组长:魏志航

组员:兰天阳，彭尹垚，张世纪，叶翔昊，谢泽川

总体情况：

两位成员完成任务，四位成员未完成java连接数据库的操作，其中还有两位未理解usb编程任务的题意，因此未完成。

**魏志航**

今天主要的学习内容有：

1.java类的三大特性：封装、继承和多态

2.java继承中的抽象类和接口

3.java多态对象的转型

4.mysql的基本操作

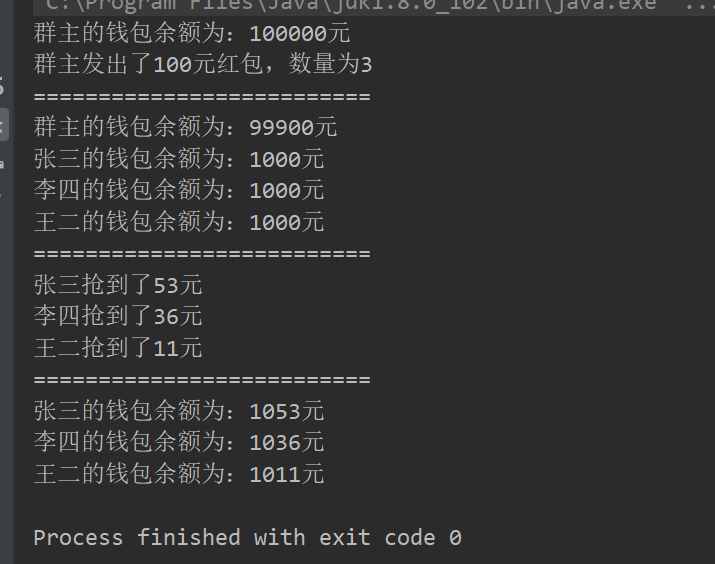
其他任务完成的有：

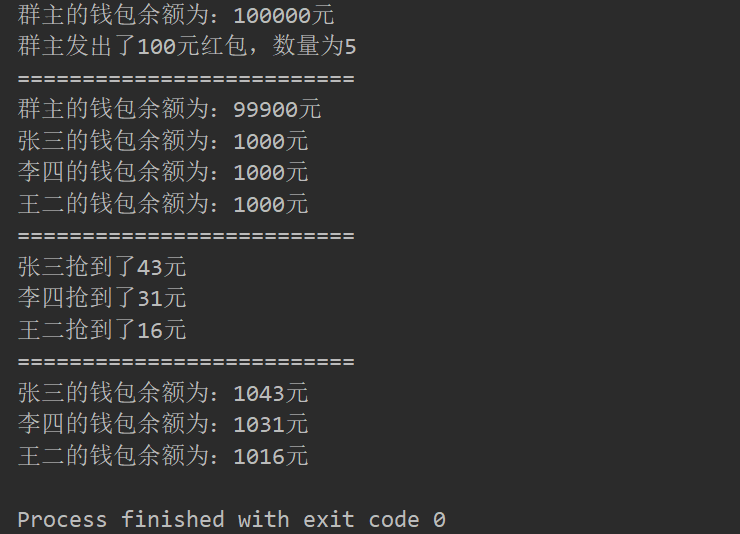
1.安装mysql

2.安装navicat

学习任务完成如下：

1.抢红包





public class Day5 {  
 //static int redpacket=0;  
 public static void main(String[] args) {  
 //创建群主  
 Manager man =new Manager("群主");  
  
 //创建成员  
 Member mem1 = new Member("张三");  
 Member mem2 = new Member("李四");  
 Member mem3 = new Member("王二");  
  
 //群主发红包  
 man.show();  
 man.send(100,5);  
 System.*out*.println("==========================");  
 man.show();  
 mem1.show();  
 mem2.show();  
 mem3.show();  
 System.*out*.println("==========================");  
 mem1.receive();  
 mem2.receive();  
 mem3.receive();  
 System.*out*.println("==========================");  
 mem1.show();  
 mem2.show();  
 mem3.show();  
  
 }  
  
}

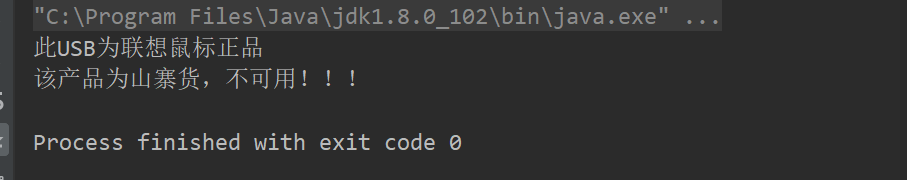
import java.util.Random;  
  
public class Member extends Person  
{  
 public Member(String name)  
 {  
 this.name=name;  
 this.money=1000;  
 }  
 public void receive()  
 {  
 if(*rednum*>0) {  
 Random ran=new Random();  
 int number= ran.nextInt(*rednum*);  
 int redd =*redmoney*.get(number);  
 *redmoney*.remove(number);  
 this.money += redd;  
 System.*out*.println(name+"抢到了"+redd+"元");  
 *rednum*--;  
 }  
 else  
 {  
 System.*out*.println("红包已经抢没了");  
 }  
  
