中山大學 SUN YAT-SEN UNIVERSITY 实 验 报 告

实验人：赖奕恺　学号：19335093　日期：2020 年 6 月 11 日星期四

院（系）：数据科学与计算机学院　专业（班级）：计算机大类（行政 3 班教务 2 班）

实验题目：

设计一个校园卡管理系统，根据用户的不同实现增删改查功能操作。

一. 实验目的

本实验面向 C++语言的初学者。
主要让实验者熟悉面向对象的编程思想以及类的使用。

二. 实验环境

操作系统：Ubuntu、Windows

2.1编程语言和开发工具

编程语言： ANSI C/C++
开发工具： Visual Studio Code、GDB、g++（前期）和Dev C++（后期）

2.2 编码规范

使用主流的编码规范：Google C++Style Guide（中文版）

1. 自动变量名称统一小写，用下划线连接。
2. 数据成员后跟一个下划线与自动变量区分
3. 类名统一用驼峰命名法。

三. 实验内容

见附件。

四.分析与设计

4.1、需求分析：

此软件要满足两类用户的需求：

校园卡管理员（Administrator）：管理员具有高级权限，可以对学生的校园卡进行增删改查的操作，既要有批量操作也要有单张操作。

学生：具有普通权限，只能对自己的校园卡进行消费、挂失、激活（改）和查询（查）。

双方各司其职，功能分配合理，不缺失不冗余。

系统主菜单包含 2 个子菜单，一个功能点：

（1） 进入管理员菜单。在主菜单界面用户输入 1 后，进入管理员菜单登录界面。

（2） 进入学生菜单。在主菜单界面用户输入 2 后，进入学生菜单登录界面。

（3） 退出系统。在主菜单界面用户输入 3 后，退出整个系统。

管理员菜单包含 8 个功能点。

（1） 管理员登录及验证。

（2） 添加学生。

（3） 给学生绑定新校园卡。

（4） 删除学生。

（5） 查询学生

（6） 查询校园卡

（7） 给学生充值校园卡。

（8） 返回主菜单。

学生菜单包含 5 个功能点：

（1） 学生登录及验证。

（2） 查看校园卡信息

（3） 激活校园卡并重设密码

（4） 查看自己的信息

（5） 消费。

（6） 挂失。

（7） 查询消费记录

（8） 查询充值记录

4.2 设计思路：

从用户需求出发，定制友好的交互界面，定义符合实际情况的功能，从而选取合适的方式管理相关数据，并注意实用性，维护代码的易读性，提升易维护性。

4.3 设计要点：

1. 对同一层次相近结构和功能的类进行抽象，利用多态灵活调用相关函数。
2. 定义大量出现的可定制的交互函数，提升交互的整洁度。
3. 将交互界面与底层数据管理分离，从而提高了代码的层次性。

4. 将数据和功能分离定义，从而方便将数据看为一个整体，提高了易写度和可读性。

5. 利用文件操作，实现了数据的储存。真正使管理系统具有实用性。

6. 功能做到合理化，从而保证了程序内的数据及其关系始终合理。

7. 尽量使不同概念的操作互相分离，使得代码层次明显，并使封装严密合理。

4.4、类结构设计

4.2.1 数据类型

学生类定义学生属性。

SchoolCard 类定义校园卡相关属性。

4.2.2 数据管理类

SystemManage 类管理 Manager 类对象和 manager_list.dat 文件；管理 Student 类对象和 student_list.dat 文件;管理 SchoolCard 类对象和 schoolcard_list.dat 文件。

4.2.3 系统功能封装。

SystemFunction 类封装管理员的相应功能；封装学生的相应功能。

4.2.3 菜单封装。

MainMenu 主菜单类：包含管理员菜单和学生菜单两个数据成员。

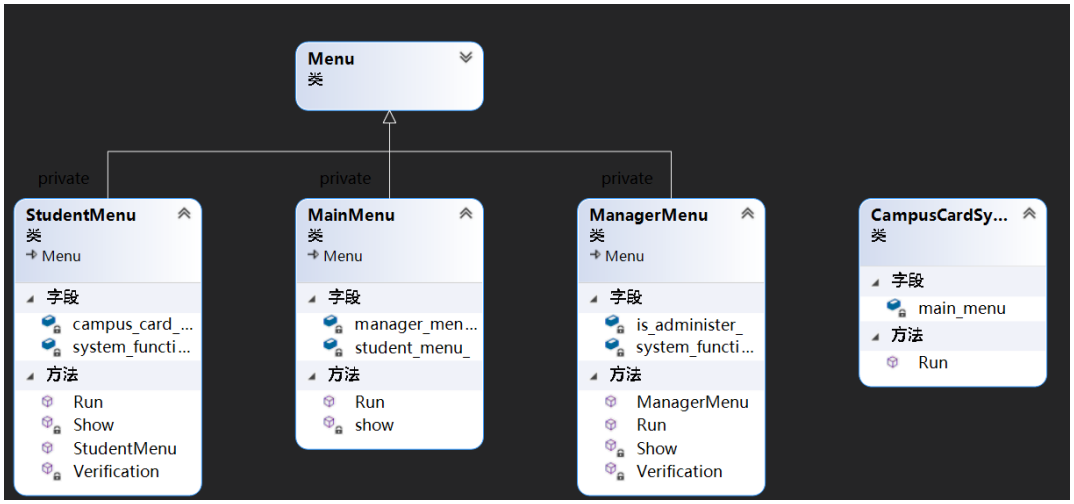
ManagerMenu 管理员菜单类：包含功能容器，能显示菜单，并调用相应功能。

