《数据结构与算法基础》课程项目

实施报告

题目： **虚拟智慧图书馆**

组 号：

任课教师：管业鹏

组 长： 15124540 姚乾浩 工作量比例：20%

成员： 15124537 涂阳明 工作量比例： 20%

成 员： 15124532 赵云 工作量比例： 20%

成 员： 15124504 林嘉敏 工作量比例：20%

成 员： 15122805 夏晓艺 工作量比例：20%

联系方式： 18217333260

yaoqianhaott@gmail.com

1. 项目要求

设计一个智能图书管理系统，模拟现有图书馆的借阅、续借、归还，图书入库、出库，查询、排序等功能。

1. 设计思想

参考一些大型图书馆比如上海图书馆、浦东新区图书馆等，都有一个检索系统，通过这个系统，借书者只需要输入书名、作者名、署名任一信息进行检索，即快速找出需要的图书的编号、书的位置、是否已借出等。

本项目要求的智能图书管理系统实质上与它类似，只是寻找位置等信息可以简化，而借书、还书等操作能够加入到系统中。

因此需要做的实际是一个关系型数据库，能够方便的检索，而为了用户友好，还需要设计一个简单易懂的GUI。

1. 实现方式

使用python进行编程，用其自带的SQLite数据库，tkinter编写GUI。

优点在于开发快速，并且相当轻量，并且可以跨平台。

缺点的话就是很丑，真的很丑。

1. 底层原理

SQLite中使用的数据结构是B树，使得无论在插入、删除、搜索、修改时，时间复杂度大致都在O（logn）。可以说是非常优秀了。

B树：

1.树中每个结点最多含有m个孩子（m>=2）；

2.除根结点和叶子结点外，其它每个结点至少有[ceil(m / 2)]个孩子（其中ceil(x)是一个取上限的函数）；

3.若根结点不是叶子结点，则至少有2个孩子（特殊情况：没有孩子的根结点，即根结点为叶子结点，整棵树只有一个根节点）；

4.所有叶子结点都出现在同一层，叶子结点不包含任何关键字信息(可以看做是外部接点或查询失败的接点，实际上这些结点不存在，指向这些结点的指针都为null)；

5.每个非终端结点中包含有n个关键字信息： (n，P0，K1，P1，K2，P2，......，Kn，Pn)。其中：  
       a)   Ki (i=1...n)为关键字，且关键字按顺序升序排序K(i-1)< Ki。   
       b)   Pi为指向子树根的接点，且指针P(i-1)指向子树种所有结点的关键字均小于Ki，但都大于K(i-1)。   
       c)   关键字的个数n必须满足： [ceil(m / 2)-1]<= n <= m-1。

B树比二叉搜索树的优点在于对于数据量大的情况，树的高度会比二叉搜索树小得多。

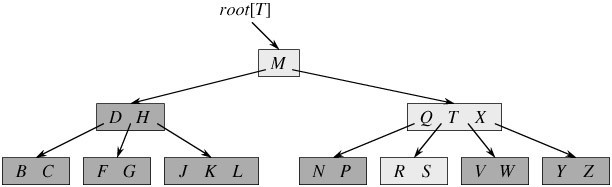
给出B树的高度公式：

 若B树某一非叶子节点包含N个关键字，则此非叶子节点含有N+1个孩子结点，而所有的叶子结点都在第I层，我们可以得出：

1. 因为根至少有两个孩子，因此第2层至少有两个结点。
2. 除根和叶子外，其它结点至少有┌m/2┐个孩子，
3. 因此在第3层至少有2\*┌m/2┐个结点，
4. 在第4层至少有2\*(┌m/2┐^2)个结点，
5. 在第 I 层至少有2\*(┌m/2┐^(l-2) )个结点，于是有： N+1 ≥ 2\*┌m/2┐I-2；
6. 考虑第L层的结点个数为N+1，那么2\*(┌m/2┐^(l-2)）≤N+1，也就是L层的最少结点数刚好达到N+1个，即： I≤ log┌m/2┐((N+1)/2 )+2；

　所以当B树包含N个关键字时，B树的最大高度为l-1（因为计算B树高度时，叶结点所在层不计算在内），即：l - 1 = **log┌m/2┐((N+1)/2 )+1**。

这个B树的高度公式从侧面显示了B树的查找效率是相当高的。



1. 使用工具的介绍
2. python的sqlite3接口以及sqlite语句

接口：

|  |  |
| --- | --- |
| 函数 | 作用 |
| sqlite3.connect(‘%s’) | 连接数据库 |
| connection.cursor(‘%s’) | sqlite3的自定义类，用于对数据库进行读写 |
| cursor.execute(‘%s’) | 调用sqlite语句 |
| connection.commit() | 提交修改 |
| connection.close() | 关闭数据库连接 |

