|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 综合体育场馆管理系统V1.0  项目设计报告   |  |  | | --- | --- | | 文档版本 | 1.1.0 | | 文档状态: | 编辑 | | 项目作者(排名不分前后): | 高德润 靳浩然 黄一宸 张硕 王冠祺 | | 负责人: | 张硕 | | 创建日期: | 2023年9月10日 | | 更新日期: | 2023年9月15日 | |

修订历史

|  |  |  |  |
| --- | --- | --- | --- |
| **日期** | **版本** | **修改者** | **描述** |
| 2023-9-12 | 1.0.0 | 靳浩然 | 报告框架与主体内容 |
| 2023-9-13 | 1.0.1 | 张硕 | 主体内容修缮 文字流重排版 |
| 2023-9-15 | 1.1.0 | 张硕 | 补充详细模块介绍 定稿 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**目录**

[1 简介 4](#_Toc145689914)

[1.1 编写目的 4](#_Toc145689915)

[1.2 使用对象 4](#_Toc145689916)

[1.3 整体业务功能 4](#_Toc145689917)

[2 总体设计概述 5](#_Toc145689918)

[2.1 业务逻辑设计 5](#_Toc145689919)

[2.2 系统总体设计 5](#_Toc145689920)

[2.3 界面设计 6](#_Toc145689921)

[2.4 协作、代码及接口定义规范 8](#_Toc145689922)

[2.5 数据方案及数据结构描述 9](#_Toc145689923)

[2.6 开发及运行环境 11](#_Toc145689924)

[3 模块详细设计 12](#_Toc145689925)

[3.1 模块功能概述 12](#_Toc145689926)

[3.2 模块详述 15](#_Toc145689927)

[3.2.1 主入口 15](#_Toc145689928)

[3.2.2 数据存取模块 15](#_Toc145689929)

[3.2.3 输入校验模块 18](#_Toc145689930)

[3.2.3 用户业务模块 20](#_Toc145689931)

[3.2.3 管理员业务模块 21](#_Toc145689932)

# 简介

本项目为北京邮电大学国际学院电信工程及管理专业2023年秋小学期“计算机实习”课程的团队实践项目。项目选题**“综合体育场馆管理系统”**，严格按照课程要求的程序设计规范与分工协作要求开展实践活动。项目分工均匀，在充分分析项目需求和成员已有基础的前提下，项目各功能模块与业务函数拆分科学合理；数据持久化方案可用性高；代码可复用性强、耦合度较低；整体代码风格及规范一致，注释丰富，可读性好，二次开发及维护难度较低；针对异常输入和其他异常情况的测试用例设计及测试开展较为完备，程序鲁棒性强。圆满完成了课程的项目要求。

## 编写目的

本文档作为结项时提交的设计报告，完整阐释了项目的设计思路及各项细节，包含软件界面设计、数据存储方案、功能模块拆分、需求实现情况及用户操作手册等章节，供课程结项时项目评审使用，也可作为用户使用时的参考手册，二次开发时开发人员/测试人员的参考资料。

## 使用对象

文档主要作课程设计项目结项评审时的提交材料。

## 整体业务功能

依照课程要求，项目面向综合体育场馆管理员与预约场地的一般顾客用户，同时提供了主管用户组作为管理员的上级权限。

主管可填写场馆名称及管理员的个人信息，添加新的场馆并为其分配管理员。

管理员为体育场馆的管理者，可向当前管理的场馆中添加场地，填写场地的名称、位置、运动类型、准入年龄、租金等基本信息，可以查看当前场馆中所有场地的预约情况，查看全场营收、最受欢迎运动等统计信息，以及编辑自身信息。

用户在填写自身姓名、年龄、性别、邮箱、手机号等基本信息后，注册成为顾客，在输入正确信息登入系统后，可以按照场馆名称、场地名称、运动类型等查询场地，可根据租金、预定量等查看场地排序，能根据是否有空余场地过滤查询结果；可以通过场地ID预定场地使用；可以查询自己过去预定的订单并取消预定；可以修改自己的手机号、邮箱、地址等个人信息。

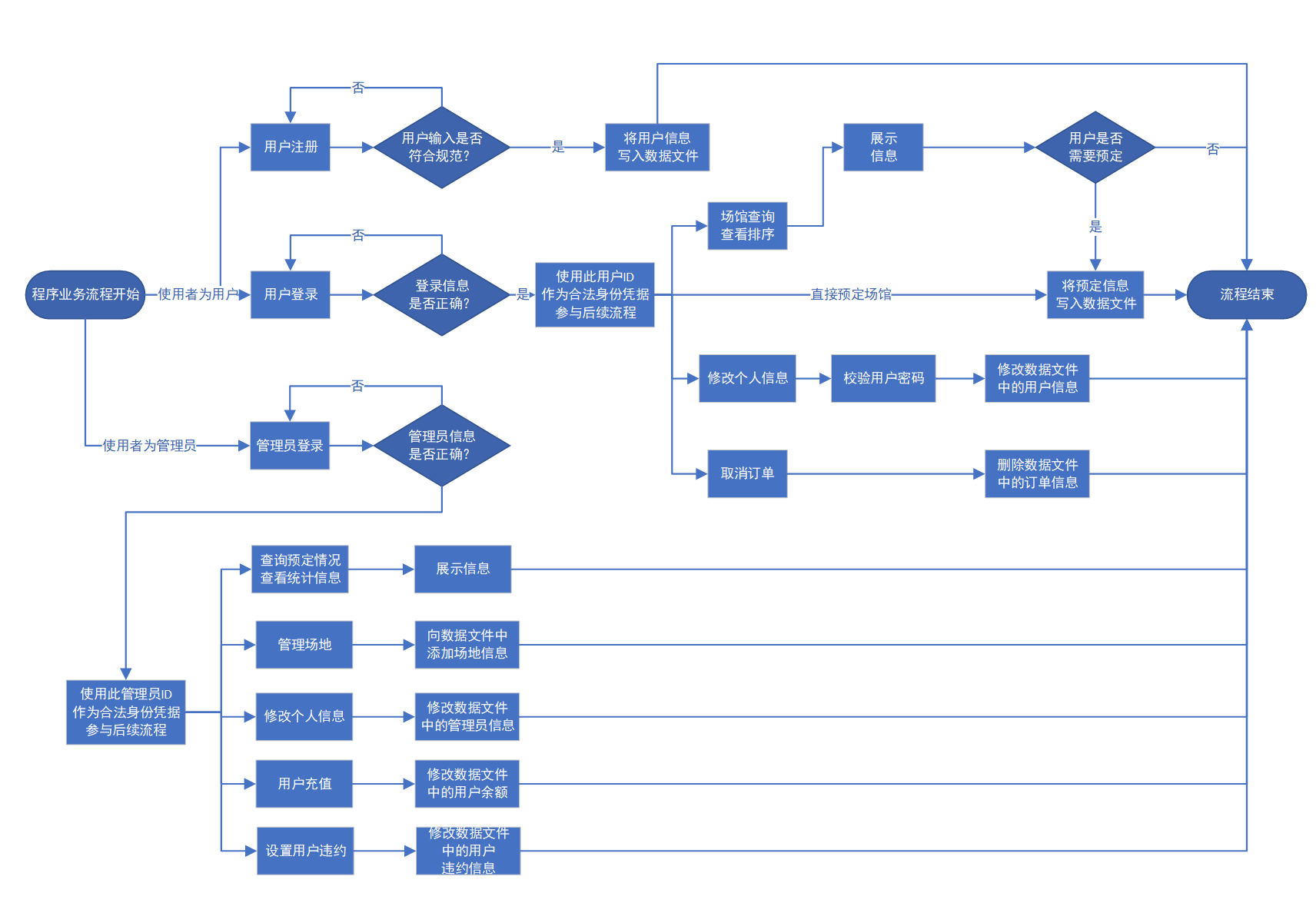
程序整体业务有完备的校验流程，会在核对用户信息合法后再进行正常业务流程，能够应对如输入不规范、预约日期冲突等常见问题，具有一定的抗击打能力，鲁棒性较强。

