**华中科技大学网络空间安全学院**

**程序设计综合课程设计**

**报告**

题目: 智慧校园之校园卡食堂应用模拟

班 级：

学 号：

姓 名：

成 绩：

指导教师：

**完成日期： 2022年3月16日**

**目 录**

**[一、系统需求分析 1](#_Toc5472)**

**[二、总体设计 4](#_Toc24865)**

**[三、数据结构设计 8](#_Toc28769)**

**[四、详细设计 13](#_Toc29508)**

**[五、系统实现 43](#_Toc35)**

**[六、运行测试与结果分析 59](#_Toc25249)**

**[七、总结 81](#_Toc707)**

**[八、参考文献 84](#_Toc25275)**

**[附录：程序使用说明 85](#_Toc31791)**

# 一、系统需求分析

该系统要解决的主要问题是根据数据文件和规定的数据说明，构建一个校园卡的模拟系统，实现校园卡管理、食堂应用、汇总分析的基本功能。

该系统的全局时间都是相同的，在进入三个模块前可进行修改。

1. 校园卡管理：可以进行校园卡的开户、销户、发卡、挂失、解挂、补卡操作，可以对已经开户的校园卡进行充值。提供以学号或姓名等单项数据的模糊匹配的校园卡账户查询和挂失、补卡等历史信息的回溯。需要支持读取数据文件进行批量操作的能力。可以满足10000人的校园卡管理，每位同学可以补卡100次；提供模糊匹配的姓名信息查询；形成操作日志；提供批量操作能力。
2. 开户：输入姓名和学号（不能与以往重复），进行开户。
3. 销户：输入学号进行销户。已经销户的信息不能再进行其他操作。
4. 发卡：输入学号，对刚开户的用户初次分配唯一卡号。
5. 挂失、解挂：输入学号，改变当前账户下校园卡的状态。
6. 补卡：输入学号，为当前学号账户分发新的校园卡，并将帐下其他校园卡设为挂失状态。
7. 卡号设置：卡号流水号从12346开始，每位同学按学号的顺序，流水号递增；补卡时，流水号同样递增。
8. 充值：输入学号，为账户充值。账户余额上限999.99元。
9. 模糊匹配搜索：？代表一个字符或一个汉字，\*表示多个字符或多个汉字，或代表空。
10. 食堂应用：有99个以内的食堂窗口，收费设备循环保存本窗口6万条最近的收费数据记录。消费时，可以设定当前消费窗口，对指定的校园卡进行收费；每次消费支付后，收费设备显示当天该窗口累计交易次数和累计收费金额。
11. 窗口：消费前可以选择1-99号窗口。
12. 窗口消费：输入消费的卡号和消费金额，系统检查该校园卡是否正常的卡，检查余额是否足够，然后进行实际的余额扣减；如果本次消费成功，则生成并保存具有完整信息项的消费记录，并显示当前该窗口在当天的累计消费次数和消费金额。
13. 窗口消费记录：食堂窗口收费设备单独记录本窗口的数据；每个窗口数据最多保存6万条，循环覆盖保存。
14. 密码输入（扩展）：如果发现当前时段的消费累计金额超过20元，需要提示输入校园卡的消费密码，密码输入正确后，对当前时段的累计消费金额清零处理。食堂消费时段为：每天有早中晚3个时段（7-9点，11-13点，17-19点）。校园卡密码默认值为“8888”。
15. 汇总分析：按时间递增顺序汇总各食堂窗口收费设备的消费记录。提供模糊的学号、姓名信息进行查询，对指定的学号统计给定起止日期内的累计消费金额。
16. 有序汇总：对当前各收费窗口的数据进行有序汇总，选择一种多路归并排序和其他的排序算法进行实现。
17. 查询：可以组合时间段、学号、姓名、消费金额范围对消费记录信息进行查询。
18. 统计：可以统计规定日期内固定学号的消费次数和消费金额。
19. 分析（扩展）：在汇总的数据中，给定一位同学的学号，可以分析出哪些同学与该同学经常一起在食堂排队购餐。
20. 校验（升级）：消费记录汇总后，将会被保存到数据文件中。设计数据的校验码，克服充值和消费数据被简单篡改，且校验码规则过于简单而被破解的情况。并提出和实施一个能够检测出某些充值和消费数据行被直接删除或伪造增加的技术方案。
21. 运行时间计算：需要计算汇总排序所耗费的时间、每次组合查询耗费的时间。

**二、总体设计**

整个校园卡模拟系统（CampusCard）总体上分为四个部分设计。

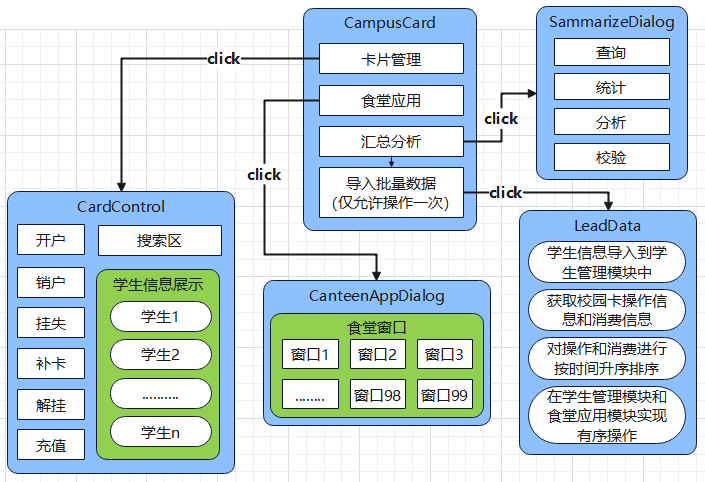


图2-1 校园卡模拟系统总设计框架

第一部分是学生信息及校园卡的管理模块。

学生的信息、学生校园卡的信息以及学生的开户、销户、挂失、解挂、补卡、充值等功能的具体实现函数由一个类（CardOpration）来存放管理。

这部分功能的可视化实现存放在类（CardControl）中，所建窗口提供开户等功能的按键，按键的click事件能够响应信息输入窗口，完成信息输入后则调用CardOpration中对应功能的完成操作信息的导入。

窗口中还提供学生信息的展示功能，以及根据学号或姓名搜索具体学生并展示的功能。这两功能的实现都依赖于CardOpration中所存储的学生信息来实现。

为具体的展示学生的历代校园卡号，在学生信息展示的表中，可以提供双击信息行展示某个学生的卡信息的功能。

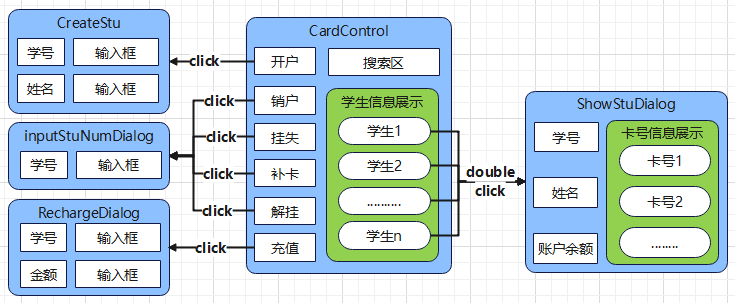


图2-2 学生信息管理系统框架

第二部分是食堂应用模块。

食堂窗口的消费功能、1-99号窗口的信息和消费记录、具体时段超金额的密码输入功能存放在一个类（CanteenOpration）中。另外为方便频繁的获取由“,”隔开的语句，其中也可以提供仿python的字符串切片功能。

食堂99个窗口的可视化由一个类（CanteenAppDialog）实现，双击窗口实现具体窗口功能的可视化，可查看消费日志和输入卡号进行消费操作。

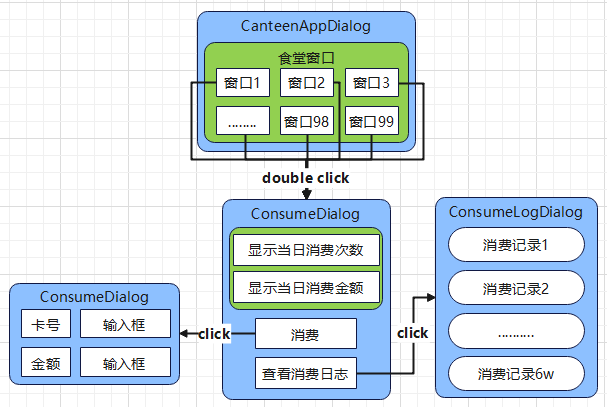


图2-3 食堂应用模块系统框架

第三部分是食堂总消费记录的汇总分析模块。

汇总分析模块首先从食堂应用模块获取99个窗口各自所保存的6w条消费记录，进行归并汇总，得到按照时间有序的消费记录，这部分功能的实现、对其它功能的选择、有序记录的保存由一个类（SammarizeDialog）封装。

该模块需要实现的查询、统计、分析、校验四个功能及各自的可视化分别存放在四个类（SearchDialog、StatisticsDialog、AnalyzeDialog、CheckFileDialog）中。

查询功能实现搜索并显示由学号、姓名、时间、消费金额四个条件组合限定的消费记录；统计功能实现搜索并显示固定学号对象在限定日期内的消费次数和消费金额；分析功能实现搜索并显示经常与固定学号对象一起排队消费的对象信息；校验功能实现检测保存有序消费序列的文件是否遭到修改并显示出被修改的地方。

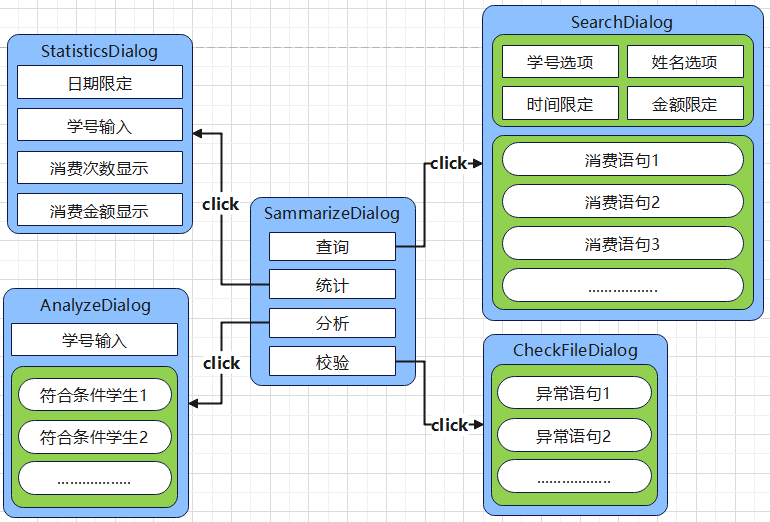


图2-4 汇总分析模块系统框架

第四部分主要是实现是批量数据的导入。

为满足模拟系统可导入批量操作文件操作的功能，提供类（LeadData）实现学生信息(students.txt)、学生校园卡操作信息(cz.txt)、消费信息(xf.txt)项学生信息管理系统和食堂应用管理系统的导入。

**三、数据结构设计**

1. Student：学生信息结构体，主要使用在学生信息管理模块的类CardOpration中的（map<string,Student>）students成员中，以学生学号(string)为key，学生信息(Student)为value，存放所有导入系统内的学生信息。其中，Student的成员cardhead是对应学生的所有校园卡卡号的链表的头节点，学生正在使用中的校园卡卡号或最新的校园卡卡号总会是该链表的首节点。

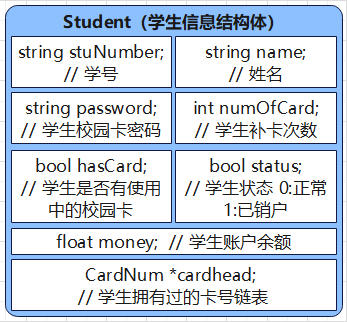


图3-1 学生信息结构体

1. CardNum：卡号链表结点结构体，作为Student结构体的成员使用，其中保存的仅是校园卡的卡号，没有校园卡的其他信息。

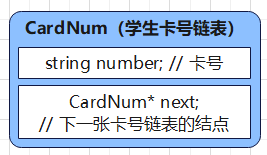


图3-2 学生卡号结构体

1. Card：校园卡信息结构体，同样主要使用在学生信息管理模块的类CardOpration中的（map<string, Card>）cards成员中，以校园卡卡号为key，校园卡信息为value，存放已经发放过的校园卡信息。

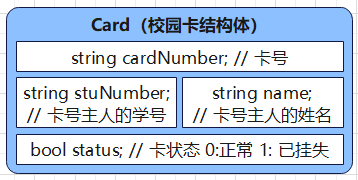


图3-3 校园卡结构体

1. Win：食堂窗口结构体，主要使用于CanteenOpration的（Win）win[99]成员中，存储1-99号窗口的部分信息。

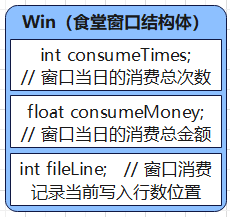


图3-4 食堂窗口结构体

1. winXfStr：食堂窗口消费记录信息链表结点结构体，主要用于食堂应用模块来保存每个食堂窗口的6w条消费记录。每个食堂窗口都有一个由该结点构成的双向循环链表。

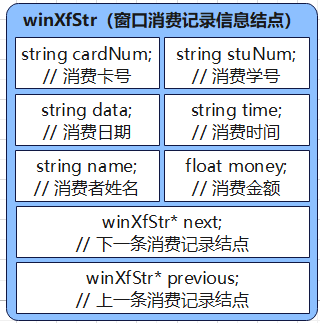


图3-5 窗口消费记录信息节点结构体

1. winXfHead：食堂窗口消费记录链表的“头结点”，主要使用于食堂应用模块的CanteenOpration类中的（map<int, WinXfHead>）m\_winXfRecords成员中，以窗口号-1为key，窗口消费记录的头节点为value，确定保存记录的双向循环链表的写入位置和信息数量。这里所谓的“头节点”即，winXfHead中的成员now总是指向对应记录链表的下一个信息写入位置的结点。

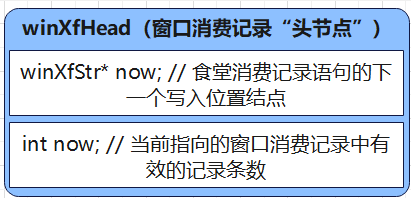


图3-6 窗口消费记录“头节点”结构体

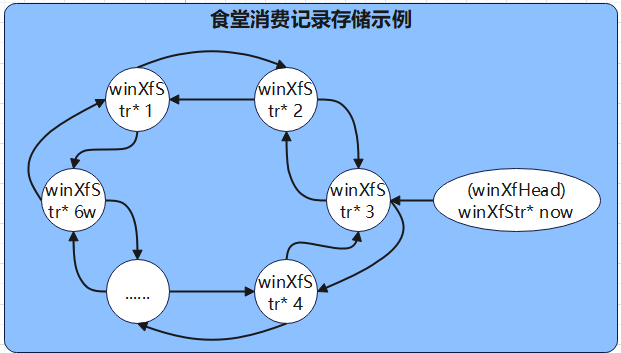


图3-7 窗口消费记录存储示意图

1. opLogHead：操作日志语句“记录节点”结构体，主要用于保存操作日志语句链表的头节点和尾节点。既方便从头读取整个操作日志语句，又方便利用尾插法加入操作日志语句。

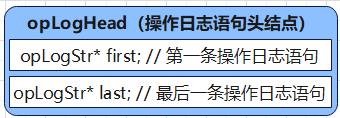


图3-8 操作日志语句头节点结构体

1. OpLogStr：操作日志语句链表节点结构体，每个节点中仅保存一句完整的操作语句和下一个节点的指针。

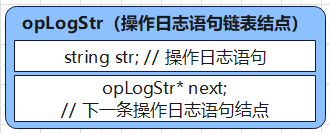


图3-9 操作日志语句链表节点结构体

1. sortedXf：食堂所有窗口的有序消费记录信息的链表节点结构体，每个节点中保存消费记录中包含的各种信息以及下一条记录的节点指针。

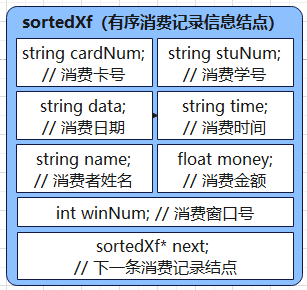


图3-10 有序消费记录信息节点结构体

1. xfStr：消费语句的链表结构体，这里的消费语句是指从xf.txt文件中读取的一条记录语句。

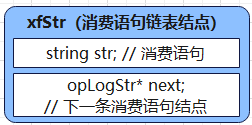


图3-11 消费语句链表节点结构

**四、详细设计**

1. 学生信息及校园卡管理系统模块

该模块实现的主要功能是有关学生的一些操作功能，开户、销户、挂失卡、解挂卡、补卡、账户充值等。

功能的主要操作对象是该模块的类CardOpration中的成员变量（map<string, Student>）students和（map<string, Card>）cards。