Sqlite语句：

|  |  |
| --- | --- |
| 语句 | 作用 |
| CREATE TABLE \*\*（\*\* \*\*,\*\* \*\*……） | 在数据库中创建表 |
| INSERT INTO \*\*(\*\*,\*\*……)\VALUES (\*\*,\*\*……) | 在表中插入数据 |
| SELECT \*\* FROM \*\* | 从表中选择数据 |
| DELETE \*\* FROM \*\* | 从表中删除数据 |
| WHERE \*\*=/LIKE \*\* | 选择条件 |
| ORDER BY \*\* ASC/DECS | 按条件排序 |

1. python的tkinter语句

|  |  |
| --- | --- |
| 函数 | 作用 |
| tk.Tk() | 窗口主体 |
| tk.Entry（） | 输入文本框 |
| tk.ListBox() | 列表框 |
| tk.Menubar（） | 菜单 |
| tk.Button（） | 按键 |
| tk.Label() | 标签，可放文字，图片 |
| tk.scrollbar() | 滚动条 |
| ttk.Combobox() | 下拉选择框 |
| tk.title（） | 窗口标题 |
| Tk.geometry() | 窗口大小 |

1. 功能介绍

用户系统：

对于游客：只支持搜索功能

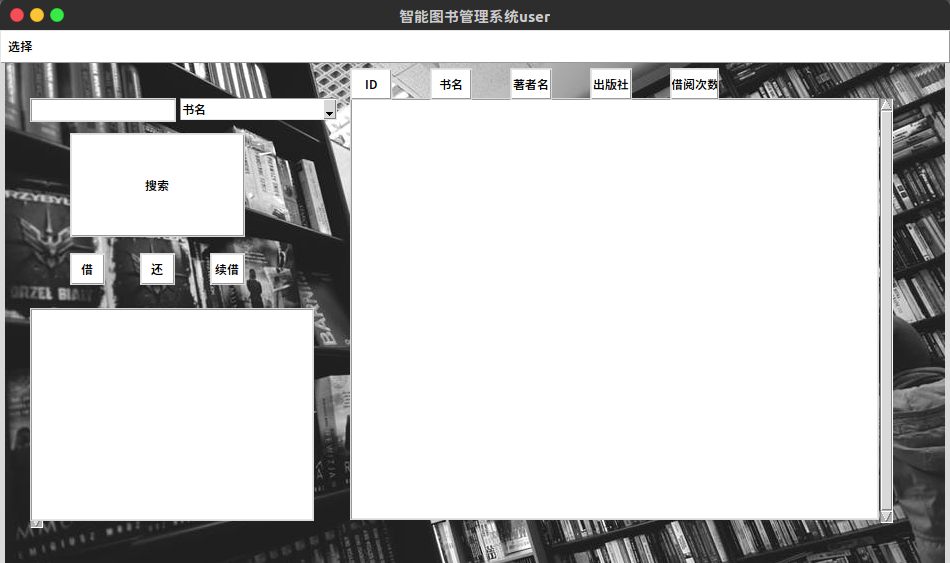
对于管理员：在游客的基础上，支持书籍的入库、出库功能

对于用户：在游客的基础上，支持借书、还书、续借功能

细节上，对于入库出库支持批量操作，对于搜索支持排序(安装ID升序，书名、著者、出版社拼音字典序，管理员有入库时间升序，以及热度降序)。

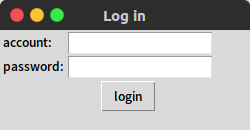
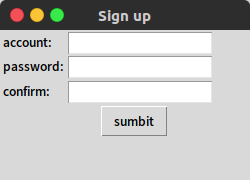
以下分别为三者的界面，仅有略微区别







以下为登录以及注册窗口



1. 实现过程与心得体会

由于暑假刚刚学了python，所以在最初的时候看到这个项目就决定使用python来实现。于是上网找了找，发现python标准库就自带了sqlite3的数据库接口以及tkinter的GUI接口，不用安装什么包，就就有点草率的决定了（后面证实这确实是一个败笔）。在了解了两个库的函数之后，马上就开始着手写了，结果搞出了不成熟的第一个作品，功能上倒是都能实现，但是界面实在不友好，丑也就算了（最后的完成作依旧丑），实际上借书还书都很不便利，需要先查询要借的书的ID之后，通过输入ID来借书。后来参考了网上一些作品最后完成的样子，发现我们做的实在是不上台面，就重新做了一个。