 }  
  
  
}

import java.util.ArrayList;  
  
public abstract class Person implements Worker  
{  
 int money;  
 String name;  
 static ArrayList<Integer> *redmoney*= new ArrayList<Integer>();  
 static int *redpacket*=0;  
 static int *rednum*=0;  
 @Override  
 public void show() {  
 System.*out*.println(name+"的钱包余额为："+money+"元");  
 }  
}

import java.util.Random;  
  
public class Manager extends Person  
{  
  
 public Manager(String name)  
 {  
 this.name=name;  
 this.money=100000;  
 }  
  
 public void send(int sum,int num)  
 {  
 this.money-=sum;  
 *redpacket*=sum;  
 *rednum*=num;  
 System.*out*.println(name+"发出了"+sum+"元红包，数量为"+num);  
 for(int i=0;i<*rednum*;i++)  
 {  
 if(num>1) {  
 Random ran = new Random();  
 int redd = ran.nextInt(sum-num+1)+1;  
 if(i==1)//用随机数套随机数的办法防止第一个分配的随机数数值过大的概率太大  
 {  
 int randd= ran.nextInt(sum-num+1)+num;  
 redd = ran.nextInt(randd-num+1)+1;  
 }  
 *redmoney*.add(redd);  
 sum -= redd;  
 num--;  
 }  
 else if (num==1)  
 {  
 *redmoney*.add(sum);  
 sum =0;  
 num--;  
 }  
  
 }  
  
  
 }  
  
}

public interface Worker {  
 void show();  
}

**2.USB**



public class UsbDemo  
{  
 public static void main(String[] args) {  
 //插入鼠标  
 *clientUsb*(new MouseUsb("联想鼠标"));//正品  
 //插入麦克风usb  
 *clientUsb*(new MicroPhoneUsb("索爱麦克风"));//山寨货  
 }  
  
 public static void clientUsb(UsbThing usbThing) {  
 //自定义功能  
 //判断是否正品货山寨货  
 //如果是正品就输出，此USB为联想鼠标正品  
 //否则输出该产品为山寨货，不可用！！！  
 int real=0;  
 if (usbThing instanceof MouseUsb) {  
 MouseUsb usbThing1 = (MouseUsb) usbThing;  
 for(int i=0;i<usbThing1.reals.size();i++)  
 {  
 if(usbThing1.reals.get(i).equals(usbThing.name))real=1;  
 }  
 }  
 else if(usbThing instanceof MicroPhoneUsb)  
 {  
 MicroPhoneUsb usbThing1 = (MicroPhoneUsb) usbThing;  
 for(int i=0;i<usbThing1.reals.size();i++)  
 {  
 if(usbThing1.reals.get(i).equals(usbThing.name))real=1;  
 }  
 }  
 else if(usbThing instanceof KeyBoardUsb)  
 {  
 KeyBoardUsb usbThing1 = (KeyBoardUsb) usbThing;  
 for(int i=0;i<usbThing1.reals.size();i++)  
 {  
 if(usbThing1.reals.get(i).equals(usbThing.name))real=1;  
 }  
 }  
 else real=0;  
  
 if(real==1) System.*out*.println("此USB为"+usbThing.name+"正品");  
 else System.*out*.println("该产品为山寨货，不可用！！！");  
  
  
 }  
  
}

public interface UsbInterface {  
 //开启USB  
 public abstract void start();  
 //运行USB  
 public abstract void run();  
 //停止USB  
 public abstract void stop();  
  
