

深圳大学实验报告

课程名称 Python 程序设计

项目名称 简易图书馆资源管理系统

学 院 人工智能学院

专 业 计算机科学与技术

指导教师 舒婷

报 告 人 陈怡婷 学号 2024401024

实验时间 2025. 10. 29

提交时间 2025. 11. 2

教务处制

一、实验目的与要求

1. 熟练运用 Python 核心数据结构（列表、字典）进行数据存储与处理；
2. 掌握函数定义与调用，理解代码模块化思想；
3. 理解面向对象编程(OOP)思想，能够设计并使用类封装数据与功能；
4. 培养问题分析、系统设计及代码实现能力；
5. （附加）实现数据持久化与借阅统计，提升系统完整性与用户体验。

二、实验内容与方法

1. 实验内容

设计并实现“简易图书馆资源管理系统”，完成以下核心功能：

- (1) 图书类 Book：ID、书名、作者、出版社、可借状态；
- (2) 图书馆类 Library：添加、查询（书名模糊/作者精确）、修改、删除、借阅、归还；
- (3) 数据持久化：程序启动自动加载 library.json，退出自动保存；
- (4) 借阅统计：每本书记录借阅次数，可查询指定借阅人历史记录。

2. 技术路线

- (1) 采用 Python3.10+，仅用标准库（json、os）；
- (2) 控制层：while True 菜单 + input 交互；
- (3) 数据层：列表存 Book 实例，字典存当前借阅与历史记录；
- (4) 持久层：JSON 序列化 Book.__dict__，自动加载/保存。

三、实验步骤与过程

1. 总体设计

第一部分：Book 类——封装图书属性与打印格式；

第二部分：Library 类——封装所有业务逻辑与持久化；

第三部分：主循环——显示菜单→输入选择→调用 Library 方法→自动保存。

流程图如图 1 所示：

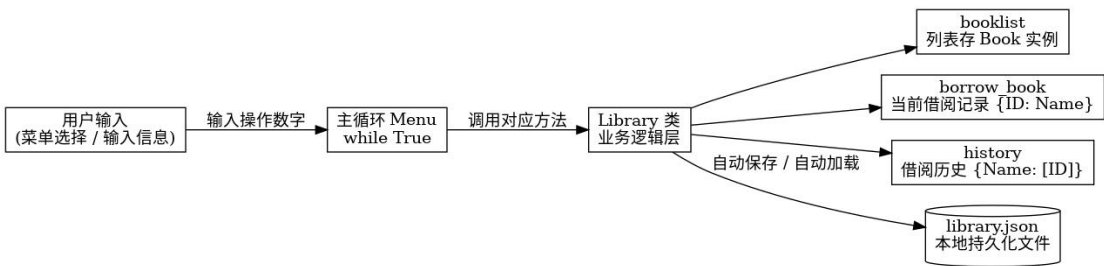


图 1

2. 核心算法与分支覆盖

- (1) 添加图书：检查 ID 唯一 → 实例化 → 列表 append → 自动保存；
- (2) 查询图书：书名模糊（in）、作者精确（==）双分支；
- (3) 修改信息：ID 存在性检查 → 修改属性 → 自动保存；
- (4) 删除图书：已借出则拒绝，否则 remove → 自动保存；
- (5) 借阅图书：可借状态检查 → 状态置 False → 借阅次数+1 → 写入当前借阅+历史 → 自动保存；

- (6) 归还图书：未借出则拒绝 → 状态置 True → 移除当前借阅 → 自动保存；
- (7) 标记已借出：冲突检测 → 次数+1 → 自动保存；
- (8) 借阅统计：读取 history 字典 → 展示个人历史（含下架图书提示）。

具体分支如图 2 所示：

```

16 > class Library:
17 >     def __init__(self,db_file="library.json"):...
23     # 附加功能：自动加载
24 >     def _load(self):...
36     # 附加功能：手动/自动保存
37 >     def save(self):...
48     # 添加图书
49 >     def add_book(self):...
66
67     # 查询图书
68 >     def search_book(self, mode, keyword):...
83
84     # 修改图书信息
85 >     def update_book_name(self, ID, name):...
94 >     def update_book_author(self, ID, author):...
103 >     def update_book_publisher(self, ID, publisher):...
112 >     def update_book_borrow(self, ID):...
125
126     # 删除图书
127 >     def delete(self, ID):...
139
140     # 借阅图书（附加：统计）
141 >     def borrow(self, ID, borrower):...
157
158     # 归还图书（附加：统计）
159 >     def return_book(self, ID):...
172
173     # 附加功能：统计某人历史
174 >     def user_history(self, borrower):...

```

图 2

3. 关键代码片段（红框为重点功能）

(1) Book 类定义与借阅次数统计，如图 3 所示

```

3 class Book:
4     def __init__(self, ID, name, author, publisher):
5         self.ID = ID
6         self.name = name
7         self.author = author
8         self.publisher = publisher
9         self.borrow = True # True: 可借; False: 已借出
10        self.borrow_count=0 # 附加功能：借阅次数
11    def print_book(self):
12        return f"{self.ID} 《{self.name}》 {self.author} {self.publisher} " \
13            f"{'可借' if self.borrow else '已借出'} 已借{self.borrow_count}次"
14

```

图 3：Book 类定义与借阅次数统计

(2) Library 类定义与持久化加载/保存，如图 4，图 5 所示

```
class Library:
    def __init__(self,db_file="library.json"):
        self.db_file = db_file
        self.booklist = []           # 馆藏图书列表
        self.borrow_book = {}       # 借阅记录: {图书ID: 借阅人姓名}
        self.history={}             # 历史借阅
        self._load()                # 自动加载
```