新版本比旧版本的优点如下：

1. 加入了滚动条，以实现超出框体的内容的查看
2. 使用了listbox（中文应该叫列表），能够直接点击需要借的书，然后点击借阅即可，不需要繁琐的查询ID，再到借书处去用输入ID的方式借书。
3. 界面的统一，把多个界面整合到一起，使用一个界面即可，友好程度大幅上升。
4. 外观稍微好看了一点，尽管依旧很丑。

实现过程中遇到的问题:

主要是在窗口实现的时候，最初写在全局的时候非常顺利，后来为了代码的简洁和复用，实现OOP的时候出了问题：封装之后主窗口上什么都没有。

上网查了各种各样的例子，无奈发现网上的例程有继承Frame的，有继承Tk的，比较混乱，去查文件，发现官方文件讲的非常简略。最后查错发现是重载构造函数的时候忘记调用父类的构造函数了。

第二是在试图美化的过程中。由于发现大家都做的非常精美，于是动了美化的心思，恰好组员有会用PS能够做一些素材的人才，弄了点图片什么的要试的时候，发现这个tkinter支持的东西非常少，并不能够作出很好看的界面。后来看了看做界面做好都是用C#来实现的，相关的库的支持会比较好一点，python的话实在不是很适合写图像界面，并且选的库也是主要面向自己写一些小程序，不太要求美观用的，所以在选择的时候实在有点草率。

1. 分工

姚乾浩：代码编写、报告整理

赵云：B树原理的了解、报告撰写

涂阳明：OOP方面的介绍、报告撰写

林嘉敏：tkinter的语句、参数整理，图片提供，报告撰写

夏晓艺：sqlite3语句的了解，报告撰写

1. 参考资料

<https://morvanzhou.github.io/tutorials/python-basic/basic/>

<http://www.runoob.com/sqlite/sqlite-syntax.html>

附录：代码

新版：

main.py

*#! /home/nero/.pyenv/versions/3.6.2/bin/python***import** better  
app=better.window(**''**)  
app.mainloop()

Better.py

**import** tkinter **as** tk  
**import** tkinter.ttk **as** ttk  
**from** PIL **import** ImageTk,Image  
**import** sqlite3  
**import** time  
**import** random  
**from** xpinyin **import** Pinyin  
**import** tkinter.messagebox  
**import** log\_in  
**import** sign\_up  
**import** \_insert  
  
p=Pinyin()  
  
**def** partition2(a, l, r, num):  
 *#print(num)  
 #print(a[l][0])* x = p.get\_pinyin(str(a[l][num]), **''**)  
 j = l  
 **for** i **in** range(l + 1, r + 1):  
 **if** p.get\_pinyin(str(a[i][num]), **''**) <= x:  
 j += 1  
 a[i], a[j] = a[j], a[i]  
 a[l], a[j] = a[j], a[l]  
 **return** j  
  
**def** randomized\_quick\_sort(a, l, r, num):  
 **if** l >= r:  
 **return** k = random.randint(l, r)  
 a[l], a[k] = a[k], a[l]  
 *# use partition3* m = partition2(a, l, r, num)  
 randomized\_quick\_sort(a, l, m - 1, num)  
 randomized\_quick\_sort(a, m + 1, r, num)  
  
**class** window(tk.Tk):  
 **def** \_\_init\_\_(self,account):  
 super().\_\_init\_\_()  
 self.tran = {}  
 self.tran[**'书名'**] = **'BOOKNAME'** self.tran[**'著者名'**] = **'AUTHORNAME'** self.tran[**'出版社'**] = **'PRESSNAME'** self.account = account  
 self.sw=0  
 self.geometry(**'950x500'**)  
 self.title(**'智能图书管理系统%s'**%self.account)  
  
 image = Image.open(**"lib4.jpg"**)  
 photo = ImageTk.PhotoImage(image)  
 self.label = tk.Label(self,image=photo)  
 self.label.image = photo *# keep a reference!* self.label.pack()  
  
  
 self.searchname = tk.Entry(self)  
 self.searchname.place(x=30, y=35, anchor=**'nw'**)  
 self.method = tk.StringVar()  
 self.selectmethod = ttk.Combobox(self, textvariable=self.method)  
 self.selectmethod[**'values'**] = (**'书名'**, **'著者名'**, **'出版社'**)  
 self.selectmethod.place(x=180, y=35, anchor=**'nw'**)  
 self.selectmethod.current(0)  
 self.searchbutton = tk.Button(self, text=**"搜索"**,width=21,height=5,background=**'white'**,command=self.search\_book)  
 self.searchbutton.place(x=70, y=70, anchor=**'nw'**)  
  
