**北京科技大学实验报告**

学院：计算机与通信工程学院 专业： 计算机类 班级：计2005

姓名：赵方程 学号：42024137 实验日期：2021 年 4月 29 日

**实验名称：**

**哈利波特书籍检索**

**实验目的：**

将哈利波特的7本书（txt格式）读入，然后在指定了人名/地名后，显示查询结果，选择指定查询结果序号（选择查询内容），能够显示指定查询结果所在位置前后的一段文字。

**实验仪器：**

计算机：Zephyrus G14

CPU：2.9GHz AMD Ryzen 7 4800HS

内存：2×8GB 3200MHz DDR4

硬盘：512GB

显卡：Nvidia Geforce RTX2060 with Max-q design

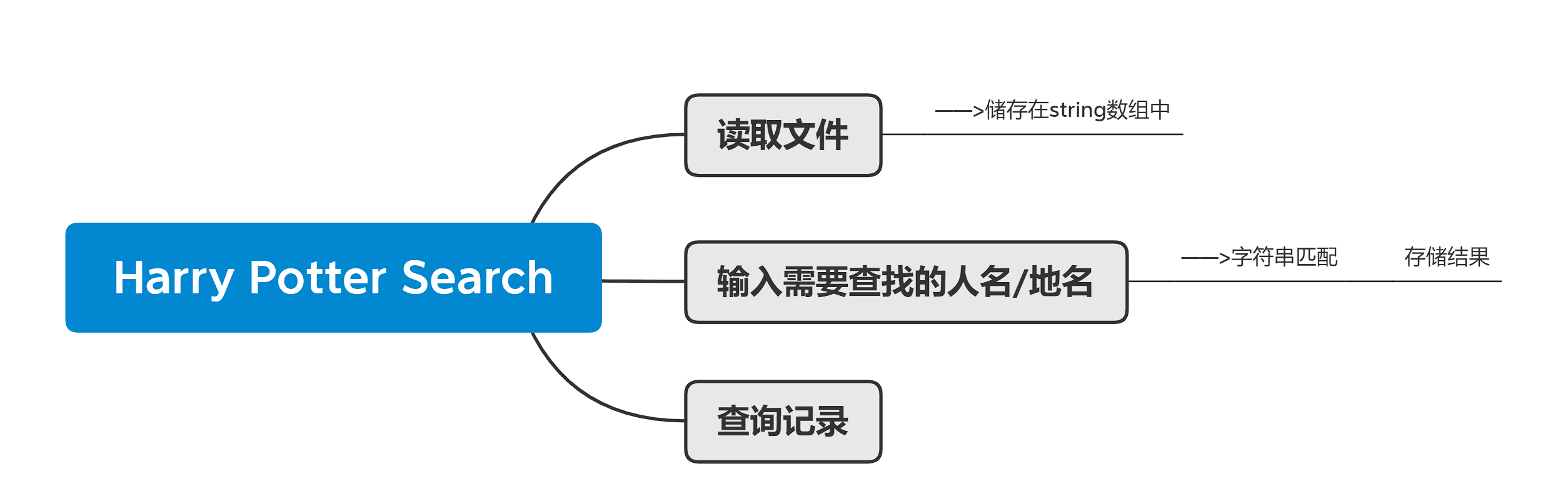
操作系统：Windows 10

编译器：MSVC 16.9.4 X64 Release

集成开发环境：Visual Studio 2019 Version16.9.4

**实验原理：**

主要的流程如下:



