

|  |
| --- |
| 第二小组 |
| 技术报告 |
|  |

|  |  |
| --- | --- |
| **项目名称** | **图书馆管理系统** |
| **组长** | **符梦鑫** |
| **小组成员** | **程旭辉 王义钧 王俊蕊 李泊林 吴鑫磊** |

**中国海洋大学**

目录

[1. 编写目的 3](#_Toc57641604)

[2. 函数架构及存储结构 3](#_Toc57641605)

[3. 技术介绍 4](#_Toc57641606)

[4. 函数与接口说明 5](#_Toc57641607)

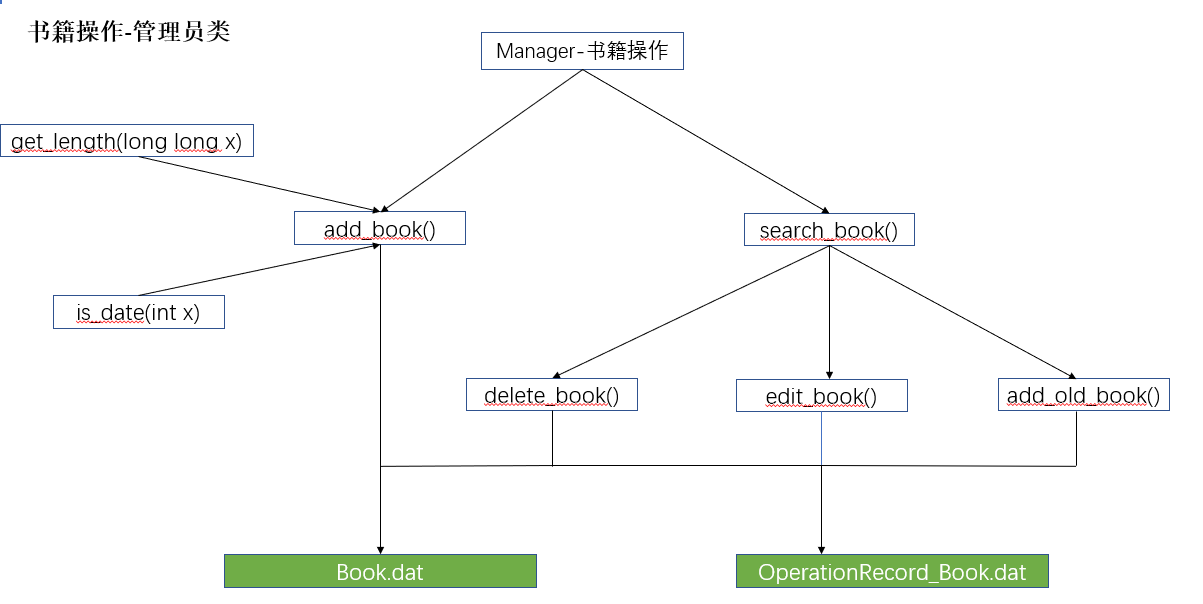
**1.编写目的**

该文档从图书馆管理系统的数据架构出发，对函数功能，函数接口，函数参数等进行了详细阐述，同时介绍了每个类及其与其他类的相关性，并且对各个类的文件存储结构进行了描述，目的在于让使用者了解该图书管理系统的工作原理、系统架构及其实现方法。

**2.函数结构及存储结构**

本系统主要分为三层：数据层、服务层以及应用层。数据层存储数据文件；服务层提供对数据层的相关函数操作，如用户和书籍的增删改查；应用层根据服务层提供的函数接口，实现良好的图书管理服务。

主要函数关系以及数据结构如下：



图示

描述已自动生成

图示

描述已自动生成

数据结构采取顺序存储，每行存储一个类的对象的数据，每个类存在一个文件中，使用到的文件有：

Book.dat

FineRecord.dat

OperationRecord\_Book.dat

OperationRecord\_User.dat

Recording.dat

User.dat

**3.技术介绍**

本图书馆管理系统采用数据层、服务层和应用层的架构方式，将三者分开，由数据层至应用层进行开发，形成一个逻辑完备的系统架构。

对于函数接口及功能开发，我们遵循了以上架构：Manager类、User类提供了对书籍、用户等位于数据层的数据的增删改查，Main函数根据上述两类提供的接口实现了完备的系统功能。数据管理方面，我们采用顺序存储方式，同时优化了表的结构，使用QuickList，加快服务层和应用层的速度，优化系统性能。