 self.scrollbar = tk.Scrollbar(self)  
 self.scrollbar.place(x=880,y=35,relheight =0.85,anchor=**'nw'**)  
  
 self.listbox = tk.Listbox(self,width=75,height=22,yscrollcommand=self.scrollbar.set)  
 self.listbox.place(x=350,y=35,anchor=**'nw'**)  
 self.scrollbar.config(command=self.listbox.yview)  
  
 self.menubar = tk.Menu(self,background=**'white'**)  
 self.menu = tk.Menu(self.menubar, tearoff=0)  
 self.menubar.add\_cascade(label=**'选择'**, menu=self.menu)  
 self.menu.add\_command(label=**'登录'**, command=self.login)  
 self.menu.add\_command(label=**'注册'**, command=self.signup)  
 self.config(menu=self.menubar)  
  
 self.swid=tk.Button(self,text=**"ID"**,width=2,height=1,background=**'white'**,command=self.setswid)  
 self.swid.place(x=350,y=5,anchor=**'nw'**)  
 self.swbookname = tk.Button(self, text=**"书名"**, width=2, height=1, background=**'white'**, command=self.setswbookname)  
 self.swbookname.place(x=430, y=5, anchor=**'nw'**)  
 self.swauthorname = tk.Button(self, text=**"著者名"**, width=2, height=1, background=**'white'**, command=self.setswauthorname)  
 self.swauthorname.place(x=510, y=5, anchor=**'nw'**)  
 self.swpressname = tk.Button(self, text=**"出版社"**, width=2, height=1, background=**'white'**, command=self.setswpressname)  
 self.swpressname.place(x=590, y=5, anchor=**'nw'**)  
 self.swhot = tk.Button(self, text=**"借阅次数"**, width=3, height=1, background=**'white'**,  
 command=self.setswhot)  
 self.swhot.place(x=670, y=5, anchor=**'nw'**)  
  
 **if** self.account==**'ADMIN'**:  
 self.inbook=tk.Button(self,text=**"入库"**,width=3,height=3,background=**'white'**,command=self.in\_book)  
 self.inbook.place(x=100,y=225,anchor=**'nw'**)  
 self.outbook=tk.Button(self,text=**"出库"**,width=3,height=3,background=**'white'**,command=self.out\_book)  
 self.outbook.place(x=140, y=325, anchor=**'nw'**)  
 self.infromfile=tk.Button(self,text=**"从文件"**,width=3,height=3,background=**'white'**,  
 command=self.in\_fromfile)  
 self.infromfile.place(x=190,y=225,anchor=**'nw'**)  
 self.outfromfile = tk.Button(self, text=**"从文件"**, width=3, height=3, background=**'white'**,  
 command=self.out\_fromfile)  
 self.outfromfile.place(x=230, y=325, anchor=**'nw'**)  
 self.swtime = tk.Button(self, text=**"入库时间"**, width=3, height=1, background=**'white'**,  
 command=self.setswtime)  
 self.swtime.place(x=770, y=5, anchor=**'nw'**)  
  
 **elif** self.account!=**''**:  
 self.owesb = tk.Scrollbar(self)  
 self.owesb.place(x=30, y=300, relheight=0.33, anchor=**'nw'**)  
 self.owelb = tk.Listbox(self, width=40, height=11,xscrollcommand=self.owesb.set)  
 self.owelb.place(x=30, y=245, anchor=**'nw'**)  
 self.owesb.config(command=self.owelb.xview)  
 self.showowe()  
 self.borrowbutton=tk.Button(self,text=**"借"**,width=1,height=1,background=**'white'**,command=self.borrowbook)  
 self.borrowbutton.place(x=70,y=190,anchor=**'nw'**)  
 self.returnbutton = tk.Button(self, text=**"还"**,width=1,height=1,background=**'white'**,command=self.returnbook)  
 self.returnbutton.place(x=140, y=190, anchor=**'nw'**)  
 self.renewbutton = tk.Button(self, text=**"续借"**, width=1,height=1,background=**'white'**,command=self.renewbook)  
 self.renewbutton.place(x=210, y=190, anchor=**'nw'**)  
 self.getcnt()  
  