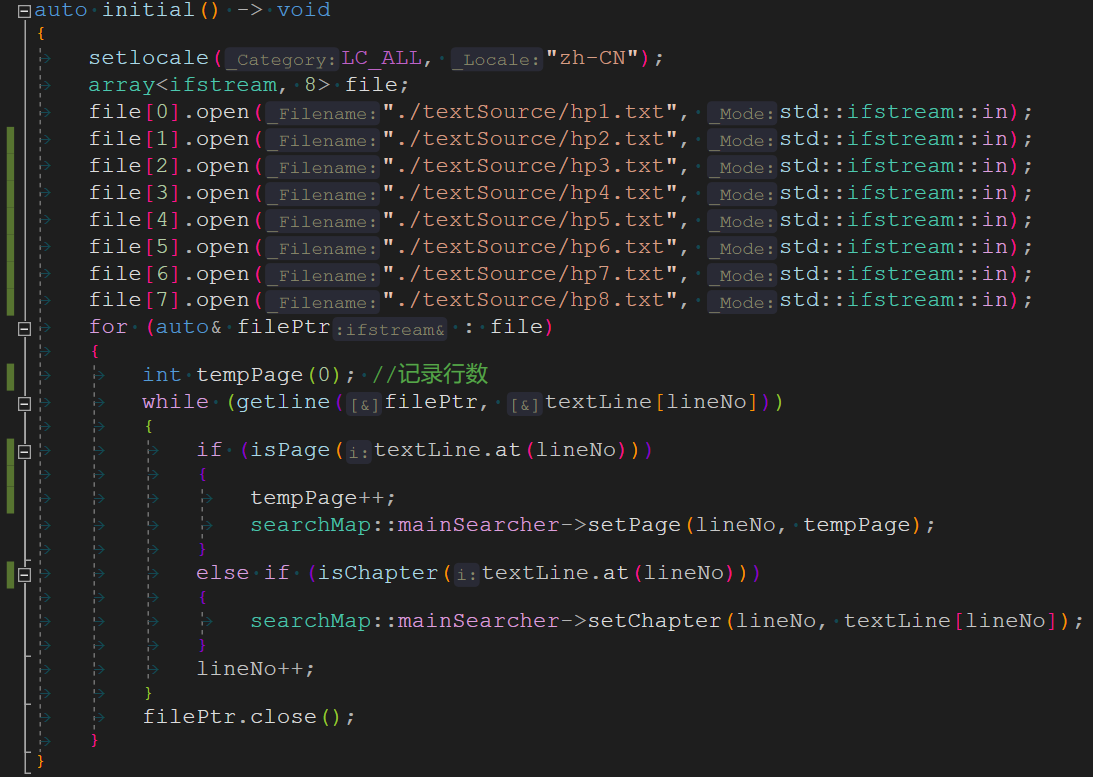
读取文件——>储存在string数组中——>输入需要查找的人名/地名——>字符串匹配——>存储结果——>查询记录

**实验内容与步骤：（字体：中文小四，英文Times New Roman小四，均不加粗）**

说明程序实现的步骤

1.读取文件并存储

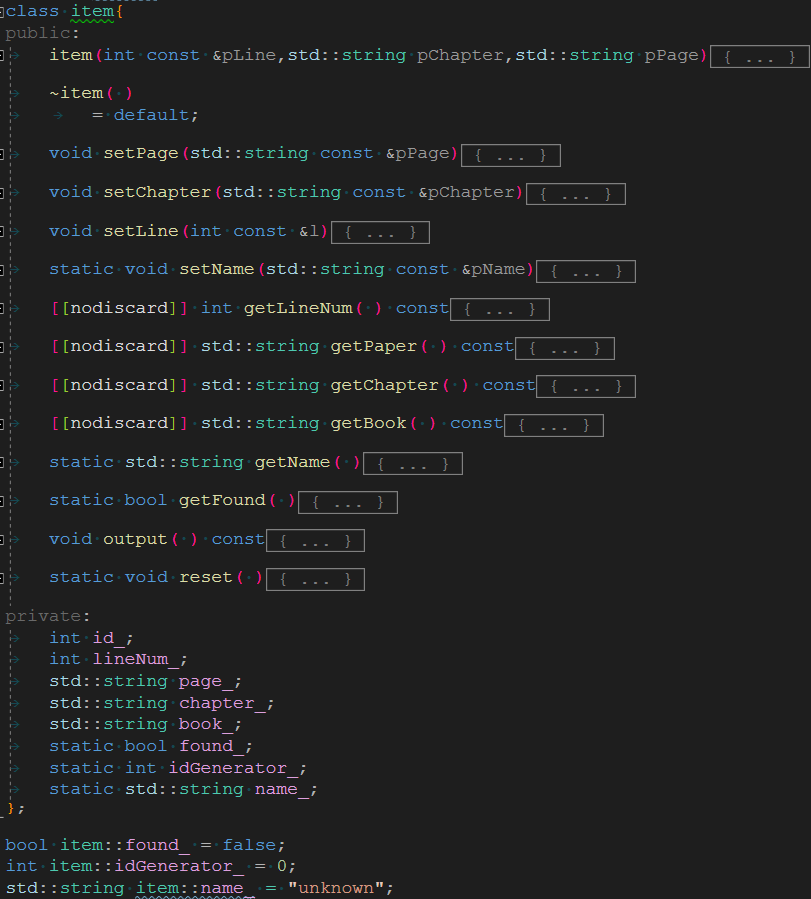
创建储存ifstream对象的数组array<ifstream,8> file,在initial()函数中依次打开,考虑到需要进行大量的查询操作,可以将其按行储存在数据类型为string的vector数组textLine中,查询时可以根据下标直接访问查询记录,时间复杂度为O(1)。读取文件时对文件内容进行判断,若为page或者chapter则将对应line存入searchMap（见下一步骤）中。储存完毕后关闭文件。



