.3 原版、改进版

无

#### 2.7.4 用到的重点知识简介

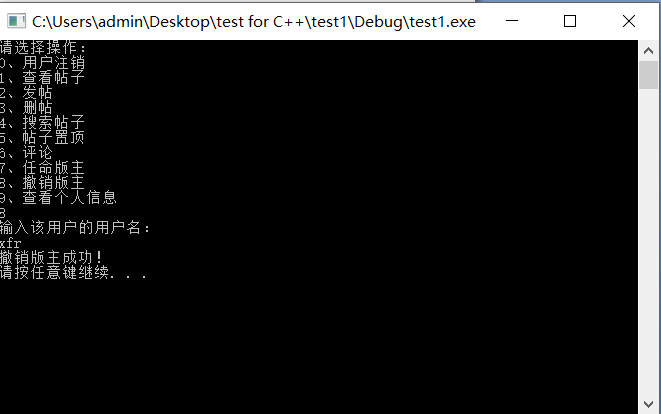
C++文件处理操作，getline读取整行与文件指针的移动。

#### 2.7.5 心得

文件处理操作极容易出现bug，在进行这个功能的调试过程中，多出一个空格或者多出一个空号都将出现很大的影响。所以在进行操作之前最好对读取的字符串进行判断，如是否为空或者是否为换行符，这样利于接下来一些非正常读取的操作也可以使程序更利于调试与检测。

### 2.8 撤销版主功能

#### 2.8.1 实现截图



#### 2.8.2 实现思路和代码

void CanModerator()

{

cout << "输入该用户的用户名：" << endl;

string name;

cin >> name;

fstream file;

ofstream filetmp("temp.txt");

string getName;

char ch;

file.open("users.txt");

bool flag = false;

while (!file.eof())

{

getline(file, getName, '\t');

filetmp << getName << '\t';

if (name == getName)

{

string getPass;

string getID;

string getkind;

getline(file, getPass, '\t');

filetmp << getPass << '\t';

getline(file, getID, '\t');

filetmp << getID << '\t';

getline(file, getkind, '\n');

if (getkind == "OrdinaryUser"||"Admin")

{

cout << "该用户不为版主！" << endl;

cout << "请重新输入用户名：" << endl;

cin >> name;

filetmp.clear();

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

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

continue;

}

flag = true;

filetmp << "OrdinaryUser";

if (!file.eof())

{

filetmp << endl;

}

cout << "撤销版主成功！" << endl;

}

else

{

file.get(ch);

while (ch != '\n' && !file.eof())

{

filetmp << ch;

file.get(ch);

}

if (!file.eof())

filetmp << endl;

}

}

if (!flag)

{

cout << "不存在该用户！" << endl;

}

file.close();

filetmp.close();

ifstream in("temp.txt");

ofstream out("users.txt");

while (in.get(ch))

{

out << ch;

}

in.close();

out.close();

remove("temp.txt");

}

撤销版主的操作与任命版主的操作基本相同，关键在于对于读取到的第四个属性即头衔的判断，如果读取到的头衔为普通用户或者为管理员，则要提示错误信息说明不为版主，要输入版主的用户名。

#### 2.8.3 原版、改进版

无

#### 2.8.4 用到的重点知识简介

C++文件处理的操作，getline等函数使用。

#### 2.8.5 心得

### 2.9 注册功能

#### 2.9.1 实现截图



#### 2.9.2 实现思路和代码

void Register()

{

string name, password, read;

cout << "创建的用户名为：" << endl;

cin >> name;

cout << "请输入密码：" << endl;

cin >> password;

int error = -1;

fstream File;

File.open("users.txt");

try

{

if (!File)

{

error = 0;

throw error;

}

}

catch (int)

{

cout << "Can't open the file!" << endl;

return ;

}

while (!File.eof())

{

getline(File, read, '\n');

}

//ID号加1操作（000000->000001）

vector<string> stu;

stringstream sstr(read);

stringstream change1, change2;

int id;

char temp[10];

string ID;

string token;

while (getline(sstr, token, '\t'))

{

stu.push\_back(token);

}

change1 << stu[2];

change1 >> id;

id++;

sprintf\_s(temp, "%06d", id);

