# C++实践

## 图书管理系统：

**设计背景：**

首先作为大学生，对图书的借阅是家常便饭，因此，我想通过本学期所学的C++知识，简单地设计一个图书管理系统，不求能够落地实现，重在将知识运用起来

**设计思想：**

**1. 学生类（student）的设计：**

图书的使用主体是学生，因此在这里对学生类作了定义，数据成员包括姓名+学号；函数成员中包含重载<运算符，用于匹配后面map容器的使用。

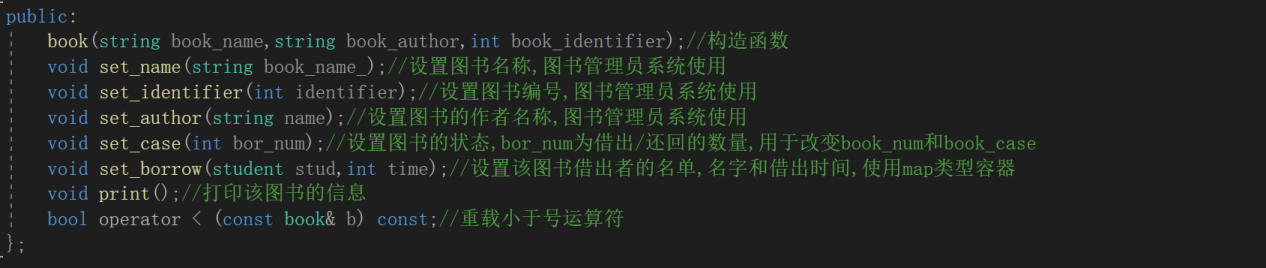
核心代码截图：



1. **图书类（book）的设计：**

图书的信息包括书名，作者名，图书编号，图书状态等，这里我对借出该图书的人用了一个map数组，其中的关键词是学生类，其数值为该学生借出该图书的时间

其中的函数成员包括：



其中各种设置函数，只需修改书籍对应的数据成员即可。重载小于号操作符也是用于匹配后面存放书类的vector数组中的排序操作。

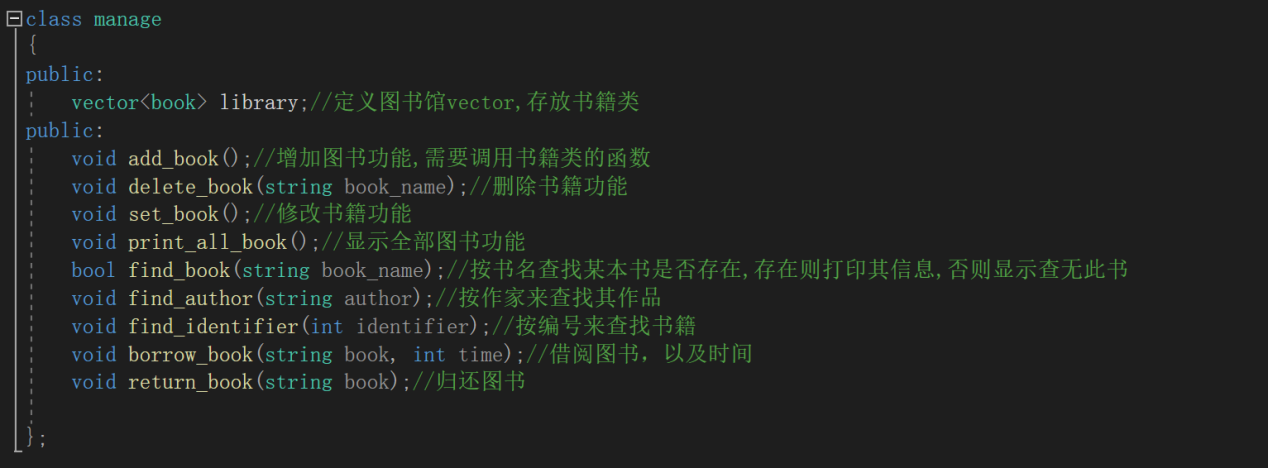
图书状态设置函数的代码实现：

bor\_num为借出或者换回的数量，通过正负号来标识，然后用图书存量对应去相加或者相减，当书籍数量为0时，不能借出。



**3.图书管理系统类（manage）的设计：**

数据成员和函数成员如下：



图书馆中必定有大量的书籍，因此定义一个能够存放书籍类的可变容量的容器，这里十分自然地能够联想到使用vector数组，但记住，需要对书籍类重载一个<操作符，用于数组的排序函数。

1.其中的增加图书功能只需输入图书的相应的信息，便可以将书籍加入图书馆。

2.删除书籍只需对vector数组中的元素进行移除操作即可。

3.修改书籍功能则通过书籍类的接口，来对其数据成员进行修改。设计这个函数的初衷是为了当加入书籍信息错误时，能够通过其他函数来修正。

4.打印全部图书功能只需遍历书籍类数组，然后输出即可。

5.查找图书功能，这里设置了三种方式，分别是按照书名，作者以及编号，满足了用户的绝大部分情况，使得该图书管理系统更加真实，甚至后期还能够按照关键字检索，通过检索书名中的关键词来查找图书，也只需要一个KMP算法即可。