**4.函数与接口说明**

以下是系统中主要的类和它们提供的函数接口：

|  |  |  |
| --- | --- | --- |
| Manager类 | Manager类主要提供了书籍、用户的数据管理 | |
| 接口 | 实现功能 | 对接函数及参数意义 |
| Manager(char newName[20], char newPassword[20]) | 构造Manager对象 | 参数意义依次为：管理员名称，管理员密码 |
| void show\_menu() | 展示管理员界面 | - |
| User register\_user() | 注册用户 | search\_user()  check\_user(User& user) |
| void add\_user() | 添加用户 | search\_user()  check\_user(User& user) |
| void delete\_user() | 删除用户 | search\_user()  check\_user(User& user) |
| void search\_user() | 搜索用户 | save\_password(int id, char password[])  edit\_user\_password()  check\_user(User& user)  delete\_user()  register\_user()  add\_user() |
| bool check\_user(User& user) | 查询用户是否注册 | save\_password(int id, char password[])  search\_user()  delete\_user()  register\_user()  add\_user()  参数意义：用户 |
| void edit\_user\_password() | 修改用户密码 | save\_password(int id, char password[])  search\_user() |
| void show\_record() | 展示借阅记录 | - |
| bool save\_password(int id, char password[]) | 保存用户密码 | search\_user()  check\_user(User& user)  edit\_user\_password()  参数意义：新用户密码 |
| void exit() | 退出管理员界面 | - |
| void show\_operation\_record\_book() | 展示管理员对图书操作记录 | - |
| void show\_operation\_record\_user() | 展示管理员对用户操作记录 | - |
| void search\_black\_list() | 查询黑名单用户 | remove\_black\_list()  add\_black\_list() |
| void add\_black\_list() | 将用户加入黑名单 | search\_black\_list() |
| void remove\_black\_list() | 将用户移除黑名单 | search\_black\_list() |
| void add\_old\_book() | 增加已有书籍 | search\_book() |
| void deal\_fine\_record() | 处理用户欠款 | - |
| void add\_book() | 添加书籍 | - |
| void delete\_book() | 删除书籍 | search\_book() |
| void edit\_book() | 编辑书籍 | search\_book() |
| void search\_book() | 查找书籍 | edit\_book()  delete\_book()  add\_book()  add\_old\_book() |

|  |  |  |
| --- | --- | --- |
| Library类 | Library类提供了对界面操作与跳转的函数接口 | |
| 接口 | 实现功能 | 对接函数及参数意义 |
| Library() | Library初始化 | 初始化manager信息 |
| void begin() | 显示图书管理系统的登录界面，分别跳转至用户注册页、用户登录页和管理员登录页 | sign\_up()  sign\_in()  manager\_sign\_in() |
| void sign\_up() | 用户注册跳转 | manager.register\_user() |
| void sign\_in() | 用户登录跳转 | manager.check\_user(\_user) |
| void manager\_sign\_in() | 管理员登录 | manager.show\_menu() |