ID = temp;

OUser NewUser;

NewUser.SetUser(name,password,ID,"OrdinaryUser","");

//cout << NewUser ;

File.seekg(0, ios::end);

File << endl << NewUser;

File.close();

}

注册功能为新加的功能，在用户输入完用户名和密码后，读写users.txt至最后一行然后将用户信息输出到文件中。稍微复杂点的操作是ID号的赋值，即当前文件最后一个ID号+1，利用stringstream和sprintf的操作，将“000000”变为“000001”，用户的头衔默认为普通用户。

#### 2.9.3 原版、改进版

无

#### 2.9.4 用到的重点知识简介

1.C++文件处理的操作，getline等函数使用。

2. stringstream数据类型的转化，字符串转整型，整型转字符串。

3.字符串格式化输出，不满0左边补0操作。

4.try catch异常处理操作

#### 2.9.5 心得

在该功能的实现上学习到了C++中不同数据类型相互转换的方法以及字符串格式化输出的基本操作，对字符串的操作有了更深的认识。

# 题目二

## 二题题目以及需求分析

“by银”论坛高级版

## 各功能实现模块

描述：在程序一的基础上的添加如下功能：

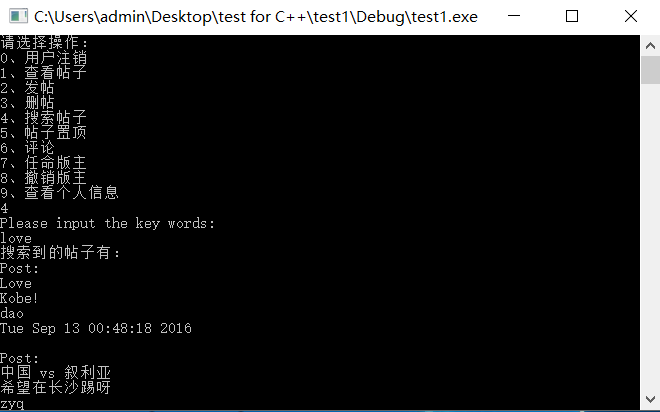
1. 搜索功能（通过函数重载实现）：

在User类中重载“查看帖子列表”操作，使之可以根据帖子名的关键字进行全局搜索。

1. 帖子置顶功能：版主可以选择其负责版块下的某篇帖子将其设置为置顶，设为置顶的帖子应在帖子列表第一页的顶部显示。

### 搜索功能

#### 2.1.1 实现截图



#### 2.1.2 实现思路和代码

void CheckPost(vector<forum> Forums, string search)

{

vector<forum>::iterator iter1;

vector<post>::iterator iter2;

vector<A>::iterator iter3;

vector<A> find;

int found = 0;

int \*\*c = NULL;

int i, j, m, n;

for (iter1 = Forums.begin(); iter1 != Forums.end(); iter1++)

{

for (iter2 = (\*iter1).f\_post.begin(); iter2 != (\*iter1).f\_post.end(); iter2++)

{

string title = (\*iter2).get\_title();

//LCS求两个字符串最大公共子序列

m = title.length();

n = search.length();

c = new int\*[m];

for (i = 0; i < m; i++)

{

c[i] = new int[n];

}

for (i = 0; i < m; i++)

{

for (j = 0; j < n; j++)

{

c[i][j] = 0;

}

}

for (i = 0; i < m; i++)

{

for (j = 0; j < n; j++)

{

if (i == 0 || j == 0)

{

if (title[i] == search[j])

{

c[i][j] = 1;

}

else

{

if (i > 0)

{

c[i][j] = c[i - 1][j];

}

if (j > 0)

{

c[i][j] = c[i][j - 1];

}

}

}

else if (title[i] == search[j])

{

c[i][j] = c[i - 1][j - 1] + 1;

}

else if (c[i - 1][j] > c[i][j - 1])

{

c[i][j] = c[i - 1][j];

}

else

{

c[i][j] = c[i][j - 1];

}

}

}

int count = 0;

count = c[m - 1][n - 1];

if (count > 0)

{

A add((\*iter2), count);

found = 1;

find.push\_back(add);

