卡通人物

中度可信度描述已自动生成

C++程序设计课程设计

实验报告

**2023/2024(2)**

徽标

描述已自动生成

实验题目 游乐园管理系统

学生姓名

学生学号

学生班级

任课教师

提交日期

**计算机科学与技术学院**

**游乐园管理系统 实验报告**

**一、课程设计内容**

游乐园管理系统（Amusement park management system）

基本要求：能从文件读出游乐场管理信息，并能将管理信息保存到文件

游乐场的基本信息包括：游乐设备编号、游乐设备名称、管理人员、消费金额、消费日期

功能要求：

1、系统以菜单方式工作；

2、增加、修改和删除游乐场消费记录；

3、查询游乐场某个游乐设备某日的收入总额；查询游乐场某日的收入总额；

4、显示所有游乐设备的收入情况；

5、能根据某时段的游乐设备收入的高低排序；能根据某日的游乐设备的收入高低排序；

6、可以把信息存入文件，也可以在系统中读出并处理。

**二、运行环境**

**软件环境**：

操作系统：Windows 11

开发软件：Microsoft Visual Studio 2022

平台：x64

语言标准：C++17 & C17

字符集：Unicode

子系统：窗口 (/SUBSYSTEM:WINDOWS)

**硬件环境**：

处理器：13th Gen Intel(R) Core(TM) i5-13500H 2.60 GHz

内存：16.0GB

系统类型：64位操作系统

**三、实验课题分析**

**1. 需求分析**

课题主要需求为：(1)菜单 (2)信息输入、处理、输出 (3)数据储存、读取，文件操作

**2. 功能结构分析**

分为两版，详见各板块

**四、前言**

在我看来，程序课程设计不仅仅是对学生能力的考验，更是一次机会，一次练手的机会，一次驱动自己学习新知识的机会。因此，为了在充分锻炼自己基本编程能力的同时学到新东西，我设下本次课设的**个人目标**。

本次课设我将分为两个迭代版本。

**第一版：**

仅使用C++的基本功能满足课设需求，交互界面使用最原始的控制台输出，实现课设要求的所有功能，即游乐园管理系统的核心功能

**第二版：**

不说功能如何，一个粗糙的控制台程序在表面上就很丑，所以，为了保护我和老师的眼睛(●ˇ∀ˇ●)，也是为了更方便、灵活、舒适的交互，在第一版的功能基础上，~~我将学习并使用Qt制作一个更加精美的图形界面，实现锦上添花。~~后经考虑，觉得Qt与本次考查目标关系不大，而且专门为了这个学它的话有点浪费时间精力，遂决定使用windows.h这个之前用过比较熟悉的库简单优化一下界面。

**PS；第一版是在写程序的同时写的报告，故演示图片体现的只是当时进度的代码，不完全代表最终成品情况。第二版为程序写完后写的报告，故代码均为最终版。**

**第一版**

**功能结构分析**

(1)菜单模块：

利用控制台输出或图形界面呈现菜单交互界面，提供用户与系统进行交互的途径，将代码调用呈现为可视化的交互方法

设定相关指令函数，由键盘输入或鼠标点击调用函数，通过菜单实现用户与系统的交互、系统间各个功能模块间的联动、操作

(2)交易模块：管理个人消费和设备收入记录

(3)用户模块：记录用户信息

(4)设备模块：管理设备信息，处理设备名称、个数、收入等信息

(5)时间模块：存储时间信息

(6)文件模块：实现程序对文件的操作

菜单为最核心的功能模块，通过它调用各个成员函数，创建各数据模块的对象

主功能1：消费记录

显示：显示游乐园消费记录

增加：增加游乐园消费记录

修改：修改某个指定的游乐园消费记录(包含删除)

保存：手动保存对消费记录的修改，使其同步到文件内

返回：返回上一级菜单

主功能2：收入情况

显示：按记录编号顺序显示所有游乐园设备的某日收入

查询：显示某个游乐设备某日收入 显示游乐园某日收入

排序：按某时段/某日收入高低排序并显示游乐设备的某时段/某日对应收入

返回：返回父级菜单

主功能3：用户信息

显示：显示用户信息

增加：增加新用户

修改：修改某个指定的用户信息(包含删除)

保存：手动保存对用户信息的修改，使其同步到文件内

返回：返回上一级菜单

主功能4：设备信息

显示：显示游乐园设备信息

增加：增加新设备

修改：修改某个指定的游乐园设备信息(包含删除)

保存：手动保存对设备信息的修改，使其同步到文件内

返回：返回上一级菜单

主功能5：退出系统：若有数据未保存，自动保存数据，随后结束程序运行

**模块的流程图**（这种东西真的画不来┭┮﹏┭┮）

操作反馈到控制台

控制台

时间

交易

菜单

用户

文件

设备

各模块一级级调用

程序对文件的读写

**模块功能的实现以及实验调试、测试、运行记录及分析**

1.**构建菜单模块框架 创建Menu.h和Menu.cpp 创建Menu类**

构建静态成员函数Menu\_Main,Menu\_Consume, Menu\_Income用于打印各级菜单界面(先写个简单界面的用于调试，后续优化)

创建静态成员变量Menu\_Choice作为选择功能的标记

调试：进行这一步时给我报了一个错



经过调试发现是因为我在Menu.h中对该静态成员属性进行了定义，而我的Menu.cpp与APMS.cpp（含有main函数）中 都包含了Menu.h头文件,故推测是编译器在编译时进行了两次编译发生重定义的错误，只需将定义写到Menu.cpp中即可，之后经过测试发现所有的类成员均应在.cpp文件中定义以避免重定义错误（知识get√）（另：如果为多态需将声明与定义写在同一个.hpp文件中）