实现1-7功能的是CardOpration类中的成员函数，实现8-10功能的是CardControl类中的成员函数。

1. 学生开户功能（createStu）

该功能需要提前传入参数：date（日期）、time（时间）、studentNum（学生学号）、name（学生姓名）。

创建学生的成员信息的填入包含：name、stuNumber、money、status、password、numOfCard、hasCard、cardhead的初始化。

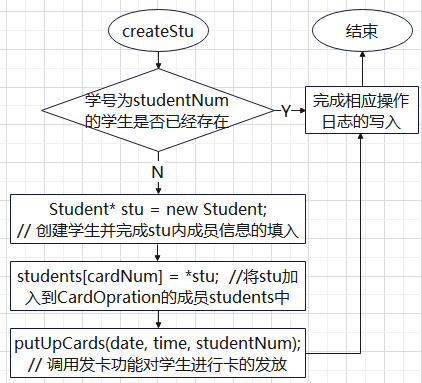


图4-1-1 开户功能函数流程图

1. 学生发卡功能（putUpCards、addCard）

发卡功能函数putUpCards需要提前传入参数：date（日期）、time（时间）、studentNum（学生学号），该功能仅在学生开户成功的同时调用，完成卡的发放。

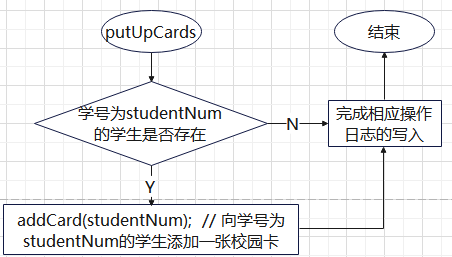


图4-1-2 发卡功能函数流程图

发卡的关键步骤是添加学生卡，该部分由addCard函数实现，传入参数有：studentNum（学生学号）。

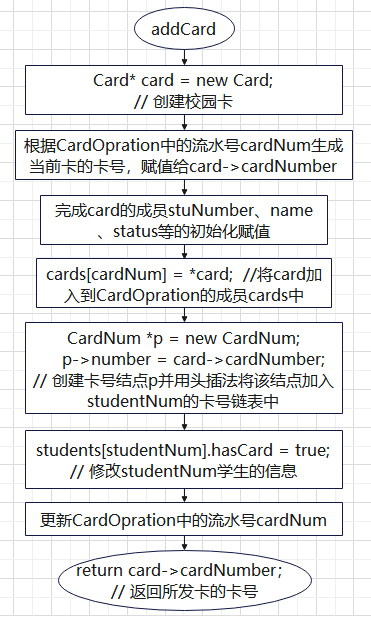


图4-1-3 addCard函数流程图

1. 学生销户功能（deleteStu）

该功能需要提前传入参数：date（日期）、time（时间）、studentNum（学生学号）。

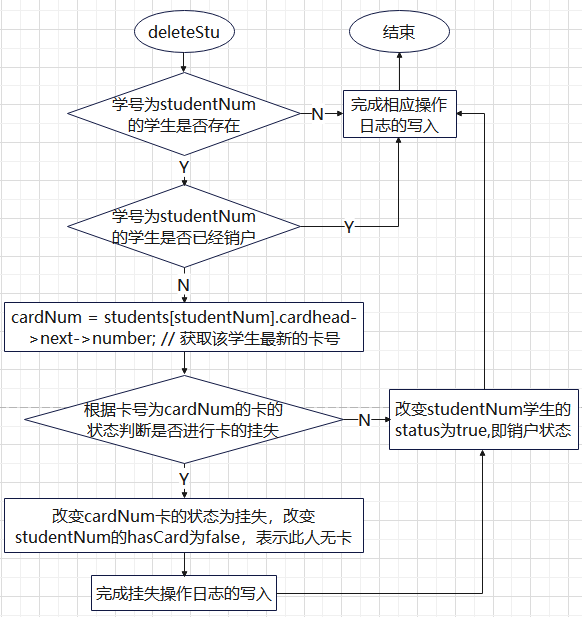


图4-1-4 销户功能函数流程图

1. 学生挂失功能（lossSolution\_1）

该功能需要提前传入参数：date（日期）、time（时间）、studentNum（学生学号）。

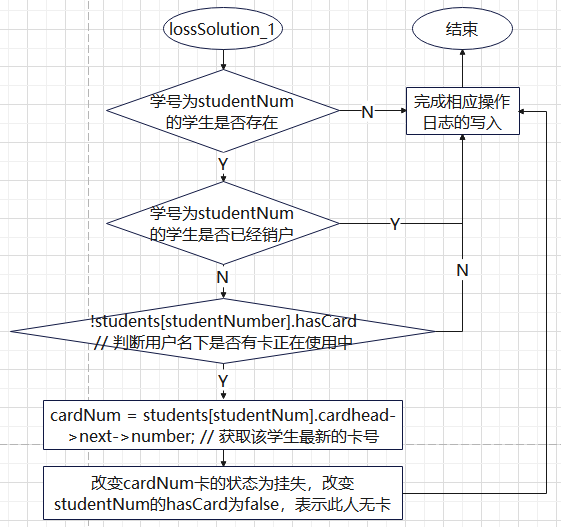


图4-1-5 挂失功能函数流程图

1. 学生解挂功能（lossSolution\_2）

该功能需要提前传入参数：date（日期）、time（时间）、studentNum（学生学号）。

该操作流程和挂失流程十分相似，不同点只在于对studentNum学生是否有卡处于使用中所做出的操作。挂失功能是有卡使用则挂失成功，否则失败；解挂功能是无卡使用则可解挂最新卡，否则失败。

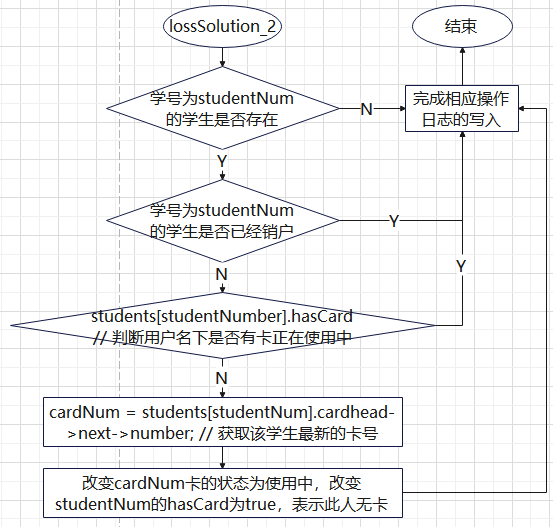


图4-1-6 解挂功能函数流程图

1. 学生充值功能（recharge）

该功能需要提前传入参数：date（日期）、time（时间）、studentNum（学生学号）、addMoney（充值金额）。

学生账户金额有上限MAXMONEY，账户的总金额不能超过这个值。

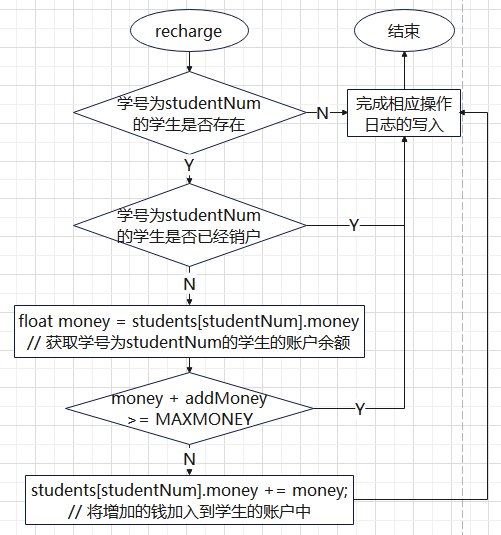


图4-1-7 充值功能函数流程图

1. 学生补卡功能（makeUpCard）

该功能需要提前传入参数：date（日期）、time（时间）、studentNum（学生学号）。

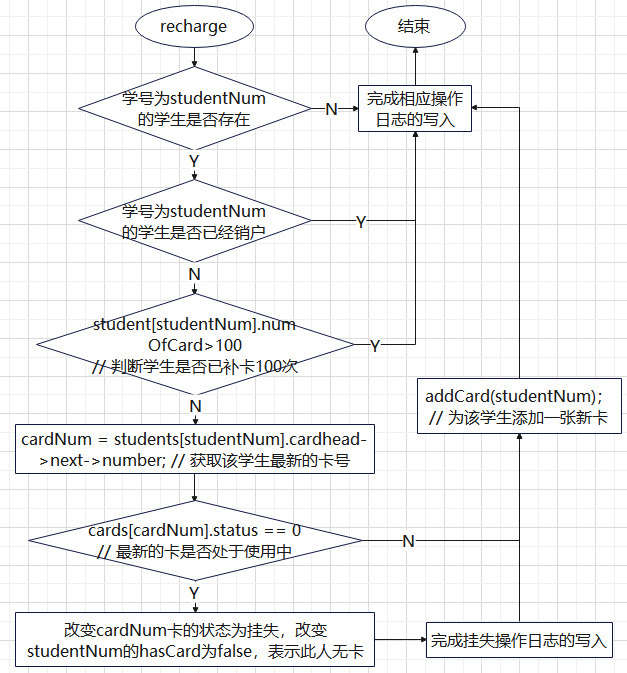


图4-1-8 补卡功能函数流程图

1. 根据学号模糊查询学生的功能主要实现函数（matchWithNum）

该功能需要提前传入参数：stuNum（待比对的学号，仅包含数字）、input（输入的内容，仅包含数字、\*和?），实现单个人学号与输入内容的匹配。

该功能的模糊搜索功能，大致实现为：在input中，\*号代表0到多个字符，?代表一个字符，在此基础上若stuNum和input能匹配成功则返回true，否则返回false。

此功能的实现关键在于maski，maskj的使用，即当遇见\*号时，他们记录下当时i，j的位置，之后的匹配遵循着\*匹配0-多个字符逐一比对，看是否能匹配成功。

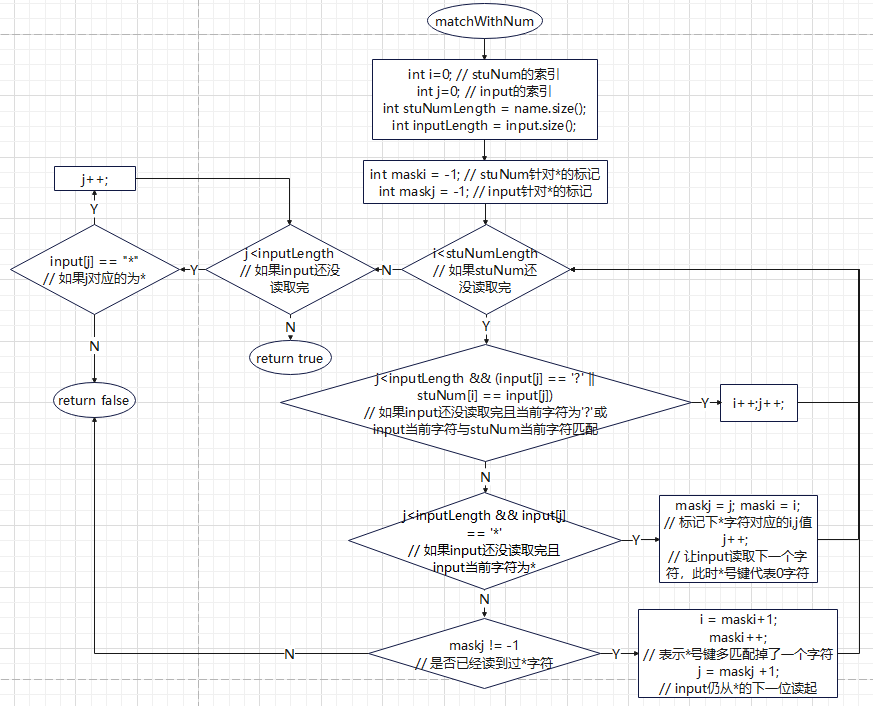


图4-1-9 学号模糊匹配功能函数流程图

1. 根据姓名模糊查询学生的功能主要实现函数（matchWithName）

该功能需要提前传入参数：name（待比对的姓名，仅包含汉字）、input（输入的内容，仅包含汉字和?），实现单个人姓名与输入内容的匹配。

该功能的模糊搜索功能，大致实现为：如果input中包含?，则?仅代表一个字，且input和name只能一对一完整比对，比对成功返回true，否则返回false；如果input中不包含?，则只需input是name的子串即可比对成功返回true，否则返回false。

且需注意，name.find(input)%2==0的意义在于避免出现两个汉字的四字节编码的中间两字节编码恰好对应一个汉字编码的错判情况。

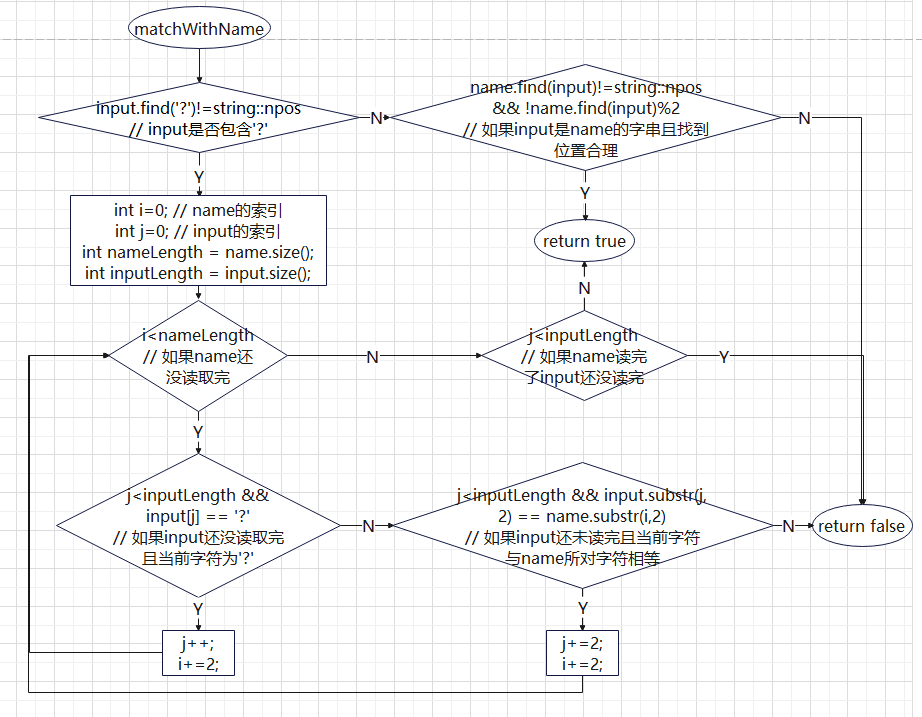


图4-1-10 姓名模糊匹配功能函数流程图

1. 在8、9的的基础上实现的完整模糊搜索功能（on\_searchButn\_clickde）

此功能函数在对应的搜索按钮searchButn触发click时调用。

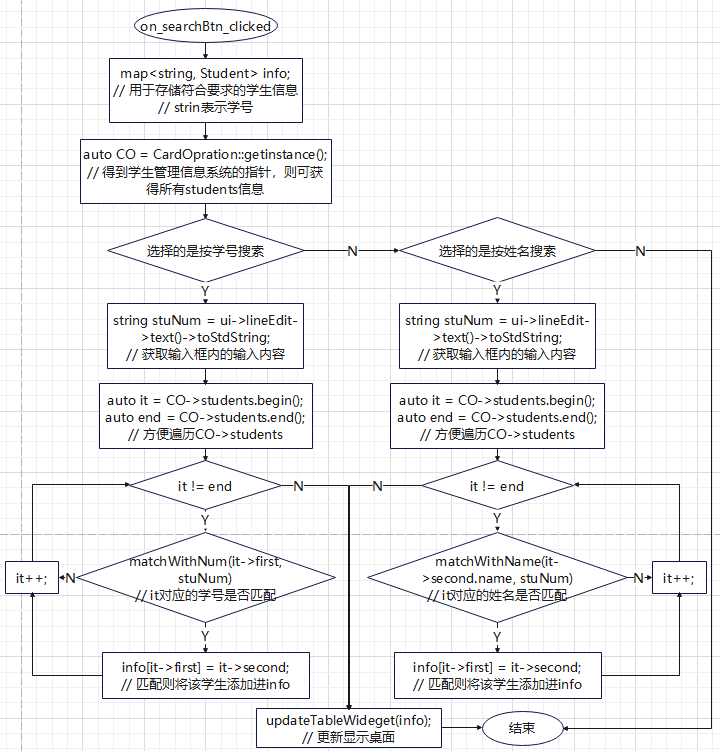


图4-1-11 搜索功能函数流程图

1. 食堂应用管理模块

该模块实现的主要功能是消费功能和消费日志功能。

消费功能的操作对象是学生信息管理模块中类CardOpration中的成员变量（map<string, Student>）students和（map<string, Card>）cards，以及该模块的类CanteenOpration中的成员（Win，消费窗口）win[99]、（string，日期）m\_date、（string，时间）、（char，时段）m\_timeFrame。