# 总体设计概述

## 业务逻辑设计

项目设计可满足一般大型体育场馆的管理需求，能够完成基本的账户控制及场馆查询与预定管理，对用户输入的信息有基本的校验。

面向使用者的整体业务逻辑设计大致如下图所示。

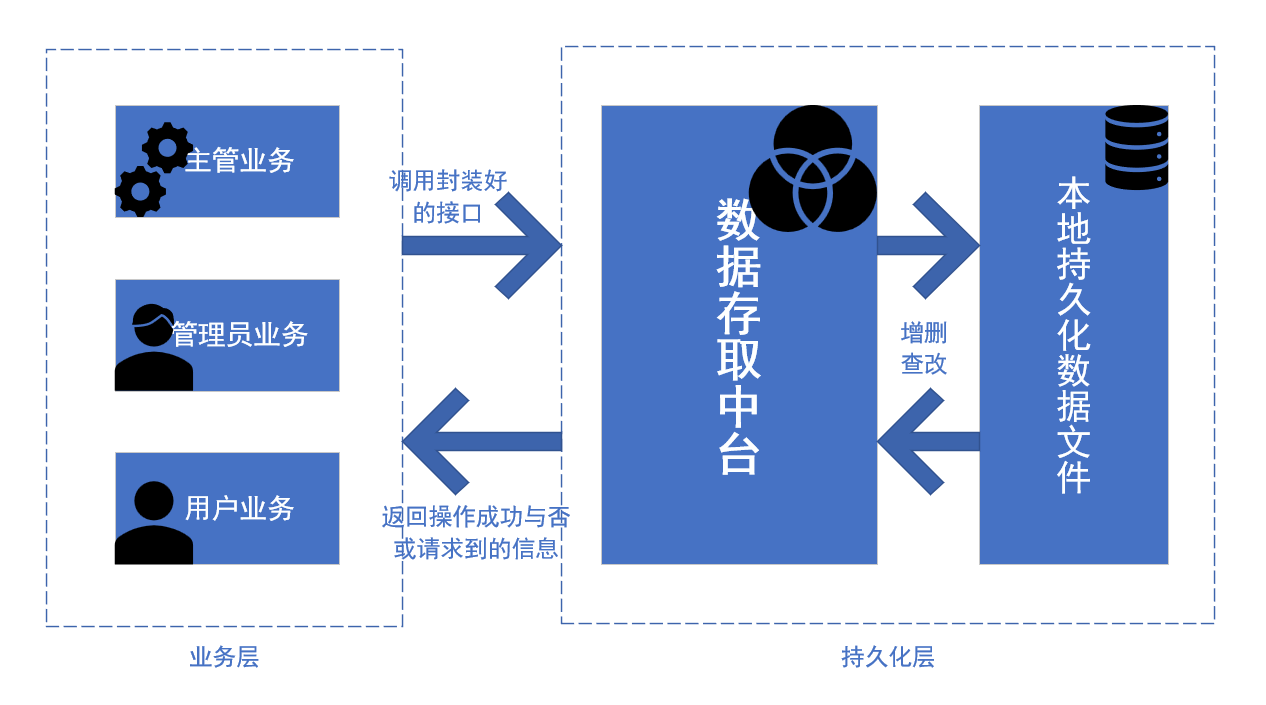


**图 2-1 业务逻辑图**

## 系统总体设计

软件完全使用C语言编写，未使用任何C++特性（面向对象、SDL等）实现功能。考虑到软件功能实现与信息存取高度相关，且需要的信息表类交错性强，软件采取界面与业务逻辑分离、数据操作与用户端流程分离的设计思路，将主管、场馆管理员和用户的业务逻辑代码与独立设计的数据存取中台（持久化层）接入，实现数据格式统一化。