2. 创建类、vector数组

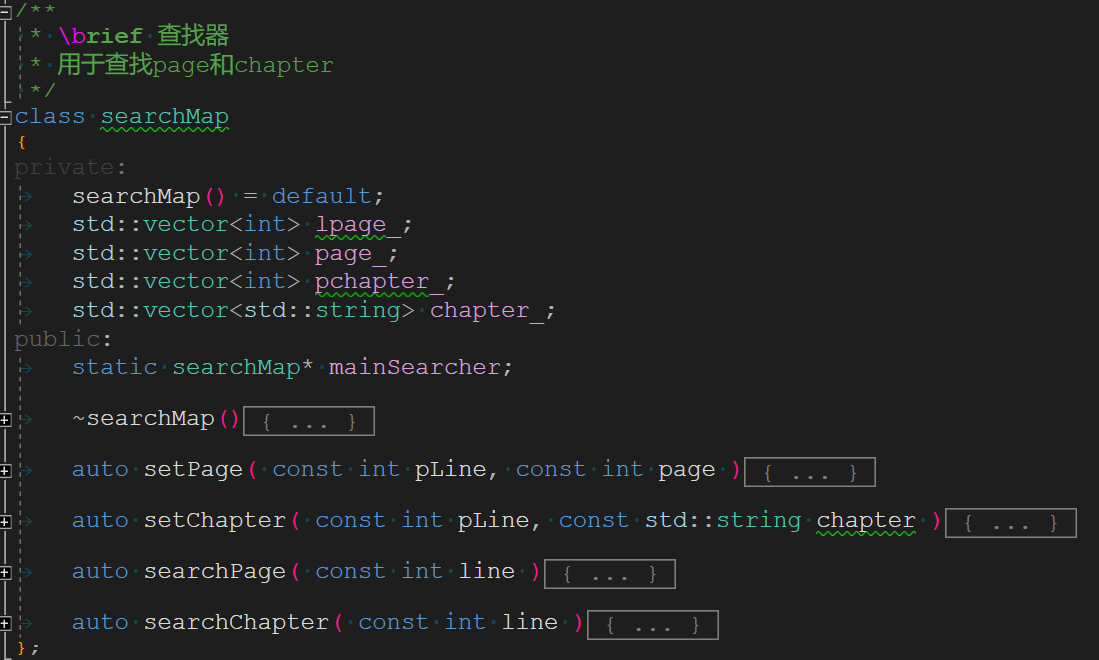
创建一个item类,用于记录某条特定的查找记录的数据,如序号、行数、页数、章节、书名等等，即私有成员：id，line，page，chapter，book。同时需要一个序号生成器即私有静态成员IdGenerator，以及所有item类对象共享的查找名称，即私有静态成员name以及是否被查找到的记号found。此外item类需提供对上述id，line，page，chapter，book访问的成员函数：getPaper() getLineNum() getChapter() getBook() getName()…...以及用于输出的函数output()。同时，由于记录一定是查找到之后再进行创建的，所以item的构造函数中要有形参line, chapter, page,并在构造函数中根据line判断该记录来自于哪本书。当未查找到时传入参数line=-1,chapter=”none”,page=”none”