创建静态成员变量Menu\_Flag作为菜单级数的标记 当该变量相同时调用Menu\_Return将返回相同的父级菜单（便于后续扩展）

构建静态成员函数Menu\_Exit实现功能-退出系统

构建静态成员函数Menu\_Return实现功能-返回父级菜单

调试：测试返回功能时发现直接退出程序，未能执行功能，随后发现编译器给出警告

图形用户界面, 应用程序, Teams

描述已自动生成

加上小括号就好了(￣▽￣)”

文本

描述已自动生成图片包含 文本

描述已自动生成文本

描述已自动生成 电脑屏幕截图

描述已自动生成 电脑萤幕画面

描述已自动生成文本

描述已自动生成文本

描述已自动生成

2.**多线程(thread)的使用&BUG的解决**

创建好菜单框架后，我忽然发现，我的菜单模块和一般的在主函数中写循环的方式有很大区别，很明显我的代码就是一个函数调下一个再调下一个……虽然这样很好看、简洁，可读性也不错，但每一个调用的函数都始终不会返回，**函数的资源也就无法释放**，然后…随着使用时间的延长，操作次数的增加，**程序占用内存就会越来越大，直至崩溃**。这是一个很大的BUG，我不得不去找解决办法，最初我想的是就**按一般的做法写循环**，但是这样的话十分**不利于后续扩展**，当菜单级数增加，主函数中的代码就会越来越乱，越来越复杂，可读性很差。此外，例如返回父级菜单这样的功能要想实现难度也会增加。因此，我必须找到其他的方法。最后，我忽然回想起之前做字符动画（送给朋友的礼物）时用到了**多线程（thread）**，那么，只要我在**调用每个菜单功能都以创建子线程的方式来做**，那就可以**回收资源**了！

调试：但是，为了使子线程创建新的子线程后继续运行完毕并销毁，我需要使用**.detach()**成员函数，这样的话主线程在创建子线程后运行完毕会导致**主进程终止引起程序终止**，**子线程会被同时强制销毁**。我必须找到一种方法使主进程在所有子线程终止前不会终止，最开始我想将主线程单独终止而不引起主进程的终止，但查了半天都找不到方法，最后只能无奈放弃，在main函数中写了一个**休眠函数的循环保活主线程**。/(ㄒoㄒ)/~~

这里暂时没有使用并发多线程操作，所以暂时不用写互斥锁

文本

描述已自动生成

文本

描述已自动生成

如图，每调用新的菜单时会退出上一个线程

文本

描述已自动生成

3.**消费记录---增加&显示&修改&保存 文件读取/写入**

创建**交易类Trade** 用于记录交易金额，交易时间，获得收入的设备编号，进行消费的用户ID

创建**时间类Time** 记录日期（年月日）和时刻（时分秒）

创建**用户类User** 记录用户ID和名称（便于后续扩展功能）

创建**设备类Device**记录设备编号/名称/其管理人员

创建**文件类File**准备构建文件操作函数

在Menu中创建静态成员属性**vector<Trade> Menu\_Trade** 用于记录交易数据

构建静态成员函数Menu\_Consume\_Add实现增加消费记录的功能

构建静态成员函数Menu\_Consume\_Display实现显示消费记录的功能

构建静态成员函数Menu\_Consume\_Modify实现修改（包括删除）消费记录的功能

构建静态成员函数Menu\_Consume\_Save实现保存消费记录的功能

调试：最初我在Add函数中调用了一个追加写入文件的函数，想实现对文件的实时修改，但发现这样在实现修改与删除功能时极其困难，故后改为在程序开始运行时调用读取文件的函数，在程序终止时调用写入文件的函数，以实现对交易数据的存储。

除此之外，最初我的Trade中有Time对象，而Time中有**string**类型的成员属性，这在我构建文件操作函数时给我带来了极大的困难----我通过函数将Menu\_Trade以二进制形式写入文件时未发生异常，但每当我**读取该二进制文件，程序就会出错**。又是一顿猛查，我发现要实现我的目标，只能使用一种麻烦的方法，而且效果还不好，就将string类型改为了char[]

但我的文件写入逻辑有点小BUG，正如注释所说的那样

图片包含 图形用户界面

描述已自动生成

思来想去没有想到解决办法，防呆设计我暂时搞不出来，因此修改文件写入逻辑---即除了在正常退出时自动保存外，在每次调用函数修改Menu\_Trade中的数据后，均询问是否确认修改，并且提供手动保存的接口（不设置自动保存是因为每次只修改一点都保存比较浪费性能）

文本

描述已自动生成 文本

描述已自动生成

文本

描述已自动生成文本

描述已自动生成

文本

描述已自动生成 文本

描述已自动生成

4.**用户信息&设备信息---增加&显示&修改&保存**

代码和消费记录的大差不差，具体可于附件中查看

构建静态成员函数Menu\_User\_Add实现增加消费记录的功能

构建静态成员函数Menu\_User\_Display实现显示消费记录的功能

构建静态成员函数Menu\_User\_Modify实现修改（包括删除）消费记录的功能

构建静态成员函数Menu\_User\_Save实现保存消费记录的功能

构建静态成员函数Menu\_Device\_Add实现增加消费记录的功能

构建静态成员函数Menu\_Device\_Display实现显示消费记录的功能

构建静态成员函数Menu\_Device\_Modify实现修改（包括删除）消费记录的功能

构建静态成员函数Menu\_Device\_Save实现保存消费记录的功能

5.**收入情况**

构建静态成员函数Menu\_Income\_Display实现显示收入情况的功能

构建静态成员函数Menu\_Income\_Search实现查找收入情况的功能

构建静态成员函数Menu\_Income\_Compare作为sort调用的函数

构建静态成员函数Menu\_Income\_Sort实现按收入高低排序的功能

文本

描述已自动生成

文本

描述已自动生成 文本

描述已自动生成

**小结**

在写第一版程序时，写的越多我越发现想以第一版为基础扩展更多功能很不现实，因为第一版所有的设计都是以控制台输出为基础考虑的，即使我设计功能时考虑了后续的扩展性，之后要转为图形界面仍旧需要大改代码。故决定以第一版为保底，先完成基本的课设任务，第二版将重写整个程序，所以就不给第一版太多篇幅了，大致介绍一下算了，也不投入过多精力优化了，第二版才是精髓(●’◡ ’●)