6.最后，书籍肯定是要有借有还的，因此，借阅图书只需登记个人信息，其中要调用该图书的设置状态函数。同样的，归还图书也只需要调用修改该书籍的状态信息函数即可。

以上的各种操作的选择，可以通过一个switch case来实现即可。

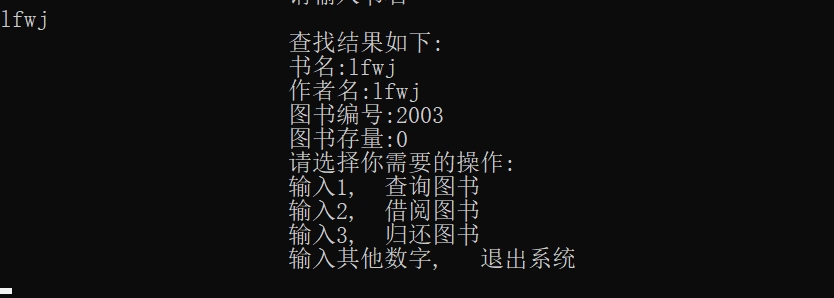
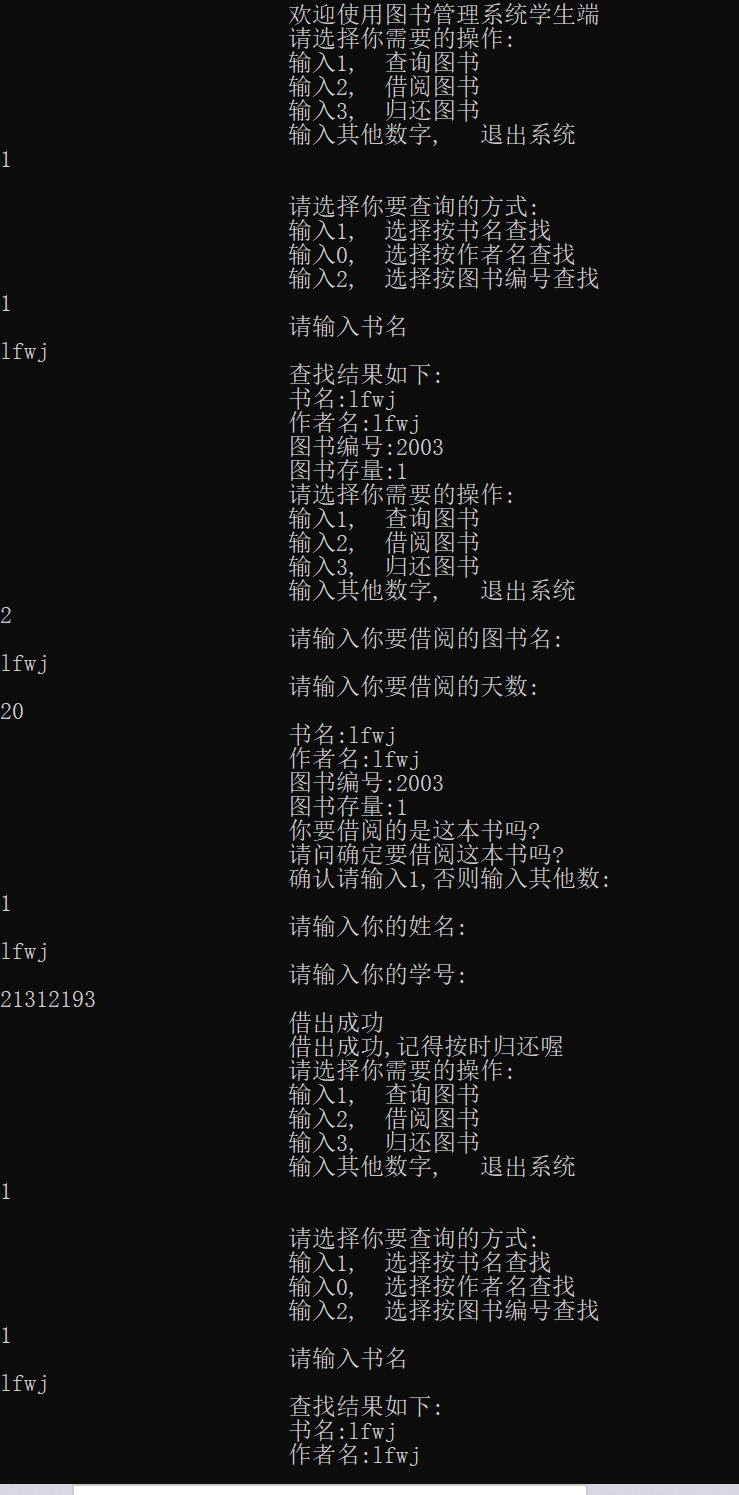
**主程序：**

在主函数中，首先需要对图书馆进行一个初始化，使得其中包含有足够的书籍，因此，我这里选择了通过读取文件的方式来实现。首先自己编写一个存储书籍信息的文本文件，然后，用文件的地址构造一个图书管理类的对象，这个对象的library（vector）中就有相应的图书信息了。之后再通过while循环以及switch case的组合，实现用户对各种功能的调用

测试代码：

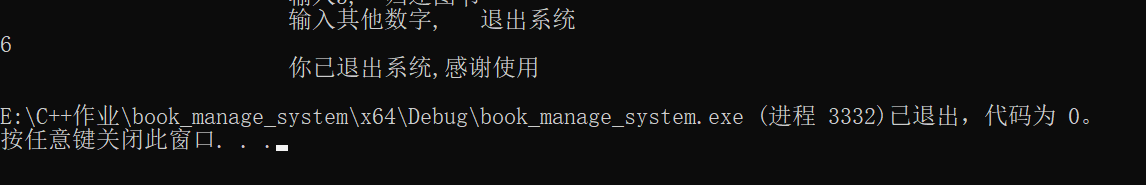
查询和借阅图书：

可以看到，当书被借出时，其存量减少1，说明系统能正常运行。



退出：

告知用户，当输入其他数字则退出



## 小游戏组合：

**Gluttonous Snake（贪吃蛇小游戏）：**

**设计背景：**

在网上的一些博客介绍中，我发现用C++就能简单地设计出贪吃蛇小游戏，虽然我个人对贪吃蛇玩的不是很多，但还是觉得学的东西能够用到自己曾经玩过的游戏上，自豪感一下子就涌上来了。