消费日志功能的操作对象是该模块的类CanteenOpration中的存储窗口消费记录的成员变量（map<int,winXfHead>）m\_winXfRecords。

以下功能函数和辅助函数都是CanteenOpration类中的成员函数。

1. 初始化窗口消费记录记录成员的函数（initWinXfRecords）

窗口消费记录的记录成员是（map<int,winXfHead>）m\_winXfRecords，它的key为窗口号-1，winXfHead是自定义结构体，具体见3-5和3-6内容及其图示。

WINNUM = 99；FILELINES = 60000 （宏定义）

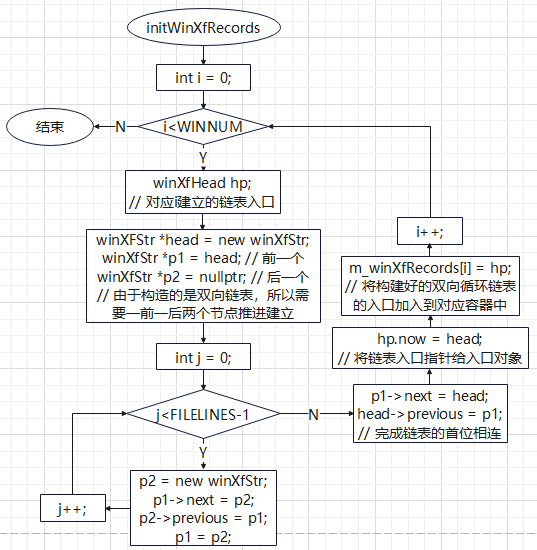


图4-2-1 initWinXfRecords函数流程图

1. 窗口消费记录的写入函数（writeXfRecords）

此函数参数有：winNum（int，窗口号-1）、cardNum（string，卡号）、date（string，日期）、（time，时间）、（conMoney，消费金额）。

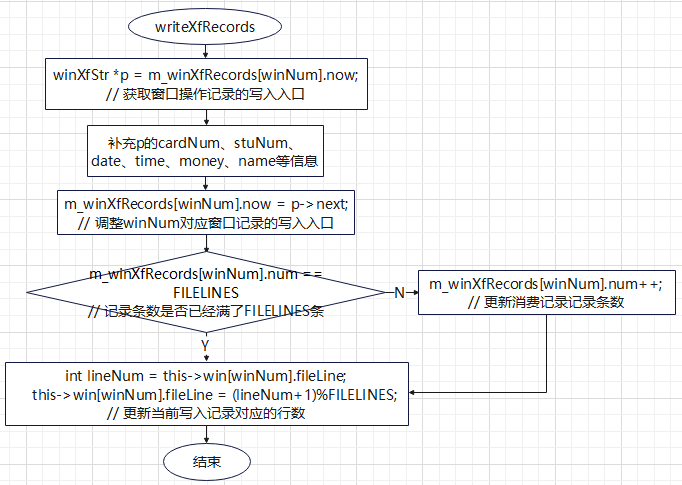


图4-2-2 writeXfRecords函数流程图

1. 日期的更新函数（updateTime）

此函数参数有：isPl（bool，是否是批量操作）、date（string，日期）、time（string，时间）。

当isPl为true时，表示批量消费操作的日期更新，不考虑时段的更新；当isPl为false时，表示个人消费操作的日期更新，需要考虑时段的更新。

注意，此处时间与日期时间组件（QDateTime）上记录的不同，组件上的时间是真实的时间流水，此处的date、time更新只是为了消费时间不回退，以及更新时段、控制个人消费的密码输入。

此中涉及的updateWin函数是CanteenOpration类的成员函数，用于将所有窗口（win[0-98]）记录的当日消费次数和消费金额清零。

（map<string，float>）m\_csMoneyInFrame是CanteenOpration类的成员变量，它的key是学生学号，value是在某个时段内该学号对应学生消费了的金额，且每成功输入一次密码该学生对应的金额会清零；该变量在每次时段更新成功时都会清空。

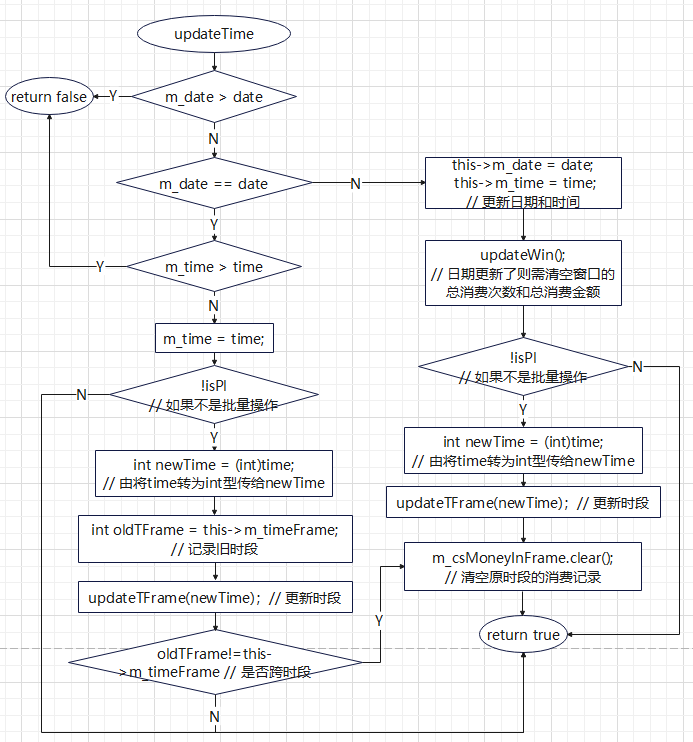


图4-2-3 updateTime函数流程图

1. 时段的更新函数（updateTFrame）

该函数的参数有：time（int，新时间）。

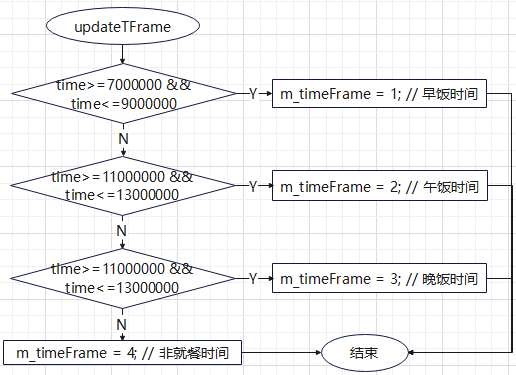


图4-2-4 updateTFrame函数流程图

1. 消费功能的实现（consume）

消费功能的主体实现函数为consume，其参数有：isPl（bool，是否是批量操作）、winNum（int，窗口号-1）、cardNum（string，卡号）、conMoney（float，消费金额）、date（string，日期）、time（string，时间）。

该消费功能是批量消费和个人消费的公用功能，通过传入参数isPl的值进行区别对待。当isPl为true时为批量消费的处理；当isPl为false时为个人消费的处理，此时比批量处理复杂些，需要考虑时段和时段内学生的消费记录的更新，来控制消费密码的输入与否。

由于该函数涉及处理较长，故将有关个人消费的密码输入的操作另作一流程图演示。

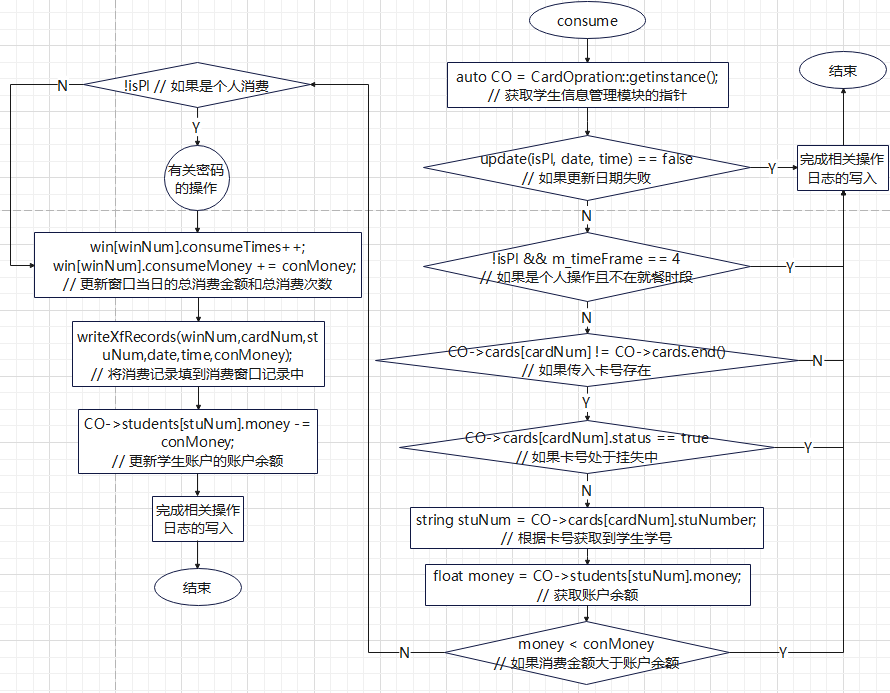


图4-2-5 消费功能函数流程图

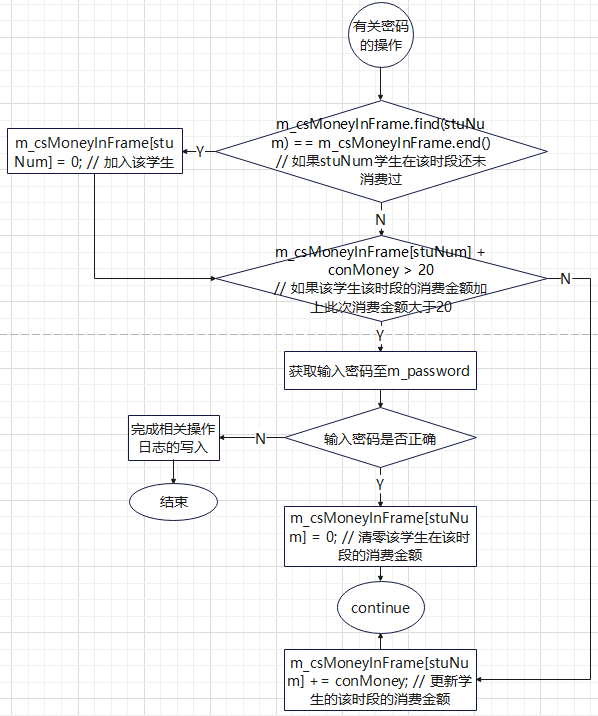


图4-2-5 消费功能函数流程图（补）

1. 汇总分析模块

该模块的实现的主要功能是对99个食堂窗口的消费记录的分析，具体需要实现的功能有：组合学号、姓名、日期、消费金额对消费记录进行搜索的功能；对固定学号限定日期计算其在这个时间段的消费金额的统计功能；针对某学生分析出经常与其一起就餐的同学的信息分析功能；预防并检测存进txt文件的消费记录是否被篡改的校验功能。

这些功能的实现之前，需要获取窗口的消费记录并将其进行排序归并。这项工作的完成在SummarizeDialog内完成，其他四个功能的调用也在该类中得以实现，但他们的具体实现都在各自的类中完成：SearchDialog（搜索功能）、StatisticsDialog（统计功能）、AnalyzeDialog（分析功能）、CheckFileDialog（校验功能）。

SearchDialog类中的成员变量主要有：（CanteenOpration\*）m\_COP（食堂管理系统的类指针，在SearchDialog的构造函数内便完成了他的初始化，指向食堂管理类的全局指针变量，其中有食堂窗口消费记录等重要成员变量） 、 是（map<int, pair<int, winXfStr\*> >）m\_allReacords（key为窗口号-1，value是对应窗口的消费记录信息，value的第一个值是该号窗口的记录条数，第二个值是指向该号窗口记录的最后一条写入记录的指针） 、 （sortedXf\*）m\_sortedRecords（所有窗口消费记录归并后有序的记录的存放链表的头节点）。

说明一下，多次提到窗口号-1，是因为窗口的下标总是比窗口号少1的，1-99号窗口对应下标是0-98。

1. 对成员变量m\_allRecords的初始化函数（init）

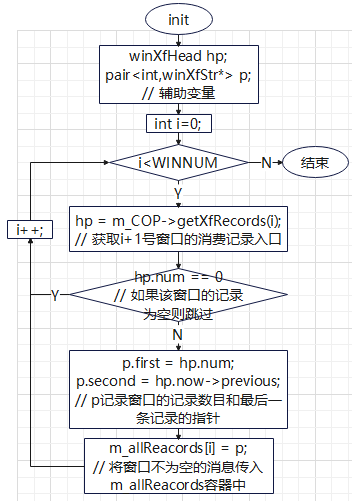


图4-3-1 init函数流程图

1. 所有窗口消费记录的归并排序同时赋值m\_sortedRecords的功能（sortWinXf、insert\_vec）

SortWinXf函数是99个窗口消费记录归并排序的主体函数，主要利用循环每次从多路尾部挑选出时间最大者，利用头插法插入到最后的有序消费记录链表中，最后得到按时间升序的有序消费序列。

insert\_vec是辅助函数，其参数有：（int，窗口号-1）i、（string，日期时间字符串）str、（vector<pair<int, string>>&, 需要插入修改的容器）vec。实现将一个值折半插入到有序的vec容器中的功能，便于挑选出多路尾元素中的最大者。

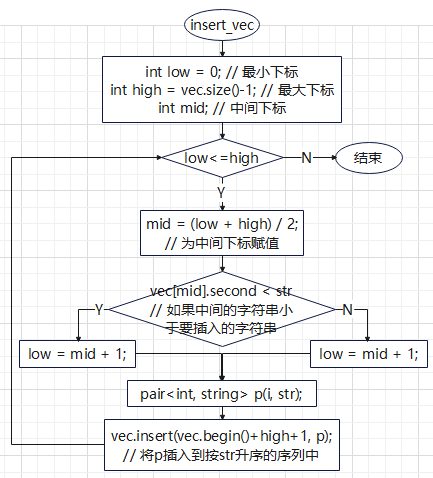


图4-3-2 insert\_vec函数流程图

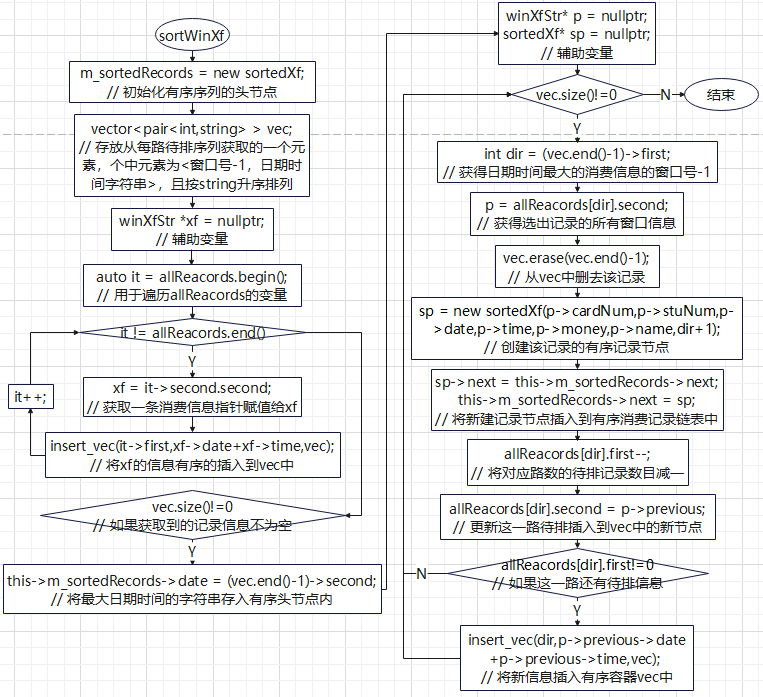


图4-3-3 SortWinXf函数流程图

1. 搜索功能实现说明

搜索功能的实现在类SearchDialog中完成，主要操作对象为从SummarizeDialog类中获取的m\_sortedRecords（有序消费记录链表的头节点）。