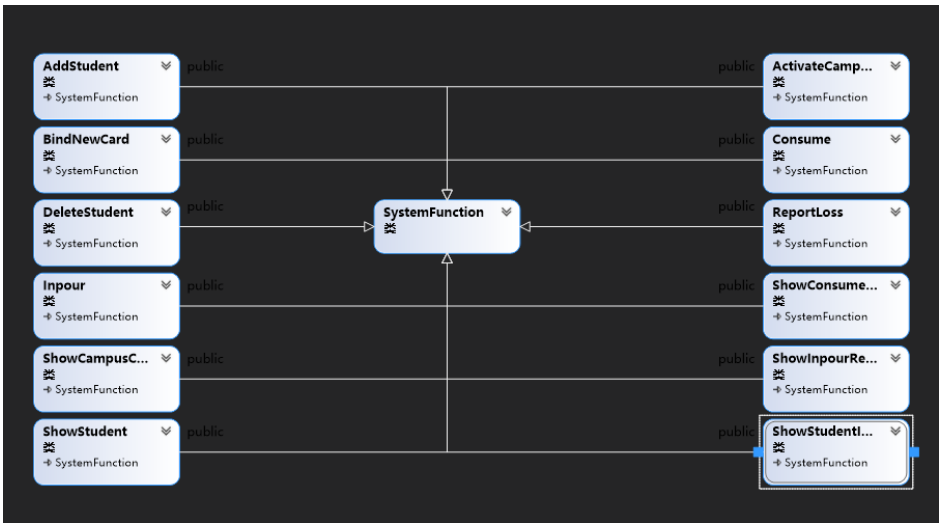
StudentMenu 学生菜单类：包含功能容器。

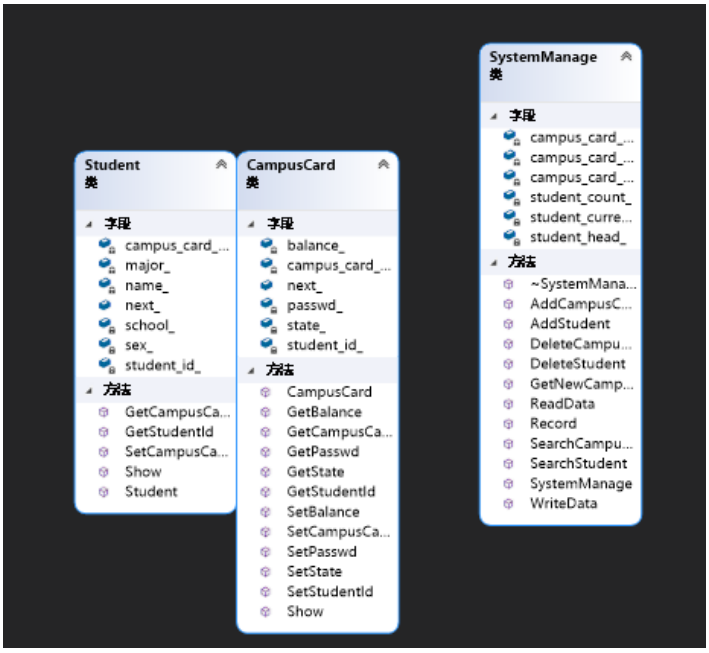
4.2.4 系统封装。

CampusCardSystem 校园卡管理系统类：封装整个系统，成员包括菜单。提供运行函数，可以对数据管对象进行初始化，并运行菜单。

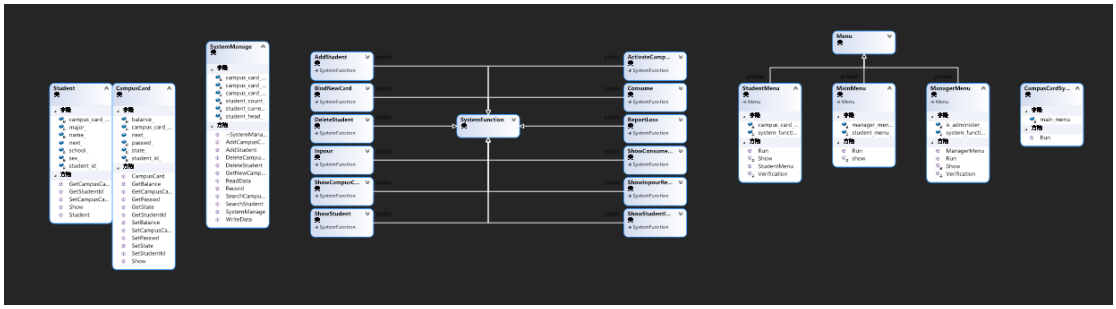
类关系图:







总览：



4.5、细节设计

接口设计：

1. UI：将用户输入的数据和选择统一为 int 整形，从而用一个 GetInt 函数统一了获取输入的 IO 界面。

数据成员设计：

1. 校园卡与学生的绑定：各自类内存储对方的 id 信息，这样只需经过管理类的查询功能就能将两个对象联系起来。
2. 菜单类的功能函数容器：用一个 vector 容器（system_functions_）储存了各个在构造函数内创建添加的功能类（SystemFunction）。方便了对功能进行增删。
3. 校园卡必须要有卡号：唯一的校园卡卡号对应唯一的校园卡实体，这保证了校园卡的唯一性不会因补办卡而被破坏，从而提升了现实中的安全性。

成员函数：

1. 数据管理类提供了各种数据管理的功能，程序的整个声明周期只有一个公共的数据管理类，避免了管理的混乱。
2. 功能类仅提供一个获取功能名字、执行相应功能的接口，封装严密合理。利用了多态性，灵活地根据用户选择对不同功能进行调用，并能让系统自动显示，更方便了功能的删减，调序。
3. 数据类提供输出运算符和各种值的 get，set 函数方便对值进行管理。

五、实验结果

见测试数据附件 TEST.txt

测试意图：测试相关功能的可用和验证交互界面的合理性。

测试过程中，发现了如下用户不友好问题：

1. 输入无法退出。解决方案：-1 退出。
2. 有些输入未被验证，导致程序意外退出。解决方案：增加验证语句。
3. 读取数字的语句能读取非法的字母。解决方案：检查 cin 的有效性。
4. 界面拥挤。解决方案：多加空行。

还发现了几个 bug：

1. 当可执行文件位于文件夹外部时，就无法读写文件。解决方法：更改程序内文件读写路径。
2. 函数可以接受输入，但毫无反应。分析：没有意外退出，应该是正常退出，可能是验证模块的判断出了问题。解决方法：改正了==写成=的错误。
3. 文件读写前部混乱，后部空行增多。解决方法：将写操作的换行符放在了输入其他信息之前。
4. 打印记录时会重复打印一行。分析：查看了数据文件，没有读写错误，应该是打印出了问题。解决：原来是应为跳出循环的条件为 eof，而读入最后一个字符时的时候未达到 eof，再次读入文件流对象被置为 eof，自动变量的值就会被保留，从而继续执行输出语句时就会重复输出。只需要在读入后检查 eof 即可。

六、设计心得。

1. 本次设计是对我这一年来所学的大检验，让我体会到了实操的重要性，更让我意识到了平时实操的缺乏是导致我熟练度不够的重要原因。以后要多尝试刷力扣。
2. 设计要自顶向下，层次分明。数据管理和用户交互要相互分离。
3. 要注重对程序模式的分析，提取出复用率高的代码，适度增加可定制性，使其复用率更高。
4. 写代码应该在思考全面后再行动手，这样能避免无用功。
5. 意识到了代码复用的强大之处，需要再接下来多多学习 STL 的使用和实战。
6. 要善于分析，将实现目标分解。
7. 更要多于他人交流，增长见识。
8. 不能过度的进行功能拆分，再拆分的同时需要对工程规模进行评估。一开始做的非常大，写了快 40MB，后面发现很多功能其实都是形式上的格式占了大部分码量，果断缩减后编译速度快了两倍多。在拆分功能的基础上，还需要对它们进行组合、聚类，这样才能在层次感的基础上使程序更加紧凑易读。
9. 最基础的自顶向下设计应该有上中下三个层次，最上层的就是与用户交互的界面，中层则是对各个函数的调用，组合。而下层（底层）就是对定义的数据结构进行管理。但是每一层内都可能有更丰富的层级，比如中层，本次代码复用率不高的重要原因就是中层庞大，各个函数功能互有交叉但是却处在平行位置，更糟糕的是整个中层甚至承担了大部分 UI 的工作。这点是我以后着重需要注意的。不过现在已经做到了底层与 UI 的分离，已是一大进步。接下来只需要注意中层的代码复用即可。
10. 最大的收获是自我解决问题的能力。在这次大作业中，我积极和成绩优秀的同学交流，受益匪浅，也看到了差距，而这个差距就是自我解决问题的积极性和行动力。
11. 让我意识到了熬夜伤身，不能熬夜写代码。需要保持最好的精力，才有最高的效率。

收获：

1. 对数据结构的初步认识。
2. 学习了部分编码规范。
3. 获取了设计制造实用软件的设计，编写，调试的丰富经验。

4. 熟悉了面向对象的设计模式。
5. 学会了基本的文件操作。
6. 熟练了一些基本操作如输入检查，非头插法链表的增删改查。

不足之处：

1. 对 STL 不熟练，只会简单使用顺序容器。
2. 对关联容器陌生。
3. 对泛型编程非常陌生。
4. 对 gdb 的命令行操作仍不熟悉。
5. 对 git 命令陌生。