PS:第一版的相关.cpp/.h/.dat文件也将被打进压缩包中，方便老师参考。

**第二版**

我在第二版的程序中预设了一个超级管理员账户，账号密码均为SuperAdmin以供测试

**功能结构分析**

虽说图形界面不是考查重点，但是一个图形窗口所能承载的功能远比控制台要多，也更能体现设计与代码能力，故第二版使用win32编程重新设计整个程序。

APMS.cpp文件中有程序入口函数，为主文件

创建general.h头文件 定义程序中用到的所有宏

定义Data类完成对各数据的文件操作

定义MyWnds类 用于创建/控制各种窗口/控件

定义Time类 用于记录时间数据

定义Permission类 用于记录权限数据  
定义Door类 用于登入/退出程序

定义Account类 用于记录账户数据

定义Device类 用于记录设备数据

定义Notification类 用于记录公告数据

定义Trade类 用于记录交易数据

定义Recharge类 用于记录充值数据

**模块的流程图**

Door(登录/注册)

MyWnds(显示登录/注册界面)

进入程序

APMS(程序入口)

Data(操作数据)

MyWnds(显示主窗口)

切换显示的界面

Account(用户信息)

Trade(交易信息)

Device(设备信息)

**模块功能的实现以及实验调试、测试、运行记录及分析**

/\*由于第二版的代码量过于庞大，故贴图时会省略不重要的代码，且除了最重要那部分均只贴声明/类定义，具体实现代码要去文件中查看\*/

**1.图形界面的构建**

定义MyWnds类 用于创建/控制各种主要窗口/控件（涉及数据信息如用户信息/设备信息等的子窗口定义放在了相关类中，方便调试） 由于该类不会用于创建对象，故成员函数与变量均使用static修饰

**1.1主窗口**

1.1.1设定主窗口大小

为了实现窗口自适应大小，将主窗口的初始大小设定为用户显示器大小的0.7倍

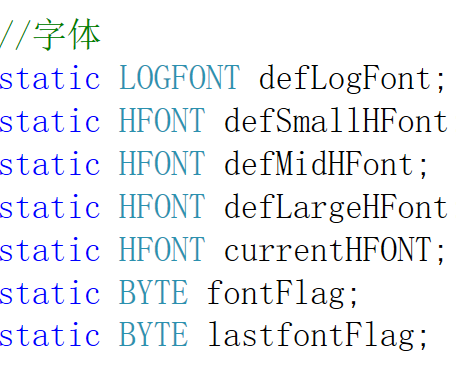
文本, 信件

描述已自动生成 文本

描述已自动生成

1.1.2创建大/中/小字体用于实现自适应窗口

图片包含 徽标

描述已自动生成文本

描述已自动生成

这三种字体会在窗口大小变化时根据其大小自动调节，使窗口尺寸变化后的界面更加舒服

1.1.3创建主窗口

文本

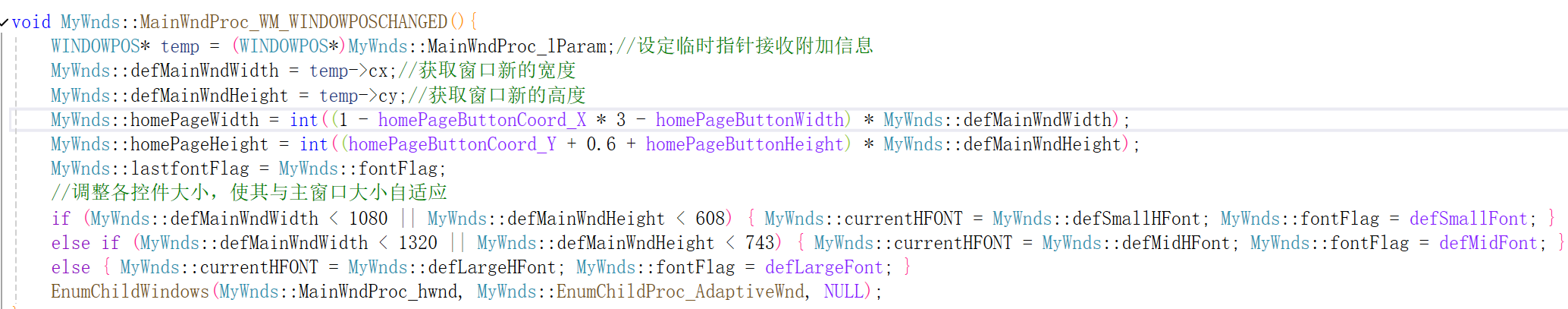
描述已自动生成MainWnd()为创建窗口的函数  
MainWndProc()为主窗口进行消息循环时回调的窗口过程函数  
以MainWndProc\_为前缀的函数均为对处理主窗口各消息的代码的封装,\_后的均为消息名

* WM\_COMMAND用于处理点击按钮的消息 在该函数中根据点击的按钮的ID执行对应按钮的功能
* WM\_WINDOWPOSCHANGING与WM\_WINDOWPOSCHANGED用于处理拖拽窗口大小的消息