类SearchDialog还有成员变量（sortedXf\*）m\_result，是用于存放符合要求的消费记录信息的链表表头。

搜索功能主要实现过程为：

如果选择了时间限定，则根据限定的初始时间定位到m\_sortedRecords恰当的日期处，然后从这一记录节点开始，遍历链表，将之后节点的记录和其他三个选择的限定条件进行比对，符合的则创建一个有相同消费信息的节点用头插法添加到m\_result中，直至搜索到节点的消费信息时间大于限定的结束为止。 如果没有限定时间，则是给我根据其他三个限定的选择对整个m\_sortedRecords链表遍历，寻找符合要求的消费节点，将其信息添加到m\_result中。注意，如果四个限定都没有选择，则不会开始搜索操作。

最后将m\_result信息显示到可视化的tableWidget上。

1. 统计功能实现说明

统计功能的实现在类StatisticsDialog中完成，主要操作对象为从SummarizeDialog类中获取的m\_sortedRecords（有序消费记录链表的头节点）。

统计功能主要实现过程为：

首先定义变量：（int，消费总次数）allTimes=0、（float，消费总金额）allConMoney=0，然后判断输入的学号是否存在在学生信息管理系统中的students容器中，如果不存在，则结束接下来的操作。

如果存在，则根据时间限定，对m\_sortedRecords链表遍历，找到时间大于等于限定的初始时间的节点，从这里开始，对接下来消费记录节点信息内的学号和输入的学号比对，如果一致，则allTimes++、allConMoney加上这个节点内的消费金额，到消费记录节点的时间大于限定的结束信息为止。

最后显示allTimes和allConMoney到该功能对应的窗口显示组件上。

1. 分析功能实现说明

分析功能的实现在类AnalyzeDialog中完成，主要操作对象为从SummarizeDialog类中获取的m\_sortedRecords（有序消费记录链表的头节点）和m\_allReacords（99个窗口的消费记录入口对象的集合）。

分析功能的主要操作其实也是遍历两个m\_sortedRecords和m\_allReacords这两个成员变量，从两种不同的方式搜查到经常与输入学号对应的学生在食堂消费的学生信息。

最后的搜索结果保存进成员变量m\_info（map<string(学号字符串), int(记录次数)>）中。

首先考虑与输入学号对应的学生同窗口排队的学生。遍历m\_allReacords，从1-99号窗口，每个窗口都从记录的入口对象所指的食堂消费记录结点开始，向previous方向读取至所有有效记录都读取完毕，若结点的学号信息与输入学号一致，则将该结点的previous和next结点学生信息存进m\_info中。

然后考虑与输入学号对应的学生相邻窗口相近时间排队消费的学生。遍历m\_sortedRecords，由于该节点所指链表为单链表，不方便索引到目标节点的前一个节点，故每次都是读取到三个节点信息，分别为pre\_p、p、next\_p，从关系为pre\_p->next==p，p->next==next\_p，从m\_sortedRecords为pre\_p开始，至next\_p为空结束。若p对应节点的学号为输入学号，则将pre\_p和next\_p节点对应的学生加入m\_info中。当next\_p为空后，还需要进行一次判断，若此时p符合要求，需要将pre\_p对应的学生加入到m\_info中。

完成两次遍历后需要对m\_info进行一次筛选，将key为空字符串或学号为输入学号的成员删除。

搜索分析完成后，调用updateTable函数，将m\_info中保存的学生的信息显示到该窗口界面的表组件上。

1. 校验功能实现说明

检验功能主要分为两部分，一部分为防止保存有序消费记录的数据文件all\_xf.txt中的单独行数据被恶意篡改，一部分为检测出数据是否遭到行删减或行增加修。

前部分在SummarizeDialog类中实现，在该类中完成消费记录的排序后，会遍历记录链表，将每个节点中的日期、时间、消费卡号、消费金额信息整合，进行语句后加密再写入同目录下的数据文件all\_xf.txt中，从而使试图篡改具体信息的人无法看懂文件内容，无法带目地修改消费内容。

加密流程如下图所示（其中，num表示一个0-2的随机数，str为待加密语句；Y=24，S=18，F=5，ALLLENGTH=26，KEYLENGTH=3(宏定义)）：

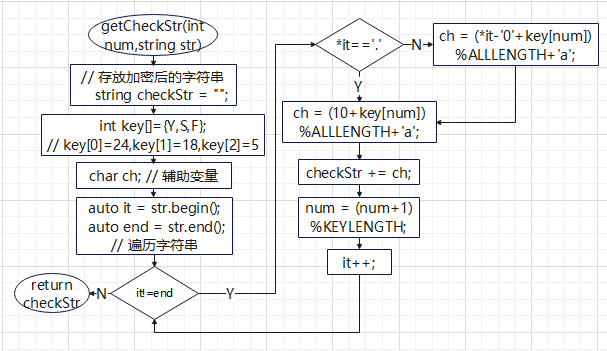


图4-3-4 语句加密函数流程图

后部分功能在类CheckFileDialog中实现，主要操作对象为从SummarizeDialog类中获取的m\_sortedRecords（有序消费记录链表的头节点）。

功能实现也是依赖于遍历比对，打开数据文件all\_xf.txt，与m\_sortedRecord所指链表的首节点开始，同时遍历进行比对。先对文件中读取到的一行数据进行解密，获取其中消费信息，与链表节点所指信息比对，若比对吻合，则继续往后遍历；若不吻合，则需考虑到增加和删减两种可能。 若是删减，则文件中的语句，在当前节点的后续节点中，一定有与其吻合的节点信息。则链表继续往后读，文件不动，若找到吻合的节点信息，则从不吻合处起至吻合处前的节点信息均是被删除的信息，保存进deleteVec（vector<string>）中；若链表读至尾部还未找到吻合节点信息，则该条文件语句是被增加的，存进addVec（vector<string>）中，链表读取位置重新定位到不吻合位置，文件读取下一行。如此读取，直至文件和链表有一方读取完毕为止。

最后需要处理尾部信息，若是链表读完而文件未读完，则文件中的剩余语句均是被增加的信息，存进addVec；若是链表未读完而文件读完，则链表的剩余节点信息就是被删除的信息，存进deleteVec中。

最后调用updateTable函数将addVec和deleteVec中的信息全部显示到该窗口界面的表组件上。

1. 批量数据的导入模块

该模块的实现类是LeadData，主要实现的就是将测试文件中的信息导入到整个系统中。主要信息有：students.txt文件中的学生序号和姓名信息、cz.txt文件中学生的相关操作（补卡、挂失、销户等）、xf.txt文件中的食堂消费信息。

未导入数据，在LeadData中设置了辅助成员变量：（xfStr\*）m\_opration[WINNUM+1]（0-98用于存储99个窗口的消费语句信息的链表头节点，99用于存储学生的操作信息的链表头节点）；（xfStr\*）m\_sortedOp（根据时间升序排序后的消费和操作语句信息的链表头节点）。

WINNUM = 99（宏定义）。

1. students.txt内的信息读取和导入

此部分由LeadData的成员函数initStu实现。

该部分实现简单：首先获得学生管理类（auto CO = CardOpration::getinstance()）和全局日期时间类（auto DT = SetDateTime::getinstance()）的指针；再打开students.txt文件,利用while循环逐行读取，每读取一行，则获取相应姓名（name）和学号（stuNumber），同时从DT中获取日期（date）和时间（time），调用CO内的开户函数，开户一名学生（CO->createStu(date,time,stuNumber,name)），直至文件读取完毕。

1. cz.txt和xf.txt内的信息读取

此部分由LeadData的成员函数getOpration实现。

此前需要先初始化好成员变量m\_opration[WINNUM+1],即m\_opration[0-99] = new xfStr，m\_opration[0-99]->next = NULL，m\_opration[0-99]->str = 窗口号的字符串形式。

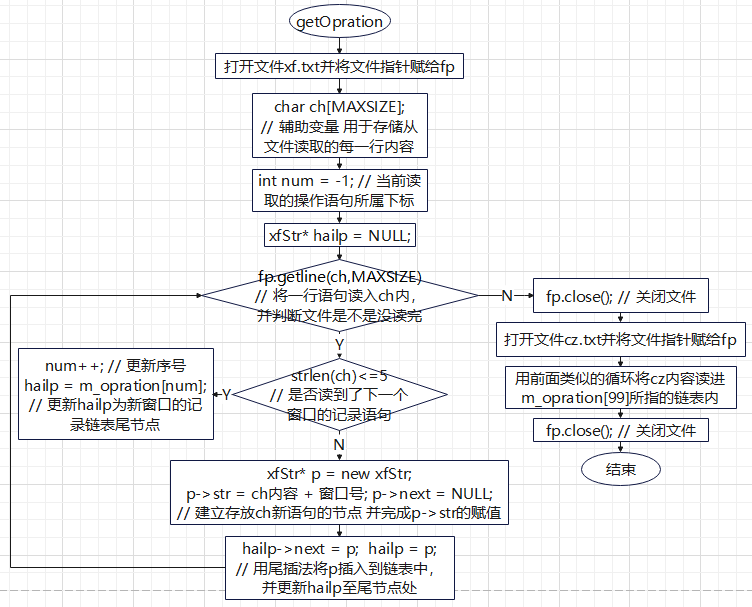


图4-4-1 getOpration函数流程图

1. cz.txt和xf.txt信息的排序

此处排序即对m\_opration[0-99]这100路链表内的语句根据其中的日期时间信息进行多路归并排序，有序信息存在以m\_sortedOp为头节点的链表中。此处不多赘述，这里多路归并的算法与汇总分析模块中所述的归并功能算法一致。

1. cz.txt和xf.txt信息的导入（操作实现）

实现操作大致为：

首先获得学生管理类（auto CO = CardOpration::getinstance()）和全局食堂应用类（auto COP = CanteenOpration::getinstance()）的指针。

然后遍历有序的m\_sortedOp链表，每读一个结点就根据其中的信息进行相关操作。结点的语句中包含“充值”，则利用其中信息调用学生管理体系中的充值功能（CO->recharge(参数....)），挂失、解挂、销户、补卡等类似。如果语句结点都不包含以上功能文字，包含窗口号，则调用食堂应用中的批量消费功能（COP->consume（true，参数....） // 传入true给isPl，说明此处进行的是批量消费操作）。

**五、系统实现**

1. 该程序是在Qt5.14.2的平台上搭建的。
2. 调用到的库有：
3. #include <iostream>
4. #include <iomanip>
5. #include <QDialog>
6. #include <QWidget>
7. #include <QApplication>
8. #include <QObject>
9. #include <QTableWidget>
10. #include <QAbastractItemView>
11. #include <QStringList>
12. #include <QDir>
13. #include <QDateTime>
14. #include <QTimer>
15. #include <QMessageBox>
16. #include <QTextCodec>
17. #include <QRegExpValidator>
18. #include <fstream>
19. #include <sstream>
20. #include <map>
21. #include <set>
22. #include <qabstractspinbox.h>
23. 主要函数原型
24. 主窗口模块(CampusCard)

//========================================================

// 功能：该类窗口界面上的退出按钮的click()信号的槽函数， // 关闭窗口界面

// 输入参数：空

// 返回值：空

//=========================================================

void on\_quit\_clicked();

//========================================================

// 功能：该类窗口界面上的学生管理按钮的click()信号的槽函数， // 打开学生管理的窗口界面

// 输入参数：空

// 返回值：空

//========================================================

void on\_CardControl\_clicked();

//========================================================

// 功能：该类窗口界面上的食堂应用按钮的click()信号的槽函数， // 打开食堂应用的窗口界面

// 输入参数：空

// 返回值：空

//========================================================

void on\_CanteenApp\_clicked();

//========================================================

// 功能：该类窗口界面上的汇总分析按钮的click()信号的槽函数， // 打开汇总分析的窗口界面

// 输入参数：空

// 返回值：空

//========================================================

void on\_Summary\_clicked();

//========================================================

// 功能：该类窗口界面上的导入批量数据按钮的click()信号的槽 // 函数，选择是否导入批量数据

// 输入参数：空

// 返回值：空

//========================================================

void on\_inputAllBtn\_clicked();

//========================================================

// 功能：该类窗口界面上的导入批量数据按钮的click()信号的槽 // 函数，选择是否导入批量数据

// 输入参数：空

// 返回值：空

//========================================================

void on\_inputAllBtn\_clicked();

//========================================================

// 功能：将操作日志语句链表内的内容全部写进同目录下的 // oplog.txt文件中

// 输入参数：空

// 返回值：空

//========================================================

void writeOpLog();

//========================================================

// 功能：调用批量数据导入类LeadData中的函数来导入文件中的 // 批量操作数据

// 输入参数：空

// 返回值：空

//========================================================

void updateData();

1. 学生信息管理模块(CardOpration)

//========================================================

// 功能：实现学生的开户功能

// 输入参数：日期；时间；学号；姓名

// 返回值：空

//========================================================

void createStu(string date,string time,string studentNum, string name);

//========================================================

// 功能：实现学生的销户功能

// 输入参数：日期；时间；学号

// 返回值：空

//========================================================

void deleteStu(string date, string time, string studentNum);

//========================================================

// 功能：对学生校园卡的挂失功能

// 输入参数：日期；时间；学号

// 返回值：空

//========================================================

void lossSolution\_1(string date, string time, string studentNum);

//========================================================

// 功能：对学生校园卡的解挂功能

// 输入参数：日期；时间；学号

// 返回值：空

//========================================================

void lossSolution\_2(string date, string time, string studentNum,string cardNum = "");

//========================================================

// 功能：对学生校园卡的补卡功能

// 输入参数：日期；时间；学号

// 返回值：空

//========================================================

void makeUpCard(string date, string time, string studentNum);

//========================================================

// 功能：对学生校园卡的充值功能

// 输入参数：日期；时间；学号；充值金额

// 返回值：空

//========================================================

void recharge(string date, string time, string studentNum, float addMoney);

//========================================================

// 功能：向对应学号的学生名下添加一张卡片，并将该卡添加到学 // 生信息系统的cards容器中

// 输入参数：学号

// 返回值：被发放卡片的卡号

//========================================================

string addCard(string studentNum);

//========================================================

// 功能：对新开户的学生进行发卡的功能

// 输入参数：日期；时间；学号

// 返回值：空

//========================================================

void putUpCards(string date, string time, string studentNum);

1. 学生信息管理窗口模块(CardControl)

//========================================================

// 功能：学生学号的模糊匹配，input为输入内容（其中\*代表任意 // 0-n个字符,?代表任意一个字符），stuNum为待比对的学 // 号，进行input和stuNum的匹配并返回匹配结果

// 输入参数：待比对学号；输入内容

// 返回值：匹配成功与否

//========================================================

bool matchWithNum(string stuNum, string input);

//========================================================

// 功能：学生姓名的模糊匹配，input为输入内容（其中?代表任意 // 一个字符），name为待比对的姓名，进行input和stuNum // 的匹配并返回匹配结果

// 输入参数：待比对学号；输入内容

// 返回值：匹配成功与否

//========================================================

bool matchWithName(string name, string input);

//========================================================

// 功能：更新该类的窗口界面的学生信息显示组件stuTable中所 显示的信息

// 输入参数：需要显示到表中的学生信息的键值对集合

// 返回值：空

//========================================================

void updateTableWideget(map<string,Student> students);