}

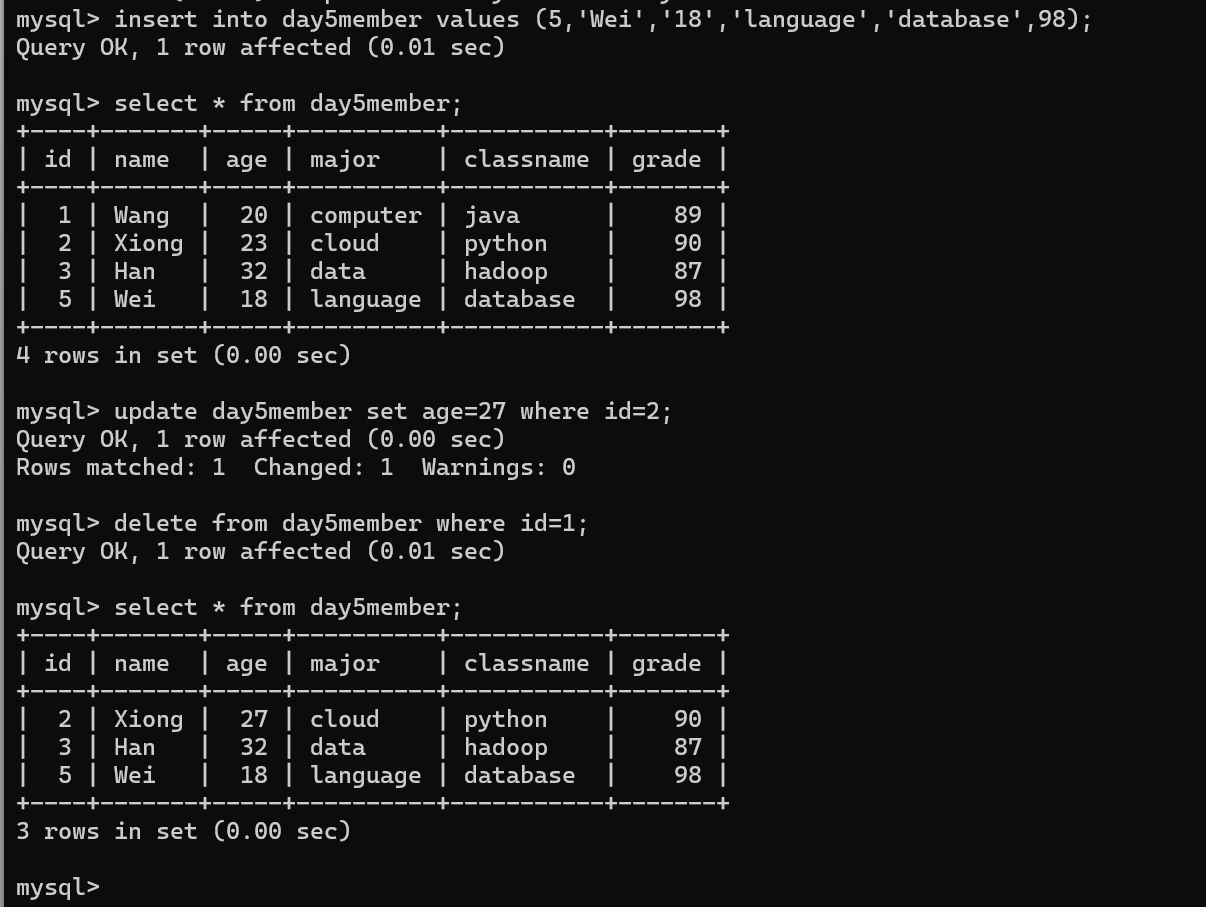
import java.util.ArrayList;  
  
public class UsbThing {  
 String name;  
}

import java.util.ArrayList;  
import java.util.Arrays;  
  
public class KeyBoardUsb extends UsbThing implements UsbInterface  
{  
 ArrayList<String> reals=new ArrayList<String>(Arrays.*asList*("联想键盘", "雷蛇键盘", "罗技键盘"));  
 public KeyBoardUsb(String name)  
 {  
 this.name=name;  
 }  
  
  
 @Override  
 public void start() {  
 System.*out*.println(name+"已启动！");  
 }  
  
 @Override  
 public void run() {  
 System.*out*.println(name+"正常运行！");  
 }  
  
 @Override  
 public void stop() {  
 System.*out*.println(name+"已断开！");  
 }  
}

import java.util.ArrayList;  
import java.util.Arrays;  
  
public class MicroPhoneUsb extends UsbThing implements UsbInterface  
{  
 ArrayList<String> reals=new ArrayList<String>(Arrays.*asList*("联想麦克风", "漫步者麦克风", "苹果麦克风"));  
 public MicroPhoneUsb(String name)  
 {  
 this.name=name;  
 }  
  
 @Override  
 public void start() {  
 System.*out*.println(name+"已启动！");  
 }  
  
 @Override  
 public void run() {  
 System.*out*.println(name+"正常运行！");  
 }  
  
 @Override  
 public void stop() {  
 System.*out*.println(name+"已断开！");  
 }  
}

public class MouseUsb extends UsbThing implements UsbInterface  
{  
  
 ArrayList<String> reals=new ArrayList<String>(Arrays.*asList*("联想鼠标", "雷蛇鼠标", "罗技鼠标"));  
 public MouseUsb(String name)  
 {  
 this.name=name;  
 }  
  