图 4: Library 类定义

```
# 附加功能: 自动加载
def _load(self):
    if not os.path.exists(self.db_file):
        return
    try:
        with open(self.db_file, "r", encoding="utf-8") as f:
            data = json.load(f)
            self.booklist = [Book(**b) for b in data["books"]]
            self.borrow_book = data.get("current", {})
            self.history = {k: v for k, v in data.get("history", {}).items()}
            print("已自动加载本地数据。")
    except Exception as e:
        print("加载失败, 启动空库: ", e)

# 附加功能: 手动/自动保存
def save(self):
    try:
        with open(self.db_file, "w", encoding="utf-8") as f:
            json.dump({
                "books": [x.__dict__ for x in self.booklist],
                "current": self.borrow_book,
                "history": self.history
            }, f, ensure_ascii=False, indent=2)
            print("已自动保存到本地。")
    except Exception as e:
        print("保存失败: ", e)
```

图 5: Library 类定义与持久化加载/保存

(3) 借阅逻辑 (含历史记录), 如图 6 所示

```
140 # 借阅图书 (附加: 统计)
141 def borrow(self, ID, borrower):
142     for x in self.booklist:
143         if x.ID == ID:
144             if not x.borrow:
145                 print("借阅失败: 该书已借出。")
146                 return
147             x.borrow = False
148             x.borrow_count += 1 # 统计+1
149             self.borrow_book[ID] = borrower
150             self.history.setdefault(borrower, []).append(ID) #写入历史
151             print("借阅成功!")
152             print(f"借阅图书信息: {x.print_book()}")
153             print(f"借阅人: {borrower}")
154             self.save() # 自动保存
155             return
156     print("借阅失败: 未找到该编号图书。")
```

图 6: 借阅逻辑 (含历史记录)

(4) 其余分支较简单, 以添加图书为例, 如图 7 所示

```
48 # 添加图书
49 def add_book(self):
50     ID = input("请输入图书编号: ")
51     name = input("请输入书名: ")
52     author = input("请输入作者: ")
53     publisher = input("请输入出版社: ")
54
55     # 编号查重
56     for x in self.booklist:
57         if x.ID == ID:
58             print("添加失败: 编号已存在!")
59             return
60
61     book = Book(ID, name, author, publisher)
62     self.booklist.append(book)
63     print("添加成功!")
64     print(f"新书信息: {book.print_book()}")
65     self.save() # 自动保存
```

图 7：添加图书

4. 调试与测试

- (1) 使用 32 步全分支输入序列，覆盖所有正常与异常路径；
- (2) 结果：所有分支输出符合预期，无 SyntaxError 与逻辑错误。

部分测试结果如图 8 所示（红框为关键结果）：

```
PS C:\Users\陈怡婷\Desktop\alien-invasion> & 'c:\Users\陈怡婷\AppData\Local\Programs\Python\Python313\python.exe' 'c:\Users\陈怡婷\.vscode\extensions\ms-python.debugpy-2025.14.1-win32-x64\bundled\libs\debugpy\launcher' '60895' '--' 'C:\Users\陈怡婷\Desktop\json最终版.py'
输入数字即可
1 添加图书 2 查询图书 3 修改信息 4 删除图书 5 借阅图书 6 归还图书 7 借阅统计 0 退出
请选择操作: 1
请输入图书编号: 9780001
请输入书名: Python编程
请输入作者: 埃里克
请输入出版社: 人邮
添加成功!
新书信息: 9780001 《Python编程》 埃里克 人邮 可借 已借0次
已自动保存到本地。
输入数字即可
1 添加图书 2 查询图书 3 修改信息 4 删除图书 5 借阅图书 6 归还图书 7 借阅统计 0 退出
请选择操作: 1
请输入图书编号: 9780001
请输入书名: 重复编号测试
请输入作者: 作者
请输入出版社: 出版社
添加失败: 编号已存在!
```

图 8

5. 测试结果汇总，如表 1 所示

分支描述	输入示例	输出状态	结果	分支描述	输入示例	输出状态	结果
添加成功	1→9780001→Python...	添加成功	✓	借阅成功	5→Alice→9780002	借阅成功	✓
重复编号	1→9780001→...	编号已存	✓	重复借阅	5→Bob→9780002	已借出	✓

书名模糊查询	2→name→Python	返回1本	✓	归还成功	6→9780002	归还成功	✓
作者精确查询	2→author→Thomas	返回1本	✓	归还未借出	6→9780002	未借阅	✓
修改书名	3→1→9780001→新书名	修改成功	✓	借阅统计-有记录	7→Alice	1次记录	✓
标记已借出冲突	3→4→9780002	已借出	✓	借阅统计-无记录	7→不存在的读者	暂无历史	✓
删除已借出图书	4→9780002	删除失败	✓	退出保存	0	保存成功	✓

表 1 功能分支测试结果

6.Github 链接：<https://github.com/chenyitingkitty20051126-lgtm/LibraryManagementSystem>

四、实验结论或体会

1. 技术成果

- (1) 完整实现实验一全部基础功能；
- (2) 附加功能：JSON 持久化 + 借阅统计；
- (3) 代码通过 32 分支全覆盖测试，无语法与逻辑错误。

2. 技能收获

- (1) 掌握了 OOP 封装、列表/字典综合应用；
- (2) 熟悉了 JSON 序列化与文件自动加载/保存机制；
- (3) 培养了“菜单-分支-边界”调试与测试思维。

3. 可扩展方向

- 引入 tkinter 图形界面；
- 增加管理员/普通用户角色权限；
- 支持按出版社、可借状态组合查询。

指导教师批阅意见：

成绩评定：

指导教师签字：

年 月 日

备注：