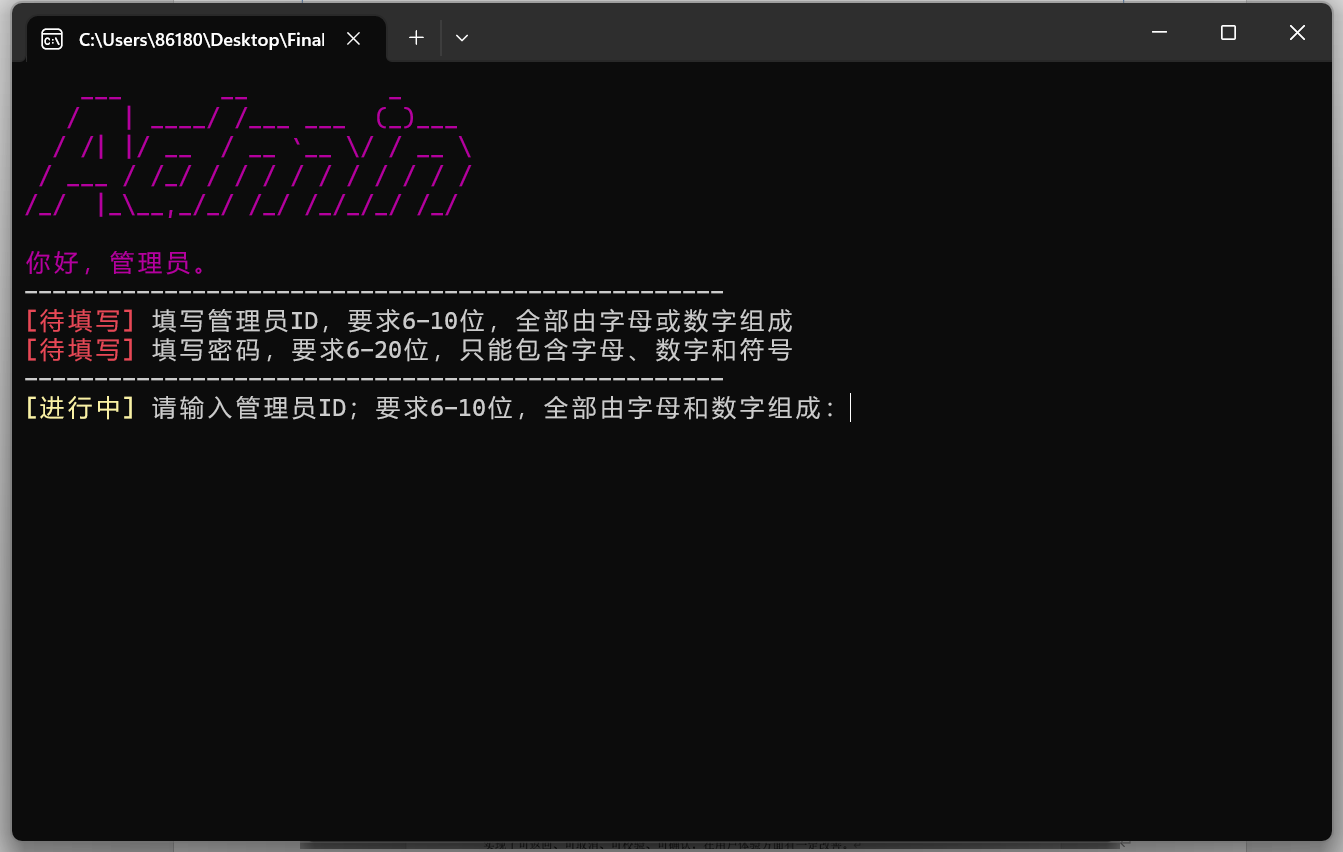
系统整体架构如图所示：



**图 2-2 系统架构图**

## 界面设计

用户界面完全使用终端命令行中的TUI（Terminal user interface，终端用户界面）方案，在每级用户菜单顶部通过ASCII字符画展示当前菜单层级，通过printf函数的 \033[m 控制符实现彩色打印，高亮显示关键信息和操作按键信息。在用户体验方面有一定改善。



**图 2-3 软件界面–管理员登录**



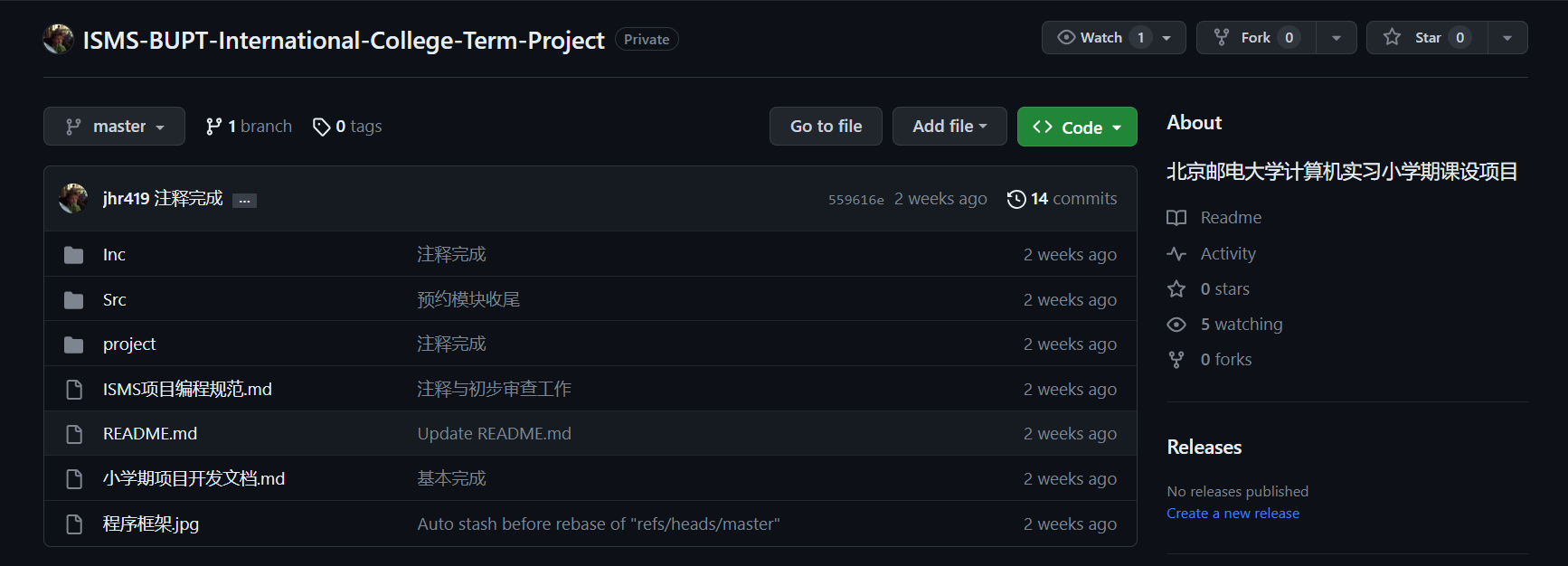
**图 2-4 软件界面–用户操作项列表**



**图 2-5 软件界面–用户预约场地**

## 协作、代码及接口定义规范

项目采用**需求分析->模块拆分->分工开发->单元测试->整合重构->整体测试**的思路开展分工协作。分工开发阶段，成员采用Git工具进行版本控制，待各模块单元开发完毕后，由一名成员进行代码的整合重构。