 @Override  
 public void start() {  
 System.*out*.println(name+"已启动！");  
 }  
  
 @Override  
 public void run() {  
 System.*out*.println(name+"正常运行！");  
 }  
  
 @Override  
 public void stop() {  
 System.*out*.println(name+"已断开！");  
 }  
}

3. 在终端中操作mysql进行增删查改



4.java连接mysql数据库的操作尚未完成，11点交完日志过后继续攻坚。

**彭尹垚**

日志：

今天学习了类的继承，多态，抽象类，接口。

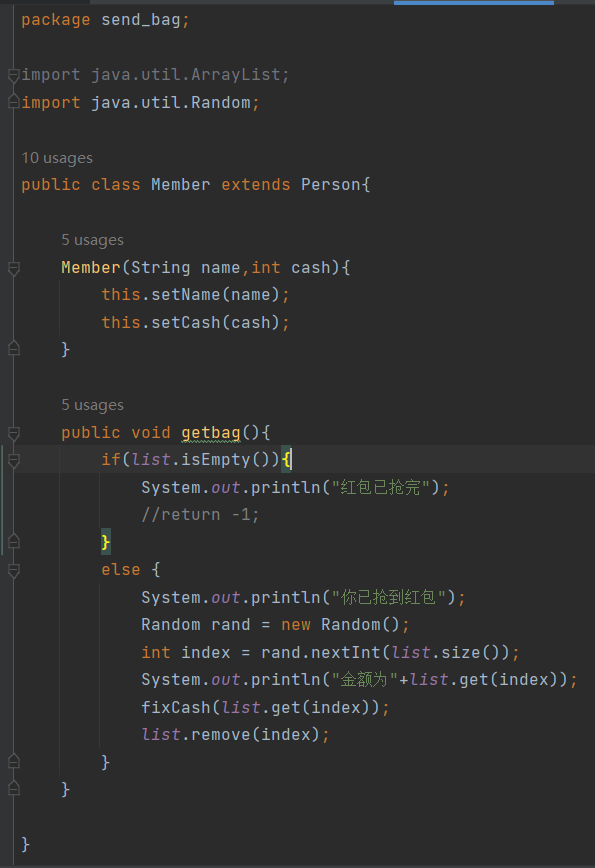
并以此实现了群组发送红包功能以及usb题目

1. 抢红包项目

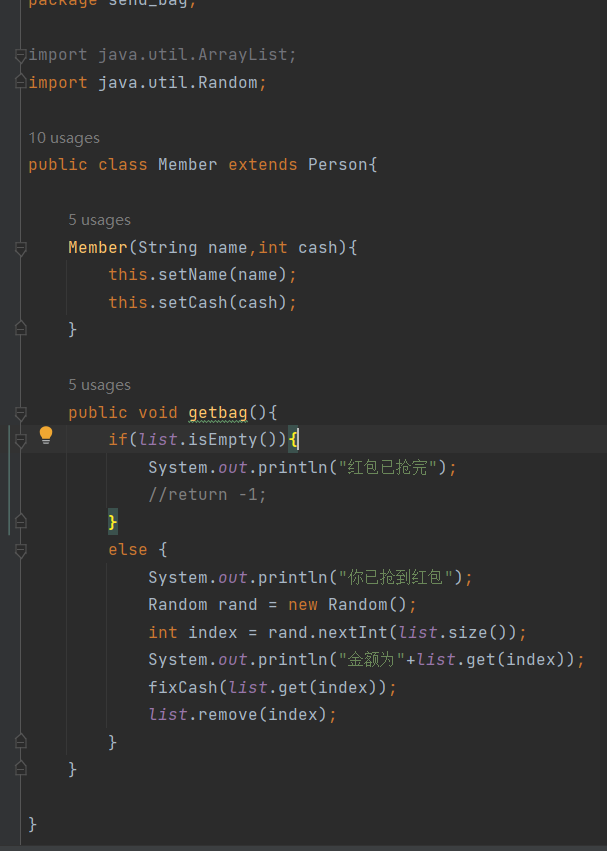
父类people：



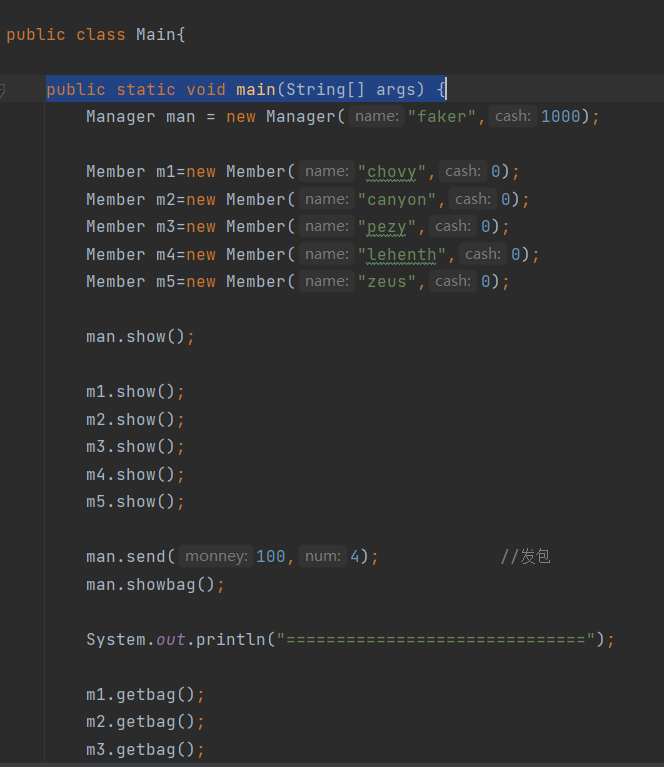
子类Manager：

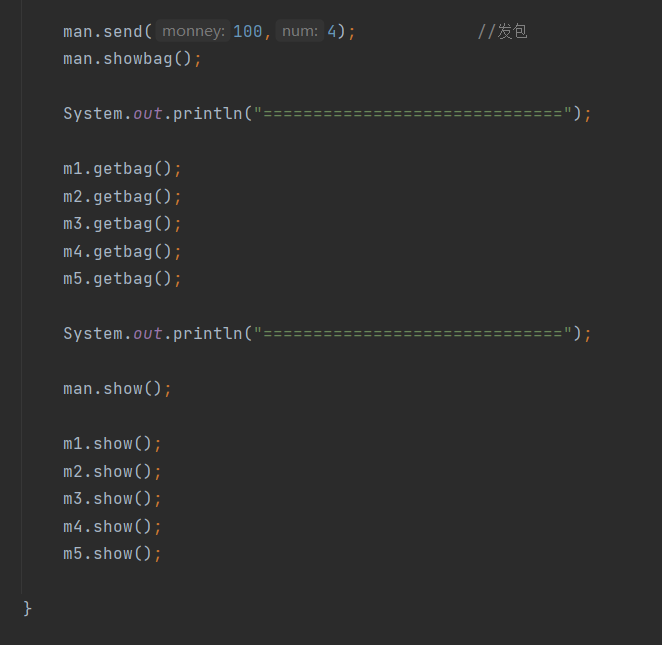


子类Member：

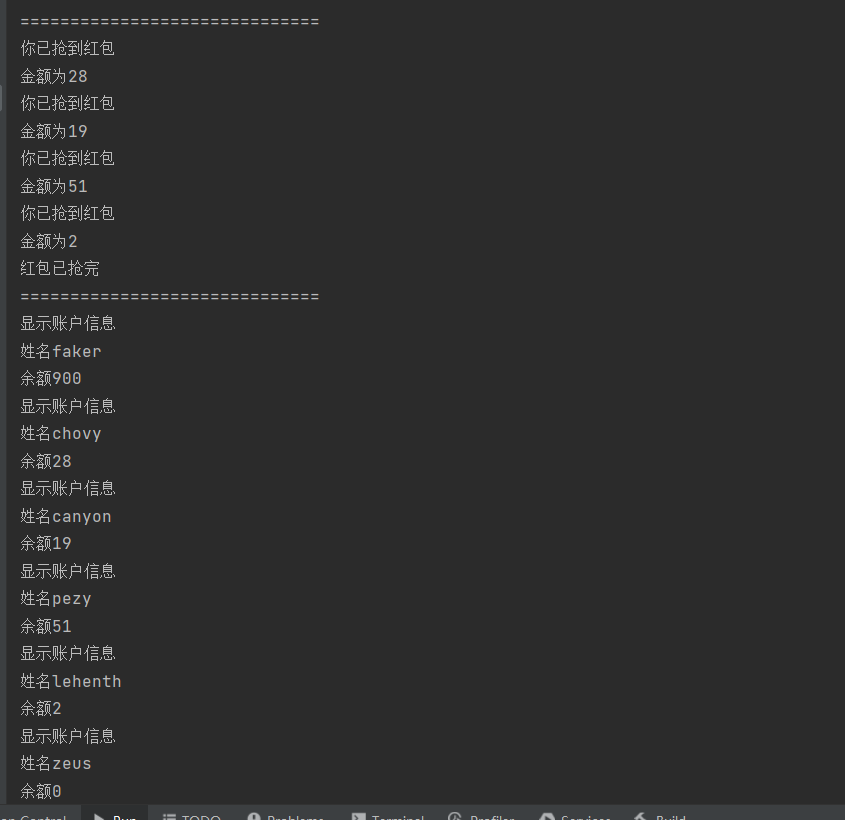
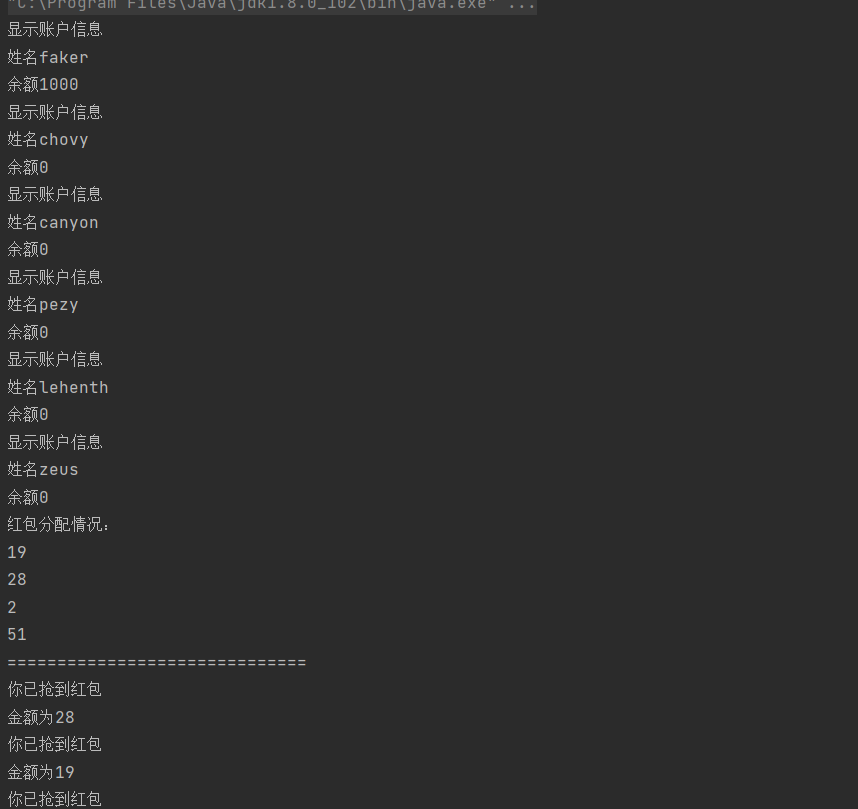