 **def** setswid(self):  
 self.sw=0  
 self.search\_book()  
  
 **def** setswbookname(self):  
 self.sw=1  
 self.search\_book()  
  
 **def** setswauthorname(self):  
 self.sw=2  
 self.search\_book()  
  
 **def** setswpressname(self):  
 self.sw=3  
 self.search\_book()  
  
 **def** setswhot(self):  
 self.sw=5  
 self.search\_book()  
  
 **def** setswtime(self):  
 self.sw=6  
 self.search\_book()  
  
 **def** login(self):  
 self.destroy()  
 app=log\_in.login()  
 app.mainloop()  
 **return  
  
 def** signup(self):  
 self.destroy()  
 app=sign\_up.signup()  
 app.mainloop()  
 **return  
  
 def** in\_book(self):  
 app=\_insert.insert()  
 app.mainloop()  
 **return  
  
 def** in\_fromfile(self):  
 f = open(**'inbook.txt'**, **'r'**)  
 lines = f.readlines()  
 f.close()  
 conn = sqlite3.connect(**"book.db"**)  
 c = conn.cursor()  
 **for** i **in** lines:  
 l=i.split()  
 id=1  
 cursor = c.execute(**"SELECT ID FROM BOOKINFO"**)  
 **for** row **in** cursor:  
 **if** row[0] != id:  
 **break** id = id + 1  
 *#print(id)  
 #localtime = time.asctime(time.localtime(time.time()))* c.execute(**"INSERT INTO BOOKINFO (ID,BOOKNAME,AUTHORNAME,PRESSNAME,STATUS,INSERTTIME) \  
 VALUES (%d,'%s','%s','%s','可借',%f)"** % (id, l[0], l[1], l[2], time.time()))  
 conn.commit()  
 conn.close()  
 self.search\_book()  
 tkinter.messagebox.showinfo(**'message'**, **'Success!'**)  
  
 **def** out\_book(self):  
 fail=()  
 **if** self.listbox.curselection()==fail:  
 **return** value = self.listbox.get(self.listbox.curselection())  
 id=int(value[0])  
 conn = sqlite3.connect(**"book.db"**)  
 c = conn.cursor()  
 cursor = c.execute(**"SELECT STATUS FROM BOOKINFO WHERE ID = %d"** % id)  
 status = **None  
 for** row **in** cursor:  
 status = row[0]  
 **if** status==**'可借'**:  
 c.execute(**"DELETE FROM BOOKINFO WHERE ID=%d"**%id)  
 conn.commit()  
 conn.close()  
 self.search\_book()  
 tkinter.messagebox.showinfo(**'message'**, **'Success!'**)  
 **else**:  
 conn.close()  
 tkinter.messagebox.showinfo(**'message'**, **'The book is rent!'**)  
  
 **def** out\_fromfile(self):  
 f = open(**'inbook.txt'**, **'r'**)  
 lines = f.readlines()  
 f.close()  
 conn = sqlite3.connect(**"book.db"**)  
 c = conn.cursor()  
 **for** i **in** lines:  
 l = i.split()  
 c.execute(**"DELETE FROM BOOKINFO WHERE BOOKNAME = '%s' AND AUTHORNAME = '%s' AND PRESSNAME='%s'"**%(l[0],l[1],l[2]))  
 conn.commit()  
 conn.close()  
 self.search\_book()  
 tkinter.messagebox.showinfo(**'message'**, **'Success!'**)  
  
 **def** search\_book(self):  
 searchname = self.searchname.get() **or None  
 if** searchname == **None**:  
 **if** self.account==**'ADMIN'**:  
 conn = sqlite3.connect(**"book.db"**)  
 c = conn.cursor()  
 cursor = c.execute(**"SELECT ID,BOOKNAME,AUTHORNAME,PRESSNAME,STATUS,BORROWTIME,INSERTTIME FROM BOOKINFO"**)  
 output=[]  
 **for** row **in** cursor:  
 output.append(row)  
 conn.close()  
 *# print(output)* **if** self.sw!=0:  
 self.mysort(output)  
 self.listbox.delete(**'0'**, **'end'**)  
 **for** item **in** output:  
 self.listbox.insert(**'end'**, str(item[0])+**' '**+item[1]+**' '**+item[2]+**' '**+item[3]+**' '**+item[4]+**' '**+item[5]+**' '**+time.asctime(time.localtime(item[6])))  
 **return** conn = sqlite3.connect(**"book.db"**)  
 c = conn.cursor()  
 *# print("SELECT BOOKNAME,AUTHORNAME,PRESSNAME,STATUS from BOOKINFO WHERE %s LIKE '%%%s%%'"%(str.upper(method.get()),searchname))* cursor = c.execute(**"SELECT ID,BOOKNAME,AUTHORNAME,PRESSNAME,STATUS,BORROWTIME,INSERTTIME FROM BOOKINFO WHERE %s LIKE '%%%s%%' ORDER BY ID ASC"** % (self.tran[self.method.get()], searchname))  
 output = []  
 **for** row **in** cursor:  
 output.append(row)  
 conn.close()  
 *# print(output)* **if** self.sw!=0:  
 self.mysort(output)  
 self.listbox.delete(**'0'**,**'end'**)  
 **for** item **in** output:  
 **if** self.account==**'ADMIN'**:  
 self.listbox.insert(**'end'**, str(item[0])+**' '**+item[1]+**' '**+item[2]+**' '**+item[3]+**' '**+item[4]+**' '**+item[5]+**' '**+time.asctime(time.localtime(item[6])))  
 **else**:  
 self.listbox.insert(**'end'**, str(item[0]) + **' '** + item[1] + **' '** + item[2] + **' '** + item[3] + **' '** + item[4]+**' '**+item[5])  
  