|  |  |  |
| --- | --- | --- |
| User类 | User类提供了用户界面借还书、修改用户信息的函数接口 | |
| 接口 | 实现功能 | 对接函数及参数意义 |
| User(int new\_id, char new\_name[], char new\_gender[], int new\_grade, int new\_major, int new\_borrow\_num, int new\_history\_borrow\_num, int new\_state, int new\_status) | 初始化用户 | 用户ID、姓名、性别、年级、专业、借书数、历史借书数、状态、身份 |
| string get\_major  (int \_majorID) | 利用键值映射  读取院系名称 | Manager.add\_user()  Manager.register\_user()  参数为专业ID |
| void set\_major  (string major) | 利用键值映射  获取院系ID | Manager.add\_user()  Manager.register\_user()  参数为专业名 |
| void show\_user() | 展示用户基本信息 | 在User类均可被调用 |
| void show\_menu() | 展示用户界面菜单 | borrow\_book()  return\_book()  inquire\_history  inquire\_now()  change\_info()  renew()  declare() |
| User\* find\_user\_locate  (int id) | 定位用户在文件中的位置，获取用户信息 | 在User类均可被调用  参数为用户ID |
| void borrow\_book() | 借书 | find\_user\_locate  (int id) |
| void return\_book() | 还书 | find\_user\_locate  (int id) |
| void inquire\_history() | 获取历史借阅记录 | Record.show\_record() |
| void inquire\_now() | 获取当前借阅记录 | Record.show\_record() |
| void change\_info() | 更改用户信息 | save\_user\_info() |
| void renew() | 续借 | Record.show\_record() |
| void declare() | 书籍申报丢失 | 无 |
| bool get\_fine\_record\_id  (int search\_id) | 获取罚款记录中的用户ID | 参数为用户ID |
| void set\_black\_list() | 生成黑名单  在用户登录时调用 | set\_user\_state  (int state)  show\_menu() |
| void show\_delay\_book() | 在黑名单用户  查看逾期未归还的书籍 | Book.show\_book() |
| int calculate\_day  (char return\_tmp[256],  char tmp[256]) | 计算日期差  在罚款与黑名单中比较是否超出日期 | set\_black\_list()  declare()  参数为借书日期和还书日期 |
| int include\_chinese  (char\* str) | 查看字符串是否  包含中文 | change\_info()  参数为当前字符串 |

|  |  |  |
| --- | --- | --- |
| Record类 | Record类提供了对用户和书籍进行读取与存储的函数接口 | |
| 接口 | 实现功能 | 对接函数及参数意义 |
| Record(int new\_id\_book, char new\_name\_book[20], int new\_id\_user, char new\_name\_user[20], int new\_book\_state, int new\_record\_state) | 初始化Record | 书籍ID、名称、用户ID、用户名、书籍状态、记录状态 |
| void recording() | 封装记录文件的读写函数  将当前记录写入记录文件中 | 在Record类中均可被调用 |
| void set\_return\_tmp() | 设置记录  还书日期 | time\_t()  时间函数 |
| void show\_record() | 记录展示 | 在Record类中均可被调用 |

|  |  |  |
| --- | --- | --- |
| Book类 | Book类提供了对书籍操作的函数接口 | |
| 接口 | 实现功能 | 对接函数及参数意义 |
| Book() | 初始化Book | 无参构造函数Book在所有函数中均可被调用 |
| Book(int \_id, char\* \_name, char\* \_writer, int \_state, char\* \_press, int \_publication\_date, double \_price, char\* \_category, long long \_isbn, int \_renew\_time) | 初始化Book | 书籍ID、名称、作者、状态、出版社、出版日期、价格、分类、ISBN、续借次数 |
| void show\_book() | 展示书籍信息 | 在Book类中均可被调用 |
| void reset\_renew\_time() | 重新初始化  书籍续借次数 | User.return\_book() |
| void set\_renew\_time() | 设置书籍续借次数+1 | User.renew() |

|  |  |  |
| --- | --- | --- |
| Manager类 | Manager类提供了对用户和书籍所有操作的函数接口 | |
| 接口 | 实现功能 | 对接函数及参数意义 |
| Manager(char newName[20], char newPassword[20]) | 初始化Manager | 管理员的用户名、密码 |
| show\_menu() | 展示管理操作菜单 | 在Manager类中均可被调用 |
| User register\_user() | 用户注册 | sign\_up() |
| void add\_user() | 管理员新增用户 | User.get\_major  (int \_majorID) |
| void delete\_user() | 管理员删除用户 | FindUserLocate(int id) |
| void edit\_user\_password() | 管理员修改  用户密码 | FindUserLocate(int id)  save\_password  (int id, char password[]) |
| void search\_user() | 管理员搜索用户 | FindUserLocate(int id)  User.show\_user() |
| bool check\_user  (User& user) | 用户登录信息确认 | Library.sign\_in()  参数为登录用户 |
| bool save\_password() | 将修改后的密码写入用户文件 | 在Manager类中均可被调用 |
| User\* FindUserLocate(int id) | 找到用户在文件中的位置 | 在Manager类中均可被调用  参数为用户ID |
| int IncludeChinese  (char\* str) | 检测字符串中是否包含中文 | add\_user()  register\_user()  参数为当前字符串 |
| void search\_black\_list() | 管理员  搜索用户黑名单 | remove\_black\_list() |
| void remove\_black\_list() | 管理员  移除用户黑名单 | search\_black\_list() |
| bool check(int date) | 检查日期是否符合  实际规范 | add\_user()  register\_user()  add\_book()  参数为当前输入日期 |
| void add\_book() | 管理员添加书籍 | 在Manager类中均可被调用 |
| void search\_book() | 管理员搜索书籍 | Book.show\_book() |
| void delete\_book() | 管理员删除书籍 | 在Manager类中均可被调用 |
| void edit\_book() | 管理员修改书籍 | check\_user  (int date) |
| void show\_record() | 展示所有借阅记录 | Record.show\_record() |
| void show\_operation\_record\_book() | 展示所有管理员对书籍操作记录 | OR\_B.show\_record() |
| void show\_operation\_User\_book() | 展示所有管理员对用户操作记录 | OR\_U.show\_record() |
| void deal\_fine\_record() | 管理员  处理罚款信息 | 在Manager类中均可被调用 |
| void add\_old\_book() | 管理员  增加已存在书籍 | add\_book() |
| int get\_length  (long long x) | 获取字符串长度 | 在Manager类中均可被调用  参数为当前字符串 |