创建一个数据类型为item的vector用于储存所有的查找记录数据。



item类

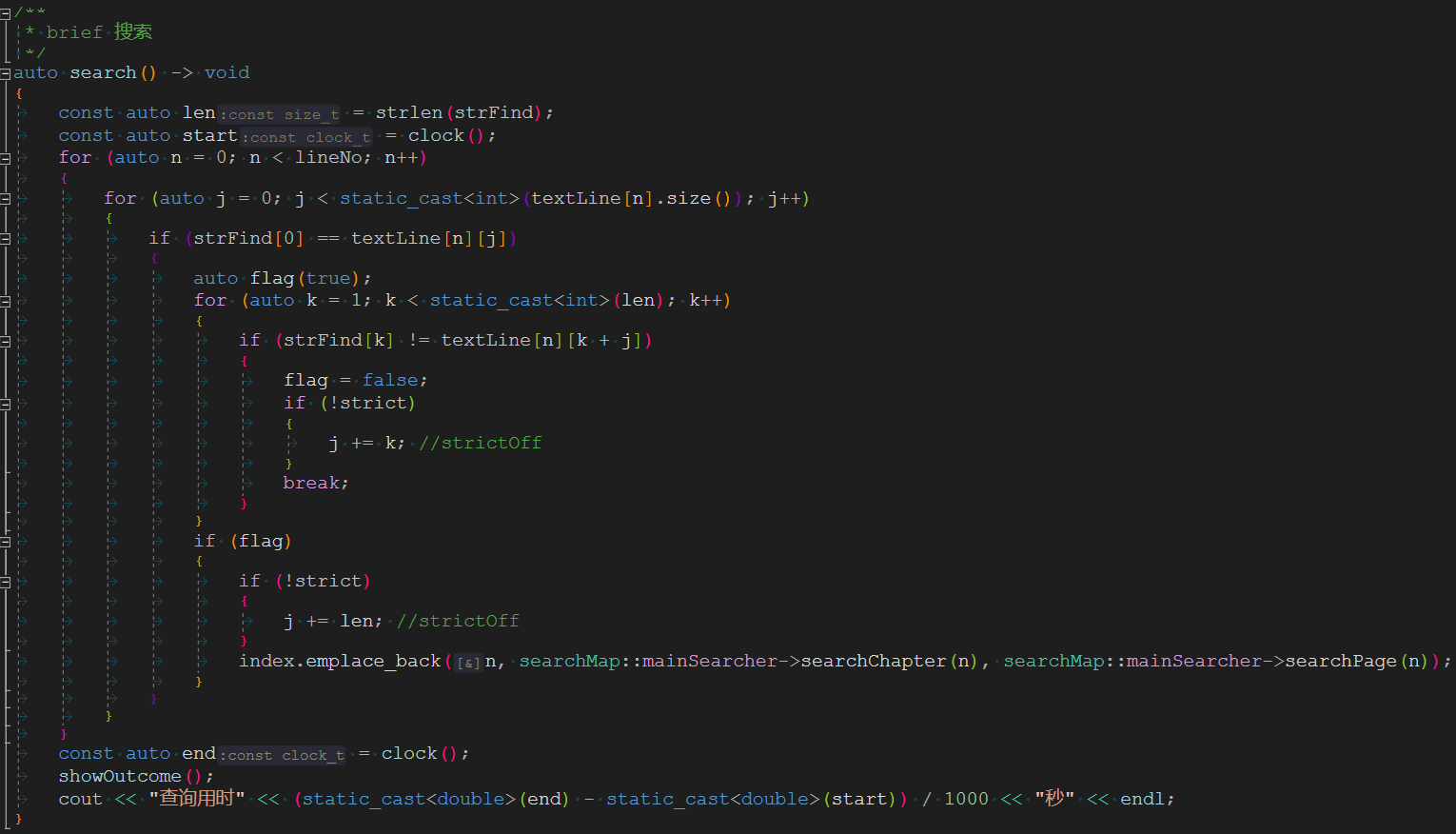
创建一个searchMap类，令其构造函数为私有，为公有searchMap\* 类型的变量mainSearcher分配内存。提供接口传入page及对应的结尾line和传入chapter及对应的开始line。提供接口接收line返回对应的page/chapter。