 **def** showowe(self):  
 conn = sqlite3.connect(**"./userinfo/%s.db"** % self.account)  
 c = conn.cursor()  
 cursor = c.execute(**"SELECT ID,BOOKNAME,AUTHORNAME,PRESSNAME,RETURNTIME,STATUS FROM OWE ORDER BY ID ASC"**)  
 owe = []  
 **for** row **in** cursor:  
 owe.append(**'%d %s %s %s %s %s'** % (  
 row[0], row[1], row[2], row[3], time.asctime(time.localtime(float(row[4]))), row[5]))  
 conn.close()  
 self.owelb.delete(**'0'**, **'end'**)  
 **for** item **in** owe:  
 self.owelb.insert(**'end'**, item)  
  
 **def** borrowbook(self):  
 fail = ()  
 **if** self.listbox.curselection() == fail:  
 **return** self.getcnt()  
 **if** self.cnt==10:  
 tkinter.messagebox.showinfo(**'message'**, **'不能再借了!'**)  
 **return** value = self.listbox.get(self.listbox.curselection())  
 id=0  
 i=0  
 **while** value[i]!=**' '**:  
 id = id\*10+int(value[i])  
 i=i+1  
 print(id)  
 conn = sqlite3.connect(**"book.db"**)  
 c = conn.cursor()  
 cursor = c.execute(**"SELECT BOOKNAME,AUTHORNAME,PRESSNAME,STATUS,BORROWTIME FROM BOOKINFO WHERE ID=%d"** % id)  
 needstatus = **None** needbookname = **None** needauthorname = **None** needpressname = **None** borrowtime=**None  
 for** row **in** cursor:  
 needbookname = row[0]  
 needauthorname = row[1]  
 needpressname = row[2]  
 needstatus = row[3]  
 borrowtime=row[4]  
 **if** needstatus == **'可借'**:  
 c.execute(**"UPDATE BOOKINFO SET STATUS = '不可借',BORROWTIME=%d WHERE ID=%d"** % (id,borrowtime+1))  
 conn.commit()  
 conn.close()  
 timel = time.time()  
 timel = timel + 60 \* 60 \* 24 \* 30 *# 30days* conn = sqlite3.connect(**"./userinfo/%s.db"** % self.account)  
 c = conn.cursor()  
 c.execute(**"INSERT INTO OWE (ID,BOOKNAME,AUTHORNAME,PRESSNAME,RETURNTIME,STATUS) \  
 VALUES (%d,'%s','%s','%s','%s','%s')"** % (  
 id, needbookname, needauthorname, needpressname, str(timel), **'可续借'**))  
 conn.commit()  
 conn.close()  
 tkinter.messagebox.showinfo(**'message'**, **'Success!'**)  
 self.search\_book()  
 self.showowe()  
 **else**:  
 conn.close()  
 tkinter.messagebox.showinfo(**'message'**, **'这本书已经被借走了!'**)  
 **return  
  
 def** returnbook(self):  
 fail = ()  
 **if** self.owelb.curselection() == fail:  
 **return** value = self.owelb.get(self.owelb.curselection())  
 id = 0  
 i = 0  
 **while** value[i] != **' '**:  
 id = id \* 10 + int(value[i])  
 i = i + 1  
 conn = sqlite3.connect(**"./userinfo/%s.db"** % self.account)  
 c = conn.cursor()  
 c.execute(**"DELETE FROM OWE WHERE ID=%d"** % id)  
 conn.commit()  
 conn.close()  
 conn = sqlite3.connect(**"book.db"**)  
 c = conn.cursor()  
 c.execute(**"UPDATE BOOKINFO SET STATUS = '可借' where ID=%d"** % id)  
 conn.commit()  
 tkinter.messagebox.showinfo(**'message'**, **'Success!'**)  
 self.search\_book()  
 self.showowe()  
  