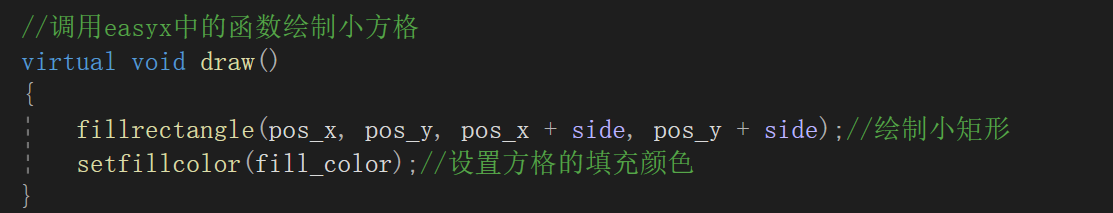
设计思路：

首先，贪吃蛇的主要内容就是一条蛇和食物，以及它所在的游戏区域，因此，我打算对这三个事物进行类的定义和实现，最后综合起来就是一个游戏。

在过程中，我一度思考，蛇应该由什么数据类型来表示，考虑到蛇有长度，故应该是一个类似数组的容器来表示蛇，并且，蛇的长度能够随着吃食物增加，因此这个应该是一个可变容量的容器。鉴于之前对vector数组的多次使用，因此，我这里初步是想通过一个vector容器来表示蛇，而其中的类型肯定不能是简单的已有的数据类型。

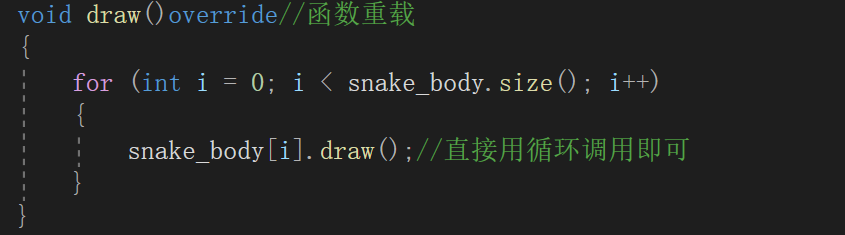
在思考过程中，我决定亲自去玩一下游戏来找一下灵感，在游戏中，我观察到，贪吃蛇是一节一节组成的，每一节像一个椭圆一样。椭圆看起来实现难度有点高，于是我选择用小矩形来替换椭圆，因此，最终确定需要设计一个小矩形类small\_rectangle。

而小矩形类中需要有哪些参数呢？首先应该有位置信息，以及小矩形的形态，这里的形态就是指颜色了。其构造函数就是要在屏幕上画出一个小矩形，因此，我查阅了一些资料，发现需要用到easyx的一些操作，在学习了相关的操作之后，我成功地将小矩形画了出来。



而对于小矩形而言，其是不会动的。如果贪吃蛇不会动，这游戏就没有意义了。为此还需要设计一个Move函数，使得小方块能够动起来，而这个操作改变的就是小方块的位置，只需在原有坐标上对横轴和纵轴进行+/-移动的距离即可。

设计好了小矩形类，那么蛇类就是对小矩形类进行一个数组的封装并加入其他的操作即可。因此，蛇类中包含数据成员蛇身：vector<small\_rectangle> snake\_body.其构造函数即先初始化一条小蛇，这里采取用三个方格进行初始化蛇身。同时，蛇身的绘制只需遍历蛇身数组中的每个小方格中的绘制函数即可。



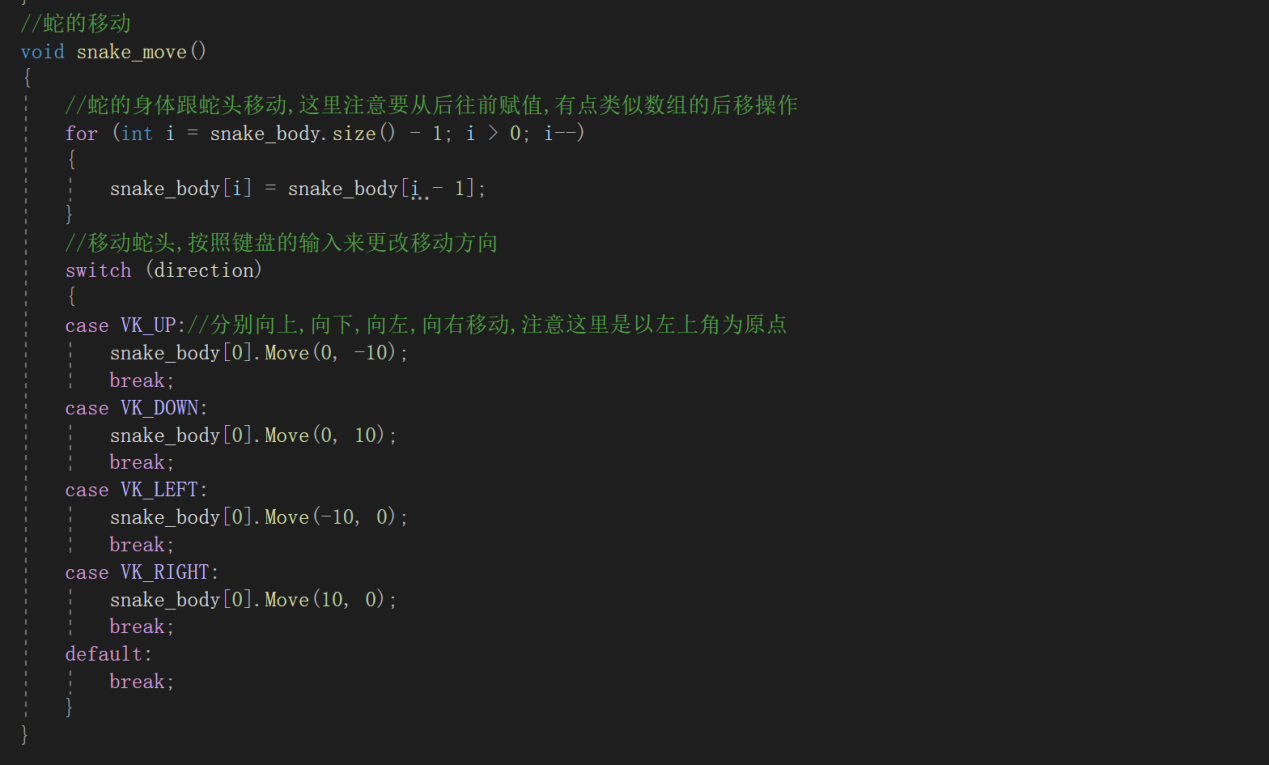
这时再来考虑食物的问题，食物，似乎也可以用一个小矩形来表示，不过其应该在小矩形的基础上还有其他函数成员，例如随机出现等，因此，这里对食物类的设计，则公有继承小矩形类即可。其中构造函数是随机生成食物所在的位置，将该参数赋给小矩形的位置即可。

