**北京科技大学实验报告**

学院：计算机与通信工程 专业：通信工程 班级：

姓名： 学号： 实验日期：2024年 6月 5日

**实验名称：个人银行账户管理系统的改进**

**实验目的：**

完善银行账户系统，并实现以下功能：

1. 增加了日期构造异常处理机制
2. 增加了账户错误的相关异常处理
3. 增加了图形界面
4. 图形界面有当月还款的提示和收入支出统计
5. 可以查询某个月按照时间，金额排序的账户查询信息

**实验仪器：**

计算机：台式电脑

CPU：12th Gen Intel(R) Core(TM) i5-12400F 2.50 GHz

内存：16GB DDR4 2667MHz

硬盘：SSD GM7 M.2 2TB

显卡：NVIDIA GeForce RTX 4060 8GB

操作系统：Windows 11 专业版

编译器：MinGW 11.2.0 64-bit for C++

**实验内容与步骤：**

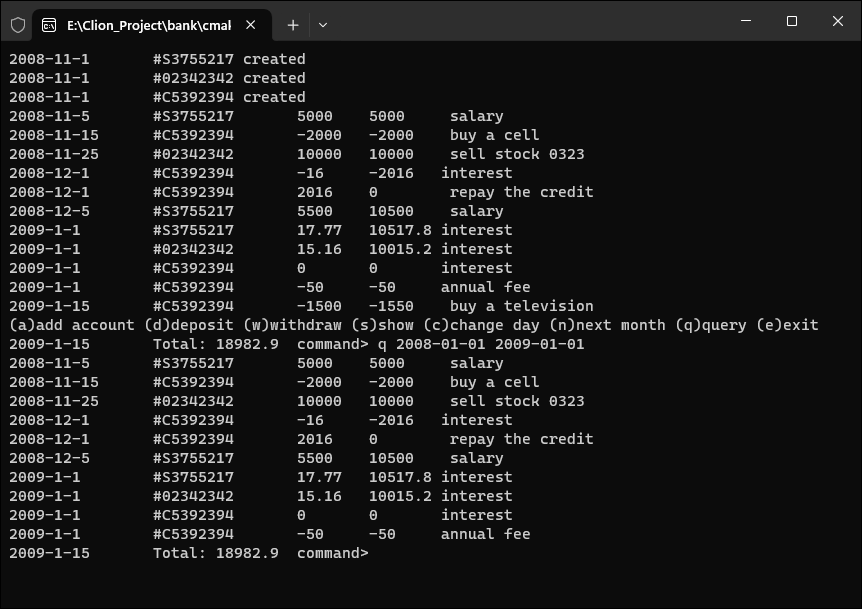
**【**功能1：增加了日期构造异常处理机制**】**

1. 设计：定义了一个新的类用来处理日期的异常，当输入的日期不合法时，抛出这个异常。
2. 实现

|  |
| --- |
| **class** **DateException** : **public** exception {  **public**:  **const** char \*what() **const** **noexcept** **override**{  **return** "日期错误，请重新输入";  }  };  下面是用来调用这个方法的：  Date::Date(int year=0,int month=0,int day=0)  :year(year),month(month),day(day){  **if** (day <= 0 || day > getMaxDay()){  **throw** DateException();  }  }; |

1. 测试
2. 测试用例1: q 2008-01-01 2009-01-01 （用户输入正确的时间）

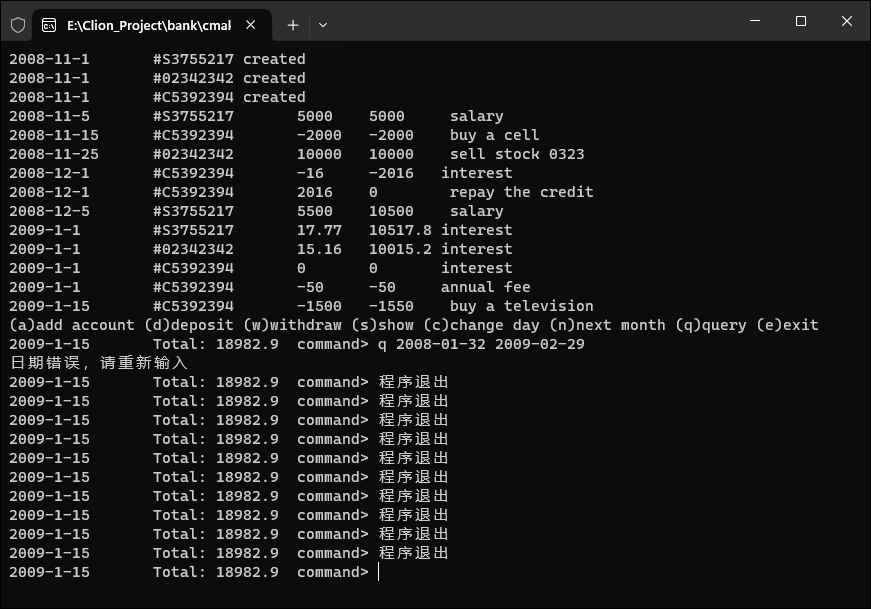
运行截图如下：



分析：该命令用于查询08年1月1日到09年1月1日的操作记录，结果表明可以成功运行

1. 测试用例 2: q 2008-01-32 2009-02-29 （用户输入了错误的时间）

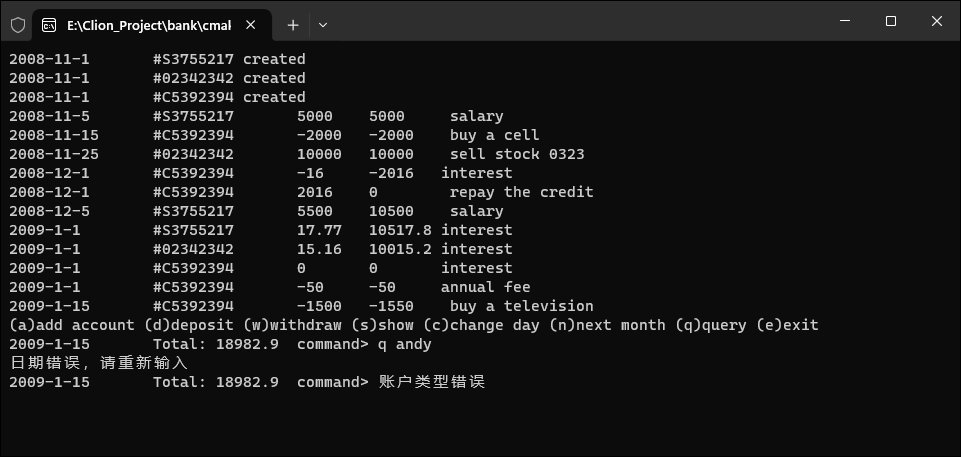
运行截图如下：



分析：由于输入了错误的日期，导致throw了异常，并且监测到并输出异常信息。

1. 测试用例3: q andy（用户输入的不是日期）

运行截图如下：



分析：由于输入的不是日期，依旧throw了该异常。

【功能2：增加了账户错误的相关异常处理】

1. 设计

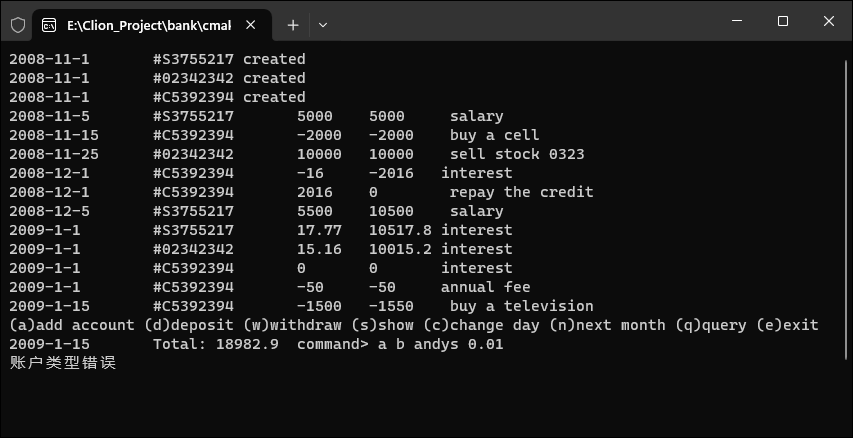
* 当增加的账户不是一个合法的参数（s 或者c）时抛出一个异常。
* 当存入或者取出的账户编号（index）不存在时抛出一个异常。
* 当从账户取钱时，余额不足，抛出一个异常。
* 当改变日期时，如果输入的日期小于今天或者超出本月最大日期，抛出异常。