 **def** renewbook(self):  
 fail = ()  
 **if** self.owelb.curselection() == fail:  
 **return** value = self.owelb.get(self.owelb.curselection())  
 id = 0  
 i = 0  
 **while** value[i] != **' '**:  
 id = id \* 10 + int(value[i])  
 i = i + 1  
 print(id)  
 conn = sqlite3.connect(**"./userinfo/%s.db"** % self.account)  
 c = conn.cursor()  
 cursor = c.execute(**"SELECT ID,RETURNTIME,STATUS FROM OWE WHERE ID = %d"**%id)  
 **for** row **in** cursor:  
 print(row)  
 **if** row[2] == **'可续借'**:  
 c.execute(**"UPDATE OWE SET RETURNTIME='%s',STATUS = '不可续借' WHERE ID = %d"** % (  
 str(float(row[1]) + 60 \* 60 \* 24 \* 30), id))  
 tkinter.messagebox.showinfo(**'message'**, **'Success!'**)  
 **else**:  
 tkinter.messagebox.showinfo(**'message'**, **'不能再借了!'**)  
  
 conn.commit()  
 conn.close()  
 self.search\_book()  
 self.showowe()  
  
 **def** mysort(self,a):  
 randomized\_quick\_sort(a,0,len(a)-1,self.sw)  
  
 **def** getcnt(self):  
 conn = sqlite3.connect(**"./userinfo/%s.db"** % self.account)  
 c = conn.cursor()  
 cursor = c.execute(**"SELECT ID FROM OWE "**)  
 self.cnt=0  
 **for** row **in** cursor:  
 self.cnt=self.cnt+1  
 conn.commit()  
 conn.close()  
 **return  
  
\_insert.py**

**import** sqlite3  
**import** tkinter **as** tk  
**import** tkinter.messagebox  
**import** time  
  
**class** insert(tk.Tk):  
  
 **def** \_\_init\_\_(self):  
 super().\_\_init\_\_()  
 self.title(**'insertbook'**)  
 self.geometry(**'300x150'**)  
  
 self.booknamelab=tk.Label(self,text=**"书名:"**)  
 self.booknamelab.place(x=1,y=1,anchor=**'nw'**)  
 self.authornamelab = tk.Label(self, text=**"著者名:"**)  
 self.authornamelab.place(x=1, y=20, anchor=**'nw'**)  
 self.pressnamelab = tk.Label(self, text=**"出版社:"**)  
 self.pressnamelab.place(x=1, y=40, anchor=**'nw'**)  
 self.bookname = tk.Entry(self)  
 self.bookname.place(x=80, y=1, anchor=**'nw'**)  
 self.authorname = tk.Entry(self)  
 self.authorname.place(x=80, y=20, anchor=**'nw'**)  
 self.pressname = tk.Entry(self)  
 self.pressname.place(x=80, y=40, anchor=**'nw'**)  
 self.insert = tk.Button(self, text=**"insert"**,relief=**'raised'**, command=self.insertbook)  
 self.insert.place(x=80, y=70, anchor=**'nw'**)  
  
 **def** insertbook(self):  
 bookname = self.bookname.get() **or None** authorname = self.authorname.get() **or None** pressname = self.pressname.get() **or None** *#localtime = time.asctime(time.localtime(time.time()))* **if** bookname == **None or** authorname == **None or** pressname == **None**:  
 **return** conn = sqlite3.connect(**"book.db"**)  
 c = conn.cursor()  
 cursor = c.execute(**"SELECT ID,BOOKNAME,AUTHORNAME,PRESSNAME,STATUS FROM BOOKINFO ORDER BY ID ASC"**)  
 id = 1  
 **for** row **in** cursor:  
 **if** row[0] != id:  
 **break** id = id + 1  
 print(id)  
 print(bookname, authorname, pressname)  
 c.execute(**"INSERT INTO BOOKINFO (ID,BOOKNAME,AUTHORNAME,PRESSNAME,STATUS,BORROWTIME,INSERTTIME) \  
 VALUES (%d,'%s','%s','%s','可借',0,%f)"** % (id, bookname, authorname, pressname, time.time()))  
 tkinter.messagebox.showinfo(**'message'**, **'Success!'**)  
 conn.commit()  
 conn.close()  
 **return***#app=insert()  
#app.mainloop()*

**log\_in.py**

**import** tkinter **as** tk  
**import** sqlite3  
**import** tkinter.messagebox  
**import** better  
  
  
**class** login(tk.Tk):  
 **def** \_\_init\_\_(self):  
 super().\_\_init\_\_()  
 self.geometry(**'250x100'**)  
 self.title(**'Log in'**)  
 self.accountlab = tk.Label(self, text=**"account:"**)  
 self.accountlab.place(x=1, y=1, anchor=**'nw'**)  
 self.passwordlab = tk.Label(self, text=**"password:"**)  
 self.passwordlab.place(x=1, y=25, anchor=**'nw'**)  
 self.account = tk.Entry(self)  
 self.account.place(x=67, y=1, anchor=**'nw'**)  
 self.password = tk.Entry(self, show=**'\*'**)  
 self.password.place(x=67, y=25, anchor=**'nw'**)  
 self.sumbit = tk.Button(self, text=**"login"**, command=self.check\_login)  
 self.sumbit.place(x=100, y=50, anchor=**'nw'**)  
  