**图 2-6 团队协作仓库（私有）**

接口定义整体采用如下基本原则：

* 动词前缀化
* 接口参数命名

全体协作成员完全遵循以下规范进行开发，确保命名统一，最大程度维持代码文本的自解释性，降低后续维护成本：

**表2-1 结构体命名规范**

|  |  |  |
| --- | --- | --- |
| **场景** | **约束** | **示例** |
| 对象信息结构体 | 在结构体对象类型后加Info作后缀 | UserInfo |
| 链表节点结构体 | 在相应的信息结构体名后加Node作后缀 | UserInfoNode |

**表2-2 变量命名规范**

|  |  |  |
| --- | --- | --- |
| **场景** | **约束** | **示例** |
| 结构体成员变量 | 对象+成员属性名，假匈牙利式命名 | usrName |
| 函数形参 | 匈牙利式 | StrName |
| 函数返回值 | 使用布尔类型（宏定义整型） | TRUE |
| 状态变量 | 下划线命名法 | Valid\_name |

**表2-3 接口函数命名规范**

|  |  |  |
| --- | --- | --- |
| **场景** | **约束** | **示例** |
| 业务功能工具函数 | 动词+操作对象，假匈牙利式命名 | printUserInfo |
| 业务实现函数 | 动词+操作对象，假匈牙利式命名 | doRegister |
| 通用工具函数 | 动词前缀，下划线命名法 | print\_status |
| 校验类工具函数 | isValid…+校验判断对象 | IsValidUserName |

## 数据方案及数据结构描述

项目使用**结构体链表**实现数据存储，单独封装数据存取中台（DataToolkit.c），封装了一系列用于增删查改的工具函数。将链表直接存储为二进制文件，进行序列化存取。每个单独的数据表存储为一个独立的文件。

存取中台基于链表和IO流文件操作封装了一系列文件读写的功能函数，依照与数据库软件类似的设计思路，使得业务逻辑层能够便利地对数据进行**增删查改**，大大降低了数据操作相关的代码工作量。

项目的数据表设计信息如下所示：

**表2-4 用户信息数据表**

|  |  |  |  |
| --- | --- | --- | --- |
| **结构体类名** | UserInfo | **链表节点** | UserInfoNode |
| **成员变量名** | **数据类型** | **含义** | **额外说明** |
| usrID | char\* | 用户ID |  |
| usrName | char\* | 用户姓名 |  |
| usrGender | int | 用户性别 | 定义0表示男性，1表示女性 |
| usrAge | int | 用户年龄 |  |
| usrPhone | char\* | 用户电话 |  |
| usrPwd | char\* | 用户密码 |  |
| usrEmail | char\* | 用户邮箱 |  |
| usrMoney | double | 用户余额 |  |
| usrLocation | char\* | 用户地址 |  |
| usrBreakAppointmentTimes | int | 违约次数 |  |

**表2-5 管理员信息数据表**

|  |  |  |  |
| --- | --- | --- | --- |
| **结构体类名** | AdminInfo | **链表节点** | AdminInfoNode |
| **成员变量名** | **数据类型** | **含义** | **额外说明** |
| adminID | char\* | 管理员ID |  |
| adminName | char\* | 管理员姓名 |  |
| adminGender | int | 管理员性别 | 定义0表示男性，1表示女性 |
| adminPhone | char\* | 管理员手机号 |  |
| adminPwd | char\* | 管理员密码 |  |
| adminEmail | char\* | 管理员邮箱 |  |
| adminVenueID | char\* | 管理员所属场馆 |  |

**表2-6 场地信息数据表**

|  |  |  |  |
| --- | --- | --- | --- |
| **结构体类名** | SiteInfo | **链表节点** | SiteInfoNode |
| **成员变量名** | **数据类型** | **含义** | **额外说明** |
| siteID | char\* | 场地ID |  |
| siteName | char\* | 场地名称 |  |
| siteLocation | char\* | 场地地址 |  |
| siteVenueID | char\* | 所属场馆 |  |
| siteSportsType | char\* | 运动类型 |  |
| siteIntro | char\* | 场地简介 |  |
| siteRent | double | 租金 |  |
| siteAllowedAgeMin | int | 最小准入年龄 |  |
| siteAllowedAgeMax | int\* | 最大准入年龄 | 不得小于最小准入年龄 |
| siteUsedTimes | int | 总使用次数 |  |

**表2-7 预约信息数据表**

|  |  |  |  |
| --- | --- | --- | --- |
| **结构体类名** | ApmtInfo | **链表节点** | AppointmentInfoNode |
| **成员变量名** | **数据类型** | **含义** | **额外说明** |
| apmtUsrID | char\* | 发起预约的  用户ID |  |
| apmtSiteID | char\* | 用户预约的  场地ID |  |
| apmtOrderDated | char\* | 发起预约的  具体时间 | 格式yyyymmddhhmmss  (eg. 20230829105133) |
| apmtStartDateh | char\* | 预约开始的  具体时间 | 格式yyyymmddhh  (eg. 2023082909) |
| apmtDurationh | int | 预约持续时长 | 单位为小时 |
| apmtUsrGender | int | 发起预约的  用户性别 | 定义0表示男性，1表示女性 |
| isUsrPresent | BOOL(int) | 用户是否到场 |  |

程序会在main流程开始时调用宏定义 initAllNodes 初始化全部数据表的链表头节点，此后即可调用封装在文件DataToolkit.c中的方法便利地进行文件操作。预封装的数据操作方法支持对数据进行增删查改存取的基本操作，也提供了一些方法便于测试时查看表情况。