前者用于限制窗口的最小尺寸文本

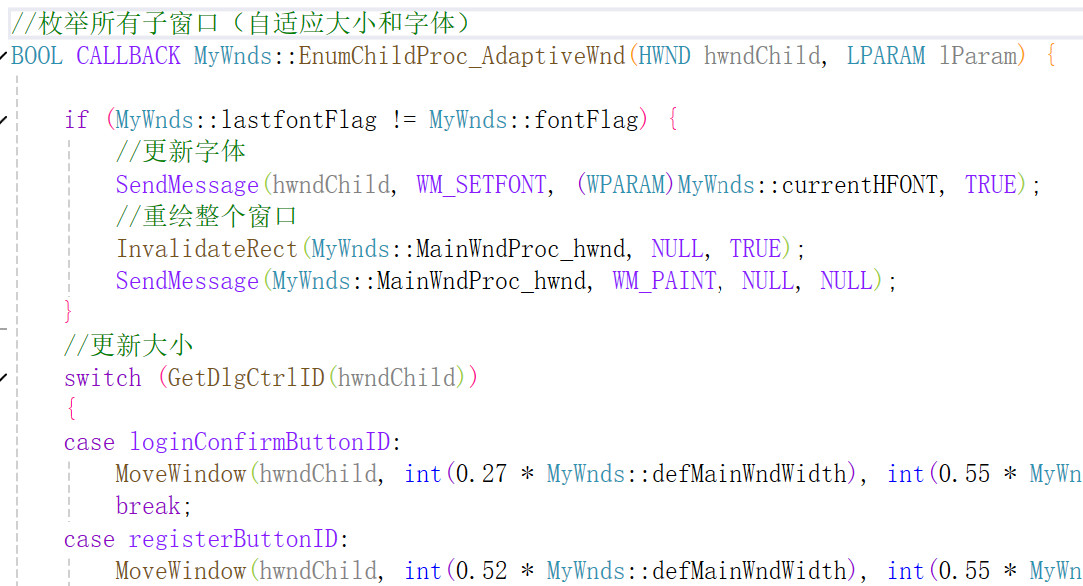
描述已自动生成

后者用于实现整个程序的自适应尺寸



最后一行的EnumChildWindows函数用于根据主窗口尺寸调整所有子窗口尺寸和字体

EnumChildProc\_AdaptiveWnd用于对每个窗口执行相应的调整尺寸和字体的代码(仅截一部分图，剩下的均为类似的case代码)



* MainWndProc\_WM\_CLOSE与 MainWndProc\_WM\_DESTROY 用于处理关闭程序的消息
* MainWndProc\_WM\_PAINT与MainWndProc\_WM\_CTLCOLORSTATIC用于绘制窗口背景 设定颜色

**1.2模态对话框**

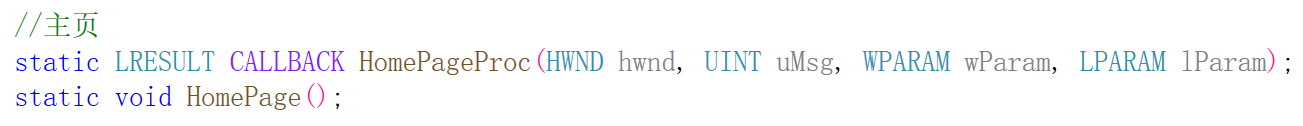
为了实现点击某个按钮弹出提示框/弹出子窗口的效果，构建一个模态对话框

图形用户界面, 文本

中度可信度描述已自动生成

前两行的函数功能与主窗口的类似，分别为窗口过程与创建窗口，剩余的则为根据功能的需要分别显示不同的界面

**1.3主页**

****

该子窗口用于显示功能提醒及声明版权（主要是实在想不出来能用来干嘛（如果不加公告栏等功能的话））

文本

描述已自动生成

**1.4用户信息**

**文本

描述已自动生成**

该子窗口用于显示登录用户的所有信息，且可在其中修改昵称和密码以及注销用户

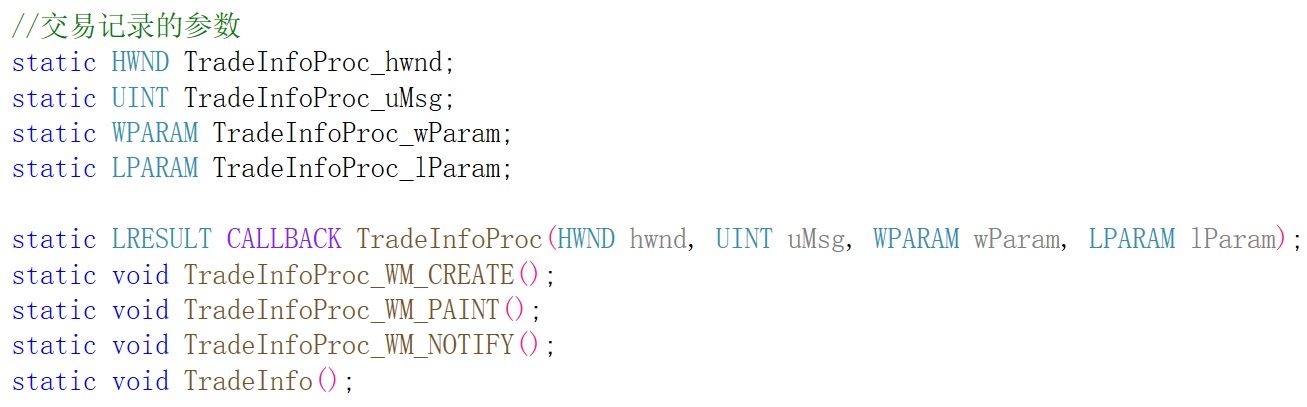
**1.5设备信息**

**图形用户界面, 文本, 应用程序

描述已自动生成**

该子窗口用于显示游乐园所有设备的信息，且可双击某个设备信息显示详情

**1.6交易信息**

****

该子窗口用于显示用户与设备间的交易信息

**1.7列表信息排序**

实现点击列表的列标题按当前列数据进行升序/降序排序的功能

图形用户界面, 应用程序

描述已自动生成

图形用户界面, 文本

描述已自动生成

ListCompareFlag用来切换升序/降序的模式，点击一次为升序时，下一次为降序

ListCompareFunc则为之后API会调用的回调函数，该函数用来具体实现排序功能（且会根据数据为字符型或数值型自动调整排序方式）

图形用户界面, 文本, 应用程序

描述已自动生成

每一个含有列表控件的子窗口，在处理WM\_NOTIFY消息时，对代码为LVN\_COLUMNCLICK的消息处理，使用ListView\_SortItemsEx API调用回调函数实现排序功能