接下来考虑蛇吃食物，蛇吃食物的方式就是蛇头碰到蛇尾，因此，这里通过对蛇身数组的首个元素的位置判断，判断其是否和食物的坐标进行碰撞即可，因此需要来一个meet函数，我选择在基类small\_rectangle中加入meet函数，其返回值设为bool类型，只需在其中判断两个小矩形的坐标是否重叠即可。

当蛇头碰到食物时，蛇的身体应该长长一段，这里只需往vector中加入一个小矩形即可。

**蛇的移动实现：**

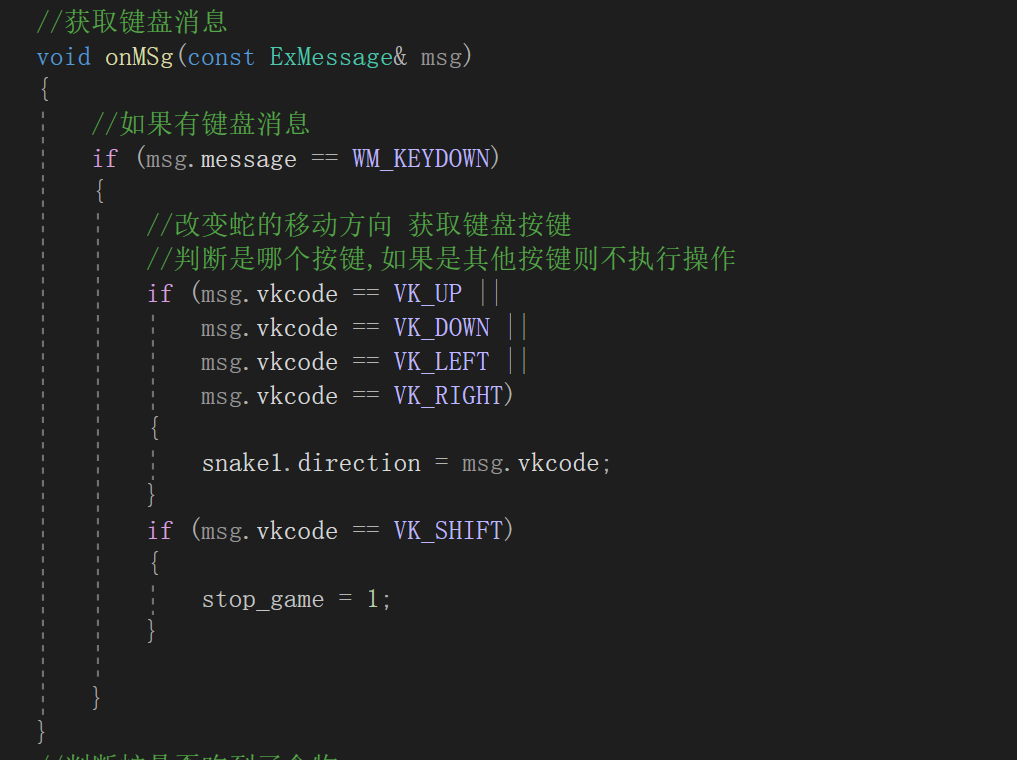
这里只需将每个小矩形的位置从后往前赋值即可，同时，在蛇类的数据成员之中，需要加入一个方向，因为蛇的移动需要一个方向，而这个方向是由用户在键盘上手动输入的，通过查询资料，我发现其实键盘上的上下左右按键对应的都是数字，因此这里可以采用switch case的方式来实现对蛇的移动方向的改变。



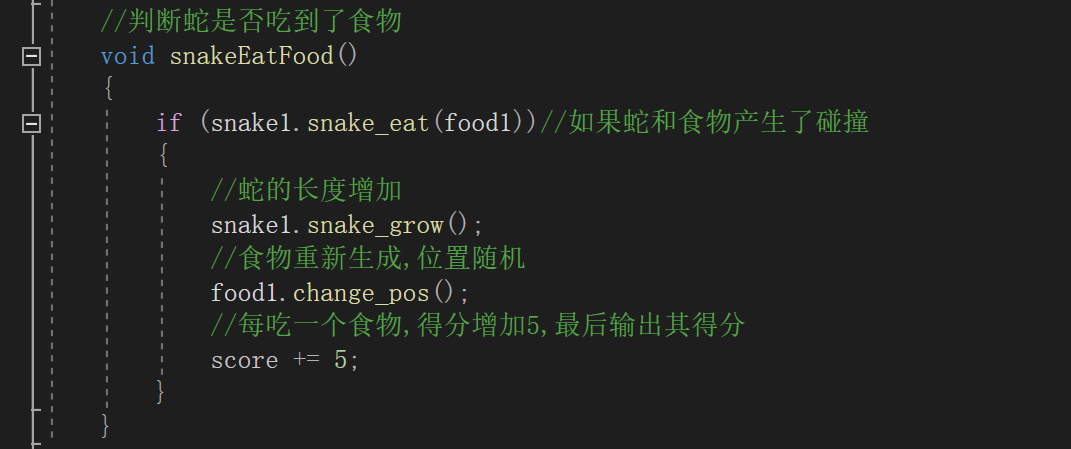
**定义游戏面板类：**

游戏面板类中的要素即为蛇和食物，因此需要在该类中声明一个蛇和食物的对象，同时，用户是在面板上进行游戏，因此需要读取用户在键盘上输入的方向控制信息，当输入其他键的信息时需要执行退出操作。