总四个表的实现类似。下表展示对数据表UserInfo用户信息的对外接口提供情况：

**表2-5 文件操作函数（以UserInfo表为例）**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **函数名** | **函数参数** | **函数返回** | **描述** | **备注** |
| InsertToUserInfo | UserInfoNode链表头节点、UserInfo用户信息结构体 | int，是否成功插入 | 向表中插入新用户 |  |
| deleteUserInfo | UserInfoNode链表头节点、char\*用户ID | int，是否成功删除 | 删除ID为某的用户信息 |  |
| queryUserInfo | UserInfoNode链表头节点、char\*用户ID | int，是否查找到用户 | 根据用户ID查询某用户的全部信息 |  |
| EditUserInfo | UserInfoNode链表头节点、UserInfo用户信息结构体 | int，是否成功编辑 | 编辑某用户的信息 |  |
| saveUserInfo | UserInfoNode链表头节点 | int，是否成功保存 | 将内存中的链表保存至文件中 |  |
| loadUserInfo | UserInfoNode链表头节点 | int，是否成功加载 | 将文件中的链表读取至内存中 |  |
| printUserInfo | UserInfoNode链表头节点 | void | 打印全部用户信息 | 仅开发测试时使用 |
| isUserExist | UserInfoNode链表头节点、char\*用户ID | int，用户是否存在 | 查询某ID是否存在 |  |

## 开发及运行环境

**开发环境：**

编译器：MingW32/GCC  
IDE：VSCode l.71.0，Visual Studio 2022，小熊猫C++   
字符编码：UTF-8编码

**运行环境:**

Windows 10及以上系统

**硬件环境:**

CPU：intel core i7-10750H  
GPU：NVIDIA GeForce RTX 3070 laptop GPU（未使用）  
主硬盘：Samsung SSD 980 1TB  
内存：Samsung 2\*8GB

# 模块详细设计

## 模块功能概述

项目将所有需实现的功能拆分为五个文件。

* **Main.c**文件为系统业务流程的主文件，包含了数据信息初始化、系统主入口、存储贯穿全业务流程的核心全局变量等。
* **StructDef.h**文件定义了数据存取需要的全部结构体。
* **DataToolkit.c**文件为系统的数据存取中台，其中封装了大量对业务数据进行增删查改和持久化文件存储、读取的函数方法。
* **UserComponent.c**文件定义了对用户输入进行校验的函数（校验模块），以及用户相关流程（用户注册、用户登录、用户查询和预定场馆等）的全部业务用代码，均封装为函数。
* **AdminComponent.c**文件定义了管理员相关流程（管理员注册、管理员登录、查看场地统计信息和新建场地等）的全部业务用代码，均封装为函数。

