上机实验综合作业--图书管理系统 实验报告

统计与金融系 PB18151833 李炳含

**概述：**

实现一个简单的图书管理系统。图书基本信息包括：图书ID，书名，ISBN，出版社，出版年月，作者。读者基本信息包括：读者ID，读者姓名，读者类型，读者联系电话。

**功能需求描述：**

（1）图书信息维护。导入书目、增加书目、删除书目、修改书目。图书购置入库（库存增加）、图书处理出库（库存减少）。

（2）读者信息维护。导入读者、增加读者、删除读者、修改读者信息。

（3）图书信息查询。根据书名、作者、标签（可选做）、出版社和ISBN查询图书。显示库存数量，根据图书库存数量显示是否可借。

（4）读者信息查询。根据读者姓名、联系方式查询显示读者信息。查询显示读者在借图书信息。查询显示读者借阅历史。

（5）图书借出。

（6）图书归还。

（7）系统关闭。保存数据之后关闭系统。这些数据在系统启动的时候自动读入系统。

**实现方法：**

1.创建两个类

class Book:  
 def \_\_init\_\_(self, book\_ID, book\_name, ISBN, pub, date, author, number):  
 self.book\_name = book\_name  
 self.book\_ID = book\_ID  
 self.ISBN = ISBN  
 self.pub = pub  
 self.date = date  
 self.author = author  
 self.number = number  
  
 def \_\_getitem\_\_(self, item):  
 if item in self.\_\_dict\_\_:  
 return self.\_\_dict\_\_[item]  
  
  
class Reader:  
 def \_\_init\_\_(self, reader\_ID, reader\_name, tel, status=0, lend=0):  
 self.reader\_name = reader\_name  
 self.reader\_ID = reader\_ID  
 self.tel = tel  
 self.status = status  
 self.lend = lend  
 self.lent = []  
  
 def \_\_getitem\_\_(self, item):  
 if item in self.\_\_dict\_\_:  
 return self.\_\_dict\_\_[item]

其中为使类与字典转换，添加getitem

2.主函数以及菜单

books = []  
readers = []  
books.append(Book('123456', 'abook', 'ISO9001', '某出版社', '2021.1', '鲁迅', 10))  
readers.append(Reader('123', 'cwk', '123456', 1, 0))  
'''  
先加入一本书和一个读者  
'''  
while True:  
 print('''  
 欢迎使用图书管理系统  
 1.图书信息维护  
 2.读者信息维护  
 3.图书信息查询  
 4.读者信息查询  
 5.图书借出  
 6.图书归还  
 0.退出  
 ''')  
 a = input('请输入选项:')  
 if a == '1':  
 manage\_book()  
 elif a == '2':  
 manage\_reader()  
 elif a == '3':  
 search\_book()  
 elif a == '4':  
 search\_reader()  
 elif a == '5':  
 lend\_book()  
 elif a == '6':  
 return\_book()  
 elif a == '0':  
 break

先创建两个空列表存放图书信息和读者信息。每有新的书或读者用append函数添加。为方便直接测试，直接人为添加例子：书“abook”和读者“cwk”

3.图书信息维护

（1）导入书目

def insert\_book():  
 print('由于没有数据，这部分未进行测试')  
 c = open(input('请输入导入数据').csv, 'r')  
 file = csv.reader(c)  
 for i in file:  
 books.append(i)

由于没有导入外部文件，这部分未进行测试，从菜单里不能直接实现功能。如果要实现可以将代码中的input改为所需文件

（2）增加书目

def add\_book():  
 name = input('书名:')  
 books.append(Book(input('ID:'), name, input('ISBN:'), input('出版社:'), input('日期:'), input('作者名:'), input('数量')))

（3）删除书目

def delete\_book():  
 name = input('请输入删除的书名:')  
 for i in books:  
 if i.book\_name == name:  
 books.remove(i)  
 break  
 else:  
 print('不存在这本书')

在book列表里根据书名找到要删除的书

（4）修改书目

def modify\_book():  
 name = input('待修改的书名:')  
 for i in books:  
 if i.book\_name == name:  
 key = input('修改内容为:ID/ISBN/出版社/出版日期/作者/数量')  
 if key == 'ID':  
 i.book\_ID = input('新的ID为:')  
 elif key == 'ISBN':  
 i.ISBN = input('新的ISBN为:')  
 elif key == '出版社':  
 i.pub = input('新的出版社为:')  
 elif key == '日期':  
 i.date = input('新的日期为:')  
 elif key == '作者名':  
 i.author = input('新的作者名为:')  
 elif key == '数量':  
 i.num = input('新的数量为:')  
 else:  
 print('错误')  
 break  
 else:  
 print('查无此书')

输入书名以及要修改的属性

（5）图书购置入库、出库

def add\_num():  
 name = input('入库书的书名:')  
 num = input('入库数量:')  
 for i in books:  
 if i.book\_name == name:  
 i.number += num  
 break  
  
  
def sub\_num():  
 name = input('出库书的书名:')  
 num = input('出库数量:')  
 for i in books:  
 if i.book\_name == name:  
 i.number -= num  
 if i.number < 0:  
 print('错误')  
 break