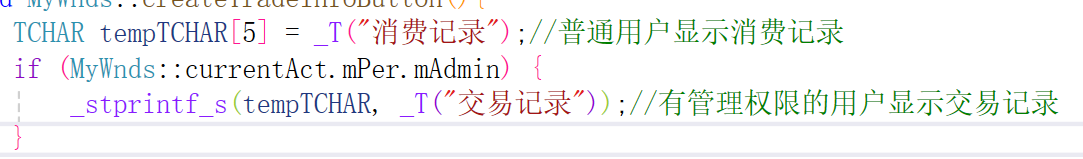
**2.权限体系的构建**

虽说课设要求的是管理系统，但我认为，作为一个程序而言仅仅只是提供管理功能未免有点浪费。同时从实用性与效益方面考虑，一个普通用户和管理员都能使用的游乐园管理系统显然比单纯的管理要好，同时也能充分利用代码。因此，我为这个程序构建了一套非常简单的权限体系，让拥有权限的管理员与游乐园的普通消费者能够使用的功能有所区别，既满足课设要求也有一定的扩展。

**2.1权限的基础**

在项目中我定义Permission类（作为Account类的成员属性）用来记录一个用户拥有的各种权限（当前的简单程序只有管理权限一个类别）,同时mAdmin的类型是BYTE(unsigned char)，其取值范围为（0-255），0作为普通用户的权限，其他数字的大小区别管理员权限的高低（不过当前的程序并没有细分高低权限的功能，只区分了0与非0的权限，这样设计方便扩展）

**2.2不同权限的功能区别**

在实现上我使用if语句判断登录用户的权限来决定按钮显示的文字（如图中消费记录与交易记录的区别，前者意味着普通用户只能查询个人的消费记录，后者则为管理员可以查询所有交易记录，这样更加贴切）以及是文本

描述已自动生成否显示一些功能（如图中只有管理员的界面上会显示增加/修改/删除以及筛选的点击项）

**2.3高权限对低权限的压制**

低权限用户无法对高权限用户执行增加/修改/删除功能，在执行前两个功能时会弹出权限不足的提示框，执行删除时不予执行

图形用户界面, 表格

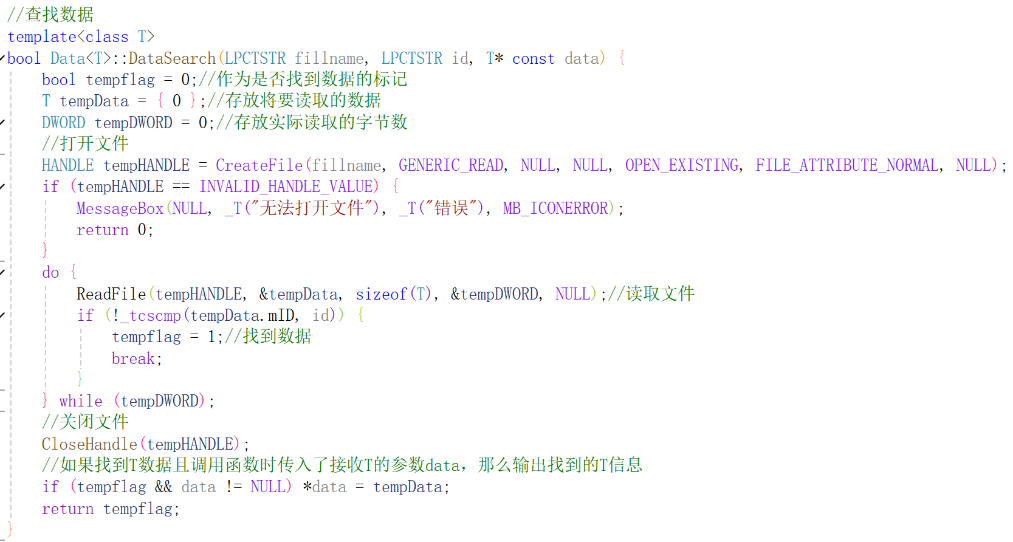
描述已自动生成

**3.各数据信息的处理**

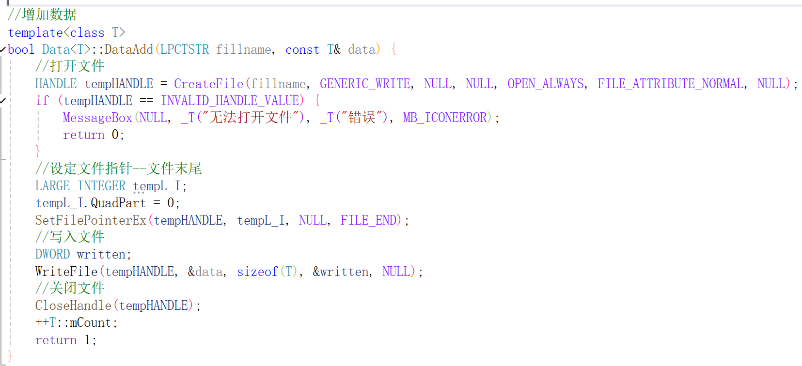
**3.1数据的查找/增加/修改/删除**

使用类模板Data完成对各数据的文件操作，提高代码复用性，每种数据都有一个成员属性mID(编号)和静态成员属性mCount，记录每条记录的唯一编号和每种数据的总记录数,方便Data进行操作