系统模块功能大致如下表所示：

**表3-1 模块功能概述**

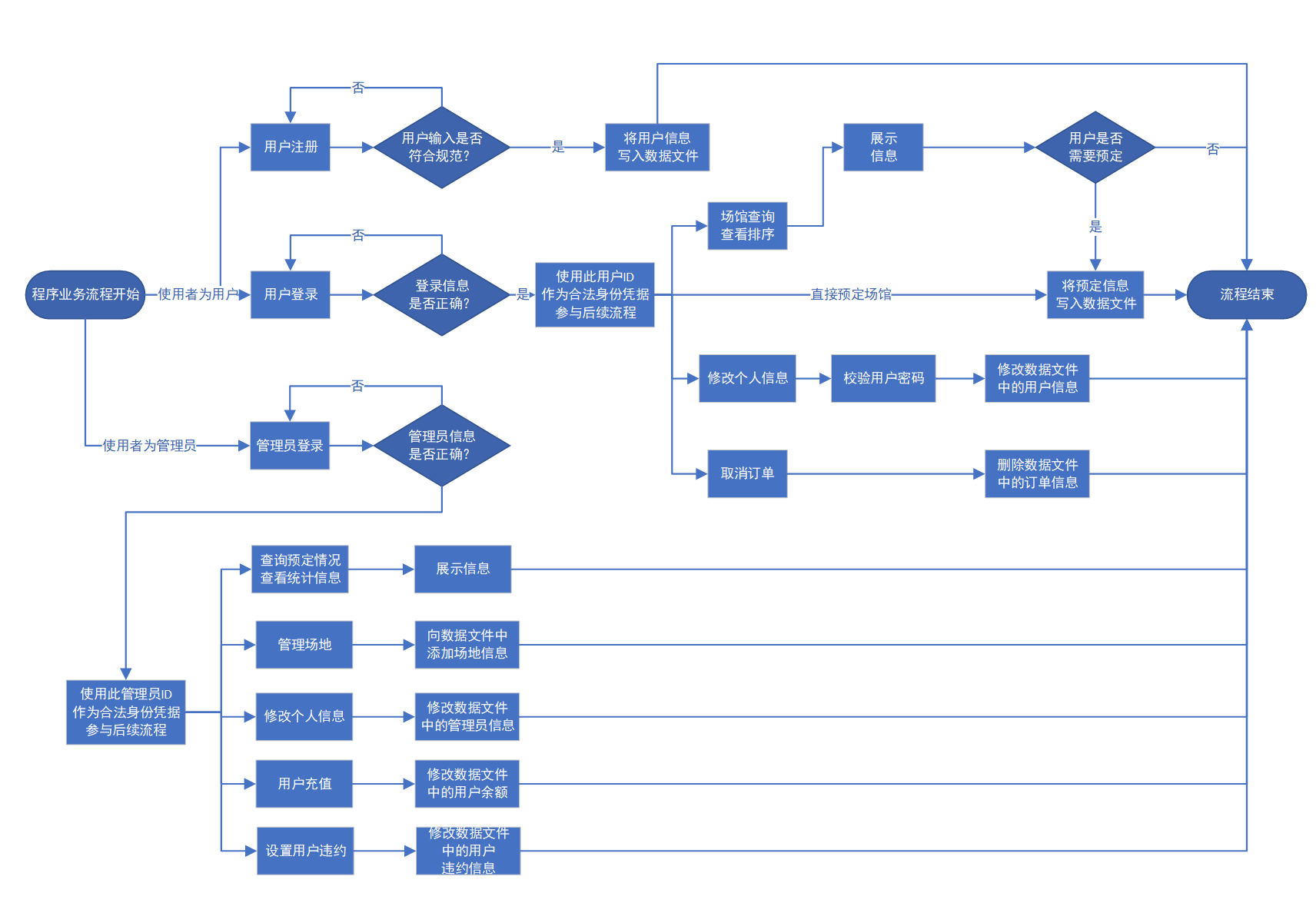
|  |  |  |  |
| --- | --- | --- | --- |
| **一级功能** | **二级功能** | **功能描述** | **约束说明** |
| 用户功能 | 注册 | 用户输入用户ID、密码、邮箱、手机号码等信息注册 | **注册使用的信息需满足以下条件：**   1. **用户ID：**要求6-10位，全部由字母或数字组成 2. **姓名：**15字符以内 3. **性别：**在男或女之间选择其一 4. **年龄：**必须是1-100间的纯数字 5. **手机号：**必须是11位的纯数字，且应以1开头 6. **电子邮箱地址：**必须有效 7. **密码：**6-20位，只能包含字母、数字和符号 8. **地理位置：**20字符以内 |
| 登录 | 用户输入注册时填入的用户ID和密码，登入管理系统 | 输入的用户ID和密码需**符合格式要求（同上）**  **用户需存在，且密码应当匹配**才可以正常登入系统 |
| 查询 | **根据场地名称查询**  用户输入待查询的场地名称，展示所有**含有用户输入字符**的场馆信息 | 输入的场地名称应在6-10个字符长度范围内  **查询后可直接跳转预定** |
| **根据场地所属区域查询**  用户输入待查询的场地所属区域，展示所有**含有用户输入字符**的场馆信息 | 输入的场地所属区域应在256个字符长度范围内  **查询后可直接跳转预定** |
| **根据场馆运动类型查询**  用户输入待查询的场地运动类型，展示所有**含有用户输入字符**的场馆信息 | 场地的运动类型应在64个字符长度范围内  **查询后可直接跳转预定** |
| **根据场馆名称查询**  用户输入待查询的场地运动类型，展示所有**含有用户输入字符**的场馆信息 | 场馆名称应在64个字符长度范围内  **查询后可直接跳转预定** |
| **查看按租金排序**  用户可查看所有场地按租金由少到多的排序 | 无需输入任何信息 |
| **查看按预定量排序**  用户可查看所有场地按预定量由多到少的排序 | 无需输入任何信息 |
| 预定场地 | **用户预定场地**  用户输入需要预约的场地ID、预约开始日期、时间和持续小时数，进行场地的预定。 | 1. 用户输入的所有信息需符合规范 2. 用户违约次数不得多于三次 3. 场地需存在 4. 用户年龄需要在场地的准入年龄范围之内 5. 用户余额 需足够支付租金， 6. 用户对场地的预约时间范围不得与其他用户（或自己）的预约时段发生冲突 7. 用户预约的时段应在可预约时长（6：00-22：00）范围内 8. 用户每次最多只可预约三个场地 |
| 修改信息 | **修改个人信息**  用户对自己注册时填写的个人信息进行修改。 | 用户生理和标识信息**（姓名、年龄、性别等）**不能修改，其余信息**（邮箱、电话、密码、地址）**可以修改。修改时输入的信息**需符合注册时的规范。** |
| 管理订单 | **查看订单**  用户可查看自己下过的预约订单。  **取消预约**  用户可取消自己先前的预约。 | * 取消预约时，用户需输入想要取消的预约信息索引号。输入的索引号应在屏幕上显示的范围内。 * 取消预约后，**不会进行退款。** |
| 管理员功能 | 登录 | 管理员输入绑定场馆并注册时填入的管理员ID和密码，登入管理系统。 | 输入的管理员ID与密码应当符合注册时的格式要求（ID应为6-10位，且仅由字母和/或数字组成，密码应为6-20位，且仅由字母、数字、符号组成）。 |
| 查看  预定信息 | **查看用户预定情况**  展示自己管理的场馆中所有场地的用户预定信息 | 无需输入信息 |
| 查看  统计信息 | 查看所管理的场馆统计信息，包括**按预定量排序、热门预约时间统计、场馆总营业额、场馆中所有场地的用户年龄层次统计、按营业额排序场地、男士/女士的最爱运动，以及场地的时均利用率** | 无需输入信息 |
| 管理场地 | **查询场地信息**  管理员可查看自己场馆中所有场地的详细信息。  **添加场地**  管理员可在自己管理的场馆中添加新的场地。 | **添加场地时，输入的信息需符合以下条件：**   1. **场馆ID：**6-10位纯字母或数字 2. **场馆名称：**64字符以内 3. **地理位置：**256字符以内 4. **运动类型：**64字符内 5. **场地简介：**100字符内 6. **场地每小时租金：**1.00-200.00的浮点数，且在输入时必须输入小数点后两位。 7. **场地最小准入年龄：**1-99的整数 8. **场地最大准入年龄：**1-100的整数 |
| 修改信息 | **修改自己的电话、密码等信息** | 管理员生理和标识信息**（姓名、性别、管理的场馆等）**不能修改，其余信息**（邮箱、电话、密码）**可以修改。修改时输入的信息**需符合管理员注册时的规范。** |
| 用户充值 | 输入用户ID和充值金额，为用户进行充值，添加余额。 | 输入的用户ID需符合规范且存在，单次充值金额最大为999.99元，且必须输入小数点后两位 |
| 设置用户违约 | 输入用户ID并确认后，可设置该用户有违约行为。 | 输入的用户ID需符合规范且存在，单次只可设置用户违约一次。 |
| 主管功能 | 添加管理员 | 添加一个新的场馆，并注册一个管理员账号，指定为该场馆的管理员。 | **添加场馆并绑定管理员时，输入的信息需符合以下条件：**   1. **管理员ID：**6-10位，全部由字母和/或数字组成 2. **姓名：**15个字符长度范围以内 3. **性别：**在男或女中选择其一 4. **手机号：**11位纯数字，以1开头 5. **电子邮箱：**必须为有效的电子邮箱地址 6. **密码：**6-20位，只含字母、数字和符号 7. **管理的场馆名称：**64个字符长度范围以内 |