//========================================================

// 功能：根据学号勾选框的选择更新输入框、搜索按钮和姓名勾选 // 框的设置

// 输入参数：选择信号（2表示选择，0表示取消选择）

// 返回值：空

//========================================================

void on\_stuNumCBox\_stateChanged(int arg1);

//========================================================

// 功能：根据姓名勾选框的选择更新输入框、搜索按钮和学号勾选 // 框的设置

// 输入参数：选择信号（2表示选择，0表示取消选择）

// 返回值：空

//========================================================

void on\_nameCBox\_stateChanged(int arg1);

//========================================================

// 功能：该类窗口界面上的搜索按钮的click()信号的槽函数，搜 // 索出与lineEdit输入内容匹配的学生并将其显示在窗口 // 组件stuTable上

// 输入参数：选择信号（2表示选择，0表示取消选择）

// 返回值：空

//========================================================

void on\_searchBtn\_clicked();

1. 食堂应用模块(CanteenAppDialog)

//========================================================

// 功能：以delim为间隔，将传入的str字符串拆分成若干个字符 // 串

// 输入参数：待切割的字符串；切割字符串的分隔字符串

// 返回值：由str切割而成的若干个字符串的集合

//========================================================

static vector<string> split(const string &str, const string &delim);

//========================================================

// 功能：实现传入卡号对应学生的消费操作

// 输入参数：是否为批量操作的标志；窗口号-1；卡号；消费金额； // 日期；时间

// 返回值：空

//========================================================

void consume(bool isPl, int winNum, string cardNum, float conMoney, string date, string time);

1. 汇总分析模块(SummarizeDialog)

//========================================================

// 功能：用多路归并将传入参数内的信息按日期时间进行排序，并 // 将排序结果存进链表中

// 输入参数：待排序的多路信息的键值对集合

// 返回值：空

//========================================================

void sortWinXf(map<int,pair<int,winXfStr\*>> allReacords);

//========================================================

// 功能：利用折半插入排序，每次插入一个元素到有序序列中(按 // 照str升序)

// 输入参数：窗口号-1，日期时间字符串，有序待插入容器

// 返回值：空

//========================================================

void insert\_vec(int i, string str, vector<pair<int, string> > &vec);

//========================================================

// 功能：对传入的字符串进行加密，并返回结果

// 输入参数：加密的起始序号；待加密的语句

// 返回值：加密后的字符串

//========================================================

static string getCheckStr(int num,string str);

//========================================================

// 功能：将有序链表中的信息结合成语句，并加密后写入数据文件 // all\_xf.txt中

// 输入参数：空

// 返回值：空

//========================================================

void writeWinXf();

1. 汇总分析的搜索功能实现模块(SearchDialog)

//========================================================

// 功能：搜索函数的辅助函数，在有时间限制的情况下对有序链表 // 进行搜索

// 输入参数：有序消费信息链表的头节点指针；其他选择的综合标 // 志；最大金额；最小金额；学号；姓名

// 返回值：空

//========================================================

void jugeWithTime(sortedXf\* p,int flag,int maxMoney,int minMoney,string stuNum,string name, string endTime);

//========================================================

// 功能：搜索函数的辅助函数，在没有时间限制的情况下对有序链 // 表进行搜索

// 输入参数：有序消费信息链表的起始搜索节点的指针；其他选择 // 的综合标志；最大金额；最小金额；学号；姓名

// 返回值：空

//========================================================

void jugeNoTime(sortedXf\* p,int flag,int maxMoney,int minMoney,string stuNum,string name);

//========================================================

// 功能：根据搜索条件的设定，遍历有序消费信息链表，搜索符合 // 要求的信息，插入到保存结果的链表中

// 输入参数：空

// 返回值：空

//========================================================

void search();

1. 汇总分析的统计功能实现模块(StatisticsDialog)

//========================================================

// 功能：该类窗口界面上的查询按钮的click()信号的槽函数，遍 // 历消费信息链表根据限定日期和学号，计算学生在该段 // 时间的消费次数和金额并显示在窗口界面的组件上

// 输入参数：空

// 返回值：空

//========================================================

void on\_inquireButton\_clicked();

1. 汇总分析的分析功能实现模块(AnalyzeDialog)

//========================================================

// 功能：该类窗口界面上的分析按钮的click()信号的槽函数，分 // 析出与输入学号对应学生经常一起在食堂排序消费的学 // 生，将他们的信息存入成员变量m\_info中

// 输入参数：空

// 返回值：空

//========================================================

void on\_analyzeBtn\_clicked();

1. 汇总分析的校验功能实现模块(CheckFileDialog)

//========================================================

// 功能：该类窗口界面上的搜索按钮的click()信号的槽函数，搜 // 索消费记录保存文件内是否有增减一行或多行的改变

// 输入参数：空

// 返回值：空

//========================================================

void on\_checkBtn\_clicked();

//========================================================

// 功能：对传入字符串进行解密

// 输入参数：加密字符串

// 返回值：解密后的字符串

//========================================================

string getXfStr(string str);

1. 数据导入模块(LeadData)

//========================================================

// 功能：根据同目录下students.txt文件内的学生信息，将学生 // 开户，信息导入到学生信息系统的students容器中

// 输入参数：空

// 返回值：空

//========================================================

void initStu();

//========================================================

// 功能：从xf.txt和cz.txt文件中获取系列操作语句

// 输入参数：空

// 返回值：空

//========================================================

void getOpration();

//========================================================

// 功能：将所有操作利用归并排序和折半插入排序根据时间的先后 // 进行排序，将排序后的结果存入到成员变量m\_sortedOp // 所指的链表中

// 输入参数：空

// 返回值：空

//========================================================

void sortOpration();

//========================================================

// 功能：将排序后的操作在学生管理和食堂应用系统中进行从首至 // 尾的实施

// 输入参数：空

// 返回值：最后一条操作的<日期，时间>

//========================================================

pair<string,string> carryOutOp();

1. 函数调用关系

之后部分凡是用到CO、COP、DT等全局指针变量，都会提前调用相应的getinstance()函数获取相应地址完成赋值，后面就不一一指出。

且图示中橙色部分都表示函数调用，蓝绿色表示窗口界面，白色部分表示函数中的部分操作。

1. main函数入口

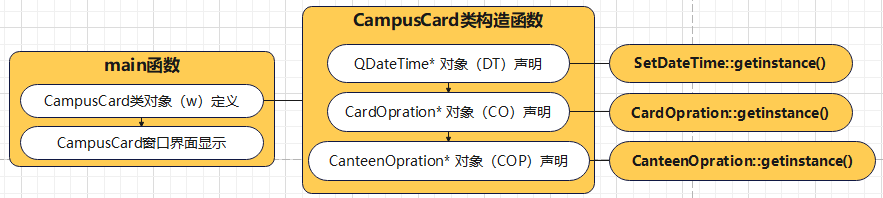


图5-4-1 main函数的程序调用

1. CampusCard窗口界面及其内部的相关函数操作图示

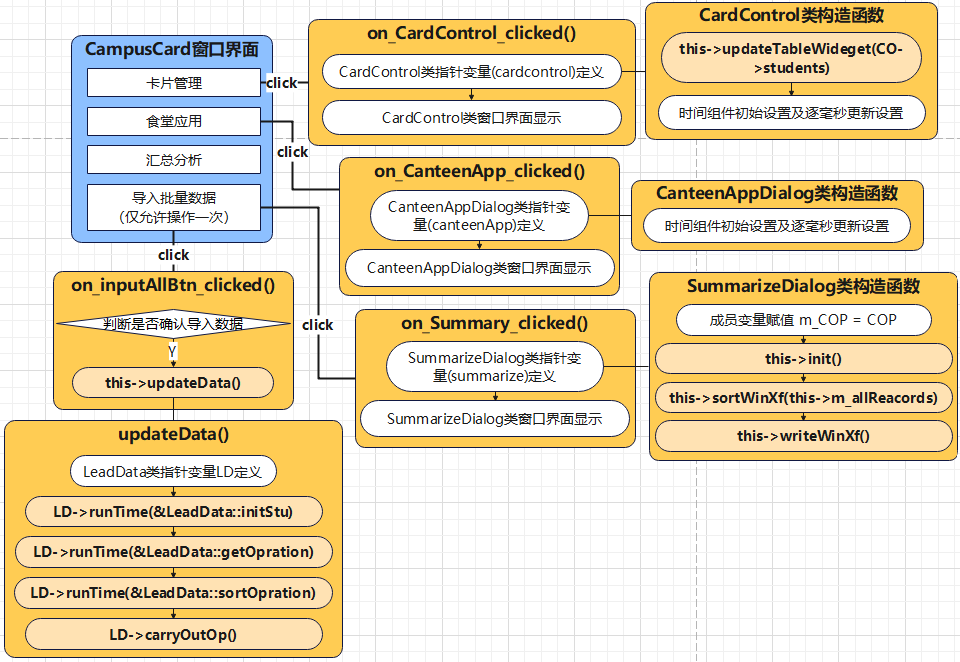


图5-4-2 CampusCard窗口界面的程序调用

1. CardControl窗口界面及其内部的相关函数操作图示

说明：

根据勾选框的不同选择，搜索按钮调用的函数中，使用的搜索比对函数会有所不同。如果勾选的是学号，会调用matchWithStuNum()；如果勾选的是姓名，会调用matchWithName()。

挂失、补卡、解挂按钮的函数操作与销户按钮类似，但在定义inPutStuNumDialog类变量时传入参数不同，1代表补卡，2代表挂失，3代表解挂，4代表销户，故在界面单击确认按钮时调用的函数也不同，分别是CO->makeUpCard()，CO->lossSolution\_1，CO->lossSolution\_2，CO->deleteStu()。