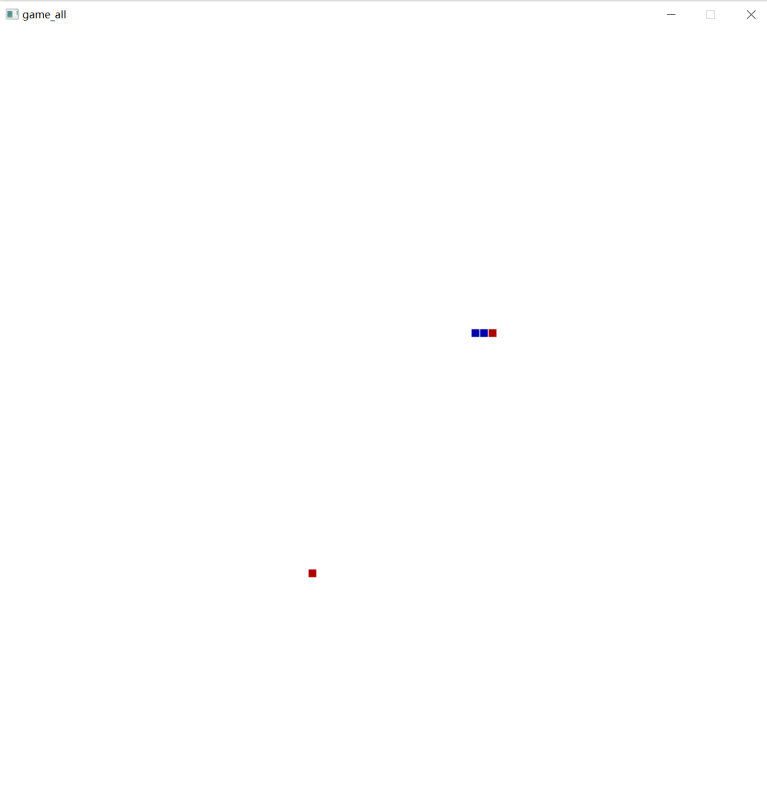
**读取键盘信息实现：**



**在蛇吃到食物之后，对食物进行更换位置：**



贪吃蛇实现效果（蛇头为红色，蛇身为蓝色）：



**Sudoku（数独）：**

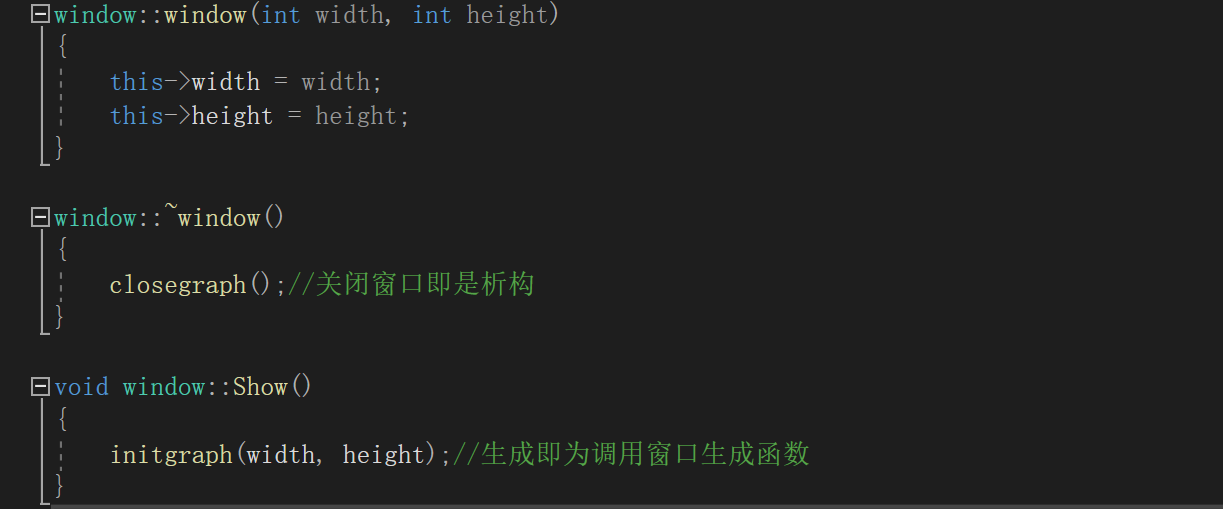
**设计背景：**

数独是一个非常大众的数学游戏，其中的要素也不是非常复杂，因此，在设计完贪吃蛇小游戏之后，我还对数独游戏进行了简单的设计。

**设计思想：**

数独页面是一个由9\*9的方格组成的，可以进行数字填入的页面。而该页面要装下 9\*9个小方格，因此我首先定义了一个窗口类，用于表示游戏页面。

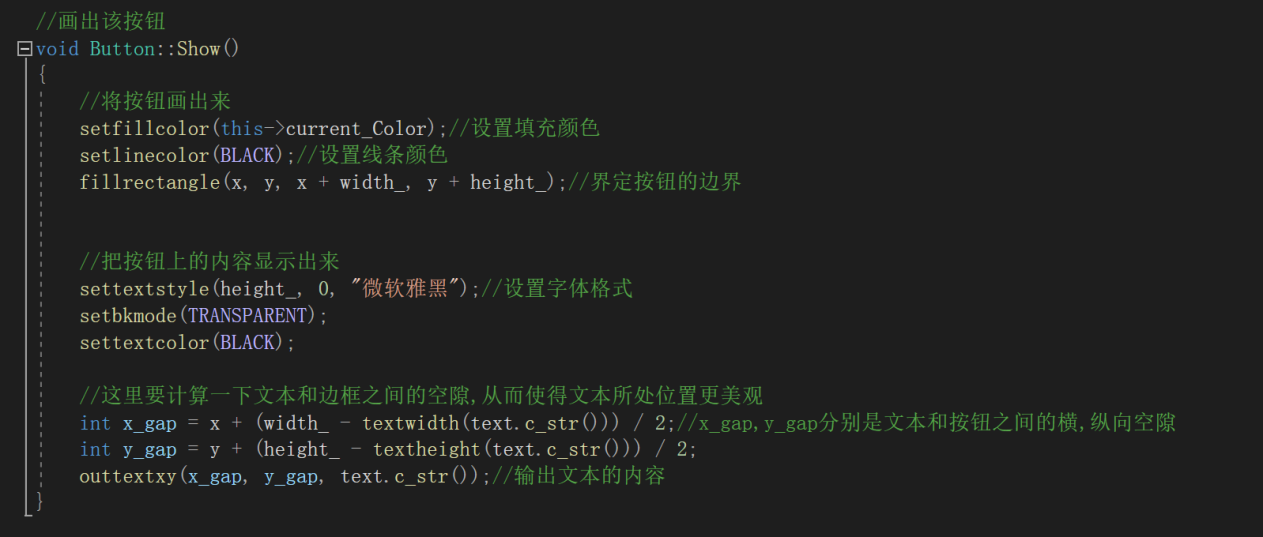