}

}

}

for (i = 0; i < m; i++)

{

delete[]c[i];

c[i] = NULL;

}

delete[]c;

c = NULL;

if (found == 1)

{

sort(find.begin(), find.end(), cmp);

cout << "搜索到的帖子有：" << endl;

for (iter3 = find.begin(); iter3 != find.end(); iter3++)

{

cout << (\*iter3).find << endl;

cout << endl;

}

}

else

{

cout << "找不到相关帖子！" << endl;

}

}

全局搜索功能是对查看帖子函数的重载，两个函数的函数名相同但是参数不同。搜索操作是基于对Forums板块容器中的搜索，使用两层迭代器循环，对板块的title与输入的关键字求最大公共子序列（LCS），若公共子序列长度不为0则视为找到，即只要关键字中某字符存在于帖子的标题中帖子即视为找到。然后将找到的帖子存在一个容器中，根据公共子序列的长度进行排序，最后输出匹配最多的帖子最先输出。

#### 2.1.3 原版、改进版

无

#### 2.1.4 用到的重点知识简介

1.C++容器的使用和迭代器的使用

2.LCS最大公共子序列算法，动态规划相关知识

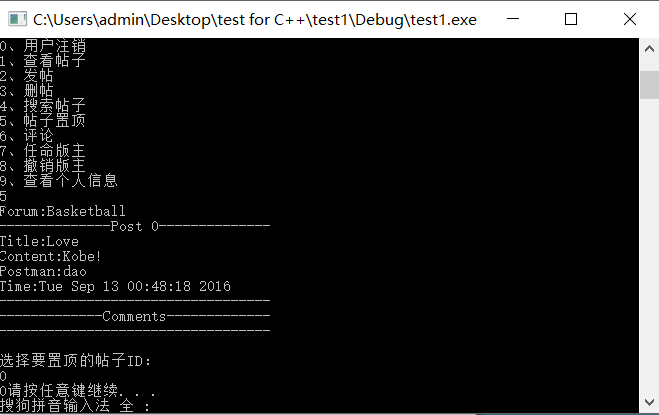
3.排序算法的使用

#### 2.1.5 心得

这个功能我想得相对复杂，为了实现求公共子序列的方法还专程复习了上学期算法设计分析所学过的动态规划的相关知识。最后根据相同匹配长度排序也是对该功能的一个完善，其实实际中的搜索引擎中的算法绝对没有这么简单，还与序列的顺序等因素有关，这还需要我继续学习以达到更全面的功能。

### 2.2 帖子置顶功能

#### 2.2.1 实现截图



#### 2.2.2 实现思路和代码

void Stick(vector<forum> &Forums)

{

int f;

if (get\_sec() == "Basketball")

{

f = 0;

}

else if (get\_sec() == "Football")

{

f = 1;

}

else

{

f = 2;

}

Forums[f].print\_screen();

int err = 1;

int num = -1;

int id;

try

{

cout << "选择要置顶的帖子ID：" << endl;

cin >> id;

vector<post>::iterator iter;

for (iter = Forums[f].f\_post.begin(); iter != Forums[f].f\_post.end(); iter++)

{

if (id == (\*iter).get\_id())

{

num = iter - Forums[f].f\_post.begin();

}

}

if (num == -1)

{

err = 0;

throw err;

}

}

catch (int)

{

cout << "输入错误！" << endl;

return;

}

post PostTmp(Forums[f].f\_post[num]);

//vector<post>::iterator iter;

for (; num > 0; num--)

{

Forums[f].f\_post[num] = Forums[f].f\_post[num - 1];

}

Forums[f].f\_post[num] = PostTmp;

}

帖子置顶功能只对于版主才能操作，当普通用户或者管理员输入该选项时会提示没有权限，主要思路便是在帖子容器中选择要置顶的帖子复制到一个新的帖子对象中，再将该帖子之前的元素全部后移一个，最后将新的帖子对象赋值给第一个元素。

#### 2.2.3 原版、改进版

无

#### 2.2.4 用到的重点知识简介

1.C++容器类的使用以及迭代器的使用

2.排序算法的借鉴

#### 2.2.5 心得