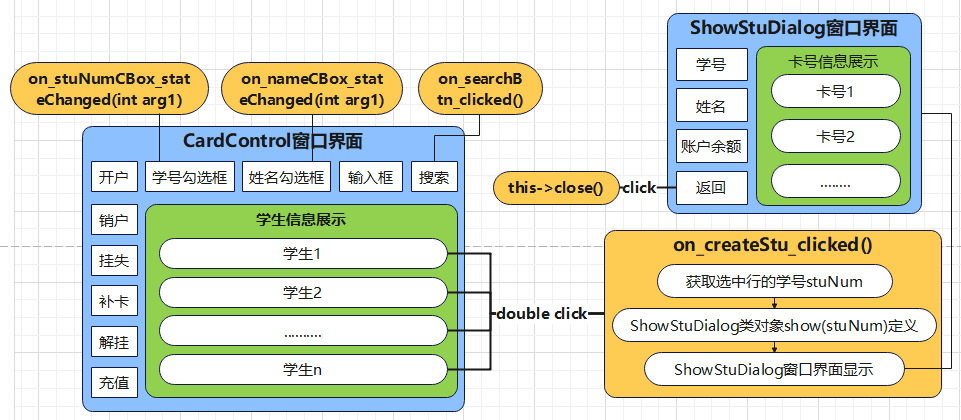


图5-4-3 CardControl窗口界面的程序调用

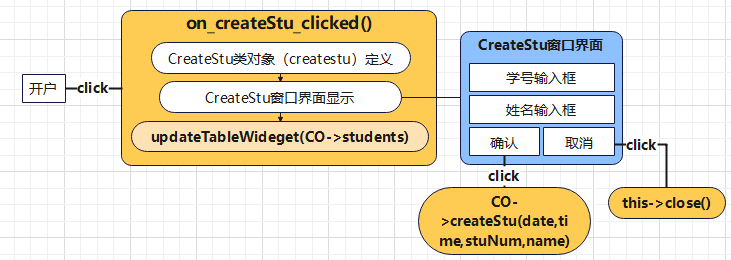


图5-4-4 开户按钮对应的程序调用

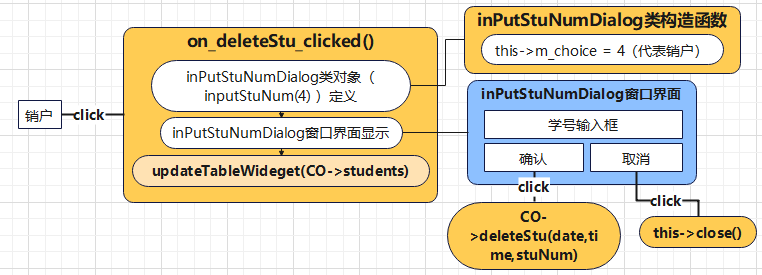


图5-4-5 销户按钮对应的程序调用

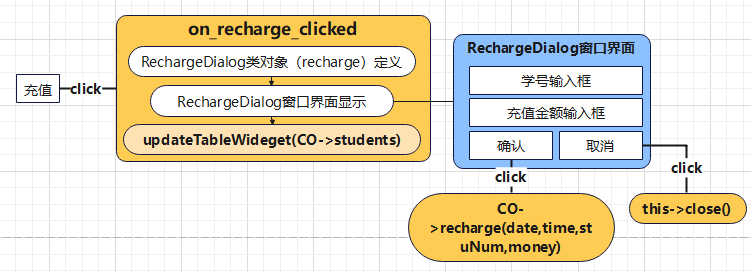


图5-4-6 充值按钮对应的程序调用

1. CanteenAppDialog窗口界面及其内部的相关函数操作图示

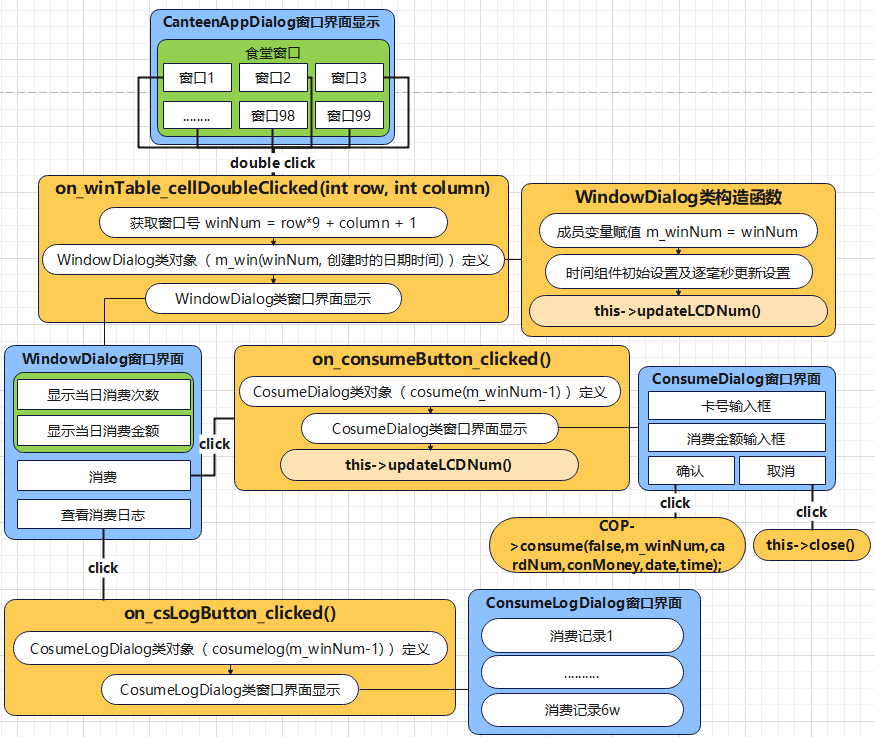


图5-4-7 CanteenAppDialog窗口界面的程序调用

1. SummarizeDialog窗口界面及其内部的相关函数操作图示

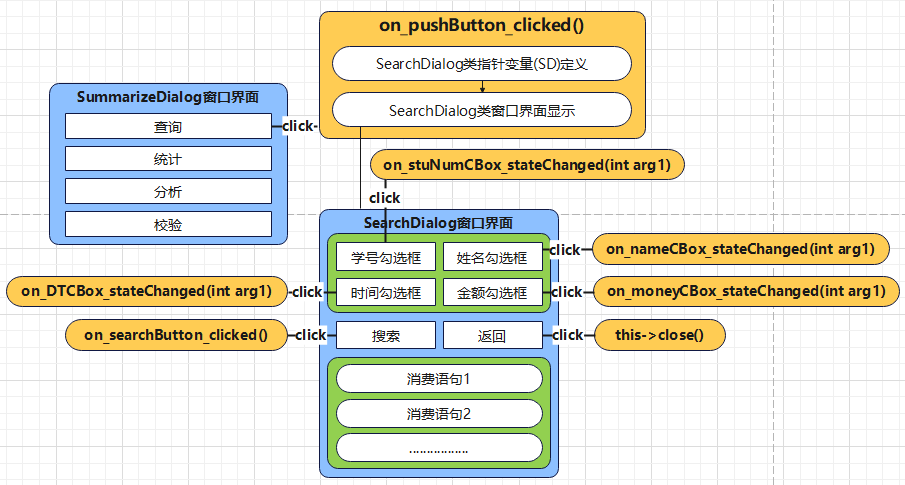


图5-4-8 SummarizeDialog窗口界面查询按钮的程序调用

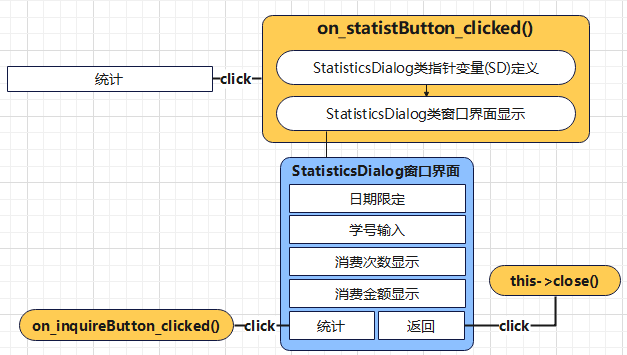


图5-4-9 SummarizeDialog窗口界面统计按钮的程序调用

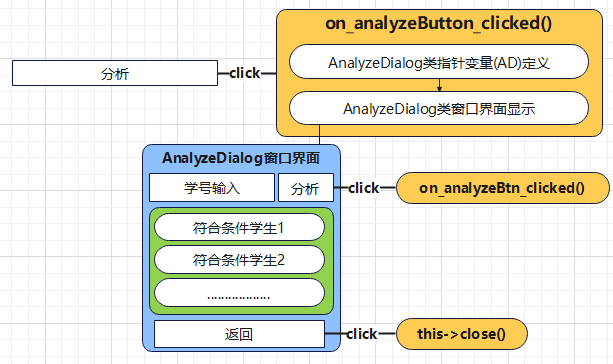


图5-4-10 SummarizeDialog窗口界面分析按钮的程序调用

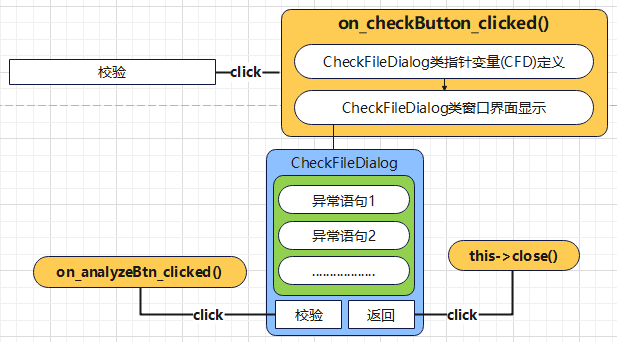


图5-4-11 SummarizeDialog窗口界面检验按钮的程序调用

# 六、运行测试与结果分析

1、程序运行初始窗口

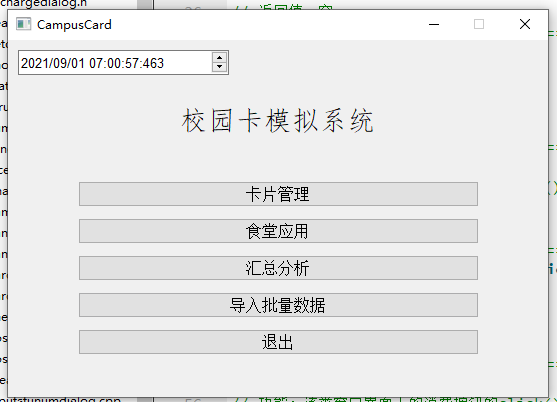


图6-1 校园卡主窗口界面

1. 选择导入批量数据

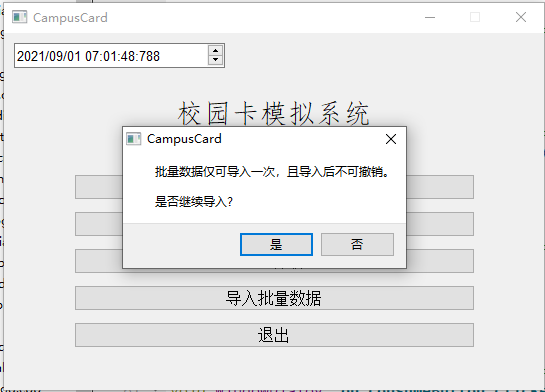


图6-2 导入批量数据按钮弹窗

1. 运行卡片管理窗口

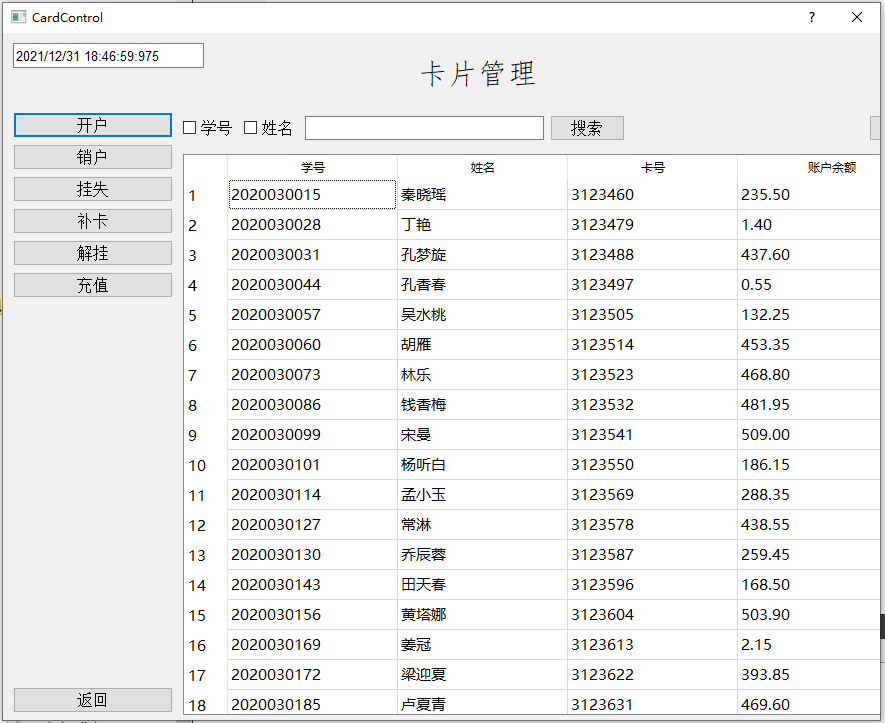


图6-3-1 导入批量数据按钮弹窗

其中各功能测试：

1. 学号搜索

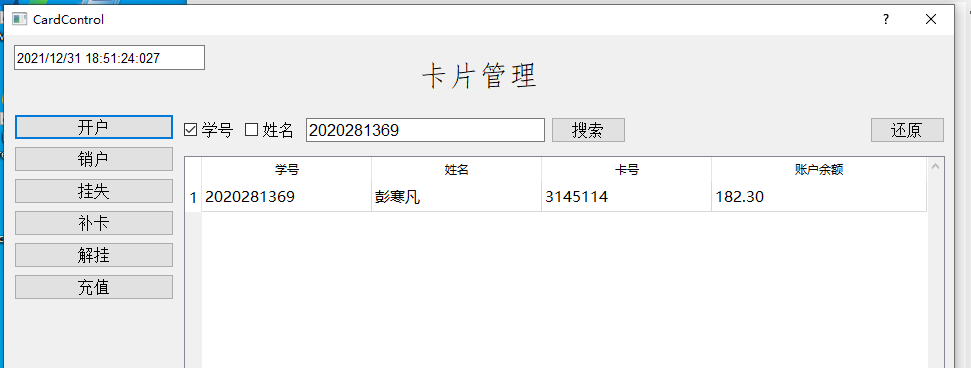


图6-3-2 根据学号准确搜索学生功能运行



图6-3-3 根据学号模糊搜索学生功能运行

1. 姓名搜索

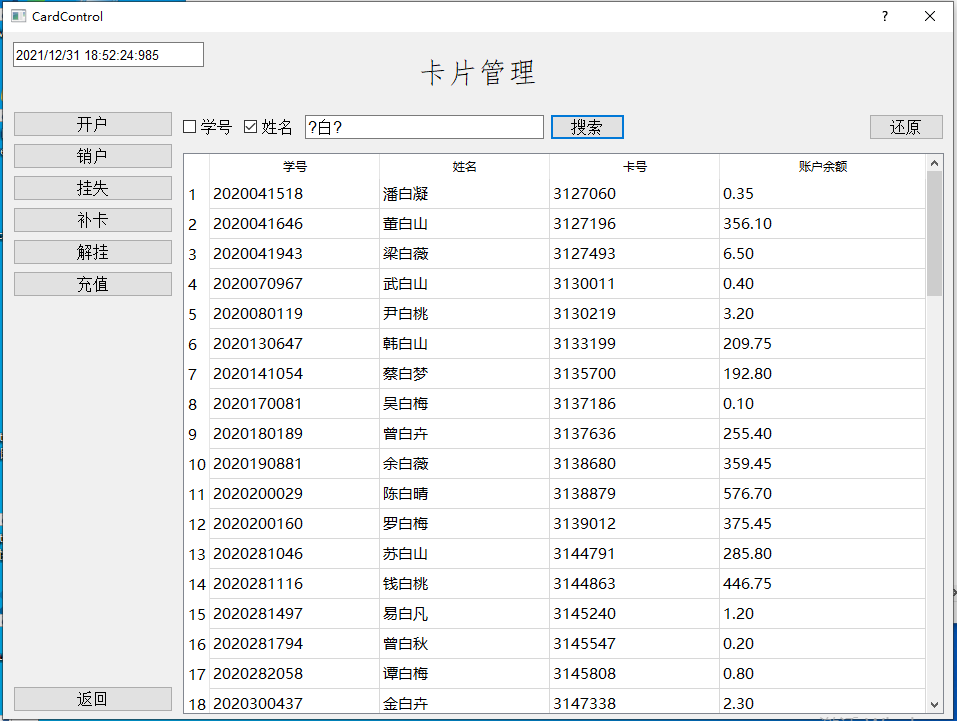


图6-3-4 根据姓名模糊搜索学生功能运行

1. 学生信息查看

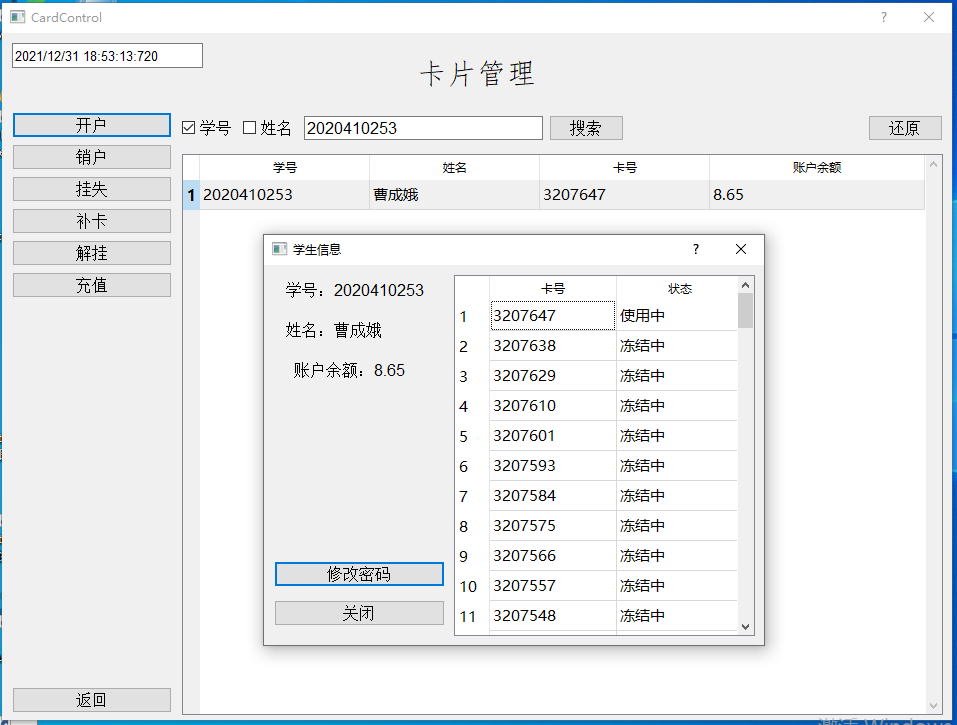


图6-3-5 学生信息查看功能运行

1. 开户

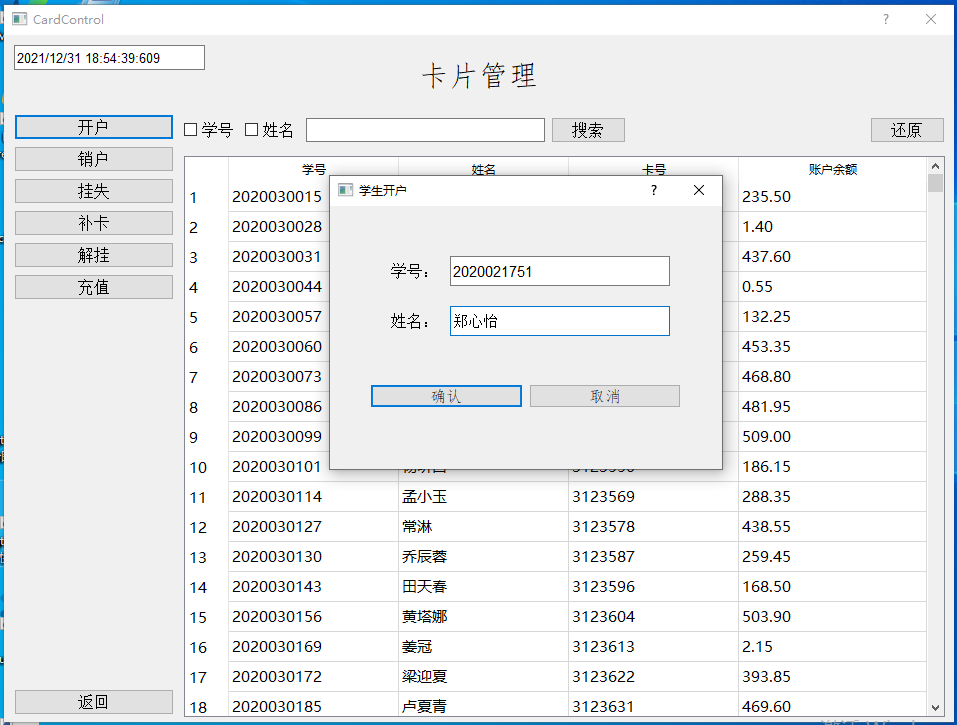


图6-3-6 学生开户功能运行弹窗

1. 充值

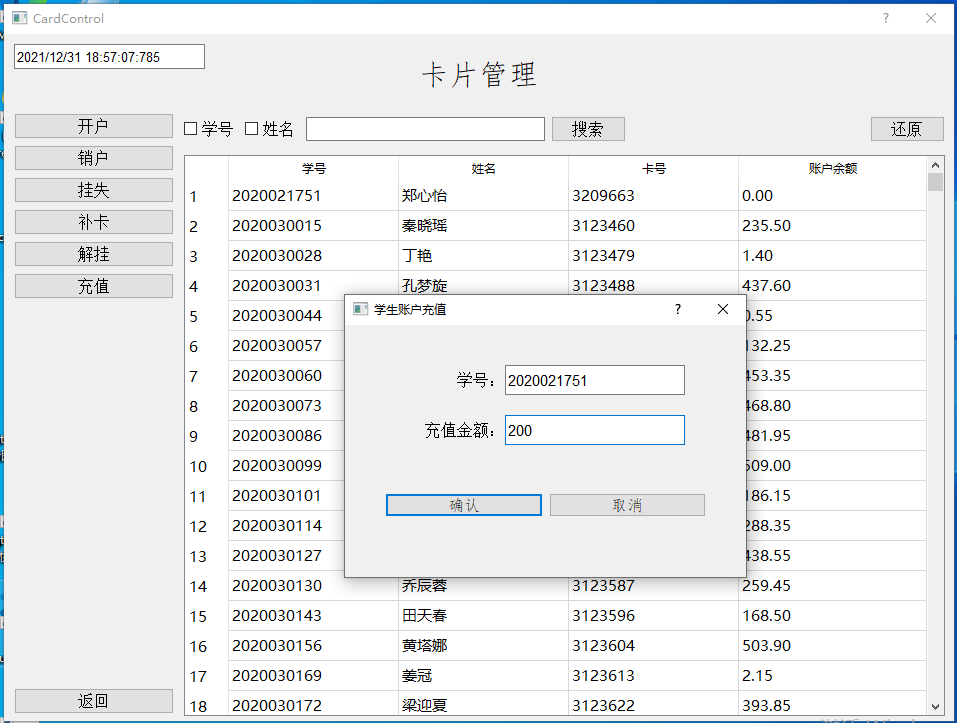


图6-3-7 学生账户充值功能运行弹窗

1. 对创建学生进行一次挂失，一次解挂，五次补卡

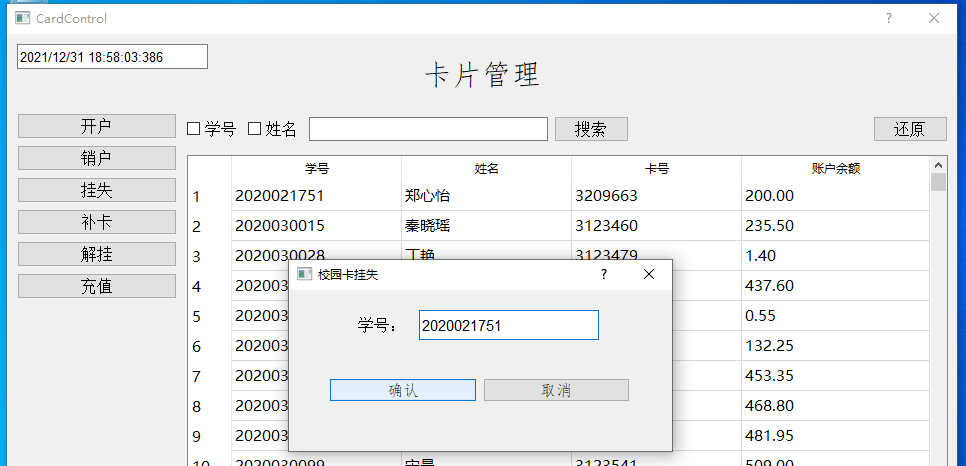


图6-3-8 学生挂失功能运行弹窗

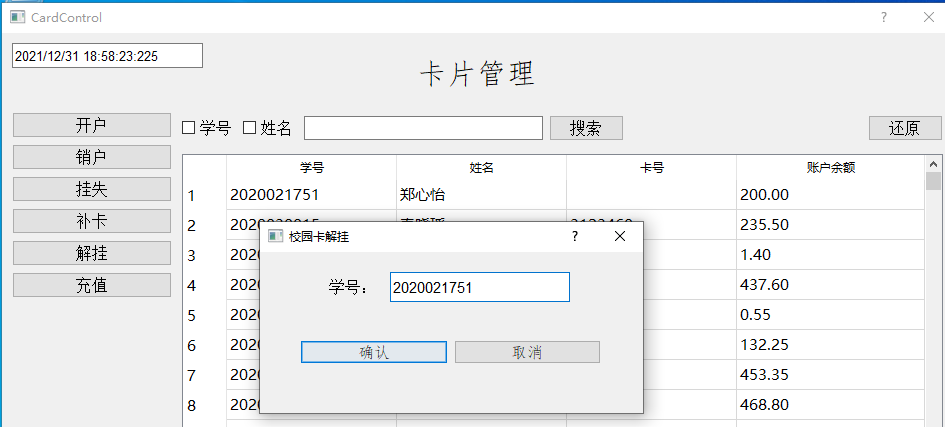


图6-3-9 学生解挂功能运行弹窗

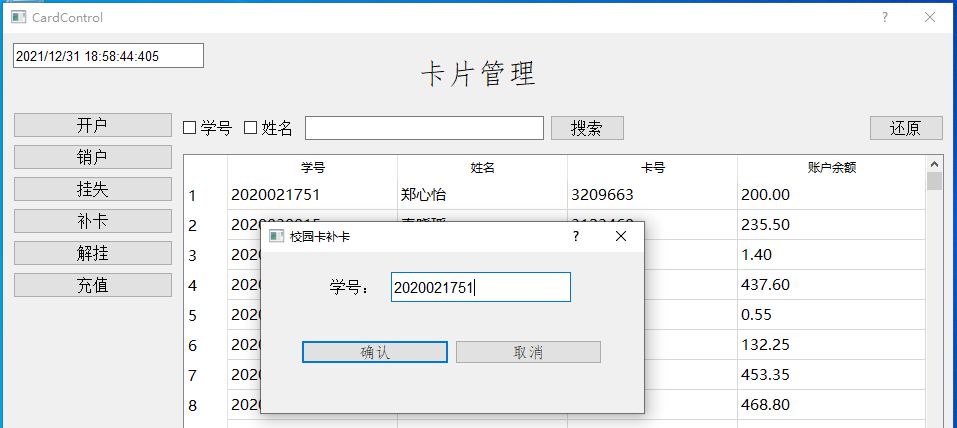


图6-3-10 学生补卡功能运行弹窗

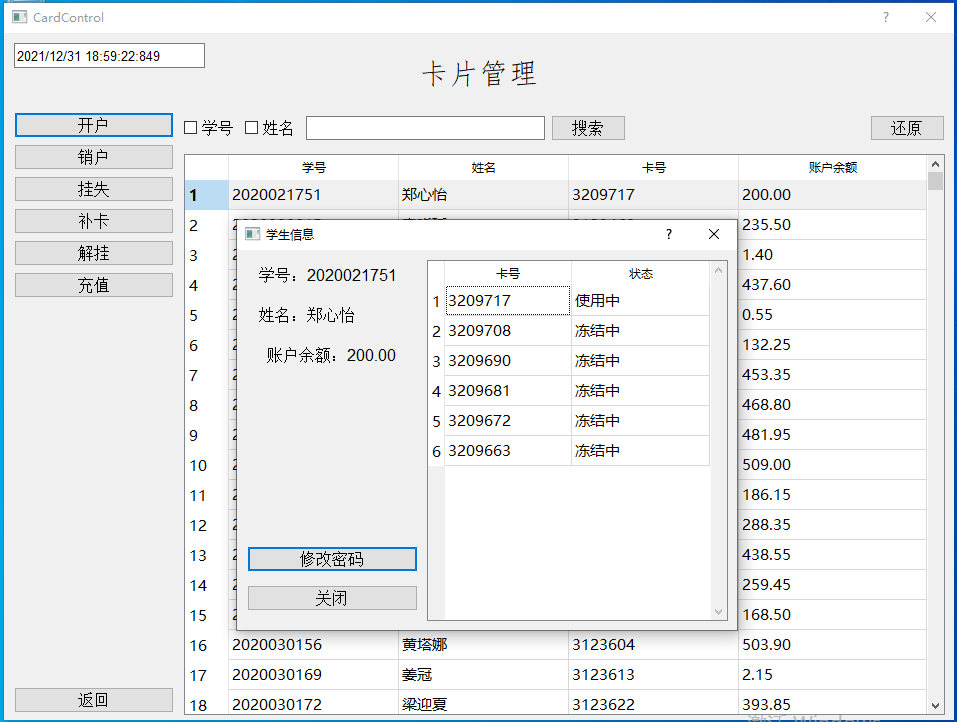


图6-3-11 挂失、解挂、补卡功能运行结果查看

1. 销户

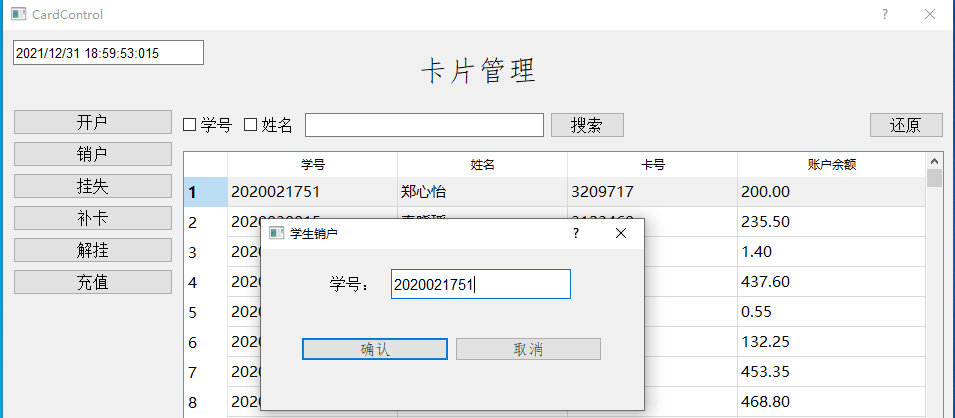


图6-3-12 学生销户功能运行弹窗

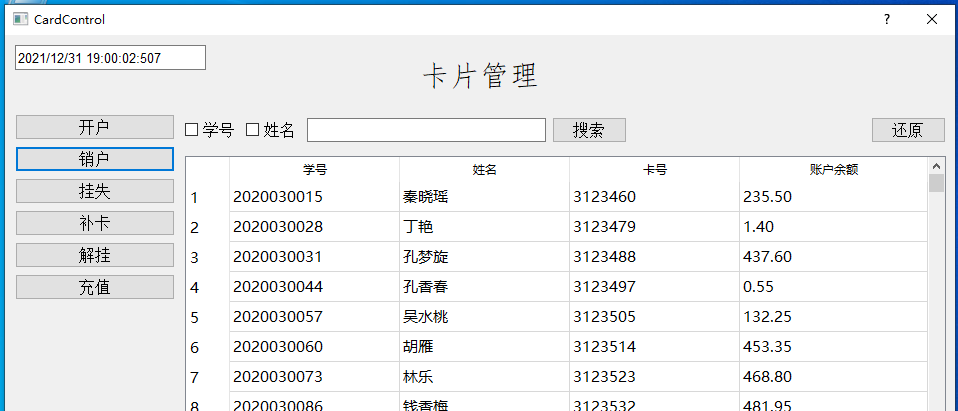


图6-3-13 学生销户功能运行结果查看

1. 运行食堂应用窗口

（1）食堂总窗口选择

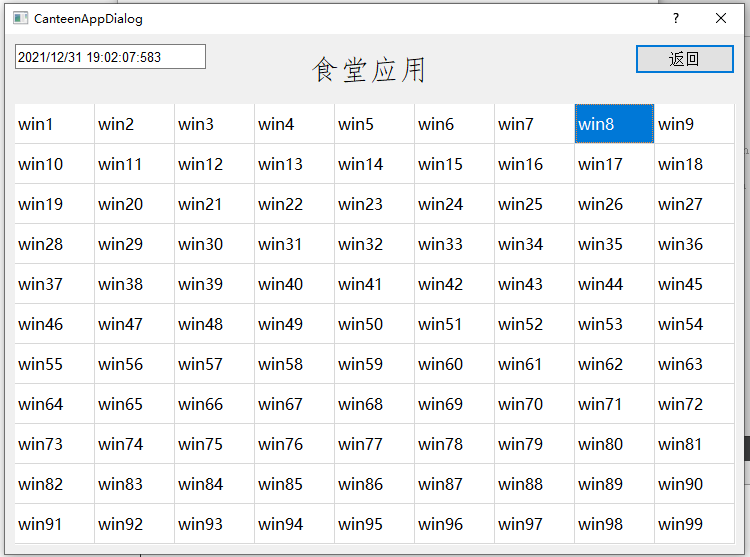


图6-4-1 食堂应用主窗口界面

1. 查看24号窗口

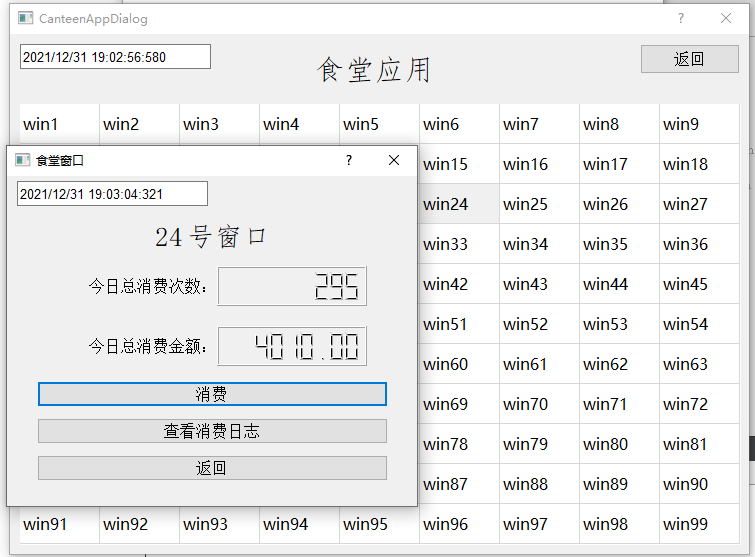


图6-4-2 食堂窗口界面

1. 查看24号窗口的消费日志



图6-4-3 食堂窗口消费日志界面

1. 在2号窗口用卡号3123488(2020030031, 孔梦旋)连续消费两次12元

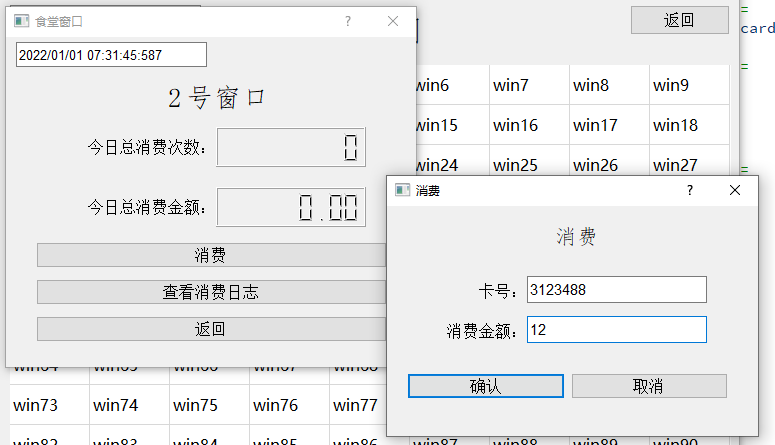


图6-4-4 食堂窗口消费界面

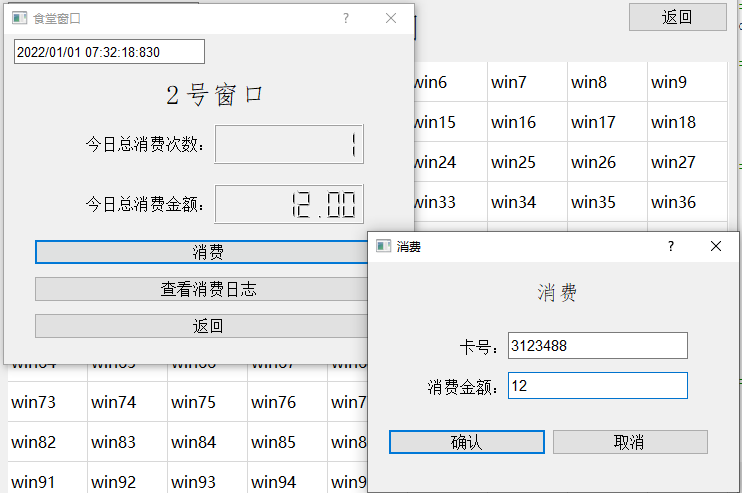


图6-4-5 食堂窗口消费界面

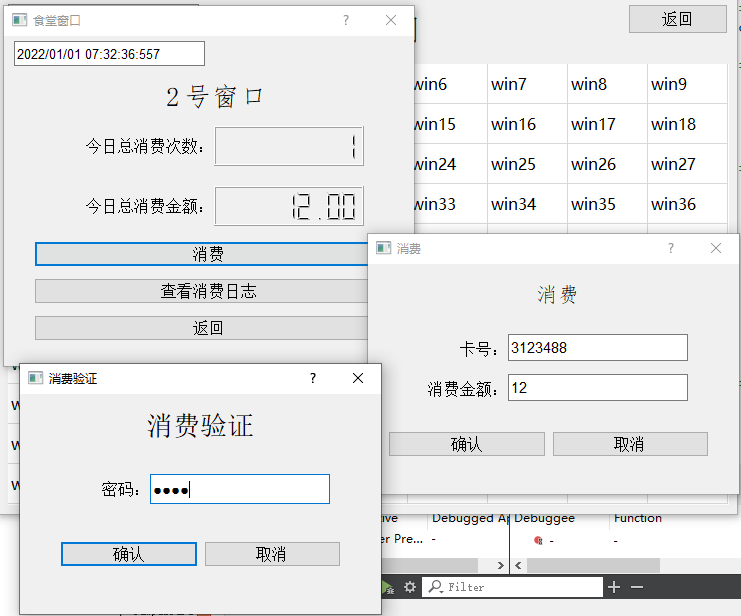