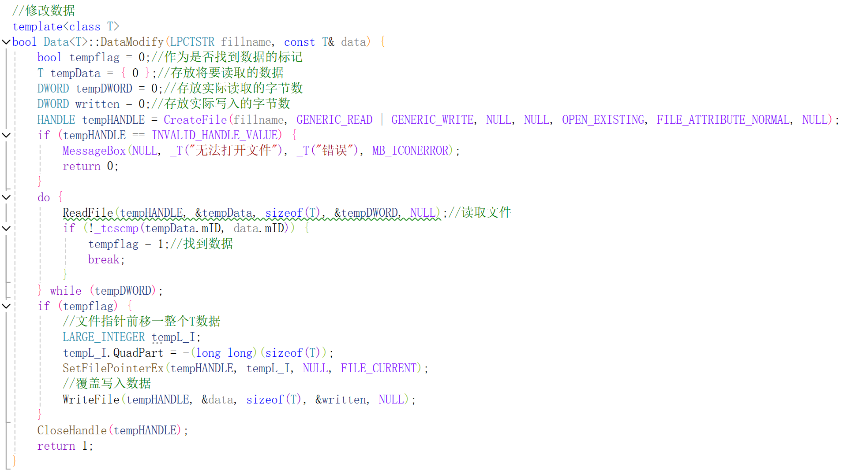
3.1.1查找

在定义中该函数的第三个参数为默认参数NULL，且此处模仿WindowsAPI中的out参数，若调用该函数时传入第三个参数，则函数在找到数据时会输出找到的对应数据信息