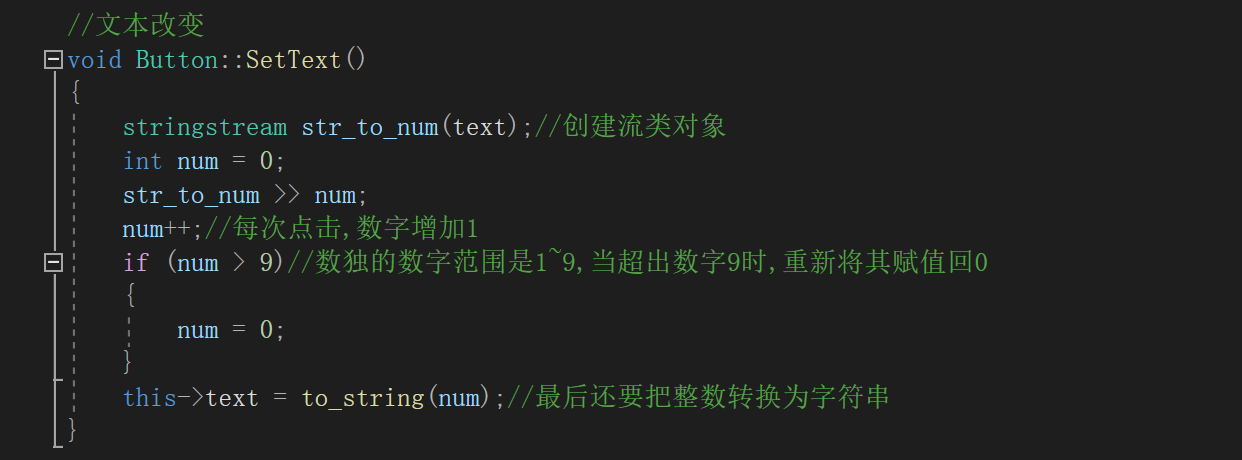
接下来对窗口内的内容进行修饰，在数独中，每个方格是最小的基本单位，其他的方格除了在数字上不同外，其他几乎完全相同，因此我对自身设计了一个按钮类（button），用于表示每个小方格，最后再用一个容器存储9\*9个按钮即可。这里需要说明的是，设置成按钮的形式有几个原因，一是避免用户往其中输入除了数字外的奇怪的内容；而是提升游戏的便捷性，按钮类要实现每点击一次就会桉顺序切换数字的功能，使用户通过鼠标点击就能玩，不用在动用键盘按键。

同时，在鼠标放到按钮上时，按钮还会变色，从而提醒用户当前所在的按钮，避免弄错。在设计时，还考虑了一些细节问题，例如，不是所有的按钮都能更改数字，只有需要用户填入的数字按钮上才能点击切换。为了突出这一特征，我在按钮的颜色上作了调整，使能改变的数字为红色，不能改变的数字为绿色。

