简易图书管理系统

1. 输入out之外命令继续操作；
2. 每种图书分别统计对应有多少本；

#encoding = utf-8

def statistics\_books(books):

""" 统计图书不同种类对应个数 """

dict\_book = {}

for i in books:

if i in dict\_book.keys():

dict\_book[i] += 1

else:

dict\_book[i] = 1

return dict\_book

def borrow(books,borrow\_book):

""" 借书方法 """

book\_name = input("请输入借书书名: ")

if book\_name in books:

user\_name = input("请输入借书人名称: ")

if user\_name in borrow\_book:

borrow\_book[user\_name].append(book\_name)

else:

borrow\_book[user\_name] = [book\_name]

books.remove(book\_name)

print("%s 已经借走 %s 书籍" % (user\_name,book\_name))

else:

print("书已经被借走")

def borrow\_search(borrow\_book):

"""查询已借书籍及借书人"""

name = input("请输入您要查询的姓名: ")

if name in borrow\_book.keys():

if len(borrow\_book[name]) == 1:

print("%s 借阅书籍为: %s" % (name,borrow\_book[name][0]))

elif len(borrow\_book[name]) == 0:

print("%s 未借阅书籍" % name)

else:

print("%s 借阅书籍为:" % name)

print("".join(["第%s本书: %s\n" % (i+1,borrow\_book[name][i]) for i in range(len(borrow\_book[name]))]))

else:

print("借书人不存在")

def send\_back\_book(books,borrow\_book):

""" 归还书籍 """

user\_name = input("输入归还人名称")

book\_name = input("输入借阅书籍名称")

books.append(book\_name)

if user\_name not in borrow\_book.keys() or book\_name not in borrow\_book[user\_name]:

return "输入有误"

borrow\_book[user\_name].remove(book\_name)

return "%s 还书 %s 成功" % (user\_name,book\_name)

def action(books,borrow\_book):

""" 主函数 """

cmd = input("""请输入命令:

输入out 退出系统;

输入search 查询书籍库存

输入 borrow 借书操作

输入 borrow\_search 查询借书人已借书籍

输入 return\_book 归还书籍: """)

while cmd != "out":

if cmd == "search":

a = statistics\_books(books)

print(a)

elif cmd == "borrow":

borrow(books,borrow\_book)

elif cmd == "borrow\_search":

borrow\_search(borrow\_book)

elif cmd == "return\_book":

print(send\_back\_book(books,borrow\_book))

else:

print("您输入命令有误，请重新输入")

cmd = input("""请输入命令:

输入out 退出系统;

输入search 查询书籍库存

输入 borrow 借书操作

输入 borrow\_search 查询借书人已借书籍

输入 return\_book 还书操作: """)

if \_\_name\_\_ == "\_\_main\_\_":

#主函数入口

books = ["平凡的世界","时间管理","墨菲定律","人性的弱点","平凡的世界"]

borrow\_book = {"jiahao":["平凡的世界","平凡的世界"],"fanyu":["墨菲定律"],"xiaoma":[]}

action(books,borrow\_book)

#encoding = utf-8

def statistics\_books(books):

""" 统计图书不同种类对应个数 """

dict\_book = {}

for i in books:

if i in dict\_book.keys():

dict\_book[i] += 1

else:

dict\_book[i] = 1

return dict\_book

def borrow(books,borrow\_book):

""" 借书方法 """

book\_name = input("请输入借书书名: ")

if book\_name in books:

user\_name = input("请输入借书人名称: ")

if user\_name in borrow\_book:

borrow\_book[user\_name].append(book\_name)

else:

borrow\_book[user\_name] = [book\_name]

books.remove(book\_name)

print("%s 已经借走 %s 书籍" % (user\_name,book\_name))

else:

print("书已经被借走")

def borrow\_search(borrow\_book):

"""查询已借书籍及借书人"""

name = input("请输入您要查询的姓名: ")

if name in borrow\_book.keys():

print(" %s借走了 %s 书籍"% (name,borrow\_book[name]))

else:

print("借书人不存在")

def action(books,borrow\_book):

""" 主函数 """

cmd = input("""请输入命令:

输入out 退出系统;