searchMap类

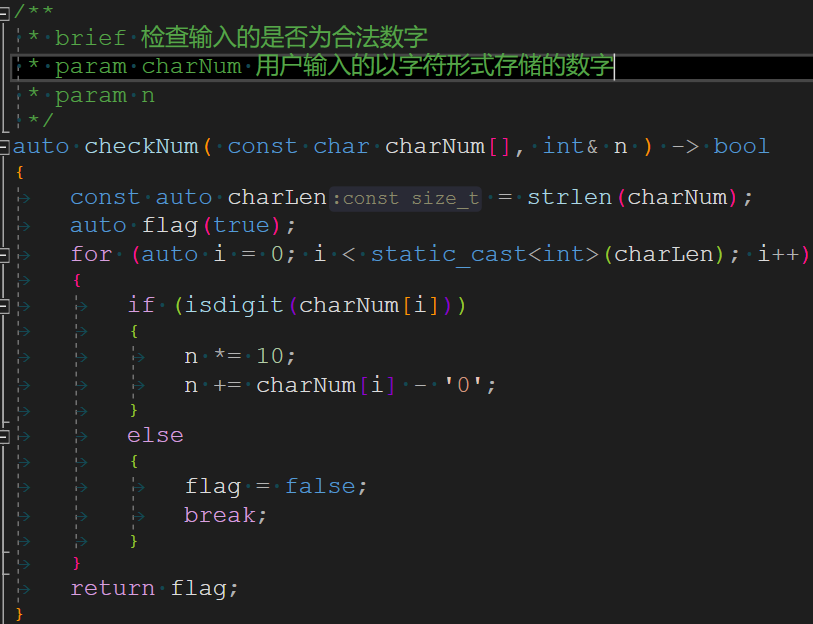
3. 字符串匹配

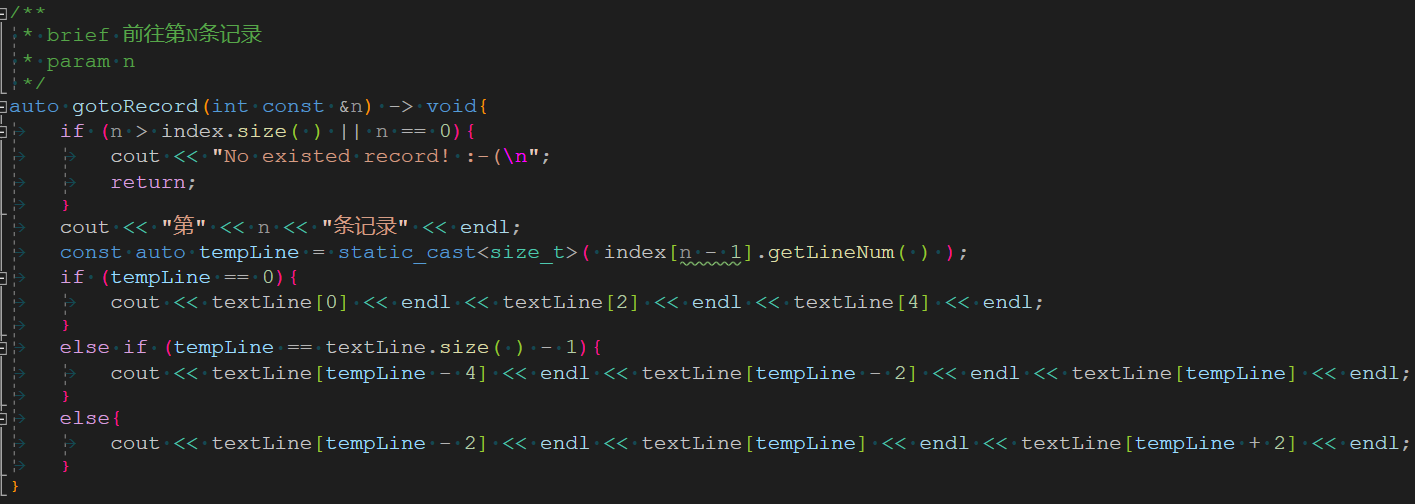
考虑到搜索人名/地名的特殊性，默认采用了并不严谨的字符串匹配方法，即在字符串失配时直接向后移动匹配成功的长度，匹配成功时直接向后移动子串的长度。这种方法在查找人名/地名时能提供较快的速度，匹配成功率也较高。但在搜索其他字符串的时候可能存在误差。故，在此引入一个默认关闭的字符串匹配严格模式（strict mode）,采用普通的算法(本来想用Boyer Moore算法的,但是试了一下不知道哪里写错了,比普通的算法还慢,就放弃了)。实际他们在release模式下速度差距很小



4. 跳转至某条记录

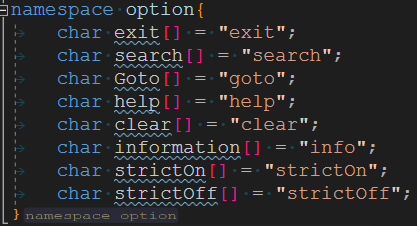
查找结束后共存在index.size()条记录,用户输入一个数,即跳转至该条记录。为了防止用户输入其他字符，构造checkNum()函数接收char型数组并判断是否是合法数字,若不是,输出” NaN\nPlease input a valid NUMBER!\n”,若是则调用gotoRecord()函数,输出”第N条记录\n”,及存储在line[index[n-1].getLineNum()]前后两行的字符串,考虑到边界条件,当查找记录在第一行或最后一行时输出textline[0]. textline[2] textline[4]或textline[index[n-1]. getLineNum()-4] line[index[n-1]. getLineNum()-2] line[index[n-1]. getLineNum()]