图6-4-6 食堂窗口时段消费超额密码输入界面

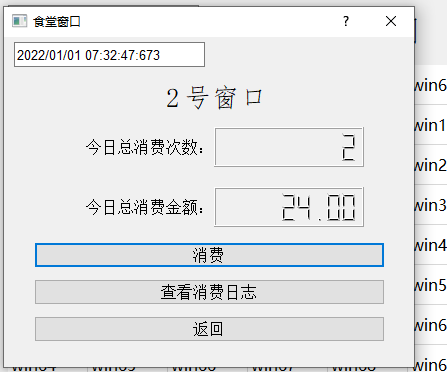


图6-4-7 消费完成后窗口界面结果显示

1. 运行汇总分析窗口

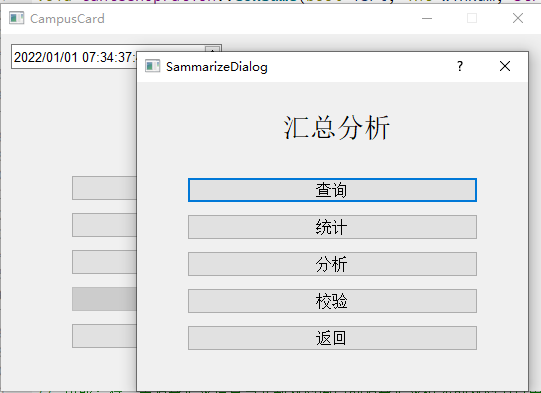


图6-5-1 汇总分析主窗口界面

1. 查询功能使用

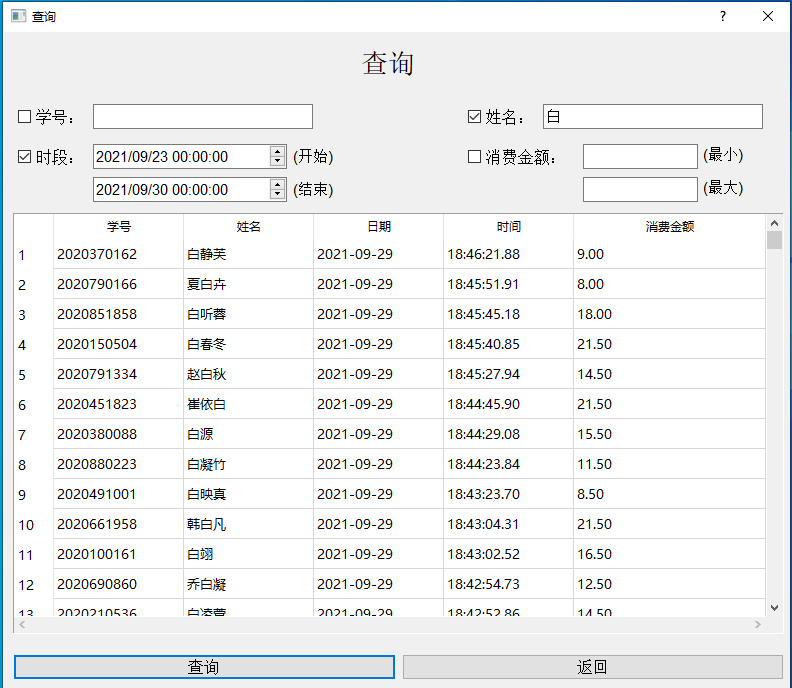


图6-5-2 汇总分析查询功能界面及使用

1. 统计功能使用

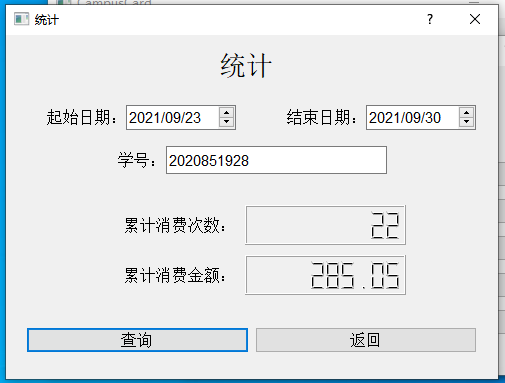


图6-5-3 汇总分析统计功能界面及使用

1. 分析功能使用

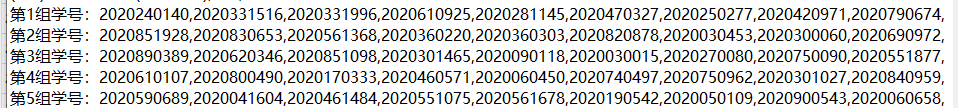


图6-5-4 分析功能的运行期望结果



图6-5-5 汇总分析分析功能界面及使用（1）



图6-5-6 汇总分析分析功能界面及使用（2）

1. 校验功能使用
2. 查看汇总消费记录的保存文件



图6-5-7 汇总分析消费记录保存数据文件部分展示

1. 从中随意删除两行，增加两行后校验

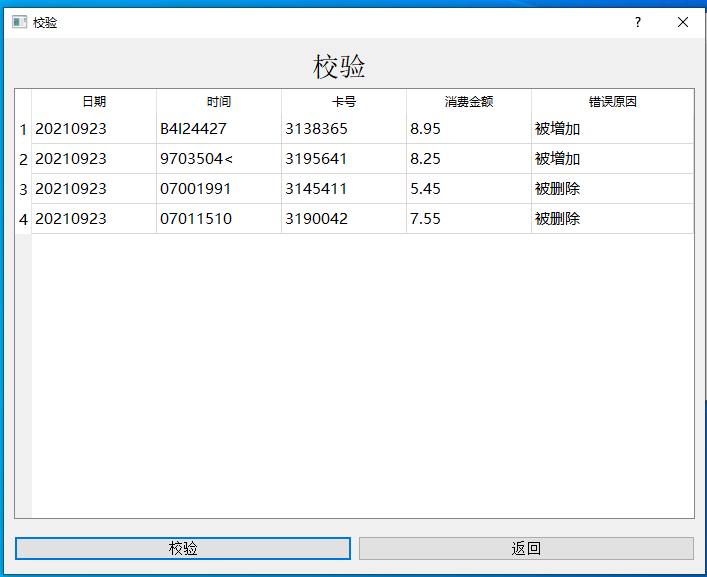


图6-5-8 汇总分析校验功能界面及使用

1. 部分功能性能及复杂度分析
2. 消费记录记录的多路归并功能

从99个窗口可以获得99路各自有序的消费记录，将其合成一条有序消费记录的链表主要采用多路归并：每次从99路的首元素中获取最大的元素，将其用头插法插入目标链表中，然后更新对应路数的下一个元素，再与其他路数进行比较，直至每一路的待排元素都为0。

设k（k<=99，表示不为空的记录路数）路的总记录条数为n，则由k路的记录生成一个有序的消费记录的时间复杂度会与n成正比。

另外考虑从k路的首元素中获取最大元素的时间复杂度，该过程采用的是折半插入排序。准备了一个vector容器有序存储待排路数的首元素信息和来自的路数，每次获取到最大元素后将其从容器中删去，替换插入其对应路数的下一个元素。（不考虑第一次从k路获取首元素的过程）每次插入新元素的比较次数大概为O(logk)，将一个元素插入到容器中的时间大致为O(k)。

则该功能的总复杂度为O(n\*(k+logk))。

空间复杂度则考虑辅助容器vector的大小和最后有序链表的大小，大致为O(n+k)。



图6-6-1 汇总分析消费记录多路归并花费时间截图

1. 汇总分析中的查询功能

该功能的主体函数进行前，会先确定组合了哪些限定因素（姓名、学号、时间、消费金额中选择1-4个），避免在整个比对过程中进行多次判断浪费时间。

查询是通过遍历有序消费记录链表来进行的，故整体的运行时间与链表的长度n有关。由于该序列是按时间有序的，故先考虑时间项是否选择。

若是选择了时间限定，则先根据时间定位到第一个符合时间范围的节点处，省去了从头节点到该节点中其他信息的比对；且当遍历到节点不服和时间限定时，则可结束遍历，省去了之后节点中的信息比对。

若没有选择时间限定，则只能遍历整个链表比对其他已选择的限定信息。

仅考虑单个节点的信息比对时，判断其是否符合要求，与选择的限定条件和对应的信息字符串长度有关（日期时间字符串长度为16，姓名字符串长度为4-8，学号字符串长度为10，消费金额字符串长度为4-5）。

时间和消费金额的比对，仅遍历字符串比较每个字符比较大小即可，但需要比较两次，与上限和下限各比较一次。

为简化搜索过程，学号的比对仅需输入学号片段是对应学号的子串即可，姓名也是。

故单项信息的比对耗费的时间复杂度是个常数。

故整个查询操作的耗费时间大致为是‘n\*常数’，故时间复杂度为O(n)。

显然该功能的空间复杂度与符合要求的记录条数有关。



图6-6-2 汇总分析中查询功能花费时间截图

1. 汇总分析中的分析功能

该功能需要遍历整个记录两次，遍历长度相同，但遍历形式不同。

前一次遍历整个有序消费记录链表，只要找到信息与输入学号对应的节点，就比对他的前后节点的信息中记录的窗口是否在该节点的窗口两侧，是则保存，这个过程中对每个节点都要进行学号比较（学号字符串长度固定为10）。

学号比较最好的情况是第一个字符就比对失败，则进入下一个节点比较；最坏的情况是比较成功或者最后一个字符才比较失败，即比较长度为10。

但总体来说，整个比较最坏的情况是10次字符比较，全部看作最坏的情况也只能是‘n\*常数’的时间消耗，故前一次遍历复杂度是O(n)。

后一次遍历，只是分别遍历每个窗口的消费记录，比对长度与前一次相同，比对过程也类似，复杂度也为O(n)。

故该功能的时间复杂度为O(n)。

显然该功能的空间复杂度与符合要求的信息数目有关。



图6-6-3 汇总分析中分析功能花费时间截图

**七、总结**

1、精神体会

我觉得很强的一个体会在于，我以前没有想过敲代码也需要坚持和锲而不舍的精神的。

