**实验报告**

**学院（系）名称：**计算机与通信工程学院

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **姓名** | 王帆 | **学号** | 20152180 | **专业** | 2015计算机科学与技术 |
| **班级** | 15-计算机1班 | **实验名称** | 综合实验项目：银行ATM系统模拟 | | |
| **课程名称** | | 高级程序设计语言II | | **课程代码** | 0667026 |
| **实验时间** | | 2016年11月10日 | | **实验地点** | 7-215 |
| **批改意见** | | | | **成绩** |  |
| **教师签字：** | | | | | |
| **一、实验目的**  根据所学习的面向对象程序设计的基本方法，使用面向对象的思想和方法对实际问题进行需求分析，根据需求分析结果完成程序的总体设计，在总体设计的基础上实现详细设计和编码调试，并对所设计的程序进行必要的测试。从而初步具备程序设计的综合能力和软件开发的初级技能。  **二、实验基本要求**  模拟实现银行ATM自动取款机系统。系统至少包括以下几方面的功能：  （1）修改密码功能；（2）余额查询功能；（3） 取款功能；（4）转账功能。  **三、概要设计分析**  程序基本功能   1. **ATM基本交互操作**   本程序最大程度上模拟了用户与ATM的基本交互操作，具体包括存款，取款，转账（伪实现）以及改密功能，并提供查询接口，可以查询余额以及账户基本信息。   1. **用户账户系统**   对于用户账户系统，本程序从信息录入与信息维护两方面出发，实现新账户交互式录入与格式化录入，以及对当前账户信息的查询与密码修改。  类的抽象  **customer：用户账户类**  实现基本的用户账户构造，查询接口提供以及数据操作（密码修改与余额操作）  **ATM：ATM功能类**  通过拷贝构造初始化对象，实现ATM基本功能模拟。    **类的关系**  **ATM类通过拷贝构造调用customer类（对象操作），实现对customer数据的操作。**  **四、具体实现分析**  **类的定义与实现**  **Class customer：**  **成员属性：**  **Protected：**  string ID  string name  int password  double money  **成员方法：**  **Public：**  构造函数：提供两种方式实例化customer类  customer(); 交互式-实例化  customer(string id,string nm,double mo); 格式化-实例化  Set函数：设置密码与金额变更  int set\_pw(); 获取密码（6位数字-整型）  void set\_pw(int pw); 设置密码  void set\_money(int mo);  Get函数：获取私有对象属性  string get\_ID();  double get\_money();  string get\_name();  int get\_pw();  Judge函数：判断逻辑正误  int judge\_pw();*//判断密码正误*  **Class ATM：**  **成员属性：**  **Protected：**  customer cm 用户类对象  **成员方法：**  **Public：**  ATM(customer& c); 构造函数：拷贝构造customer，用于操作测试（伪操作）ver 1.0  菜单函数：显示菜单与交互操作  void welcome();  void menu();  功能函数：实现对应功能  void deposit(); 存款功能  void withdraw(); 取款功能  void transfer(); 转账功能（伪转账操作）  string change\_pw(); 改密码功能  显示函数：格式化输出对应内容  void show\_deposit();  void show\_deposit(int mo); 重载show\_deposit 实现存款金额提示  void show\_withdraw();  void show\_withdraw(int mo); 重载show\_withdraw 实现取款金额提示  void show\_withdraw(int mo, int i); 重载show\_withdraw 实现取款交互提示  void show\_transfer(int i);  void show\_transfer(int mo, int id, int i); 重载show\_transfer 实现转账交互提示  void show\_money();  void show\_menu();  void show\_information();  判断函数：判断逻辑正误  int judge\_num(); 判断整100数字正误，最终获得正确数值  **关键代码分析**  **1.密文处理**  在ATM模拟的过程中，密码的获取与判断对于整个系统来讲是一个重点问题。日常使用过程中，密码作为密文形式进行表现。本模块综合运用if/while循环语句， getch() 流输入操作，实现对密码输入的加密显示与密码值传递的功能。  **本模块代码清单：**  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  int customer::set\_pw(){  char password[100];  int i=0;  do{  char ch;  ch=getch();  *//判断部分：实现退格重新输出，密文显示与屏蔽非数字键功能*  if(ch==8){ *//退格键实现*  if(i!=0){  cout<<char(8)<<" "<<char(8);  i--;  }  }  else if(ch>=’0’&&ch<=’9’){*//密文显示&屏蔽非数字键*  cout<<"\*";  password[i++]=ch;  }  *//判断部分结束*  }while(i<6);  *//转换部分：转换char\*为int*  int temp=0;  for(i=0;i<6;i++){  temp=temp\*10+(password[i]-'0');  }  *//转换部分结束*  cout<<endl;  system("pause");  return temp;*//将获取到的密码以整型返回*  }  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  **2.