5. 其他

提供clear命令刷新控制台界面,info命令输出程序基本信息,help命令输出操作提示,exit命令退出程序。

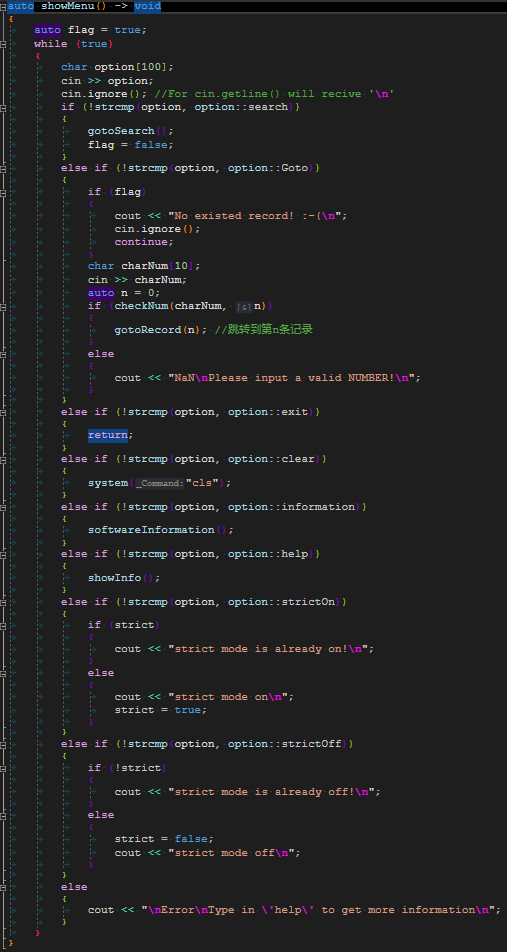


6. 用户输入错误

当无法匹配用户输入的指令时,提示"\nError\nType in \'help\' to get more information\n"。

7. 循坏接收指令

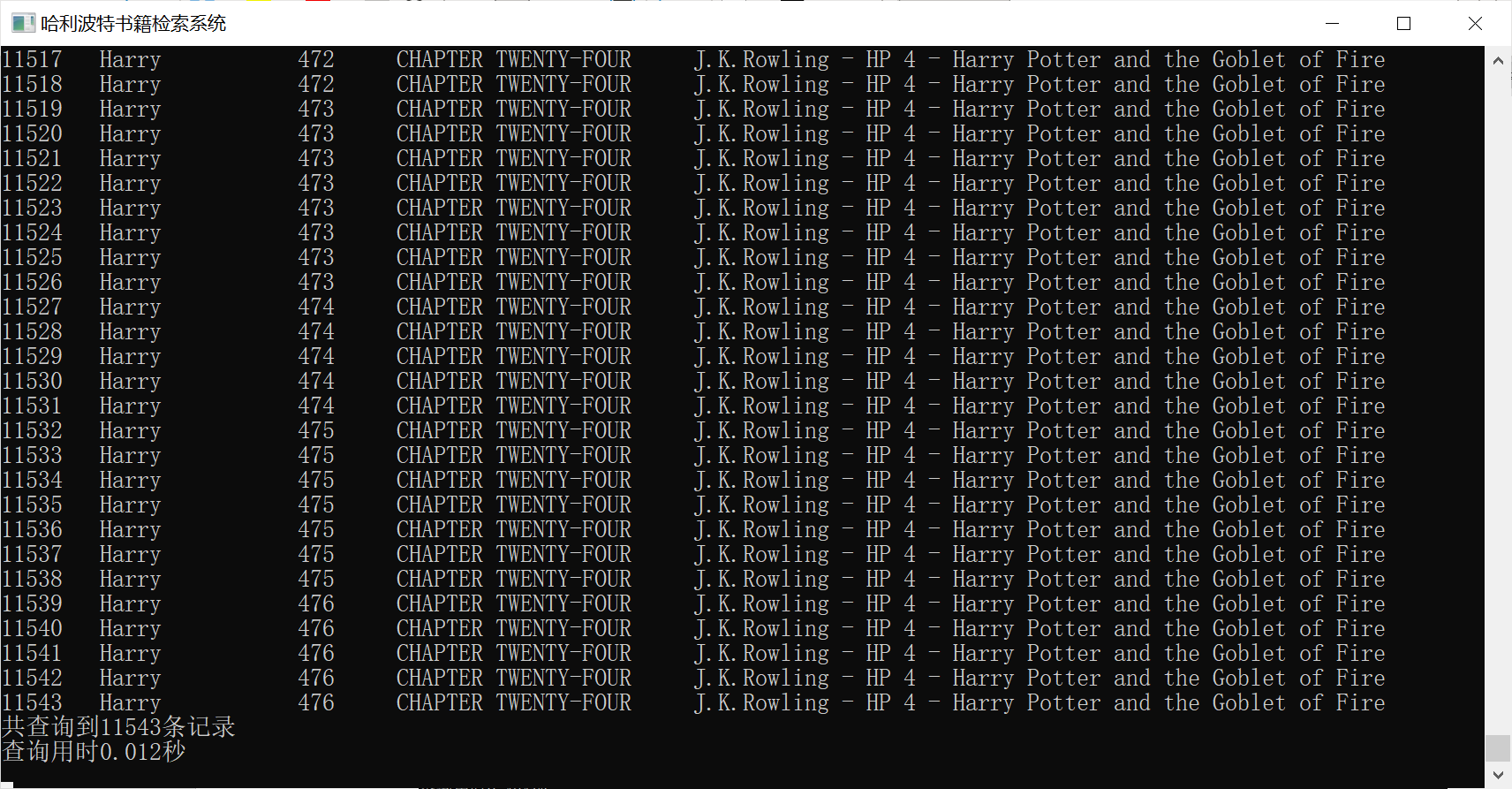
由于用户的操作是连续的,要求程序在执行完指令后能继续接收下一条指令,则应该将各个函数放入while(true)循环中,按照各个指令的使用频率将各个指令放入if(expression){expression;}else if(…){…}…中。