主函数main



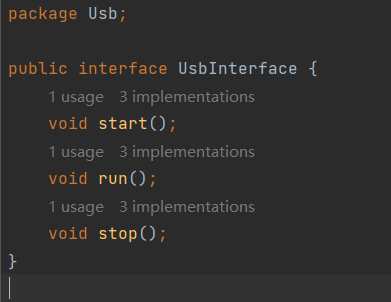


实现效果：

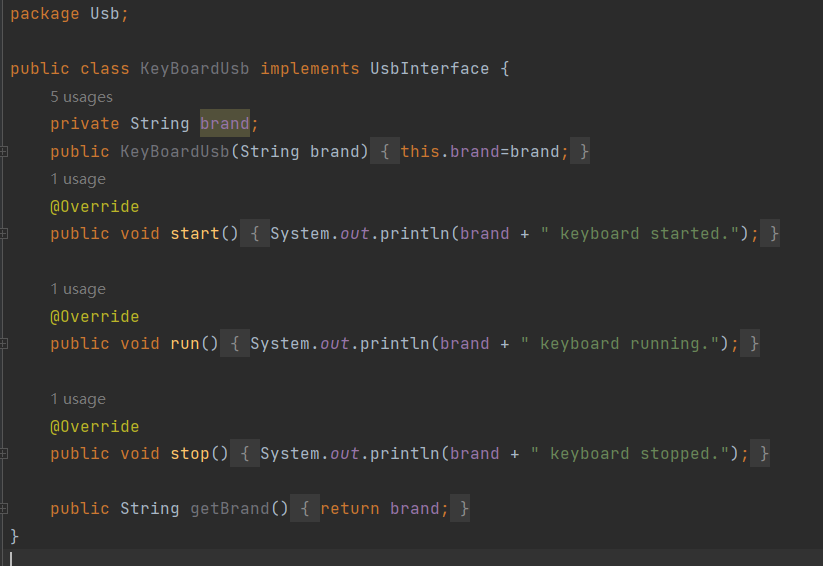


Usb检测：

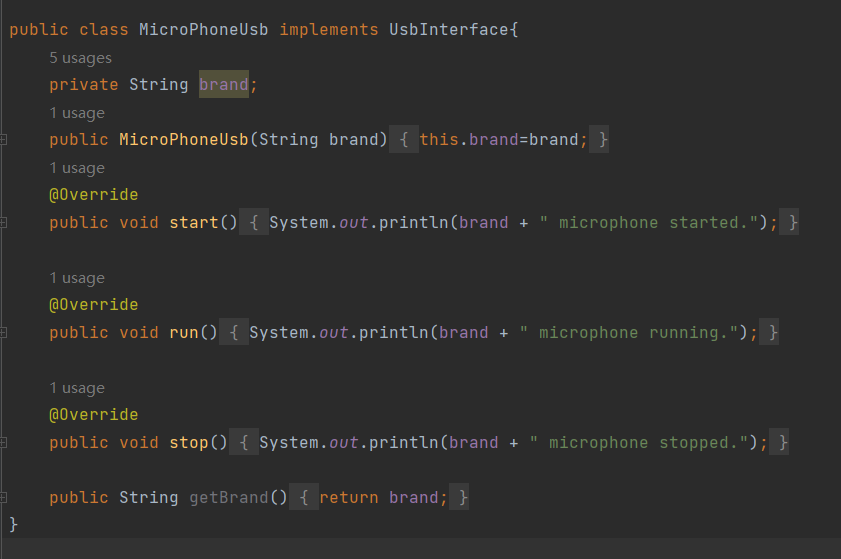
接口：