取款功能-deposit**  取款功能作为ATM最基本的功能之一，是最容易却又最难实现的部分。一方面，取款界面包括格式化操作与个性化输出，需要用到switch分支跳转功能；另一方面，为了达到最佳的模拟度，提高用户体验，需要运用getch()进行字符获取。另外，通过对交互提示语句和功能语句的拆分，对代码进行优化，提高日后升级维护的可持续性。  **本模块代码清单：**  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  void ATM::withdraw(){  *//取款操作*  char ch;  show\_withdraw();  do{  ch=getch();  switch(ch){  case 'A':{*//格式化操作：取款100元*  if(cm.get\_money()>=100){  cm.set\_money(-100);  show\_withdraw(100);  }  else show\_withdraw(100,0);  withdraw();  break;  }  *// ……（其他格式化操作省略）*  case 'E':{*//个性化操作：根据用户输入金额决定取款金额*  show\_withdraw(0);  int temp\_money=judge\_num();  if(cm.get\_money()==0){  show\_withdraw(0,2);  }  else if(cm.get\_money()>=temp\_money){  cm.set\_money(-temp\_money);  show\_withdraw(temp\_money);  }  else show\_withdraw(temp\_money,0);  withdraw();  break;  }  case 'Q':{*//返回主菜单*  menu();  break;  }  default:{*//其他键入的情况*  show\_withdraw(0,-2);  show\_withdraw();  break;  }  }  }while(1);  }  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  **五、测试用例、测试结果**  **测试用例**  **测试结果**  1.用户初始化  **1.1 交互信息录入**  根据交互提示键入对应内容，密码仅可使用0-9以及退格键操作。  **1.2 基本信息提示**  以友好的方式提示新用户信息。  2.ATM操作  **2.1 欢迎界面**  以友好的方式提示用户操作。  **2.2 登录界面**  密码正确时进入主菜单。  密码错误时交互提示重新输入密码。  **2.3 主菜单**  **2.4 存款功能**  **2.5 取款功能**  2.5.1 取款主菜单  2.5.2 定额取款提示  取款成功提示：  取款失败（余额不足）提示：  2.5.3 其他金额取款提示  取款成功提示：  取款失败（输入错误）提示：  取款失败（余额不足）提示：    **2.6 转账功能**  转账成功提示：  转账失败（余额不足）提示：  （伪转账，暂时没有实现真正的ID相互转账，本功能将在下一个版本中完成）  **2.7 余额查询功能**  **2.8 个人信息查询功能**  **2.9 密码修改功能**  密码与原密码相同提示：  两次密码输入不相同提示：  密码修改成功提示：  2.10 退卡  **六、收获、心得体会**  本次实验是高级程序设计语言II课程的结课实验。本学期，通过对C++这一个具有面向对象程序设计特性的高级程序设计语言的学习，我大致了解并基本掌握了面向对象程序设计的基本特性：多态性、抽象性、继承性与封装性。了解了对象、类、消息与方法这些面向对象程序设计的基本概念。但是通过本次实验，我最直接的感受就是：自己需要学习的东西还有很多，在C++这门课程中自己了解的还太少。  起初在收到“模拟ATM”这个题目的时候，我的内心其实是拒绝的。因为上学期的C语言结课大实验也是这个“模拟ATM”设计。但当我真正着手准备，思考本课题的时候，我发现自己低估了它的难度。因为之前有做过C语言版本的ATM模拟机的实验，所以这次一上手就有了一个比较清晰的思维架构，以及具体需要实现的功能。“一下子画了一张大饼”，可惜胃口太小，在将近两周的准备后还是发现能够实现的功能实在有限。因此，第一版我只实现了最基本的功能要求。而在本次实验之后，我将继续完善这个项目，做到第二版，加入文件操作部分的功能，从而使得模拟度更高。  另一方面，作为C++程序设计语言的结课实验，我力求在本实验中突出C++的特性。本项目我抽象了两个类，分别是用户customer类与ATM机器类，分别对应信息的整合与功能的实现，从而充分诠释面向对象程序设计语言的精髓含义—模块化设计。此外，本项目充分使用C++的多态性，对同一函数多次重载以实现各个功能。  在本实验的构思过程中，我运用百度脑图作为辅助，从类的抽象，功能实现，执行思路与待完成四个部分绘制思维导图，从而达到辅助编程的目的。平常的学习中我发现对于一个项目总会存在一些混乱点，而运用思维导图，就可以清晰地梳理出整个流程，从而更好地抽象函数与类，提高代码的编写效率。在以后的学习中，我也将继续使用这一工具来辅助学习。  本次实验也是大学期间第一次使用Dev C++ IDE进行项目开发。通过这次实验，我对Dev C++ IDE 项目开发的操作流程更加熟悉，这也为以后的学习提供了极佳的支撑。  总而言之，积跬步以行千里，一点一点的积累才是进步的源泉。  **参考资料：**  1.《C++面向对象程序设计教程（第3版）》 陈维兴，林小茶 编著 清华大学出版社  2.《21天学通C++（第4版）》 刘蕾 编著 电子工业出版社  3.CSDN：<http://www.csdn.net> 提供部分功能实现提示  4.百度脑图：<http://naotu.baidu.com> 提供思维导图工具  5.百度文库：<http://wenku.baidu.com> 提供部分功能实现提示以及其他优秀作品参考 | | | | | |