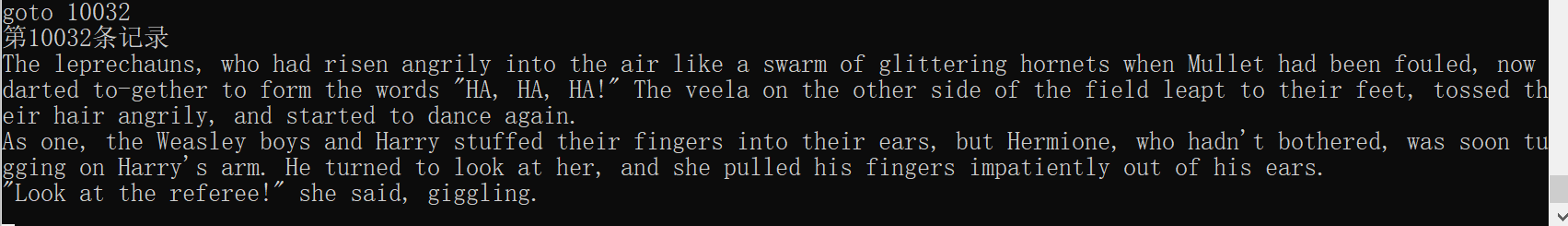
**实验结果与分析：每个步骤有运行截图和分析**

测试用例1：Harry

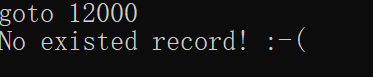
运行截图1：



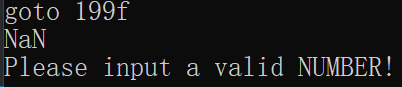
搜索Harry ,共有11543条记录



查找第10032条记录,输出上下文



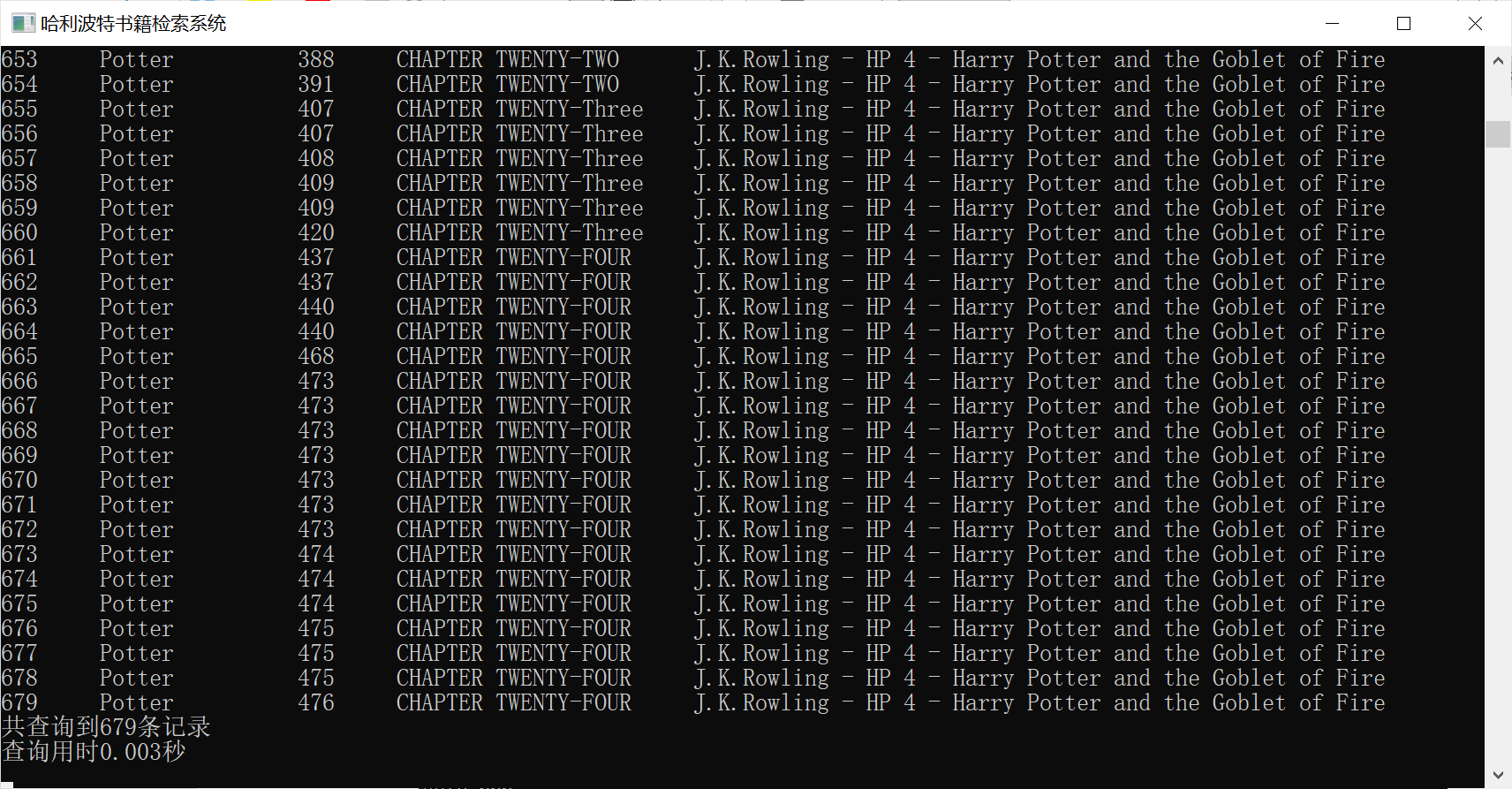
查找第12000条记录,记录不存在☹



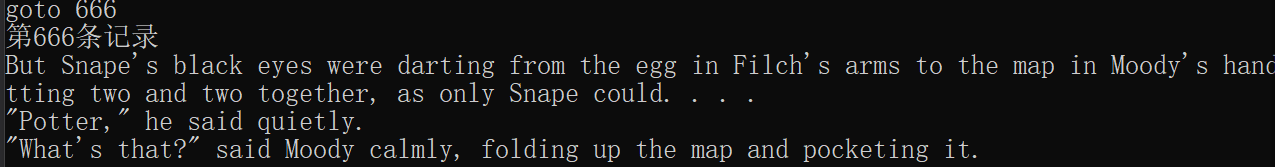
前往第199f条记录,输入非法

测试用例2：Potter

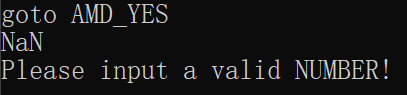
运行截图2：



搜素Potter,共有679条记录



前往第666条记录,输出上下文



前往第AMD\_YES条记录,输入非法