1. 实现

|  |
| --- |
| *// 处理命令*  **switch** (cmd) {  **case** 'a': *// 增加账户*  **try** {  cin >> type >> id; *// 读取账户类型和ID*  **if** (type == 's' || type == 'c') {  ofs << endl << cmd << " ";  ofs << type << " " << id << " ";  } **else** {  **throw** runtime\_error("账户类型错误"); *// 如果账户类型错误，抛出异常*  }  }  **catch** (runtime\_error& err) {  cout << err.what() << endl;  }  **if** (type == 's') { *// 如果是储蓄账户*  cin >> rate; *// 读取利率*  ofs << rate;  account = **new** SavingsAccount(date, id, rate); *// 创建储蓄账户对象*  } **else** { *// 如果是信用账户*  cin >> credit >> rate >> fee; *// 读取信用额度、利率和手续费*  ofs << credit << " " << rate << " " << fee;  account = **new** CreditAccount(date, id, credit, rate, fee); *// 创建信用账户对象*  }  accounts.push\_back(account); *// 将新创建的账户添加到账户数组中*  **break**;  **case** 'd': *// 存入现金*  **try** {  cin >> index\_ >> amount; *// 读取账户索引和存入金额*  getline(cin, desc); *// 读取描述信息*  **if** (index\_ < 0 || index\_ >= accounts.size()) {  **throw** runtime\_error("账户不存在"); *// 如果账户索引无效，抛出异常*  }  accounts[index\_]->deposit(date, amount, desc); *// 执行存款操作*  ofs << endl << cmd << " ";  ofs << index\_ << " " << amount << " ";  ofs << desc;  }  **catch** (runtime\_error& err) {  cout << err.what() << endl;  }  **break**;  **case** 'w': *// 取出现金*  **try** {  cin >> index\_ >> amount; *// 读取账户索引和取出金额*  getline(cin, desc); *// 读取描述信息*  **if** (index\_ < 0 || index\_ >= accounts.size()) {  **throw** runtime\_error("账户不存在"); *// 如果账户索引无效，抛出异常*  }  accounts[index\_]->withdraw(date, amount, desc); *// 执行取款操作*  ofs << endl << cmd << " ";  ofs << index\_ << " " << amount << " ";  ofs << desc;  }  **catch** (AccountException& err) {  cout << err.what() << endl; *// 捕捉账户异常并输出错误信息*  }  **catch** (runtime\_error& err) {  cout << err.what() << endl; *// 捕捉运行时异常并输出错误信息*  }  **break**;  **case** 's': *// 查询各账户信息*  **for** (size\_t i = 0; i < accounts.size(); i++) {  cout << "[" << i << "] ";  accounts[i]->show(); *// 显示账户信息*  cout << endl;  }  **break**;  **case** 'c': *// 改变日期*  cin >> day; *// 读取新的日期*  **if** (day < date.getDay())  cout << "时间不能倒流哦" << endl; *// 日期不能倒流*  **else** **if** (day > date.getMaxDay())  cout << "这个日期太大了" << endl; *// 日期超过当月最大天数*  **else**  date = Date(date.getYear(), date.getMonth(), day); *// 更新日期*  ofs << endl << cmd << " ";  ofs << day;  **break**;  **case** 'n': *// 进入下个月*  **if** (date.getMonth() == 12)  date = Date(date.getYear() + 1, 1, 1); *// 如果当前是12月，则更新为下一年1月1日*  **else**  date = Date(date.getYear(), date.getMonth() + 1, 1); *// 否则更新为下个月1日*  *// 结算所有账户*  **for** (vector<Account\*>::iterator iter = accounts.begin(); iter != accounts.end(); ++iter)  (\*iter)->settle(date);  ofs << endl << cmd << " ";  **break**;  **case** 'q': *// 查询一段时间内的账目*  **try** {  date1 = Date::read(); *// 读取开始日期*  date2 = Date::read(); *// 读取结束日期*  Account::query(date1, date2); *// 查询账目*  }  **catch** (DateException & err) {  cout << err.what() << endl; *// 捕捉日期异常并输出错误信息*  **if** (cin.fail()) cin.clear();  }  **break**;  **default**:  cout << "命令错误" << endl; *// 命令错误*  **break**;  *// 账户错误异常*  **class** **AccountException** : **public** exception{  **private**:  Account\* account;  **public**:  **explicit** AccountException(Account \*account): account(account){};  **const** char \*what() **const** **noexcept** **override**{  string s1 = "余额不足:";  string s2 = account->getId();  string s3 = to\_string(account->getBalance());  **static** string s4 = s1+s2+"**\t**balance:"+s3;  **return** s4.c\_str();  }  };  *//储蓄账户取钱操作*  void SavingsAccount::withdraw(Date &date, double amount, string desc){  **if**(amount <= getBalance()) {  record(date, -amount, desc);  acc.change(date, getBalance());  }  **else** **throw** AccountException(**this**);  }  *//信用账户取钱操作*  void CreditAccount::withdraw(Date &date, double amount, string desc) {  **if**(credit+getBalance()-amount>=0){  record(date,-amount,desc);  acc.change(date,getBalance() > 0 ? 0 : getBalance());  }  **else**{  **throw** AccountException(**this**);  }  } |