输入search 查询书籍库存

输入 borrow 借书操作

输入 borrow\_search 查询借书人已借书籍: """)

while cmd != "out":

if cmd == "search":

a = statistics\_books(books)

print(a)

elif cmd == "borrow":

borrow(books,borrow\_book)

elif cmd == "borrow\_search":

borrow\_search(borrow\_book)

else:

print("您输入命令有误，请重新输入")

cmd = input("""请输入命令:

输入out 退出系统;

输入search 查询书籍库存

输入 borrow 借书操作

输入 borrow\_search 查询借书人已借书籍: """)

if \_\_name\_\_ == "\_\_main\_\_":

#主函数入口

books = ["平凡的世界","时间管理","墨菲定律","人性的弱点","平凡的世界"]

borrow\_book = {}

action(books,borrow\_book)

字典

新增

A = {key :value}

Dict(key = value)

Dict( [ (key,value), (key,value) ] )

{}.fromkeys(可迭代数据, value) 把可迭代数据遍历一遍用每次遍历结果作为key生成 字典一个元素 key:value

a.Setdefault(key,value) key存在情况返回原字典key对应value，key不存在时返回当前参数value

删除

Pop（key,value）:key存在情况 删除key,value，返回原字典key对应value，不存在，返回当前参数value

Popitem()

Del dicta[key]

修改

Update(key,value)

A[key] = value

查找

Keys() key集合

Values() value集合

Items() key,value集合

eval("{"+"".join(["%s:%s," % (i,i+1) for i in range(1,9,2)])[:-1]+"}")

dict([(i,i+1) for i in range(1,9)])

学生管理系统

学生信息录入系统

Student\_info={int(id):{sex:value,name:value,born:value,admission\_time:value,tel:value,address:value}}

时间戳做ID

Born:” %s年%s月%s日”

admission\_time: ” %s年%s月%s日”

<tel:str>

address:str

生成一个id

Student\_info = ={int(id):{sex:value,name:value,born:value,admission\_time:value,tel:value,address:value}}

#encoding = utf-8

import time

def add\_student\_info(student\_info\_all):

id = time.time()

student\_info = {id:{"name":"","sex":"","born":"",\

"admission\_time":"","tel":"","address":""}}

for i in range(3):

name = input("请输入您的姓名: ")

if name == "":

print("输入姓名为空，请重新输入")

else:

student\_info[id].update({"name":name})

break

else:

print("您名称错误超过三次，请重新开始录入")

return

for i in range(3):

sex = input("请输入您的性别: ")

if sex == "":

print("输入性别为空，请重新输入")

else:

student\_info[id].update({"sex":sex})

break

else:

print("您性别错误超过三次，请重新开始录入")

return

for i in range(3):

tel = input("请输入您的电话: ")

if len(tel) != 11 or not tel.isdigit():

print("输入电话 %s 为格式错误，请重新输入" % tel)

else:

student\_info[id].update({"tel":tel})

break

else:

print("您电话错误超过三次，请重新开始录入")

return

for i in range(3):

born = input("请输入出生日期,格式为 年-月-日: ")

born\_tuple = tuple(born.split("-"))

if len(born\_tuple) == 3:

student\_info[id].update({"born":"%s年%s月%s日" % born\_tuple})

break

else:

print("您输入内容 %s 格式错误,正确格式为 年-月-日" % born)

else:

print("您出生日期错误超过三次，请重新开始录入")

return

for i in range(3):

admission\_time = input("请输入入校日期,格式为 年-月-日: ")

admission\_time\_tuple = tuple(admission\_time.split("-"))

if len(admission\_time\_tuple) == 3:

student\_info[id].update({"admission\_time":"%s年%s月%s日" % admission\_time\_tuple})

break

else:

print("您输入内容 %s 格式错误,正确格式为 年-月-日" % admission\_time)

else:

print("您出生日期错误超过三次，请重新开始录入")

return

address = input("请输入家庭住址: ")

student\_info[id].update({"address":address})

student\_info\_all.update(student\_info)

def action():

student\_info\_all = {}

cmd = input("请输入命令操作: ")

while cmd != "out":

if cmd == "add":

add\_student\_info(student\_info\_all)

print(student\_info\_all)

cmd = input("请输入命令操作: ")

if \_\_name\_\_ == "\_\_main\_\_":

#主函数入口

action()