|  |  |  |
| --- | --- | --- |
| FineRecord类 | FineRecord类提供了对书籍罚款的函数接口 | |
| 接口 | 实现功能 | 对接函数及参数意义 |
| FR::FR(int \_id, char\* \_name, int \_category, double \_money) | 初始化FineRecord | 用户ID、姓名、罚款分类、罚款金额 |
| void show() | 展示罚款信息 | 在FineRecord类中均可被调用 |

|  |  |  |
| --- | --- | --- |
| OperationRecord\_Book类 | OperationRecord\_Book类提供了保留管理员对书籍所有操作的函数接口 | |
| 接口 | 实现功能 | 对接函数及参数意义 |
| OR\_B::OR\_B(int book\_id1, char\* book\_name1, int state1) | 初始化OperationRecord\_Book | 书籍ID、名称、状态 |
| int get\_book\_id() | 获取书籍ID | 在OR\_B类中均可被调用 |
| char\* get\_book\_name() | 获取书籍名称 | 在OR\_B类中均可被调用 |
| int get\_state() | 获取书籍名称 | 在OR\_B类中均可被调用 |
| char\* get\_time() | 获取操作时间 | 在OR\_B类中均可被调用 |
| char\* get\_manager\_name() | 获取操作者名称 | 在OR\_B类中均可被调用 |

|  |  |  |
| --- | --- | --- |
| OperationRecord\_User类 | OperationRecord\_User类提供了保留管理员对用户所有操作的函数接口 | |
| 接口 | 实现功能 | 对接函数及参数意义 |
| OR\_U::OR\_U(int \_user\_id, char \_name[], int \_grade, int \_major, int \_state, int \_user\_state, char \_manager\_name[]) | 初始化OperationRecord\_User | 用户ID、姓名、年级、专业、用户状态、状态、操作者姓名 |
| int get\_id() | 获取用户ID | 在OR\_U类中均可被调用 |
| char\* get\_name() | 获取用户名称 | 在OR\_U类中均可被调用 |
| int get\_grade() | 获取用户年级 | 在OR\_U类中均可被调用 |
| string get\_major() | 获取用户专业 | 在OR\_U类中均可被调用 |
| char\* get\_ori\_password() | 获取用户修改前密码 | 在OR\_U类中均可被调用 |
| char\* get\_now\_password() | 获取用户修改后密码 | 在OR\_U类中均可被调用 |
| void get\_ori\_password  (char passwd[20]) | 设置用户修改前密码 | 在OR\_U类中均可被调用  参数为修改前密码 |
| void get\_now\_password  (char passwd[20]) | 设置用户修改后密码 | 在OR\_U类中均可被调用  参数为修改后密码 |
| char\* get\_manager\_name() | 获取操作者名称 | 在OR\_U类中均可被调用 |
| char\* get\_time() | 获取操作时间 | 在OR\_U类中均可被调用 |
| int get\_state() | 获取记录状态 | 在OR\_U类中均可被调用 |