1. 测试
2. 测试用例1: a b andys 0.01 （用户输入了账户类型）

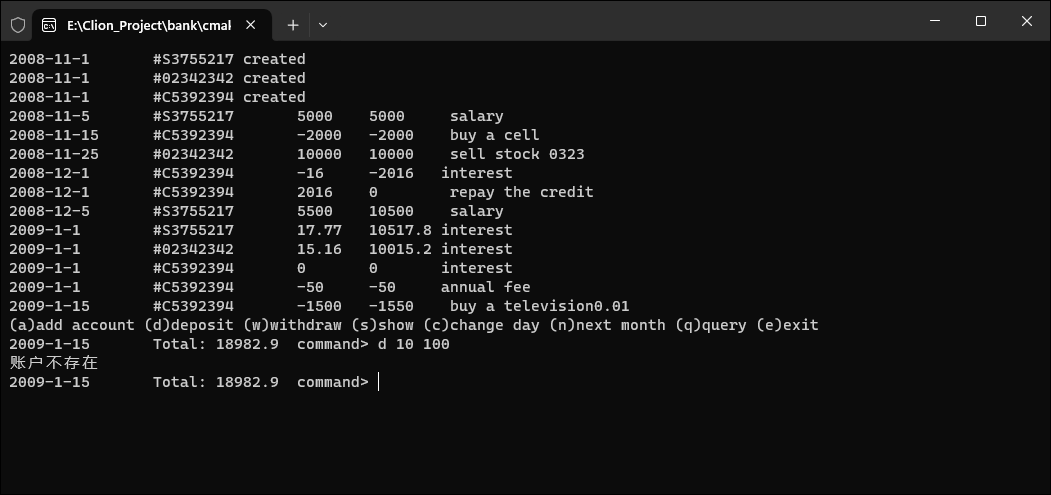
输出截图：



分析：由于输入了错误的账户类型b，抛出了异常（账户类型错误）。

1. 测试用例2:d 10 100（用户输入错误的账户序号）

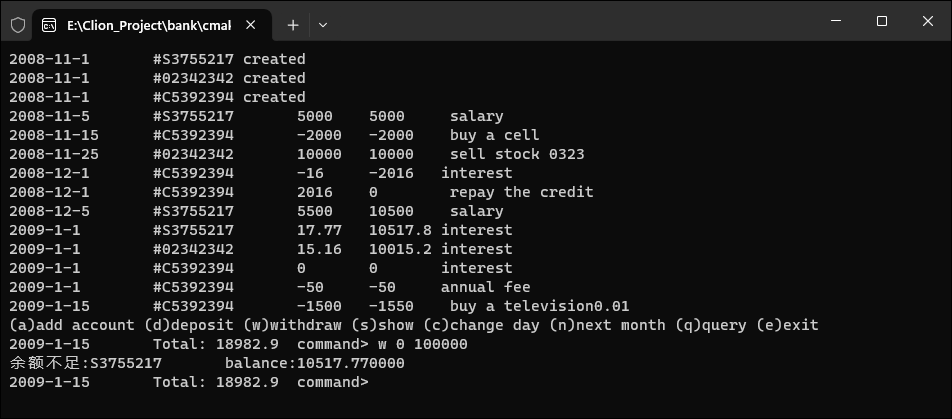
输出截图：



分析：由于输入了错误的账户序号（10），抛出了异常。w

1. 测试用例3:w 0 100000（取钱超出余额）

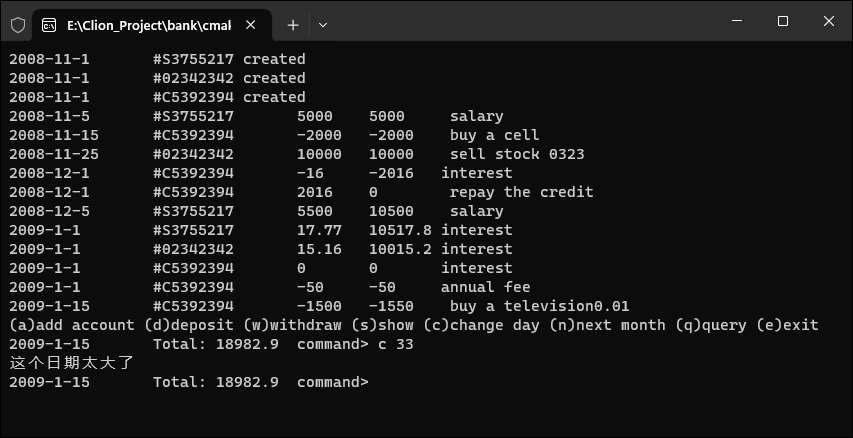
输出截图：



分析：由于取的钱数超过了账户内的余额10517.77元，抛出异常

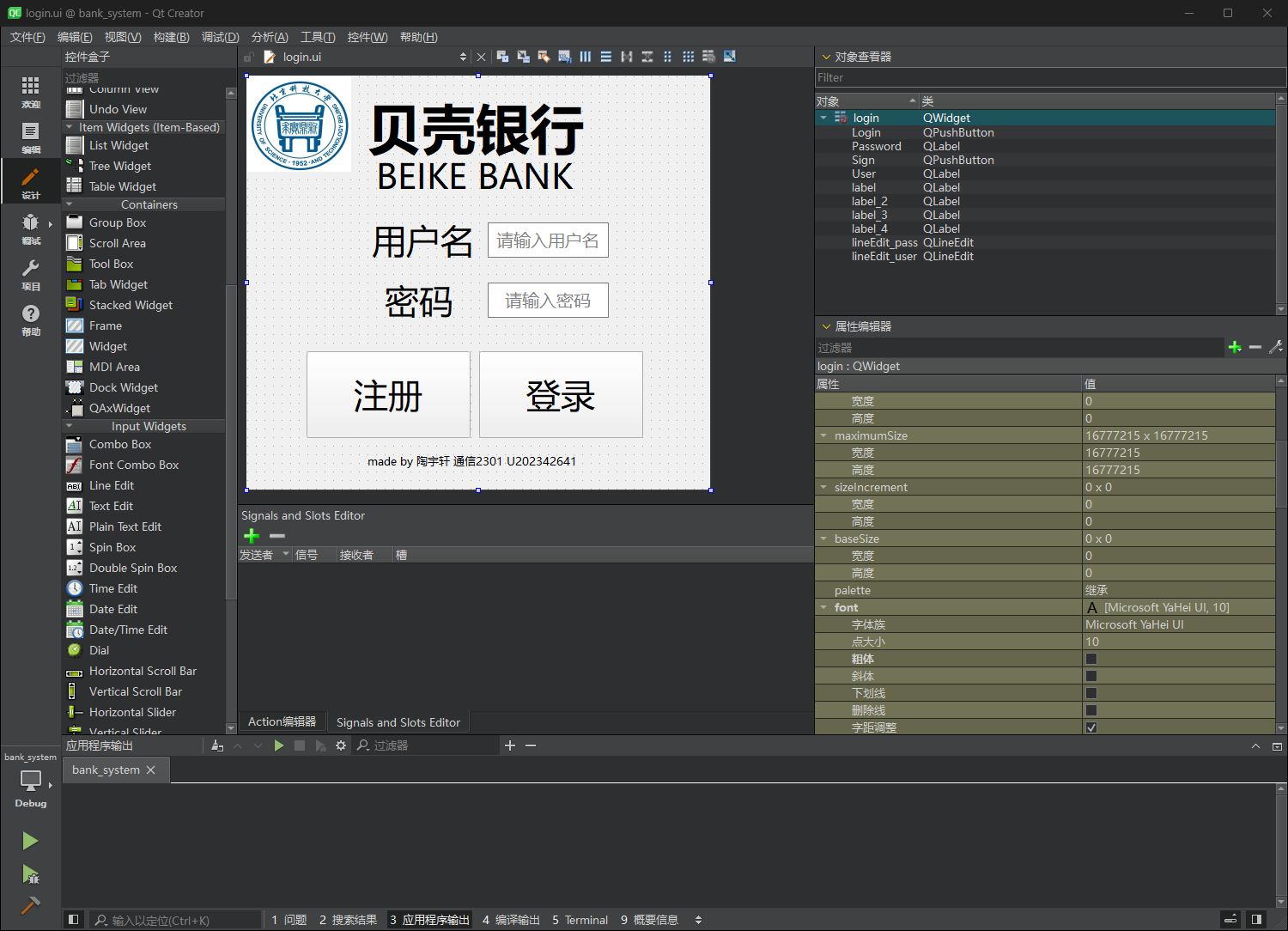
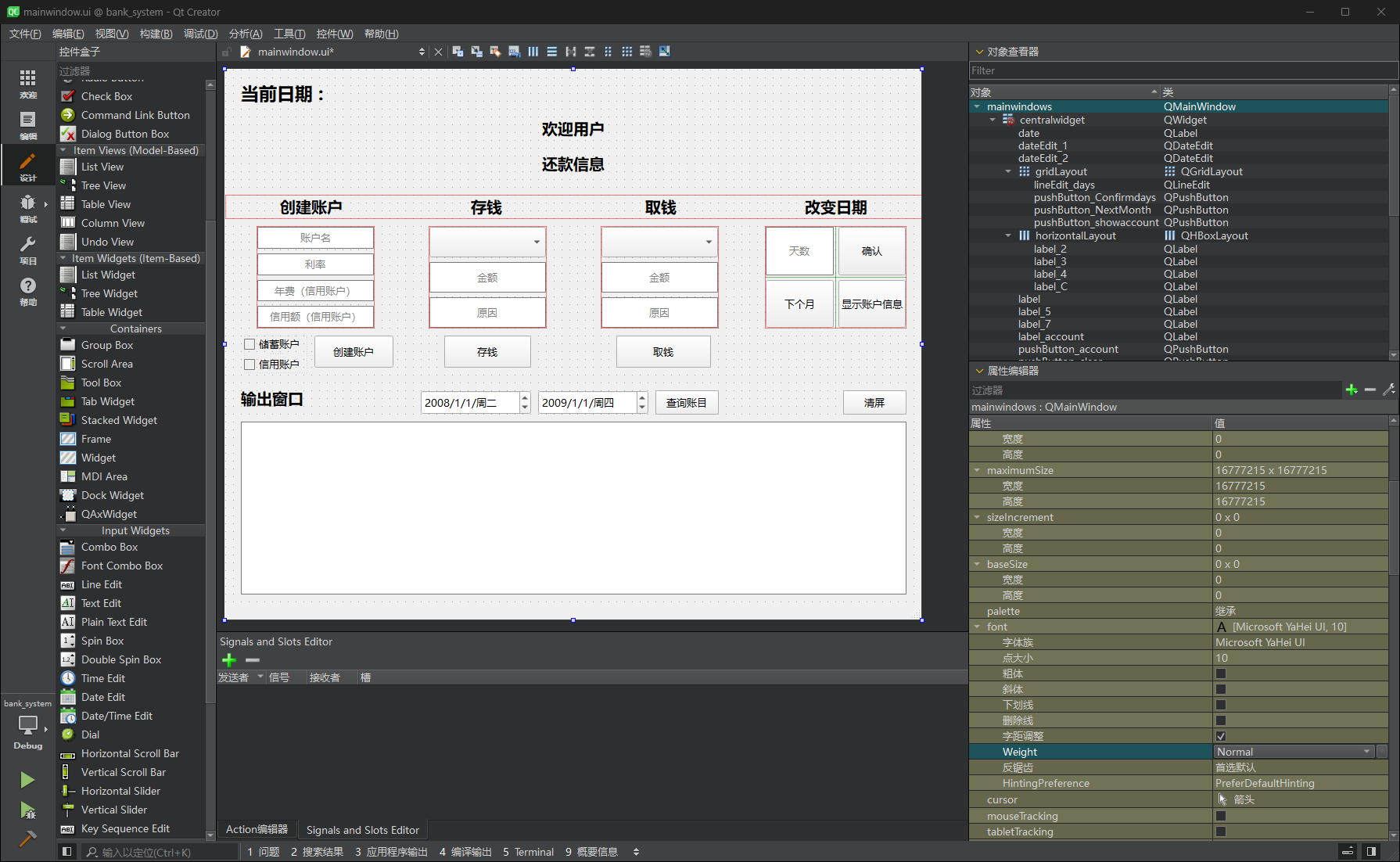
1. 测试用例4:c 33 （用户改变日期时，想要把当月天数改为50）

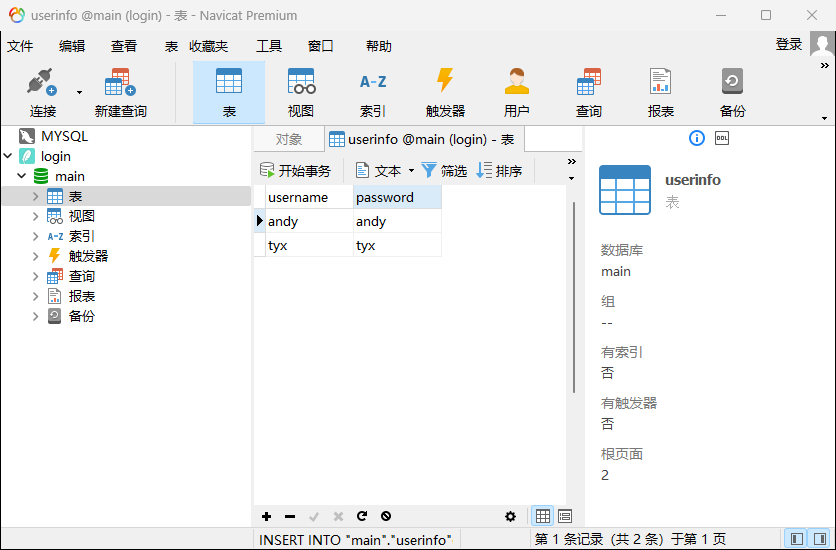
输出截图：



分析：由于33超出了1月的天数，抛出了异常。

【功能3：增加了图形页面】

1. 设计
   1. 采用qt6.6.2设计了图形页面，并且完成了相关的按钮，注册/登陆功能。以下为登录页面样式，包含了一个账户和密码的输入框和注册登陆的按钮。以下为操作页面的样式，包含欢迎语句和相关的功能输入框和按钮。当在输入框输入相应的内容，并且点按按钮，将会在下方的输出窗口中内显示对应的信息。
   2. 在QT中使用mysql 数据库存储用户登录信息，创建数据库login和一个存储用户名和密码的数据库表userinfo，存储用户信息。