button类的部分关键函数实现：



点击使得文本改变函数：



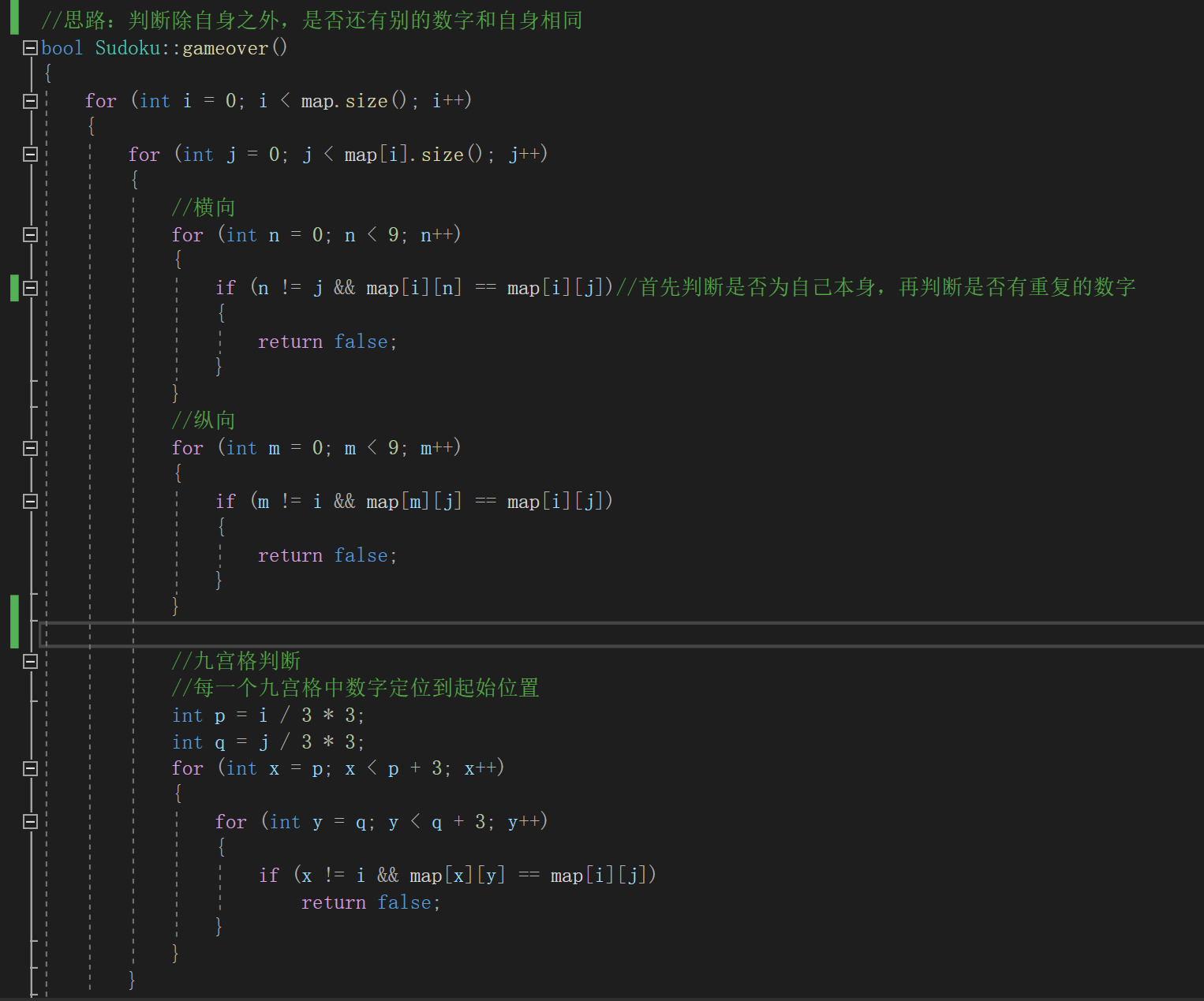
最后设计一个数独类：

其中包括对9\*9个按钮的存储实现，这里很自然地就想到用到vector容器，并且使用一个pair关联，对每个按钮上的数字是否改变做一个映射。对9\*9个button进行绘制，以及从鼠标上获取信息，使得当鼠标点击按钮时，其上的数字能够修改。

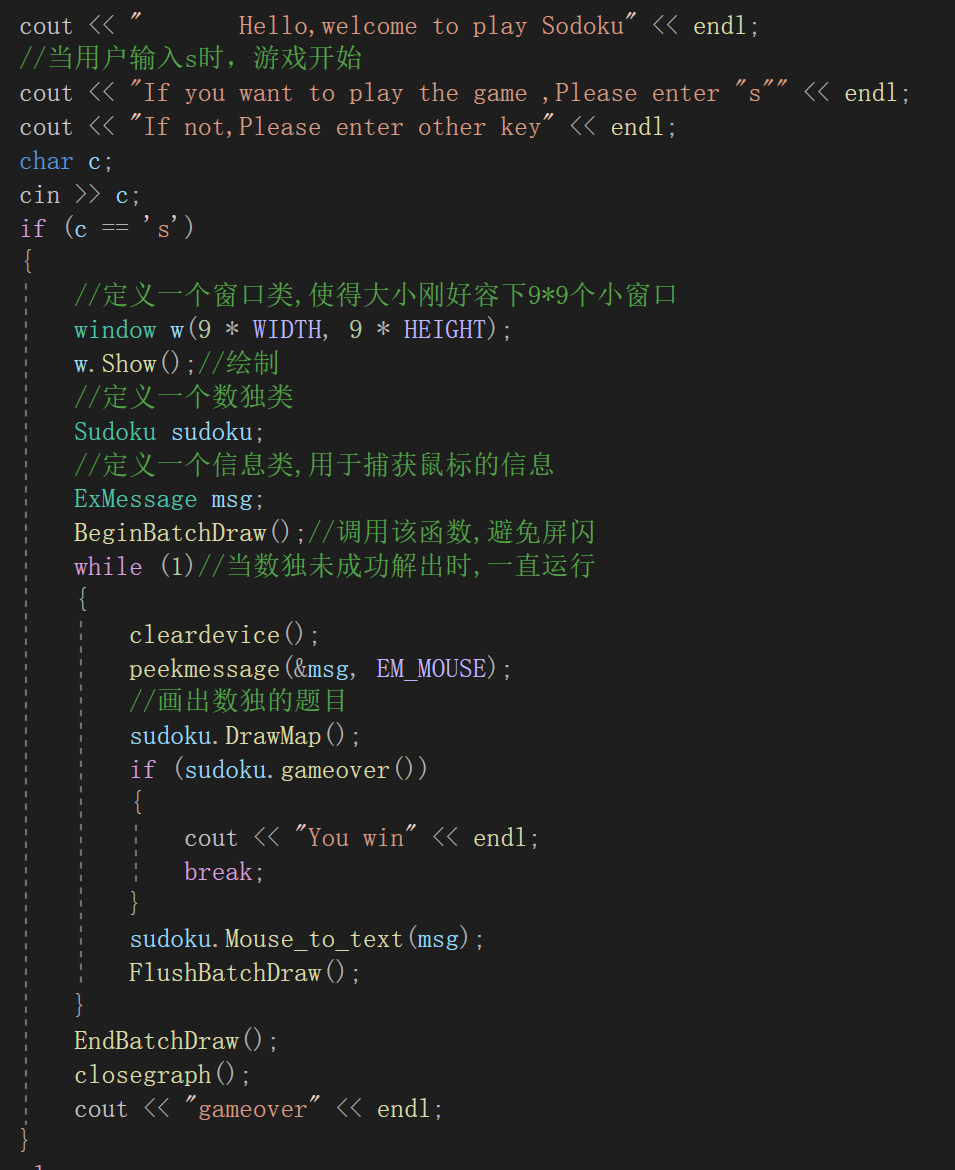
此外，还得设计一个检查函数，对每个数独类进行检查，检查玩家是否已经完全将数字天对了。