即修改数量属性，且减少后数量不能小于0

（6）分菜单

def manage\_book():  
 while True:  
 print("""  
 图书管理系统  
 1.导入书目  
 2.增加书目  
 3.删除书目  
 4.修改书目  
 5.库存增加  
 6.库存减少  
 0.退出  
 """)  
 choice = input('请选择：')  
 if choice == '1':  
 insert\_book()  
 elif choice == '2':  
 add\_book()  
 elif choice == '3':  
 delete\_book()  
 elif choice == '4':  
 modify\_book()  
 elif choice == '5':  
 add\_num()  
 elif choice == '6':  
 sub\_num()  
 elif choice == '0':  
 break

4.读者信息维护

def insert\_reader():  
 print('由于没有数据，这部分未进行测试')  
 c=open(input('请输入导入数据').csv, 'r')  
 file = csv.reader(c)  
 for i in file:  
 books.append(i)  
  
  
def add\_reader():  
 readers.append(Reader(input('读者名字'), input('读者ID:'), input('电话:')))  
  
  
def delete\_reader():  
 name = input('请输入删除的读者名:')  
 for i in readers:  
 if i.reader\_name == name:  
 readers.remove(i)  
 break  
  
  
def modify\_reader():  
 name = input('待修改的读者名:')  
 for i in readers:  
 if i.reader\_name == name:  
 key = input('修改内容为:ID/联系电话/是否为会员')  
 if key == 'ID':  
 i.reader\_ID = input('新的ID为:')  
 elif key == '联系电话':  
 i.tel = input('新的电话为:')  
 elif key == '是否为会员':  
 i.status = input('是会员请输入1，非会员请输入0')  
 else:  
 print('错误')  
 break  
 else:  
 print('查无此人')

def manage\_reader():  
 while True:  
 print("""  
 读者管理系统  
 1.导入读者  
 2.增加读者  
 3.删除读者  
 4.修改读者信息   
 0.退出  
 """)  
 choice = input('请选择：')  
 if choice == '1':  
 insert\_reader()  
 elif choice == '2':  
 add\_reader()  
 elif choice == '3':  
 delete\_reader()  
 elif choice == '4':  
 modify\_reader()  
 elif choice == '0':  
 break

原理思想与维护图书相同

5.图书、读者信息查询

def search\_book():  
 while True:  
 print("""  
 1.书名  
 2.作者  
 3.出版社  
 4.ISBN   
 0.退出  
 """)  
 choice = input('请输入查询的选项:')  
 key = input('请输入查询的内容:')  
 temp = []  
 if choice == '1':  
 for i in books:  
 if i.book\_name == key:  
 temp.append(i)  
 elif choice == '2':  
 for i in books:  
 if i.author == key:  
 temp.append(i)  
 elif choice == '3':  
 for i in books:  
 if i.pub == key:  
 temp.append(i)  
 elif choice == '4':  
 for i in books:  
 if i.ISBN == key:  
 temp.append(i)  
 elif choice == '0':  
 break  
 for j in temp:  
 for (k, v) in j.\_\_dict\_\_.items():  
 print(k, ':', v)  
  
  
def search\_reader():  
 while True:  
 print("""  
 1.读者名  
 2.读者id  
 3.联系方式  
 0.退出  
 """)  
 choice = input('请输入查询的选项:')  
 key = input('请输入查询的内容:')  
 temp = []  
 if choice == '1':  
 for i in readers:  
 if i.reader\_name == key:  
 temp.append(i)  
 elif choice == '2':  
 for i in readers:  
 if i.reader\_ID == key:  
 temp.append(i)  
 elif choice == '3':  
 for i in readers:  
 if i.tel == key:  
 temp.append(i)  
 elif choice == '0':  
 break  
 for i in temp:  
 for (k, v) in i.\_\_dict\_\_.items():  
 print(k, ':', v)

缺点是退出需要输入两次0

先建一个空列表temp，把所有查询到符合要求的选项存入，再以字典形式输出temp

6、图书借还

def lend\_book():  
 name1 = input('书名为:')  
 name2 = input('读者名为:')  
 for i in books:  
 if i.book\_name == name1:  
 break  
 for j in readers:  
 if j.reader\_name == name2:  
 break  
 if (j.status == 1) and (j.lend > 8):  
 print('借书达到上限')  
 elif (j.status == 0) and (j.lend > 4):  
 print('借书达到上限')  
 elif i.number == 0:  
 print('库存不够')  
 else:  
 j.lend += 1  
 j.lent.append(name1)  
 i.number = i.number - 1  
  
  
def return\_book():  
 name1 = input('书名为:')  
 name2 = input('读者名为:')  
 for i in books:  
 if i.book\_name == name1:  
 break  
 for j in readers:  
 if j.reader\_name == name2:  
 break  
 if name1 not in j.lent:  
 print('错误')  
 else:  
 j.lend -= 1  
 j.lent.remove(name1)  
 i.number += 1

先判断读者是否能继续借书，若借书上限满、库存不够则不能继续借。

借书后库存-1，读者借书记录储存在列表lent中

还书时判断读者是否持有这本书

**总结**

运用列表、字典、类等多种基础知识制作的图书管理系统，缺点是暂时无法在用户使用界面导入数据，其他功能可以正常使用。核心思想是把不容易遍历的类转换成容易遍历的字典，便于搜索和输出。