 **def** check\_login(self):  
 account = self.account.get() **or ''** password = self.password.get() **or ''  
 if** account ==**'' or** password==**''**:  
 **return** conn = sqlite3.connect(**"login.db"**)  
 c = conn.cursor()  
 a = **None** b = **None** cursor = c.execute(**"SELECT ACCOUNT,PASSWORD FROM LOGIN WHERE ACCOUNT='%s'"** % account)  
 **for** row **in** cursor:  
 a = row[0]  
 b = row[1]  
 **if** a == account **and** b == password:  
 tkinter.messagebox.showinfo(**'message'**, **'Success!'**)  
 conn.close()  
 self.destroy()  
 app=better.window(**'%s'**%account)  
 app.mainloop()  
 **return  
 elif** a==**None**:  
 tkinter.messagebox.showinfo(**'message'**, **'No such account!'**)  
 **else**:  
 tkinter.messagebox.showinfo(**'message'**, **'Password error!'**)  
 conn.close()  
 **return**

**sign\_up.py**

**import** tkinter **as** tk  
**import** sqlite3  
**import** tkinter.messagebox  
**import** better  
  
**class** signup(tk.Tk):  
 **def** \_\_init\_\_(self):  
 super().\_\_init\_\_()  
 self.geometry(**'250x150'**)  
 self.title(**'Sign up'**)  
  
 self.accountlab = tk.Label(self, text=**"account:"**)  
 self.accountlab.place(x=1, y=1, anchor=**'nw'**)  
 self.passwordlab = tk.Label(self, text=**"password:"**)  
 self.passwordlab.place(x=1, y=25, anchor=**'nw'**)  
 self.confirm = tk.Label(self, text=**"confirm:"**)  
 self.confirm.place(x=1, y=50, anchor=**'nw'**)  
 self.account = tk.Entry(self)  
 self.account.place(x=67, y=1, anchor=**'nw'**)  
 self.password = tk.Entry(self, show=**'\*'**)  
 self.password.place(x=67, y=25, anchor=**'nw'**)  
 self.confirm = tk.Entry(self, show=**'\*'**)  
 self.confirm.place(x=67, y=50, anchor=**'nw'**)  
 self.sumbit = tk.Button(self, text=**"sumbit"**, command=self.check\_signup)  
 self.sumbit.place(x=100, y=75, anchor=**'nw'**)  
  
 **def** check\_signup(self):  
 conn = sqlite3.connect(**"login.db"**)  
 c = conn.cursor()  
 account = self.account.get() **or None** password = self.password.get() **or None** confirm = self.confirm.get() **or None** cursor = c.execute(**"SELECT ACCOUNT,PASSWORD FROM LOGIN WHERE ACCOUNT='%s'"** % account)  
 a = **None  
 for** row **in** cursor:  
 a = row[0]  
 **if** account == **None or** password == **None or** confirm == **None**:  
 conn.commit()  
 conn.close()  
 **return  
 elif** password != confirm:  
 tkinter.messagebox.showinfo(**'message'**, **'confirm failed!'**)  
 conn.commit()  
 conn.close()  
 **return  
 elif** a == **None**:  
 tkinter.messagebox.showinfo(**'message'**, **'Success!'**)  
 c.execute(**"INSERT INTO LOGIN (ACCOUNT,PASSWORD) \  
 VALUES ('%s','%s')"** % (account, password));  
 new = sqlite3.connect(**'./userinfo/%s.db'** % account)  
 N = new.cursor()  
 N.execute(**'''CREATE TABLE OWE  
 ( ID INT PRIMARY KEY NOT NULL,  
 BOOKNAME TEXT NOT NULL,  
 AUTHORNAME TEXT NOT NULL,  
 PRESSNAME TEXT NOT NULL,  
 RETURNTIME TEXT NOT NULL,  
 STATUS TEXT NOT NULL);'''**)  
 new.commit()  
 new.close()  
 conn.commit()  
 conn.close()  
 self.destroy()  
 app=better.window(**'%s'**%account)  
 app.mainloop()  
 **return  
 else**:  
 tkinter.messagebox.showinfo(**'message'**, **'Account has already been used!'**)  
 conn.commit()  
 conn.close()  
 **return  
 return**