核心代码实现：

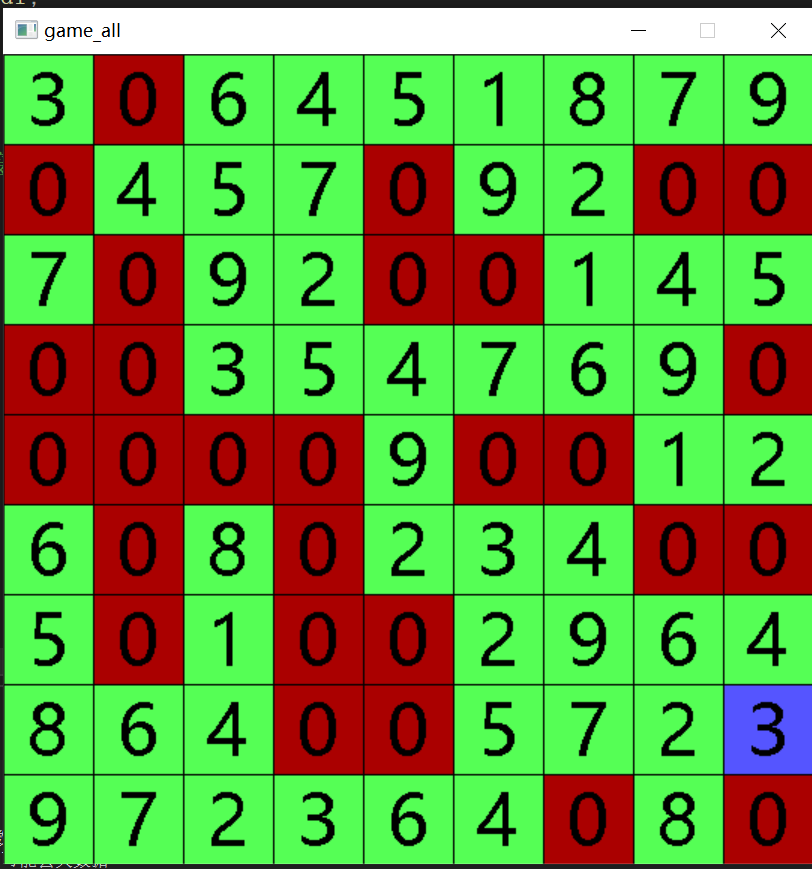
**检查函数：**



主函数代码：



运行结果：



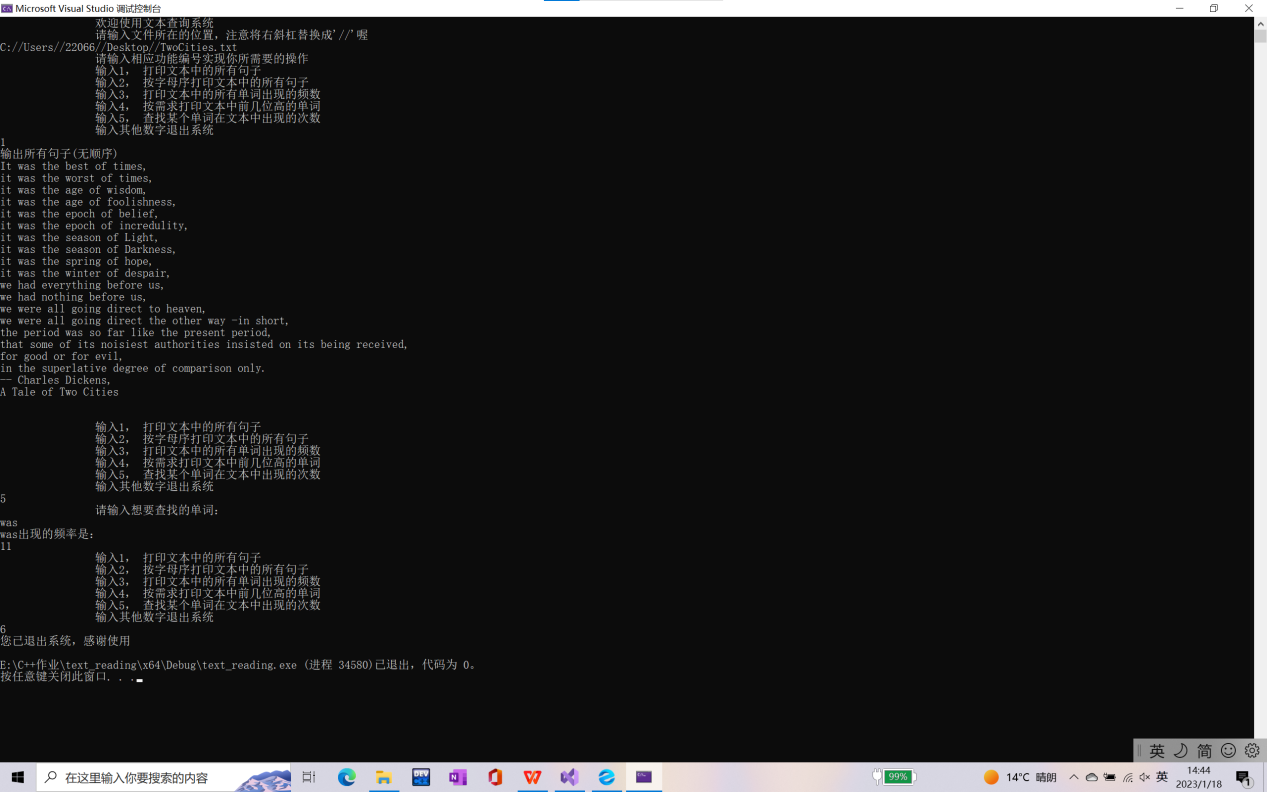
## 文本查询系统：

在实验七的基础上，我对该文本查询类进行了一些使其系统化的操作，例如通过在运行程序中手动输入需要读取的文件位置，按不同的数字进行不同的操作等。

个人认为该系统的实用价值还是比较高的，例如在一些需要进行关键字查找，搜索词频最高等功能的系统上便有用武之地。

其中的核心操作以及设计思想已经在实验七中的报告中有详细的讲解，这里就不再复制粘贴了。

运行结果如下：



## 心得总结：

我一共进行了3个项目的设计（图书管理，小游戏集合（贪吃蛇，数独），文本查询系统），通过对这三个项目的设计，我一共写了大约1800多行的代码。通过对这些项目的编写，我对本学期的C++的知识掌握更加深化了。对类的封装思想更是渗透进了我的生活中，对“类”这个字眼也有了更丰富的理解，虽然很累，但回想起来更多的还是泪。