## 模块详述

**3.2.1 主入口**

主入口位于Main.c文件中的main()函数，其中对系统主要流程进行了实现，主要包含系统主菜单、对其他模块接口的调用和文件封装相关的初始化。

核心业务流程如图所示：



**图 3-1 系统业务流程**

**3.2.2 数据存取模块**

数据存取模块中定义了统一的数据操作接口。

**表3-1 数据存取模块所有接口详情**

|  |  |  |  |
| --- | --- | --- | --- |
| **宏定义名称** | **宏函数参数** | **宏功能/定义描述** | **备注** |
| InitAllHeadNode | 无 | 初始化全部头节点  并从文件中读取信息 |  |
| BOOL | 无 | 定义BOOL为int |  |
| TRUE | 无 | 定义TRUE为1 | 方便写  返回值 |
| FALSE | 无 | 定义FALSE为0 | 方便写  返回值 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **函数名** | **函数参数** | **函数返回** | **描述** | **备注** |
| InsertToUserInfo | UserInfoNode链表头节点、UserInfo用户信息结构体 | int，是否成功插入 | 向表中插入新用户 |  |
| deleteUserInfo | UserInfoNode链表头节点、char\*用户ID | int，是否成功删除 | 删除ID为某的用户信息 |  |
| queryUserInfo | UserInfoNode链表头节点、char\*用户ID | int，是否查找到用户 | 根据用户ID查询某用户的全部信息 |  |
| EditUserInfo | UserInfoNode链表头节点、UserInfo用户信息结构体 | int，是否成功编辑 | 编辑某用户的信息 |  |
| saveUserInfo | UserInfoNode链表头节点 | int，是否成功保存 | 将内存中的链表保存至文件中 |  |
| loadUserInfo | UserInfoNode链表头节点 | int，是否成功加载 | 将文件中的链表读取至内存中 |  |
| printUserInfo | UserInfoNode链表头节点 | void | 打印全部用户信息 | 仅开发测试时使用 |
| isUserExist | UserInfoNode链表头节点、char\*用户ID | int，用户是否存在 | 查询某ID是否存在 |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **函数名** | **函数参数** | **函数返回** | **描述** | **备注** |
| InsertToAdminInfo | AdminInfoNode链表头节点、AdminInfo用户信息结构体 | int，是否成功插入 | 向表中插入新用户 |  |
| deleteAdminInfo | AdminInfoNode链表头节点、char\*管理员ID | int，是否成功删除 | 删除ID为某的用户信息 | 在系统业务中未使用到 |
| queryAdminInfo | AdminInfoNode链表头节点、char\*管理员ID | int，是否查找到管理眼 | 根据用户ID查询某管理员的全部信息 |  |
| EditAdminInfo | AdminInfoNode链表头节点、AdminInfo管理员结构体 | int，是否成功编辑 | 编辑某管理员的信息 |  |
| saveAdminInfo | AdminInfoNode链表头节点 | int，是否成功保存 | 将内存中的链表保存至文件中 |  |
| loadAdminInfo | AdminInfoNode链表头节点 | int，是否成功加载 | 将文件中的链表读取至内存中 |  |
| printAdminInfo | AdminInfoNode链表头节点 | void | 打印全部管理员信息 | 仅开发测试时使用 |
| isAdminExist | AdminInfoNode链表头节点、char\*管理员ID | int，管理员是否存在 | 查询某ID是否存在 |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **函数名** | **函数参数** | **函数返回** | **描述** | **备注** |
| InsertToSiteInfo | SiteInfoNode链表头节点、SiteInfo场地信息结构体 | int，是否成功插入 | 向表中插入新用户 |  |
| deleteSiteInfo | SiteInfoNode链表头节点、char\*场地ID | int，是否成功删除 | 删除ID为某的用户信息 | 在系统业务中未使用到 |
| querySiteInfo | SiteInfoNode链表头节点、char\*场地ID | int，是否查找到场地 | 根据用户ID查询某用户的全部信息 |  |
| EditSiteInfo | SiteInfoNode链表头节点、SiteInfo场地信息结构体 | int，是否成功编辑 | 编辑某场地的信息 | 在系统业务中未使用到 |
| saveSiteInfo | SiteInfoNode链表头节点 | int，是否成功保存 | 将内存中的链表保存至文件中 |  |
| loadSiteInfo | SiteInfoNode链表头节点 | int，是否成功加载 | 将文件中的链表读取至内存中 |  |
| printSitenfo | SiteInfoNode链表头节点 | void | 打印全部场地信息 | 仅开发测试时使用 |
| isSiteExist | SiteInfoNode链表头节点、char\*场地ID | int，场地是否存在 | 查询某ID是否存在 |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **函数名** | **函数参数** | **函数返回** | **描述** | **备注** |
| InsertToApmtInfo | AppointmentInfoNode链表头节点、ApmtInfo预约信息结构体 | int，是否成功插入 | 向表中插入新用户 |  |
| deleteApmtInfo | AppointmentInfoNode链表头节点、char\*发起预约的用户ID、char\*预约的场地ID char\*发起预约的用户ID、char\*预约开始详细时间 | int，是否成功删除 | 删除ID为某的预约信息 |  |
| queryApmtInfo | AppointmentInfoNode链表头节点、char\*发起预约的用户ID、char\*预约的场地ID char\*发起预约的用户ID、char\*预约开始详细时间 | int，是否查找到预约 | 根据此三条信息查询某预约的全部信息 |  |
| saveApmtInfo | AppointmentInfoNode链表头节点 | int，是否成功保存 | 将内存中的链表保存至文件中 |  |
| loadApmtInfo | AppointmentInfoNode链表头节点 | int，是否成功加载 | 将文件中的链表读取至内存中 |  |
| printApmtInfo | AppointmentInfoNode链表头节点 | void | 打印全部预约信息 | 仅开发测试时使用 |

**3.2.3 输入校验模块**

输入校验模块位于**UserComponent.c**文件中，定义了一系列方法用于校验用户输入是否合法。

模块接口返回值使用到了如下宏定义：

**表3-2 输入校验模块使用到的宏定义**

|  |  |  |  |
| --- | --- | --- | --- |
| **宏定义名称** | **宏函数参数** | **宏功能/定义描述** | **备注** |
| BOOL | 无 | 定义BOOL为int |  |
| TRUE | 无 | 定义TRUE为1 | 方便写  返回值 |
| FALSE | 无 | 定义FALSE为0 | 方便写  返回值 |