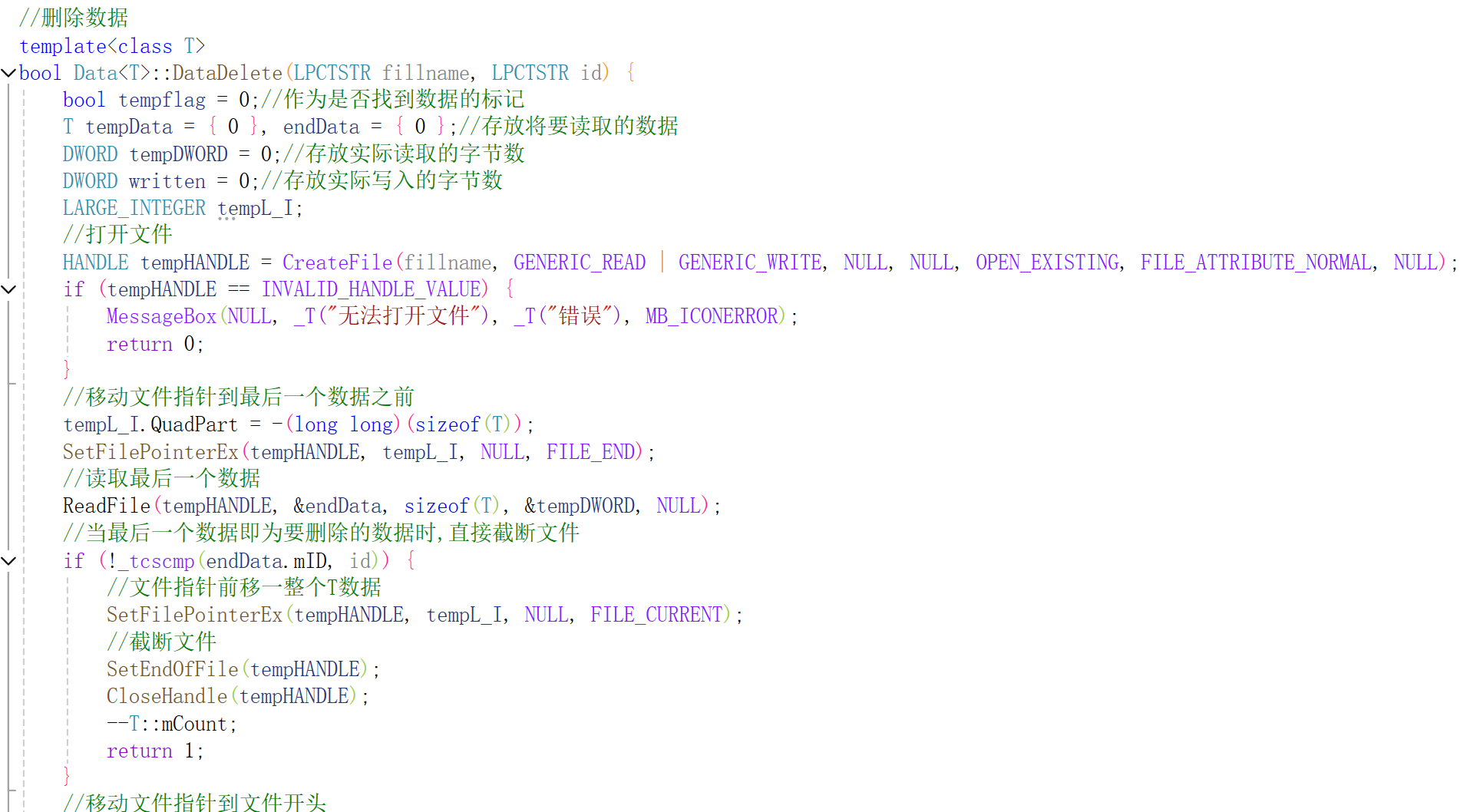
3.1.2增加

增加数据时在文件末尾直接写入新的数据

3.1.3修改

修改数据时在数据原位置覆盖写入新的数据

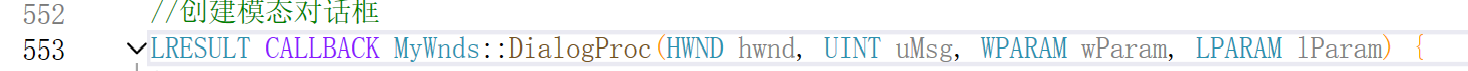
3.1.4删除

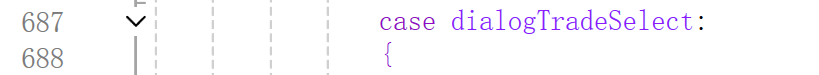


删除数据时，若位于文件末尾，直接截断文件实现删除；若位于文件中间，则将文件末尾的数据覆盖写入要删除的数据的位置，再截断文件

**3.2交易记录的筛选**

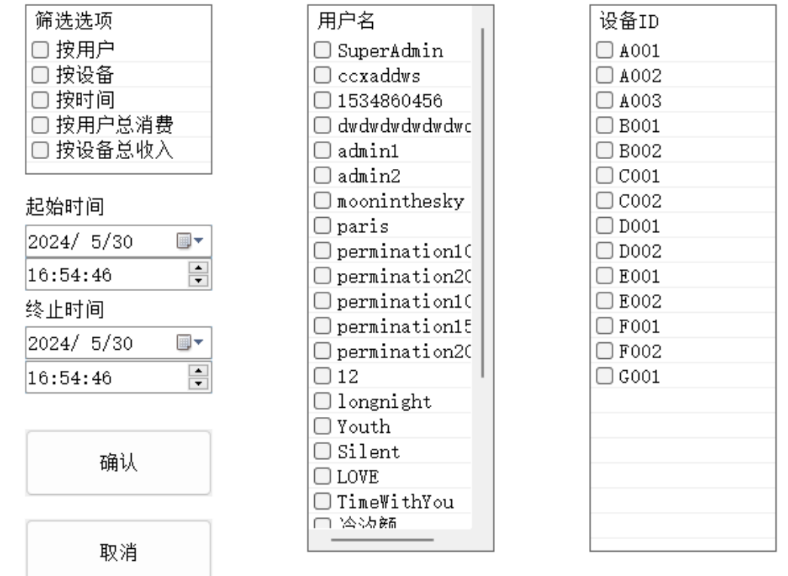
这段代码很长，如果需要具体看建议在MyWnds.cpp文件中的



中的中查看

我来说下实现思路：

首先从上至下读取对话框中这个列表的选择状态，若选中则进行相应筛选

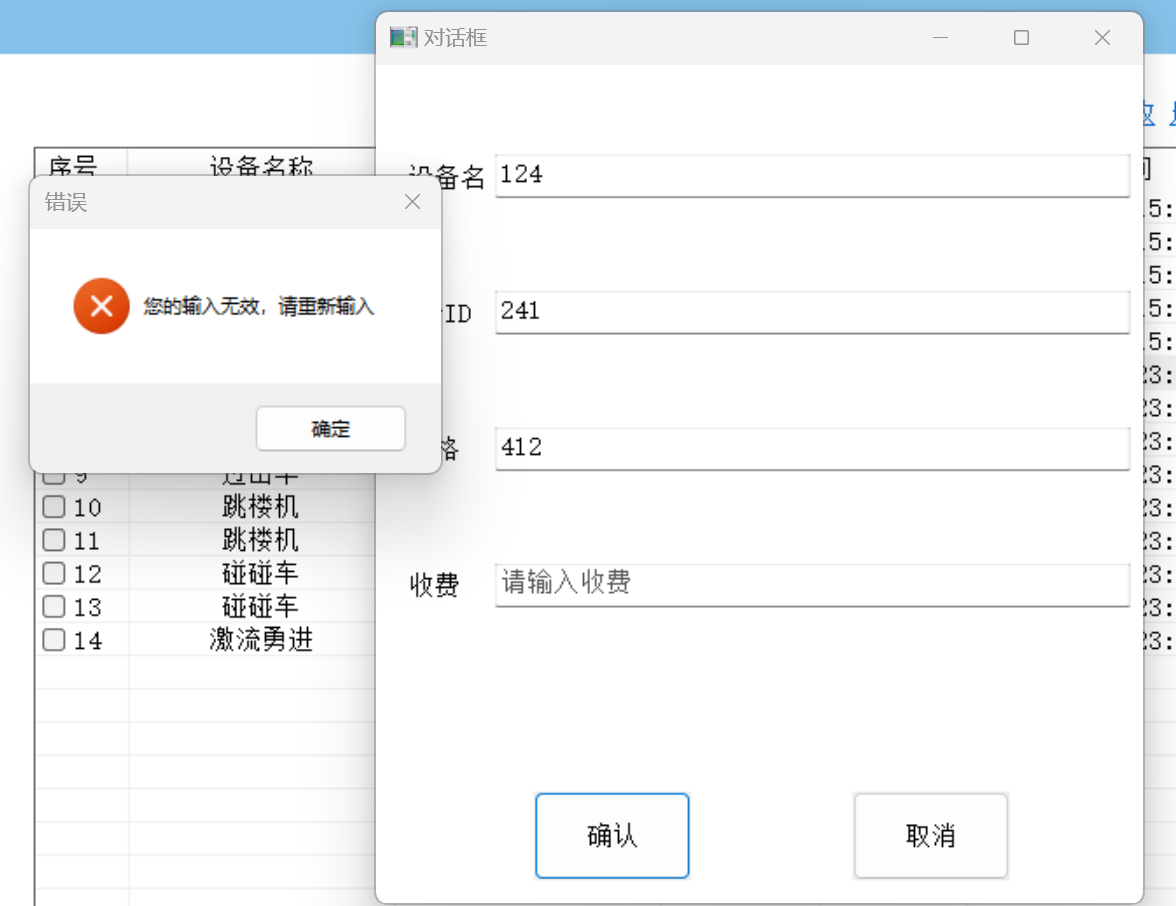
其中按用户/按设备/按世界三个可以随意选择/不选择，在选择后其对应的选择框才会发挥作用，而这三个选项的实现原理是：