1. 实现

|  |
| --- |
| **主函数**  *//main.cpp*  #include *"login.h"*  #include *<QApplication>*  #include *<QString>*  #include *<QSqlQuery>*  #include *<QDebug>*  #include *<QSqlDatabase>*  #define endl '\n'  int main(int argc, char \*argv[])  {  *//创建数据库*  QSqlDatabase db = QSqlDatabase::addDatabase("QSQLITE");  *//设置数据库名字*  db.setDatabaseName("login.db");  *//打开数据库*  **if**(db.open())  {  qDebug()<<"打开数据库成功"<<endl;  }**else**  {  qDebug()<<"打开数据库失败"<<endl;  }  *//创建一个存储user和密码的的数据库表*  QString cmd = "create table if not exists userinfo(username varchar(64),password varchar(64))";  *//执行sql语句*  QSqlQuery query;  **if**(query.exec(cmd))  {  qDebug()<<"创建成功"<<endl;  }**else**  {  qDebug()<<"创建失败"<<endl;  }  QApplication a(argc, argv);  Login w;  *//打开登录界面*  w.show();  **return** a.exec();  }  **登录注册实现**  *//login.cpp*  #include *"login.h"*  #include *"ui\_login.h"*  #include *"mainwindow.h"*  #include *<QMovie>*  #include *<QSqlQuery>*  #include *<QDebug>*  #include *<QSqlTableModel>*  #define endl '\n'  **extern** QString User;  Login::Login(QWidget \*parent)  : QWidget(parent)  , ui(**new** Ui::login)  {  ui->setupUi(**this**);  }  Login::~Login()  {  **delete** ui;  }  *//登录*  void Login::on\_Login\_clicked()  {  QString username = **this**->ui->lineEdit\_user->text();  QString password = **this**->ui->lineEdit\_pass->text();  *//创建model*  QSqlTableModel \*model = **new** QSqlTableModel;  *//绑定表*  model->setTable("userinfo");  *//过滤数据*  model->setFilter(QString("username='%1' and password='%2'").arg(username,password));  *//查询*  model->select();  *//结果*  int row = model->rowCount();  User = username;  s = **new** MainWindow;  **if**(row>0){  qDebug()<<"登录成功"<<endl;  *//打开主界面*  s->show();  **this**->hide();  }  **else**{  QMessageBox::about(NULL,"","用户名或密码错误");  s->deleteLater();  qDebug()<<"登录失败"<<endl;  }  }  *//注册*  void Login::on\_Sign\_clicked()  {  QString username = **this**->ui->lineEdit\_user->text();  QString password = **this**->ui->lineEdit\_pass->text();  QSqlQuery query;  query.prepare("select username from userinfo where username=:username");  query.bindValue(":username", username);  query.exec();  **if**(query.next())  {  QMessageBox::about(NULL,"警告","账号已注册");  }  **else**{  QString cmd = QString("insert into userinfo(username,password)values('%1','%2')").arg(username,password);  query.exec(cmd);  *//qDebug()<<"insert suc"<<endl;*  }  }  *//login.h*  #ifndef LOGIN\_H  #define LOGIN\_H  #include *"mainwindow.h"*  #include *<QWidget>*  **using** **namespace** **std**;  **namespace** **Ui** {  **class** **login**;  }  **class** **Login** : **public** QWidget  {  Q\_OBJECT  **public**:  **explicit** Login(QWidget \*parent = **nullptr**);  ~Login();  **private** slots:  void on\_Login\_clicked();  void on\_Sign\_clicked();  **private**:  Ui::login \*ui;  MainWindow \*s;  };  #endif *// LOGIN\_H*  **核心操作实现**  *//mainwindow.cpp*  #include *"mainwindow.h"*  #include *"./ui\_mainwindow.h"*  #include *"date.h"*  #include *"account.h"*  #include *"savingaccount.h"*  #include *"creditaccount.h"*  #include *<iostream>*  #include *<vector>*  #include *<fstream>*  #include *<stdexcept>*  #include *<QLabel>*  #include *<QPushButton>*  #include *<QLineEdit>*  #include *<QDebug>*  #include *<QMessageBox>*  #include *<QSqlDatabase>*  #include *<QStylePainter>*  #include *<QStylePainter>*  *// 全局变量*  QString User; *// 用户名*  Date date(2008, 11, 1); *// 起始日期*  vector<Account \*> accounts; *// 创建账户数组，初始元素个数为0*  char cmd,type;  int index\_, day;  double amount, credit, rate, fee;  string user, id, desc;  Account\* account;  Date date1, date2;  ofstream ofs;  QString account\_input[5];  QString change\_input[3];  int day\_change=0;  Date d1 ,d2;  MainWindow \*MainWindow::my = **nullptr**;  *// 文件存储格式*  *// User cmd ...*  *//读取命令*  void MainWindow::read\_cmd(){  ifstream ifs;  ifs.open("commands.txt");  **if**(!ifs.is\_open()){  qDebug()<<"无法打开文件";  **return**;  }  **while**(ifs>>cmd){  **switch** (cmd) {  **case** 'a':*//增加账户*  ifs >> user >> type >> id;  **if** (type == 's') {  ifs >> rate;  account = **new** SavingsAccount(date, id, user ,rate);  }  **else** {  ifs >> credit >> rate >> fee;  account = **new** CreditAccount(date, id, user, credit, rate, fee);  }  accounts.push\_back(account);  my->ui->comboBox\_account\_1->addItem(QString::fromStdString(id));  my->ui->comboBox\_account\_2->addItem(QString::fromStdString(id));  **break**;  **case** 'd':*//存入现金*  ifs >> index\_ >> amount;  getline(ifs,desc);  accounts[index\_]->deposit(date, amount, desc);  **break**;  **case** 'w':*//取出现金*  ifs >> index\_ >> amount ;  getline(ifs,desc);  accounts[index\_]->withdraw(date, amount, desc);  **break**;  **case** 'c':*//改变日期*  ifs >> day;  date = Date(date.getYear(), date.getMonth(), day);  **break**;  **case** 'n':*//进入下个月*  **if** (date.getMonth() == 12)  date = Date(date.getYear() + 1, 1, 1);  **else**  date = Date(date.getYear(), date.getMonth() + 1, 1);  **for** (vector<Account\*>::iterator iter = accounts.begin(); iter != accounts.end(); ++iter)  (\*iter)->settle(date);  **break**;  }  }  ifs.close();  }  **struct** **deleter** {  **template** <**class** **T**> void **operator** () (T\* p) { **delete** p; }  };  MainWindow::MainWindow(QWidget \*parent)  : QMainWindow(parent)  , ui(**new** Ui::mainwindows)  {  ui->setupUi(**this**);  my = **this**;  *//设置欢迎语*  QString theHello = "欢迎用户**\t**"+ User;  ui->label->setText(theHello);  *//读取文件*  read\_cmd();  *//显示提示信息*  show\_alert();  *//刷新日期*  refreshdate();  *//写入文件初始化*  ofs.open("commands.txt",ios::app);  }  MainWindow::~MainWindow()  {  **delete** ui;  }  *//处理命令*  void MainWindow::cmdprocess(char cmd){  *//date.show();*  *//cout << "\tTotal: " << Account::getTotal() << "\tcommand> ";*  **switch** (cmd) {  **case** 'a':*//增加账户*  **try**{  *// cin >> type >> id;*  type = account\_input[0].toStdString().c\_str()[0];  user = User.toStdString();  id = account\_input[1].toStdString();  **if**(type == 's' || type == 'c'){  ofs << endl << cmd << " " << user << " ";  ofs << type << " " << id << " ";  }  **else** **throw** runtime\_error("账户类型错误");  }  **catch** (runtime\_error& err) {  *//cout << err.what() << endl;*  }  **if** (type == 's') {  *// cin >> rate;*  rate = account\_input[3].toDouble();  ofs<<rate;  account = **new** SavingsAccount(date, id, user, rate);  }  **else** {  *// cin >> credit >> rate >> fee;*  credit = account\_input[2].toDouble();  rate = account\_input[3].toDouble();  fee = account\_input[4].toDouble();  ofs<<credit<<" "<<rate<<" "<<fee;  account = **new** CreditAccount(date, id, user, credit, rate, fee);  }  *//qDebug()<<"here\n";*  **if**(user == User.toStdString()){ *// 用户过滤*  my->ui->textBrowser\_output->append(User+ "的"+ QString::fromStdString(id)+ "账号已创建.");  my->ui->comboBox\_account\_1->addItem(QString::fromStdString(id));  my->ui->comboBox\_account\_2->addItem(QString::fromStdString(id));  accounts.push\_back(account);  }  **break**;  **case** 'd':*//存入现金*  **try**{  *// cin >> index\_ >> amount;*  index\_ = change\_input[0].toInt();  amount = change\_input[1].toDouble();  desc = change\_input[2].toStdString();  **if**(index\_ < 0 || index\_ >= accounts.size()){  **throw** runtime\_error("找不到账户<\_>");  }  accounts[index\_]->deposit(date, amount, desc);  ofs << endl << cmd << " " ;  ofs<<index\_<<" "<<amount<<" ";  ofs<<desc;  show\_alert();  **if**(accounts[index\_]->getUser() == User.toStdString()){ *// 用户过滤*  my->ui->textBrowser\_output->append(QString::fromStdString(to\_string(date.getYear())+"年"+to\_string(date.getMonth())+"月"+to\_string(date.getDay())+"日"+accounts[index\_]->getId()+"存了"+to\_string(amount)+"元,原因："+desc));  }  }  **catch** (runtime\_error& err) {  *//cout << err.what() << endl;*  QMessageBox::warning(my,tr("警告"),tr(err.what()));  }  **break**;  **case** 'w':*//取出现金*  **try**{  *//cin >> index\_ >> amount;*  index\_ = change\_input[0].toInt();  amount = change\_input[1].toDouble();  desc = change\_input[2].toStdString();  *//getline(cin, desc);*  **if**(index\_ < 0 || index\_ >= accounts.size()){  **throw** runtime\_error("找不到账户<\_>");  }  accounts[index\_]->withdraw(date, amount, desc);  ofs << endl << cmd << " ";  ofs<<index\_<<" "<<amount<<" ";  ofs<<desc;  show\_alert();  **if**(accounts[index\_]->getUser() == User.toStdString()){ *// 用户过滤*  my->ui->textBrowser\_output->append(QString::fromStdString(to\_string(date.getYear())+"年"+to\_string(date.getMonth())+"月"+to\_string(date.getDay())+"日"+accounts[index\_]->getId()+"取了"+to\_string(amount)+"元,原因："+desc));  }  }  **catch** (AccountException& err){  *//cout << err.what() << endl;*  QMessageBox::warning(my,tr("警告"),tr("你有钱吗<\_>"));  }  **catch** (runtime\_error& err) {  *//cout << err.what() << endl;*  QMessageBox::warning(my,tr("警告"),tr(err.what()));  }  **break**;  **case** 's':*//查询各账户信息*  **static** unsigned int num;  num = 1;  **for** (size\_t i = 0; i < accounts.size(); i++) {  *//cout << "[" << i << "] ";*  *//accounts[i]->show();*  *//cout << endl;*  **if**(accounts[i]->getUser() == User.toStdString()) *// 用户过滤*  my->ui->textBrowser\_output->append(QString::fromStdString("[" + to\_string(num++) + "] " + accounts[i]->show()));  }  **break**;  **case** 'c':*//改变日期*  *//cin >> day;*  day = day\_change;  **if** (day < date.getDay())  QMessageBox::warning(my,tr("警告"),tr("时间不能倒流哦 >\_<"));  *//cout << "You cannot specify a previous day";*  **else** **if** (day > date.getMaxDay())  QMessageBox::warning(my,tr("警告"),tr("这个日期不行呢 >\_<"));  *//cout << "Invalid day";*  **else**{  date = Date(date.getYear(), date.getMonth(), day);  ofs << endl << cmd << " ";  ofs<<day;  }  **break**;  **case** 'n':*//进入下个月*  **if** (date.getMonth() == 12)  date = Date(date.getYear() + 1, 1, 1);  **else**  date = Date(date.getYear(), date.getMonth() + 1, 1);  **for** (vector<Account\*>::iterator iter = accounts.begin(); iter != accounts.end(); ++iter)  (\*iter)->settle(date);  ofs << endl << cmd << " ";  **break**;  **case** 'q':*//查询一段时间内的账目*  **try**{  date1 = d1;  date2 = d2;  *//Account::query(date1, date2 ,user);*  my->ui->textBrowser\_output->append(QString::fromStdString(Account::query(date1, date2, User.toStdString())));  }  **catch** (DateException & err){  QMessageBox::warning(my,tr("警告"),tr(err.what()));  }  }  }  *//创建账户*  void MainWindow::on\_pushButton\_account\_clicked()  {  account\_input[1] = ui->lineEdit\_ID->text();  account\_input[2] = ui->lineEdit\_credit->text();  account\_input[3] = ui->lineEdit\_rate->text();  account\_input[4] = ui->lineEdit\_fee->text();  **if**(account\_input[1]==""){  QMessageBox::warning(**this**,tr("警告"),tr("请输入账号名"));  **return**;  }  **for**(**auto** &iter : accounts){  **if**(iter->getId() == account\_input[1].toStdString()){  QMessageBox::warning(**this**,tr("警告"),tr("系统中已有重名账号"));  **return**;  }  }  **if**(ui->checkBox\_S->isChecked() && !ui->checkBox\_C->isChecked()){  **if**(account\_input[3].toDouble()){  account\_input[0]='s';  cmdprocess('a');  }  **else**  QMessageBox::warning(**this**,tr("警告"),tr("请补全信息"));  }  **else** **if**(!ui->checkBox\_S->isChecked() && ui->checkBox\_C->isChecked()){  **if**(account\_input[2].toDouble() && account\_input[3].toDouble() && account\_input[4].toDouble()){  account\_input[0]='c';  cmdprocess('a');  }  **else**  QMessageBox::warning(**this**,tr("警告"),tr("请补全信息"));  }  **else**{  QMessageBox::warning(**this**,tr("警告"),tr("请选择一种账户"));  **return**;  }  }  *//存钱*  void MainWindow::on\_pushButton\_deposit\_clicked()  {  change\_input[0] = ui->comboBox\_account\_1->currentText();  **for**(unsigned int i=0; i < accounts.size(); i++){  **if**(accounts[i]->getId() == change\_input[0].toStdString()){  **if**(accounts[i]->getUser() != User.toStdString()){  QMessageBox::warning(**this**,tr("警告"),tr("这不是你的账号,请不要乱动~\_~"));  **return**;  }  change\_input[0] = (char)(i+48);  change\_input[1] = ui->lineEdit\_Depositamount->text();  change\_input[2] = ui->lineEdit\_Depositdesc->text();  MainWindow::cmdprocess('d');  **return**;  }  }  QMessageBox::warning(**this**,tr("警告"),tr("找不到该账号>\_<"));  }  *//取钱*  void MainWindow::on\_pushButton\_withdraw\_clicked()  {  change\_input[0] = ui->comboBox\_account\_2->currentText();  **for**(unsigned int i=0; i < accounts.size(); i++){  **if**(accounts[i]->getId() == change\_input[0].toStdString()){  **if**(accounts[i]->getUser() != User.toStdString()){  QMessageBox::warning(**this**,tr("警告"),tr("这不是你的账号,请不要乱动~\_~"));  **return**;  }  change\_input[0] = (char)(i+48);  change\_input[1] = ui->lineEdit\_Withdrawamount->text();  change\_input[2] = ui->lineEdit\_Withdrawdesc->text();  MainWindow::cmdprocess('w');  **return**;  }  }  QMessageBox::warning(**this**,tr("警告"),tr("找不到该账号>\_<"));  }  *//显示账户信息*  void MainWindow::on\_pushButton\_showaccount\_clicked()  {  MainWindow::cmdprocess('s');  show\_alert();  }  *//改变天数*  void MainWindow::on\_pushButton\_Confirmdays\_clicked()  {  day\_change = my->ui->lineEdit\_days->text().toInt();  **if**(day\_change){  MainWindow::cmdprocess('c');  refreshdate();  }  **else**  QMessageBox::warning(**this**,tr("警告"),tr("请输入天数>\_<"));  }  *//下个月*  void MainWindow::on\_pushButton\_NextMonth\_clicked()  {  MainWindow::cmdprocess('n');  refreshdate();  }  *//查询账目*  void MainWindow::on\_pushButton\_query\_clicked()  {  d1 = Date(ui->dateEdit\_1->date().year(), ui->dateEdit\_1->date().month(), ui->dateEdit\_1->date().day());  d2 = Date(ui->dateEdit\_2->date().year(), ui->dateEdit\_2->date().month(), ui->dateEdit\_2->date().day());  MainWindow::cmdprocess('q');  }  *//显示还款信息*  void MainWindow::show\_alert(){  double debt = 0;  double income = 0, cost = 0;  **for**(unsigned int i=0; i < accounts.size(); i++){  **if**(accounts[i]->getUser() == User.toStdString()){  **if**(accounts[i]->s\_c() && accounts[i]->get\_debt() < 0){*//信用卡*  debt -= accounts[i]->get\_debt();  }  }  }  *//cout << date.getYear()<<date.getMonth()-1<<1*  income = Account::month\_income(Date(date.getYear(),date.getMonth(),1),Date(date.getYear(),date.getMonth(),date.getMaxDay()),User.toStdString());  cost = Account::month\_cost(Date(date.getYear(),date.getMonth(),1),Date(date.getYear(),date.getMonth(),date.getMaxDay()),User.toStdString());  my->ui->label\_account->setText(QString::fromStdString("本月应还："+to\_string(debt)+" 本月收入："+to\_string(income)+" 本月支出"+to\_string(cost)));  }  *//清屏*  void MainWindow::on\_pushButton\_clear\_clicked()  {  my->ui->textBrowser\_output->setText("");  }  *//刷新日期*  void MainWindow::refreshdate(){  my->ui->date->setText(QString::fromStdString(date.show()));  }  *//mainwindow.h*  #ifndef MAINWINDOW\_H  #define MAINWINDOW\_H  #include *<QMainWindow>*  #include *<QPushButton>*  #include *<QLineEdit>*  #include *<QLabel>*  #include *<QMessageBox>*  QT\_BEGIN\_NAMESPACE  **namespace** **Ui** {  **class** **mainwindows**;  }  QT\_END\_NAMESPACE  **class** **MainWindow** : **public** QMainWindow  {  Q\_OBJECT  **public**:  MainWindow(QWidget \*parent = **nullptr**);  ~MainWindow();  **private** slots:  void on\_pushButton\_account\_clicked();  void on\_pushButton\_deposit\_clicked();  void on\_pushButton\_showaccount\_clicked();  void on\_pushButton\_withdraw\_clicked();  void on\_pushButton\_Confirmdays\_clicked();  void on\_pushButton\_NextMonth\_clicked();  void on\_pushButton\_query\_clicked();  void on\_pushButton\_clear\_clicked();  **private**:  **static** void cmdprocess(char cmd); *//处理命令*  **static** void read\_cmd();  Ui::mainwindows \*ui;  **static** MainWindow \*my;  **static** void show\_alert(); *//显示提示信息*  **static** void refreshdate(); *//更新日期*  };  #endif *// MAINWINDOW\_H*  **Ui文件**  *//mainwindow.ui*  <?xml version="1.0" encoding="UTF-8"?>  <ui version="4.0">  <**class**>mainwindows</**class**>  <widget **class**="QMainWindow" name="mainwindows">  <property name="geometry">  <rect>  <x>0</x>  <y>0</y>  <width>890</width>  <height>703</height>  </rect>  </property>  <property name="windowTitle">  <string>操作页面</string>  </property>  <widget **class**="QWidget" name="centralwidget">  <widget **class**="QPushButton" name="pushButton\_account">  <property name="geometry">  <rect>  <x>114</x>  <y>340</y>  <width>101</width>  <height>41</height>  </rect>  </property>  <property name="text">  <string>创建账户</string>  </property>  </widget>  <widget **class**="QWidget" name="layoutWidget">  <property name="geometry">  <rect>  <x>40</x>  <y>200</y>  <width>151</width>  <height>131</height>  </rect>  </property>  <layout **class**="QVBoxLayout" name="verticalLayout">  <item>  <widget **class**="QLineEdit" name="lineEdit\_ID">  <property name="enabled">  <bool>true</bool>  </property>  <property name="sizePolicy">  <sizepolicy hsizetype="Expanding" vsizetype="Expanding">  <horstretch>0</horstretch>  <verstretch>0</verstretch>  </sizepolicy>  </property>  <property name="text">  <string/>  </property>  <property name="alignment">  <set>Qt::AlignCenter</set>  </property>  <property name="placeholderText">  <string>账户名</string>  </property>  </widget>  </item>  <item>  <widget **class**="QLineEdit" name="lineEdit\_rate">  <property name="sizePolicy">  <sizepolicy hsizetype="Expanding" vsizetype="Expanding">  <horstretch>0</horstretch>  <verstretch>0</verstretch>  </sizepolicy>  </property>  <property name="text">  <string/>  </property>  <property name="alignment">  <set>Qt::AlignCenter</set>  </property>  <property name="placeholderText">  <string>利率</string>  </property>  </widget>  </item>  <item>  <widget **class**="QLineEdit" name="lineEdit\_fee">  <property name="sizePolicy">  <sizepolicy hsizetype="Expanding" vsizetype="Expanding">  <horstretch>0</horstretch>  <verstretch>0</verstretch>  </sizepolicy>  </property>  <property name="text">  <string/>  </property>  <property name="alignment">  <set>Qt::AlignCenter</set>  </property>  <property name="placeholderText">  <string>年费（信用账户）</string>  </property>  </widget>  </item>  <item>  <widget **class**="QLineEdit" name="lineEdit\_credit">  <property name="sizePolicy">  <sizepolicy hsizetype="Expanding" vsizetype="Expanding">  <horstretch>0</horstretch>  <verstretch>0</verstretch>  </sizepolicy>  </property>  <property name="text">  <string/>  </property>  <property name="alignment">  <set>Qt::AlignCenter</set>  </property>  <property name="placeholderText">  <string>信用额（信用账户）</string>  </property>  </widget>  </item>  </layout>  </widget>  <widget **class**="QPushButton" name="pushButton\_deposit">  <property name="geometry">  <rect>  <x>280</x>  <y>340</y>  <width>111</width>  <height>41</height>  </rect>  </property>  <property name="text">  <string>存钱</string>  </property>  </widget>  <widget **class**="QWidget" name="layoutWidget\_2">  <property name="geometry">  <rect>  <x>260</x>  <y>200</y>  <width>151</width>  <height>131</height>  </rect>  </property>  <layout **class**="QVBoxLayout" name="verticalLayout\_2">  <item>  <widget **class**="QComboBox" name="comboBox\_account\_1">  <property name="sizePolicy">  <sizepolicy hsizetype="Expanding" vsizetype="Expanding">  <horstretch>0</horstretch>  <verstretch>0</verstretch>  </sizepolicy>  </property>  <property name="layoutDirection">  <**enum**>Qt::LeftToRight</**enum**>  </property>  <property name="insertPolicy">  <**enum**>QComboBox::InsertAtBottom</**enum**>  </property>  <property name="placeholderText">  <string/>  </property>  </widget>  </item>  <item>  <widget **class**="QLineEdit" name="lineEdit\_Depositamount">  <property name="sizePolicy">  <sizepolicy hsizetype="Expanding" vsizetype="Expanding">  <horstretch>0</horstretch>  <verstretch>0</verstretch>  </sizepolicy>  </property>  <property name="text">  <string/>  </property>  <property name="alignment">  <set>Qt::AlignCenter</set>  </property>  <property name="placeholderText">  <string>金额</string>  </property>  </widget>  </item>  <item>  <widget **class**="QLineEdit" name="lineEdit\_Depositdesc">  <property name="sizePolicy">  <sizepolicy hsizetype="Expanding" vsizetype="Expanding">  <horstretch>0</horstretch>  <verstretch>0</verstretch>  </sizepolicy>  </property>  <property name="text">  <string/>  </property>  <property name="alignment">  <set>Qt::AlignCenter</set>  </property>  <property name="placeholderText">  <string>原因</string>  </property>  </widget>  </item>  </layout>  </widget>  <widget **class**="QPushButton" name="pushButton\_withdraw">  <property name="geometry">  <rect>  <x>500</x>  <y>340</y>  <width>121</width>  <height>41</height>  </rect>  </property>  <property name="text">  <string>取钱</string>  </property>  </widget>  <widget **class**="QWidget" name="layoutWidget\_3">  <property name="geometry">  <rect>  <x>480</x>  <y>200</y>  <width>151</width>  <height>131</height>  </rect>  </property>  <property name="maximumSize">  <size>  <width>16000000</width>  <height>16000000</height>  </size>  </property>  <layout **class**="QVBoxLayout" name="verticalLayout\_3">  <item>  <widget **class**="QComboBox" name="comboBox\_account\_2">  <property name="sizePolicy">  <sizepolicy hsizetype="Preferred" vsizetype="Expanding">  <horstretch>0</horstretch>  <verstretch>0</verstretch>  </sizepolicy>  </property>  <property name="layoutDirection">  <**enum**>Qt::LeftToRight</**enum**>  </property>  <property name="editable">  <bool>false</bool>  </property>  <property name="placeholderText">  <string/>  </property>  </widget>  </item>  <item>  <widget **class**="QLineEdit" name="lineEdit\_Withdrawamount">  <property name="sizePolicy">  <sizepolicy hsizetype="Expanding" vsizetype="Expanding">  <horstretch>0</horstretch>  <verstretch>0</verstretch>  </sizepolicy>  </property>  <property name="maximumSize">  <size>  <width>16000000</width>  <height>16000000</height>  </size>  </property>  <property name="text">  <string/>  </property>  <property name="alignment">  <set>Qt::AlignCenter</set>  </property>  <property name="placeholderText">  <string>金额</string>  </property>  </widget>  </item>  <item>  <widget **class**="QLineEdit" name="lineEdit\_Withdrawdesc">  <property name="sizePolicy">  <sizepolicy hsizetype="Expanding" vsizetype="Expanding">  <horstretch>0</horstretch>  <verstretch>0</verstretch>  </sizepolicy>  </property>  <property name="text">  <string/>  </property>  <property name="alignment">  <set>Qt::AlignCenter</set>  </property>  <property name="placeholderText">  <string>原因</string>  </property>  </widget>  </item>  </layout>  </widget>  <widget **class**="QPushButton" name="pushButton\_query">  <property name="geometry">  <rect>  <x>550</x>  <y>410</y>  <width>81</width>  <height>31</height>  </rect>  </property>  <property name="text">  <string>查询账目</string>  </property>  </widget>  <widget **class**="QDateEdit" name="dateEdit\_1">  <property name="geometry">  <rect>  <x>250</x>  <y>410</y>  <width>141</width>  <height>31</height>  </rect>  </property>  <property name="date">  <date>  <year>2008</year>  <month>1</month>  <day>1</day>  </date>  </property>  </widget>  <widget **class**="QDateEdit" name="dateEdit\_2">  <property name="geometry">  <rect>  <x>400</x>  <y>410</y>  <width>141</width>  <height>31</height>  </rect>  </property>  <property name="accelerated">  <bool>false</bool>  </property>  <property name="showGroupSeparator" stdset="0">  <bool>false</bool>  </property>  <property name="date">  <date>  <year>2009</year>  <month>1</month>  <day>1</day>  </date>  </property>  </widget>  <widget **class**="QTextBrowser" name="textBrowser\_output">  <property name="geometry">  <rect>  <x>20</x>  <y>450</y>  <width>851</width>  <height>221</height>  </rect>  </property>  <property name="font">  <font>  <pointsize>13</pointsize>  </font>  </property>  </widget>  <widget **class**="QLabel" name="label">  <property name="geometry">  <rect>  <x>10</x>  <y>50</y>  <width>871</width>  <height>51</height>  </rect>  </property>  <property name="font">  <font>  <pointsize>15</pointsize>  <bold>true</bold>  </font>  </property>  <property name="lineWidth">  <number>2</number>  </property>  <property name="text">  <string>欢迎用户</string>  </property>  <property name="alignment">  <set>Qt::AlignCenter</set>  </property>  </widget>  <widget **class**="QWidget" name="layoutWidget">  <property name="geometry">  <rect>  <x>690</x>  <y>200</y>  <width>181</width>  <height>131</height>  </rect>  </property>  <layout **class**="QGridLayout" name="gridLayout">  <item row="0" column="0">  <widget **class**="QLineEdit" name="lineEdit\_days">  <property name="sizePolicy">  <sizepolicy hsizetype="Expanding" vsizetype="Expanding">  <horstretch>0</horstretch>  <verstretch>0</verstretch>  </sizepolicy>  </property>  <property name="alignment">  <set>Qt::AlignCenter</set>  </property>  <property name="placeholderText">  <string>天数</string>  </property>  </widget>  </item>  <item row="1" column="0">  <widget **class**="QPushButton" name="pushButton\_NextMonth">  <property name="sizePolicy">  <sizepolicy hsizetype="Expanding" vsizetype="Expanding">  <horstretch>0</horstretch>  <verstretch>0</verstretch>  </sizepolicy>  </property>  <property name="text">  <string>下个月</string>  </property>  </widget>  </item>  <item row="0" column="1">  <widget **class**="QPushButton" name="pushButton\_Confirmdays">  <property name="sizePolicy">  <sizepolicy hsizetype="Expanding" vsizetype="Expanding">  <horstretch>0</horstretch>  <verstretch>0</verstretch>  </sizepolicy>  </property>  <property name="text">  <string>确认</string>  </property>  </widget>  </item>  <item row="1" column="1">  <widget **class**="QPushButton" name="pushButton\_showaccount">  <property name="sizePolicy">  <sizepolicy hsizetype="Expanding" vsizetype="Expanding">  <horstretch>0</horstretch>  <verstretch>0</verstretch>  </sizepolicy>  </property>  <property name="text">  <string>显示账户信息</string>  </property>  </widget>  </item>  </layout>  </widget>  <widget **class**="QLabel" name="label\_5">  <property name="geometry">  <rect>  <x>20</x>  <y>400</y>  <width>81</width>  <height>41</height>  </rect>  </property>  <property name="font">  <font>  <pointsize>15</pointsize>  <bold>true</bold>  </font>  </property>  <property name="text">  <string>输出窗口</string>  </property>  <property name="alignment">  <set>Qt::AlignCenter</set>  </property>  </widget>  <widget **class**="QPushButton" name="pushButton\_clear">  <property name="geometry">  <rect>  <x>790</x>  <y>410</y>  <width>81</width>  <height>31</height>  </rect>  </property>  <property name="text">  <string>清屏</string>  </property>  </widget>  <widget **class**="QWidget" name="layoutWidget">  <property name="geometry">  <rect>  <x>0</x>  <y>160</y>  <width>891</width>  <height>31</height>  </rect>  </property>  <layout **class**="QHBoxLayout" name="horizontalLayout">  <item>  <widget **class**="QLabel" name="label\_C">  <property name="font">  <font>  <pointsize>15</pointsize>  <bold>true</bold>  </font>  </property>  <property name="frameShape">  <**enum**>QFrame::NoFrame</**enum**>  </property>  <property name="frameShadow">  <**enum**>QFrame::Plain</**enum**>  </property>  <property name="text">  <string>创建账户</string>  </property>  <property name="alignment">  <set>Qt::AlignCenter</set>  </property>  </widget>  </item>  <item>  <widget **class**="QLabel" name="label\_2">  <property name="font">  <font>  <pointsize>15</pointsize>  <bold>true</bold>  </font>  </property>  <property name="frameShape">  <**enum**>QFrame::NoFrame</**enum**>  </property>  <property name="text">  <string>存钱</string>  </property>  <property name="alignment">  <set>Qt::AlignCenter</set>  </property>  </widget>  </item>  <item>  <widget **class**="QLabel" name="label\_3">  <property name="font">  <font>  <pointsize>15</pointsize>  <bold>true</bold>  </font>  </property>  <property name="text">  <string>取钱</string>  </property>  <property name="alignment">  <set>Qt::AlignCenter</set>  </property>  </widget>  </item>  <item>  <widget **class**="QLabel" name="label\_4">  <property name="font">  <font>  <pointsize>15</pointsize>  <italic>false</italic>  <bold>true</bold>  </font>  </property>  <property name="text">  <string>改变日期</string>  </property>  <property name="alignment">  <set>Qt::AlignCenter</set>  </property>  </widget>  </item>  </layout>  </widget>  <widget **class**="QSplitter" name="splitter">  <property name="geometry">  <rect>  <x>24</x>  <y>340</y>  <width>81</width>  <height>48</height>  </rect>  </property>  <property name="orientation">  <**enum**>Qt::Vertical</**enum**>  </property>  <widget **class**="QCheckBox" name="checkBox\_S">  <property name="text">  <string>储蓄账户</string>  </property>  </widget>  <widget **class**="QCheckBox" name="checkBox\_C">  <property name="text">  <string>信用账户</string>  </property>  </widget>  </widget>  <widget **class**="QLabel" name="label\_7">  <property name="geometry">  <rect>  <x>20</x>  <y>10</y>  <width>111</width>  <height>41</height>  </rect>  </property>  <property name="font">  <font>  <pointsize>17</pointsize>  <bold>true</bold>  </font>  </property>  <property name="text">  <string>当前日期 :</string>  </property>  </widget>  <widget **class**="QLabel" name="date">  <property name="geometry">  <rect>  <x>130</x>  <y>10</y>  <width>221</width>  <height>41</height>  </rect>  </property>  <property name="font">  <font>  <pointsize>17</pointsize>  <bold>true</bold>  </font>  </property>  <property name="text">  <string/>  </property>  </widget>  <widget **class**="QLabel" name="label\_account">  <property name="geometry">  <rect>  <x>10</x>  <y>100</y>  <width>871</width>  <height>41</height>  </rect>  </property>  <property name="font">  <font>  <pointsize>15</pointsize>  <bold>true</bold>  </font>  </property>  <property name="text">  <string>还款信息</string>  </property>  <property name="alignment">  <set>Qt::AlignCenter</set>  </property>  </widget>  </widget>  <widget **class**="QStatusBar" name="statusbar"/>  </widget>  <resources/>  <connections/>  </ui>  *//login.ui*  <?xml version="1.0" encoding="UTF-8"?>  <ui version="4.0">  <**class**>login</**class**>  <widget **class**="QWidget" name="login">  <property name="geometry">  <rect>  <x>0</x>  <y>0</y>  <width>539</width>  <height>481</height>  </rect>  </property>  <property name="font">  <font>  <fontweight>Medium</fontweight>  </font>  </property>  <property name="windowTitle">  <string>登录页面</string>  </property>  <widget **class**="QPushButton" name="Sign">  <property name="geometry">  <rect>  <x>69</x>  <y>320</y>  <width>191</width>  <height>101</height>  </rect>  </property>  <property name="font">  <font>  <pointsize>30</pointsize>  <fontweight>Medium</fontweight>  </font>  </property>  <property name="styleSheet">  <string notr="true"/>  </property>  <property name="text">  <string>注册</string>  </property>  <property name="icon">  <iconset theme="address-book-new"/>  </property>  <property name="autoDefault">  <bool>false</bool>  </property>  </widget>  <widget **class**="QLabel" name="label">  <property name="geometry">  <rect>  <x>150</x>  <y>90</y>  <width>229</width>  <height>51</height>  </rect>  </property>  <property name="font">  <font>  <pointsize>30</pointsize>  <italic>false</italic>  <underline>false</underline>  <fontweight>Medium</fontweight>  </font>  </property>  <property name="text">  <string>BEIKE BANK</string>  </property>  </widget>  <widget **class**="QLabel" name="label\_2">  <property name="geometry">  <rect>  <x>140</x>  <y>20</y>  <width>252</width>  <height>80</height>  </rect>  </property>  <property name="font">  <font>  <pointsize>47</pointsize>  <bold>true</bold>  </font>  </property>  <property name="text">  <string>贝壳银行</string>  </property>  </widget>  <widget **class**="QPushButton" name="Login">  <property name="geometry">  <rect>  <x>270</x>  <y>320</y>  <width>191</width>  <height>101</height>  </rect>  </property>  <property name="font">  <font>  <pointsize>30</pointsize>  <fontweight>Medium</fontweight>  </font>  </property>  <property name="styleSheet">  <string notr="true"/>  </property>  <property name="text">  <string>登录</string>  </property>  </widget>  <widget **class**="QLabel" name="Password">  <property name="geometry">  <rect>  <x>140</x>  <y>240</y>  <width>121</width>  <height>41</height>  </rect>  </property>  <property name="font">  <font>  <pointsize>30</pointsize>  <fontweight>Medium</fontweight>  </font>  </property>  <property name="text">  <string>密码</string>  </property>  <property name="alignment">  <set>Qt::AlignCenter</set>  </property>  </widget>  <widget **class**="QLineEdit" name="lineEdit\_pass">  <property name="geometry">  <rect>  <x>280</x>  <y>240</y>  <width>141</width>  <height>41</height>  </rect>  </property>  <property name="font">  <font>  <pointsize>15</pointsize>  <fontweight>Medium</fontweight>  </font>  </property>  <property name="echoMode">  <**enum**>QLineEdit::Password</**enum**>  </property>  <property name="alignment">  <set>Qt::AlignCenter</set>  </property>  <property name="placeholderText">  <string>请输入密码</string>  </property>  </widget>  <widget **class**="QLabel" name="User">  <property name="geometry">  <rect>  <x>140</x>  <y>170</y>  <width>131</width>  <height>41</height>  </rect>  </property>  <property name="font">  <font>  <pointsize>30</pointsize>  <fontweight>Medium</fontweight>  </font>  </property>  <property name="frameShape">  <**enum**>QFrame::NoFrame</**enum**>  </property>  <property name="frameShadow">  <**enum**>QFrame::Plain</**enum**>  </property>  <property name="text">  <string>用户名</string>  </property>  <property name="alignment">  <set>Qt::AlignCenter</set>  </property>  </widget>  <widget **class**="QLineEdit" name="lineEdit\_user">  <property name="geometry">  <rect>  <x>280</x>  <y>170</y>  <width>141</width>  <height>41</height>  </rect>  </property>  <property name="font">  <font>  <pointsize>15</pointsize>  <fontweight>Medium</fontweight>  </font>  </property>  <property name="alignment">  <set>Qt::AlignCenter</set>  </property>  <property name="placeholderText">  <string>请输入用户名</string>  </property>  </widget>  <widget **class**="QLabel" name="label\_3">  <property name="geometry">  <rect>  <x>0</x>  <y>0</y>  <width>121</width>  <height>111</height>  </rect>  </property>  <property name="text">  <string/>  </property>  <property name="pixmap">  <pixmap resource="icon.qrc">:/C:/Users/Administrator/Desktop/R.jpg</pixmap>  </property>  <property name="scaledContents">  <bool>true</bool>  </property>  </widget>  <widget **class**="QLabel" name="label\_4">  <property name="geometry">  <rect>  <x>140</x>  <y>440</y>  <width>244</width>  <height>16</height>  </rect>  </property>  <property name="font">  <font>  <fontweight>Medium</fontweight>  </font>  </property>  <property name="text">  <string>made by 陶宇轩 通信2301 U202342641</string>  </property>  </widget>  </widget>  <resources>  <include location="icon.qrc"/>  </resources>  <connections/>  </ui> |