我之前没有写过这样一个功能完善的大程序（对我来说算大了），这次自己写了真的觉得这个工作是很耗耐力、精力的，中间很多次我都会写烦了就休息，也算是半玩半写了。但是其实可以想到，我在很放松的时间限制和环境下写都如此烦躁不耐烦，以后的工作任务未必有这么放松的时限，工程量也比这大得多，那又如何自处？所以时刻保持着探究精神，保持锲而不舍的耐力，保持心情的平静，保持思维的冷静，保持对自己所写代码的责任意识都是很重要的，是在敲代码的成长历程中需要不断提高的。

2、课设过程中的体会和收获

在代码的编写的过程中也有很多新的收获。

对于在Qt中代码运行出的各种bug有了一些基本的辨别能力，大致能够知道报错弹窗所要表达的意义，栈溢出、空指针等，后期程序的收尾阶段没有再因为看不到弹窗意义，不知道错误方向，无脑调试浪费很多时间。

也知道了学一种新语言最快的方法应该是学以致用，对于没有用过的一些内置函数的了解方法最快的是查API，这样一般也会比上网搜可以了解得更深入具体一些。同时也要学会好好利用一个软件自带的帮助文档。再者我觉得语言本身就是一门通门门通的，只要对一两门语言掌握到位，其他语言的学习都是学个规则就可以较好的使用了。

其中我认为给我留下的印象最深的教训是，不要过于依赖一些看似可以省去很多功夫的容器，特别是在不了解容器的各种相关函数的内部运作过程的时候。

整个程序写下来，令我卡壳最久的应该是多路归并排序，最让我恼火的点是卡壳的原因不在于我写的功能算法，而在于这个过程中保存数据的容器。多路归并功能实施的算法并不复杂，但我写的最初版本，仅仅排序不涉及其他相关功能调用的情况下，竟然需要运行5000+s之久。导致这个结果的原因是，我为了偷懒，几乎所有在内存中运转的数据，都是用vector等容器存放，除了规定的需要用链表的地方，没有其他一处用到了链表。容器存放数据确实可以省去很多不必要的麻烦，但当数据很多、操作很频繁时，使用容器会大大降低整个程序的效率，需要考虑容器内置函数运行本身的复杂性。我之前一直都没有考虑过这个问题，只是一直在改进其他的地方，减少比较判断等，以期减少排序时间，最后都没有效果，最后还是被同学提醒了，将整个程序的基础结构体都进行了改动，将大部分容器的使用都用了链表替代，之后将排序的时间控制在了5-7s内。

不得不说，处理大量数据时，不能使用数据库的情况下，链表不失为一个很好的选择，所有的复杂度都是透明的，则效率的分析提高都会变得更加清晰。

3、思维模式上的收获

以往的课程任务，如果不追求编写标准的话一个cpp文件就可以解决问题，但这次的课程设计体现了很强的模块化思维，如果将整个程序用一个文件或者仅一个模块解决，我想想觉得是灾难。

通过整个程序的编写我也更喜欢使用类了，因为当需要模块时，类往往会是一种很好的分割界限和一种很好的定位方法。

还有一个很重要的思维转变在于，程序在编写前是一定需要设计的。程序小、性能要求简单的时候，可能这一步的重要性不会显现得很明显。但当整个程序很庞大，有很多要实现的功能的时候，不提前设计而写一步想一步，不仅耗时耗精力更大，而且出错的可能性更高，会导致整个程序的结构散乱不堪。设计不仅仅在于分割模块和功能，我认为更重要的在于一些贯穿各个模块的组件、功能的放置设计。

就比如说时间，就这个程序而言，整个的设定下是一定有一个唯一时间线的，那这个时间变量应该放在哪个位置就显得尤为重要， 我想了很久之后才决定将一个全局时间变量绑定程序运行后弹出的第一个窗口的日期时间组件上，后续窗口的时间显示由参数传入来传递。

另外在写报告的过程中，才真的能够意识到一个代码程序的好坏也不仅仅在于程序本身的性能，也在于代码的注释添加和他人易懂性，尤其是以后程序再大一些，多人合作的时候这一点就会显得更加重要。

最后还要提到一点，我觉得当已经可以很好利用语言的时候，想法和想法的实施方案会比语言更重要，所以多思多想，多看一些较大项目的开发代码是会很有益处的。

# 八、参考文献

1. 殷立峰,祁淑霞,房志峰.Qt C++跨平台图形界面程序设计基础(第二版)[M].北京: 清华大学出版社, 2018.
2. 严蔚敏,吴伟民.数据结构(C语言版)[M].北京: 清华大学出版社, 2019.
3. 编程小鱼酱.Qt学生信息管理系统V1.0[EB/OL]. <https://space.bilibili.com/523392138/channel/collectiondetail?sid=125683,> 2022-01-03/2022-01-15.

# 附录：程序使用说明

程序的运行功能展示都在报告的第六部分有了整体的说明，此处只说明两处处细节部分。

1. 所有输入框的输入都有设定（输入时会自带限制，不需要自己主观控制输入限制），学号输入框限制只能输入10位数字，姓名输入框只能输入2-4个汉字，消费金额输入框只能输入0-99.99的消费金额，充值金额输入框只能输入整数，模糊匹配的学号输入框只能输入数字、?、\*，模糊匹配的姓名输入框只能输入汉字和?。
2. 查看学生具体信息是双击学生管理界面的表中的信息行。
3. 只能在最新的窗口进行操作，只有关闭当前窗口才能进行
4. 时间的更改只能在最初始的窗口界面。