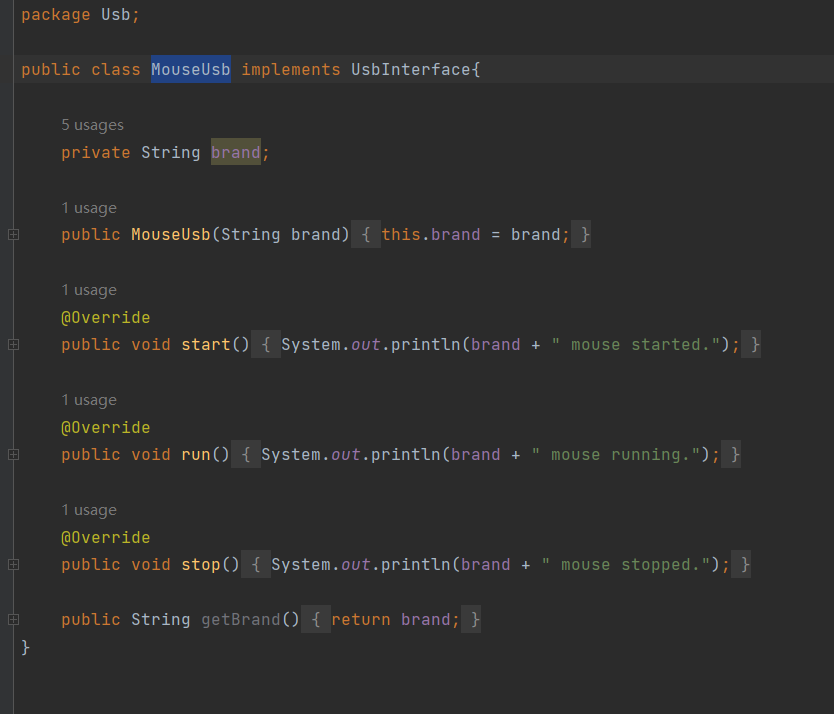
类KeyBoardUsb



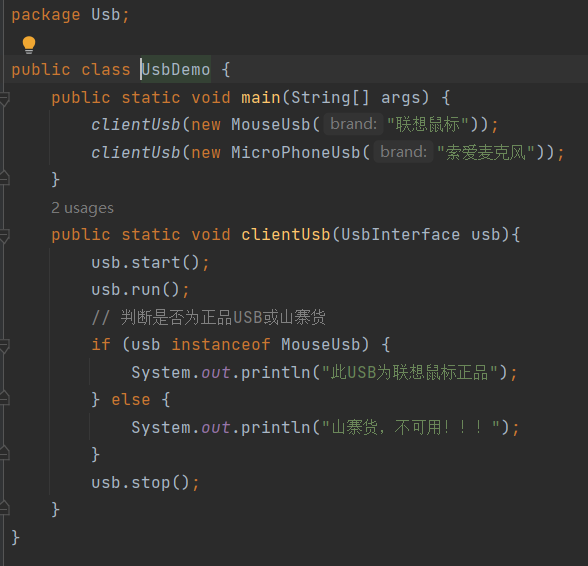
类MicroPhoneUsb：



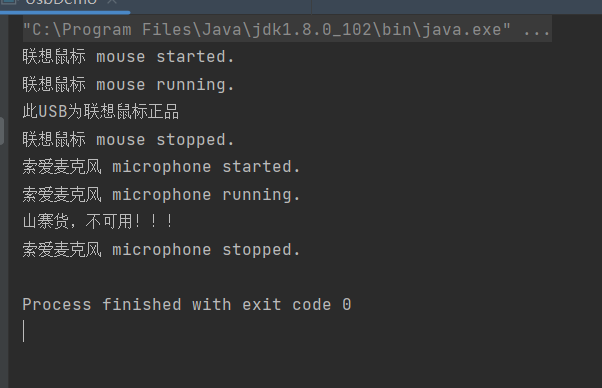