1. 测试
2. 测试用例1:在登陆页面输入账号：tyx密码：123456点击注册两次再点击登陆。

输出截图：

图形用户界面

低可信度描述已自动生成

分析：点击第二次时，由于tyx这个账户已经创建过了，所以不能再注册，弹出提示框。



分析：点击登陆，tyx成功登陆，显示用户页面，显示还款收入支出信息。

1. 测试用例2：账户名：tyxs，不是信用账户，利率为0.015

输出截图：



分析：如图所示，创建了tyxs储蓄账户。

1. 测试用例3:账户名：tyxc 是信用账户，利率为0.1 透支额度5000元，年费为10元

输出截图：



分析：如图所示，tyxc账号已创建

1. 测试用例4:再输入一个同名的tyxs账号，点击“创建账户”。

输出截图：

**图形用户界面, 应用程序

描述已自动生成**

分析：由于系统中有重名账户，弹出提示框

1. 测试用例5: 创建tyx信用账户，只输入利率为0.1

输出截图：

图形用户界面

描述已自动生成

分析：由于账户信息不全，弹出提示框

1. 测试用例6:选择存钱账户名：tyxs，存钱金额100，存钱原因：sell books，点击存钱。

输出截图：



分析：如图。

1. 测试用例6:存钱账户名：tyxc，存钱金额：1000，存钱原因：sell cars。点击存钱



分析：如图。

1. 测试用例7:从tyxc取4000元，原因，buy a house

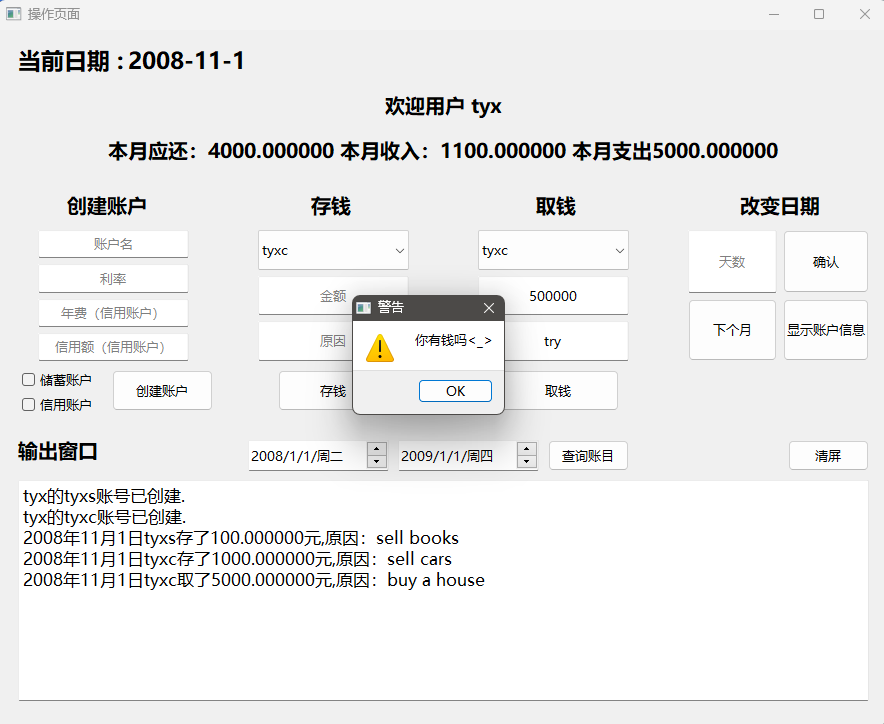
输出截图：



分析：如图。

1. 测试用例8:从tyxc账户取500000元，取钱原因：try

输出截图：



分析：账户余额加上透支额度不够取的金额，弹出提示框。

1. 测试用例9: 点击清屏，再点击“查询账户信息”

输出截图：

****

分析：如图。

1. 测试用例10:点击进入下个月，再显示账户信息

输出截图：

****

分析：进入下个月后，左上角日期发生变化，并结算了上个月的利息，进入下一年后，扣除了信用卡的年费。

1. 测试用例11: 2008-1-1～2009-1-1 点击查询时间段账目

输出截图：



分析：如图所示

【功能4：图形界面有当月还款的提示和收入支出统计】

1. 设计

在上方显示本月支出和本月收入，用一个函数来统计，并且调用qt的方法来显示在文本框上。

1. 实现

|  |
| --- |
| *//显示还款信息*  void MainWindow::show\_alert(){  double debt = 0;  double income = 0, cost = 0;  **for**(unsigned int i=0; i < accounts.size(); i++){  **if**(accounts[i]->getUser() == User.toStdString()){  **if**(accounts[i]->s\_c() && accounts[i]->get\_debt() < 0){*//信用卡*  debt -= accounts[i]->get\_debt();  }  }  }  *//cout << date.getYear()<<date.getMonth()-1<<1*  income = Account::month\_income(Date(date.getYear(),date.getMonth(),1),Date(date.getYear(),date.getMonth(),date.getMaxDay()),User.toStdString());  cost = Account::month\_cost(Date(date.getYear(),date.getMonth(),1),Date(date.getYear(),date.getMonth(),date.getMaxDay()),User.toStdString());  my->ui->label\_account->setText(QString::fromStdString("本月应还："+to\_string(debt)+" 本月收入："+to\_string(income)+" 本月支出"+to\_string(cost)));  } |

3.测试



分析：如图所示。

【功能5：可以查询某个月按照时间排序的账户查询信息】

1. 设计：与银行1～6的类似，但是时间输入改为了从qt提供的时间选择控件输入，然后点击查询时间段账目，即可。

图片包含 文本

描述已自动生成

1. 实现

|  |
| --- |
| *//查询账目*  void MainWindow::on\_pushButton\_query\_clicked()  {  d1 = Date(ui->dateEdit\_1->date().year(), ui->dateEdit\_1->date().month(), ui->dateEdit\_1->date().day());  d2 = Date(ui->dateEdit\_2->date().year(), ui->dateEdit\_2->date().month(), ui->dateEdit\_2->date().day());  MainWindow::cmdprocess('q');  } |

1. 测试：查询2009年2月的信息；

输出截图：



分析：如图所示。

**实验结论与感想**

这次的实验让我对C++类的运用有了更加深刻的理解，

也让我学到了更多有关QT的知识。