模块中包含的所有函数接口如下所示：

**表3-3 输入校验模块所有接口详情**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **函数名** | **函数参数** | **函数返回** | **描述** | **备注** |
| isValidUserID | char\*用户ID，也可是管理员和场地ID | TRUE/FALSE，是否合法 | 判断输入的ID是否合法 |  |
| isValidUserName | char\*用户姓名 | TRUE/FALSE，是否合法 | 判断输入的姓名是否合法 |  |
| isValidUserGender | char\*用户性别 | TRUE/FALSE，是否合法 | 判断输入的性别是否合法 | 只为0或1，定义0为男性1为女性 |
| isValidUserAge | char\*用户年龄 | TRUE/FALSE，是否合法 | 判断输入的年龄是否合法 |  |
| isValidUserPhone | char\*用户手机号 | TRUE/FALSE，是否合法 | 判断输入的手机号是否合法 |  |
| isValidUserPwd | char\*用户密码 | TRUE/FALSE，是否合法 | 判断输入的密码是否合法 |  |
| isValidUserEmail | char\*用户邮箱 | TRUE/FALSE，是否合法 | 判断输入的邮箱是否合法 | 这个写的真的很  符合标准 |
| isValidUserLocation | char\*用户位置 | TRUE/FALSE，是否合法 | 判断输入的地址是否合法 |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **函数名** | **函数参数** | **函数返回** | **描述** | **备注** |
| isValidVenue | char\*场地名 | TRUE/FALSE，是否合法 | 判断输入的场馆是否合法 |  |
| isValidDoubleNumStr | char\*场地租金字符串 | TRUE/FALSE，是否合法 | 判断输入的场地租金是否合法 |  |
| isValidMoneyAdd | char\*用户充值金额字符串 | TRUE/FALSE，是否合法 | 判断输入的充值金额是否合法 |  |
| isValidDate | char\*日期字符串 | TRUE/FALSE，是否合法 | 判断输入的日期是否合法 | 检测日期是否存在 |
| isValidYYYYMMDDHHDate | char\*日期字符串 | TRUE/FALSE，是否合法 | 判断输入的日期格式是否合法 | 检测日期是否符合格式 |
| isReserveDateConflict | AppointmentInfoNode头节点、char\*场地ID、char\*日期字符串、int开始时间（小时） | TRUE/FALSE，是否冲突 | 判断输入的预约日期是否冲突 |  |
| isReserveHourConflict | AppointmentInfoNode头节点、char\*场地ID、char\*日期字符串、int开始时间（小时）、int持续时间（小时） | TRUE/FALSE，是否冲突 | 判断输入的预约时长是否冲突 |  |
| isValidIndex | char\*用户输入的需取消预约索引字符串，int存在的最大索引 | TRUE/FALSE，是否合法 | 判断输入的需取消索引是否合法 |  |

**3.2.3 用户业务模块**

用户业务模块实现了用户侧的具体业务功能。

用户侧接口因涉及用户信息表的数据操作，参数**均需包含UserInfoNode头节点**。除非特殊说明，调用此模块中的接口均需包含作为首个参数，下文不再赘述。

模块中还包含一些复用性强的拆分小函数，但均为便利流程开发的过程性函数，对外适配性差，此处不作对外接口公布。

**表3-4 用户业务模块所有接口详情**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **函数名** | **函数参数** | **函数返回** | **描述** | **备注** |
| doRegister | - | void | 进行用户注册 |  |
| doLogin | - | TRUE/FALSE，是否登录成功 | 进行用户登录 |  |
| doUsrInfoEdit | - | 0 | 进行用户信息修改 |  |
| doSiteReserve | -，SiteInfoNode场地信息头节点，AppointmentInfoNode预约信息头节点 | 0 | 进行场地预约 |  |
| doSiteSearch | -，SiteInfoNode场地信息头节点，AppointmentInfoNode预约信息头节点 | 0 | 进行场地检索 |  |
| doApmtManage | -，SiteInfoNode场地信息头节点，AppointmentInfoNode预约信息头节点 | 0 | 进行订单管理 |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **函数名** | **函数参数** | **函数返回** | **描述** | **备注** |
| searchSiteByName | SiteInfoNode场地信息头节点，char\*需查询的场地名称 | void | 根据场地名称查询场地 |  |
| searchSiteByVenue | SiteInfoNode场地信息头节点，char\*需查询的场馆名称 | void | 根据场地所属场馆查询场地 |  |
| searchSiteBySoprtsType | SiteInfoNode场地信息头节点，char\*需查询的运动类型 | void | 根据场地运动类型查询场地 |  |
| searchSiteByLocation | SiteInfoNode场地信息头节点，char\*需查询的场馆地理位置 | void | 根据场地地理位置查询场地 |  |

**3.2.3 管理员业务模块**

管理员业务模块实现了管理员侧的具体业务功能。

管理员侧接口因涉及管理员信息表的数据操作，参数**均需包含AdminInfoNode头节点**。除非特殊说明，调用此模块中的接口均需包含作为首个参数，下文不再赘述。

模块中还包含一些复用性强的拆分小函数，但均为便利流程开发的过程性函数，对外适配性差，此处不作对外接口公布

**表3-5 管理员模块所有接口详情**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **函数名** | **函数参数** | **函数返回** | **描述** | **备注** |
| doAdminRegister | - | void | 进行管理员注册 | 需输入  主管密码 |
| doAdminLogin | - | TRUE/FALSE，是否登录成功 | 进行管理员登录 |  |
| doAdminInfoEdit | - | 0 | 进行管理员信息修改 |  |
| doAdminAddSite | -，SiteInfoNode场地信息头节点 | 0 | 进行添加场地 |  |
| doAdminStatistics | -，SiteInfoNode场地信息头节点，AppointmentInfoNode预约信息头节点,UserInfoNode用户信息头节点 | 0 | 进行查看统计信息 |  |
| doSetUserBreak | -，UserInfoNode用户信息头节点 | 0 | 进行设置用户违约 |  |
| doCredit | -，UserInfoNode用户信息头节点 | 0 | 为用户充值 |  |