从下至上读取交易信息列表，每读一条记录，从下至上遍历筛选对象的列表，若该记录对应的筛选对象在列表中未被选中，则删除该条记录。这样，当前三个筛选选项执行完毕，剩余的即为满足条件的交易信息。

剩下的两个选项最多只能同时选中其中一项（若选择按用户总消费，则会对交易信息列表中剩余的记录按用户分别计算其总消费并增加至列表末尾，最后全部增加完成后，删除前面的单个信息）

**4.容错处理**

一个良好的程序应当提前考虑好在实际发布后用户的各种骚操作可能会引发的问题，并提前堵住用户瞎搞/意外搞出的漏洞，以此提高体验度，并减少后续BUG的修改量。因此，我为该项目设计了各种各样的容错处理，预防BUG的产生。

* 若各种输入框输入为空，则点击确定时弹出弹框（登录/注册/增加/修改/删除）  
  
* 数据ID唯一，同一种数据不可同时出现相同的ID（注册/增加）  
  图形用户界面, 应用程序, 表格

  描述已自动生成  
  若对数据执行修改，则唯一ID对应的编辑框为只读状态无法操作  
  图形用户界面, 应用程序

  描述已自动生成
* 若点击修改时未选择具体数据，则弹出弹框阻止  
  表格

  描述已自动生成
* 图形用户界面, 应用程序

  描述已自动生成若对某数据进行修改中途直接退出/意外退出，则不保存相应修改，除非点击保存  
  此时若我点击左边的主页切换界面，则该修改不保存
* 所有的编辑框均会限制其输入字符数（不会超过存储该数据的数组的最大值）&int类型的数据对应编辑框只能输入数字

图形用户界面, 文本

描述已自动生成

设计思路

//由于文件中数据的顺序对软件使用几乎没有影响，故删除数据时采用“拆东墙补西墙”的方法，即将最后一个数据移到删除的数据的位置，减少删除数据时消耗的性能

//程序首次启动自动创建一个有最高权限的超级管理员，用户名与密码均为SuperAdmin

//根据权限决定功能

//第一版用fstream操作文件，第二版用WindowsAPI

//实现窗口控件自适应变化，控件随主窗口大小变化成比例变化

//自定义三种大小字体，随窗口大小变化自动变化，实现字体与控件大小的良好匹配

//利用宏定义每组控件的锚点控件相对于父窗口的坐标与大小的倍率，方便随时调整，每组中的每个控件坐标与大小都与锚点控件的对应值有关联

//~~进行数据处理时，实时直接操纵文件指针对文件内容查找与修改，省去用数组暂存在内存中的步骤~~（验收后知道是错误的做法）

//在不同界面间转换时，即时销毁无用控件，创建新控件，提高程序性能

//根据权限显示/隐藏功能按钮

//后续优化控件TAB自动切换聚焦

//做好防呆设计，防止用户不按常规使用卡出奇奇怪怪的BUG

BUG

充分利用GetLastError函数和打断点寻找问题原因

//直接拖拽窗口时不会自动绘制更新的区域，需在处理WM\_PAINT消息中手动调用函数实现

比如这样

//窗口类只能创建一次，多次创建要写判断条件防止重复创建相同的窗口类

//创建窗口时一定要构建好窗口过程，注意默认过程必须返回

//子窗口不要构建消息循环（与主窗口平级的模态对话框除外）

/\*未出现具体调试BUG的图是因为第二版程序写的时间比较长，且调用windowsAPI时出现的BUG很多很杂也都是小BUG，当时也(忙着修BUG)没想到要记录下来，现在也想不起来都碰到了啥奇奇怪怪的BUG，故没有图\*/

**小结**

虽然使用多态继承啥的有优势，但是忽略实用性为了多态而多态或者为了继承而继承并没有什么意义，在我的设计思路中没有什么用处的特性就不刻意去用了(比如当使用多态对代码的简化与扩展性没有帮助的时候)。

2024/5/20 今天验收了，老师指出我的第二版程序在业务逻辑方面有问题，听了老师的解释后我也认为确实如此。但是如果要改的话……基本每十行都要改一下，难度不亚于全部重写一遍。实话说，我的第一版是正常的，使用vector存储数据并在程序关闭时写入文件，所以正常的逻辑我也会写，我就不再浪费时间费很大精力去大改第二版了，这样做除了能提高一些分数并没有意义，我大可以用这些时间学别的技术，而且我的初衷也是通过课设的机会学习一些东西，而不是追求更高的分数，所以就只拿我该得的分吧。

**总结**

总的来说，本次课设对于我还是很有意义的，我不仅在第一版中练习了C++中的基本语法与知识，浅浅的了解了一下多线程操作，同时也在第二版中深入学习了windowsAPI的使用和图形界面的构建，此外，我还在实践中较好地练习掌握了debug的能力。

但与此同时，本次课设也暴露出了我的一些不足之处，尤其是第二版缺乏数据结构，这个问题是从最初构思时酿造的，我没有仔细了解使用内存存储数据的作用，一厢情愿地认为直接操作文件会更有效率，这个自以为是的想法直接导致了第二版的重大缺陷。之后，我将吸取本次课设的教训，不断提高自己，同时在最初设计程序时要严谨、务实，避免这样的错误的再次出现。

**附录：源代码**