**实验结论与感想**

整个程序的算法并不难,重要的是搞清楚思路,把功能拆分开来看,自顶向下,开始有点模块化编程的感觉了,剩下的都是一些小坑,比如cin.getline()接受了缓冲区的换行符,以及并不规范的txt文件,中英文符号掺杂.最后,实现了基本功能后,还要考虑如果用户输入了不该输入的东西怎么办,为了防止程序异常退出(异常处理还不会呢)要怎么做等等.

还有代码的规范,这种小程序其实也没啥规范好讲,都是自己的个人风格,反正大体上还是要和主流差不多,命名法啥的.然后我写的有些用了C++11或者更新版本的特性,最近刚刚看了一下,有时候就有点奇怪,不过既然是程序设计实践当就是一个尝试,反正是小程序.像是**auto** function(parameters) ->**void**{expression;}

这样的写法以前还真没见到过,可能对于一些奇怪返回值的函数来说,这样判断返回值更清晰吧.在平时写的时候这么写确实有点奇怪，但是试试也无妨（？）在实践中学习嘿嘿~(respect)然后这个程序的提升空间的话,我觉得有字符串匹配算法，还有降低showMenu(), item::setBookName() 函数的复杂度等等.

然后再有的话我觉得也可以通过多线程查找来提高速度。