类MouseUsb：



主函数main：

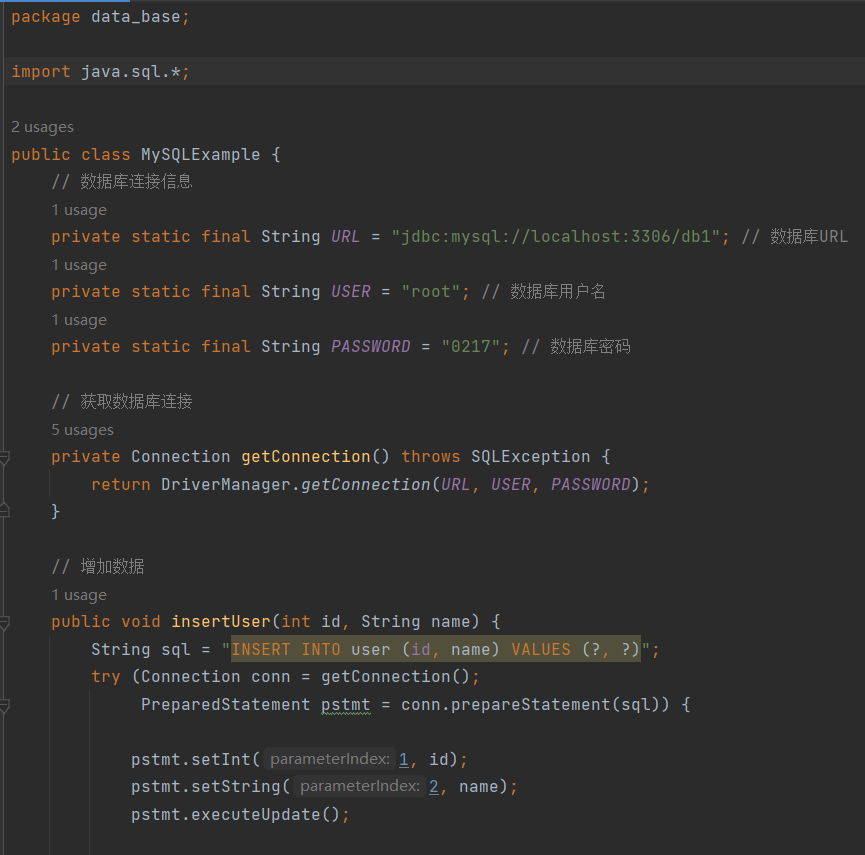


实现效果：

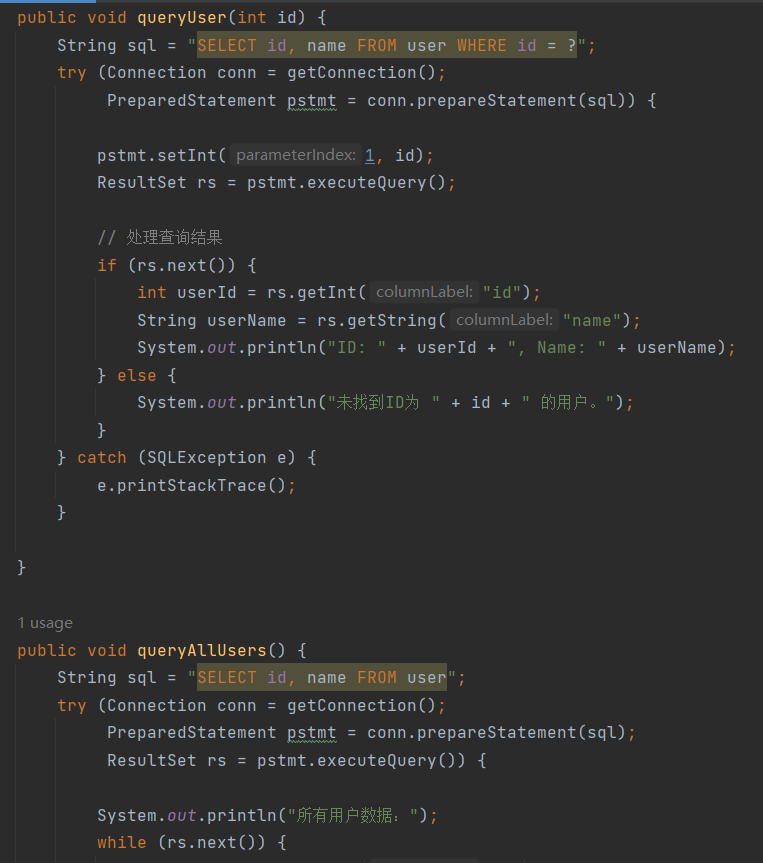


Sql数据库封装

类：









实现效果



**谢泽川**

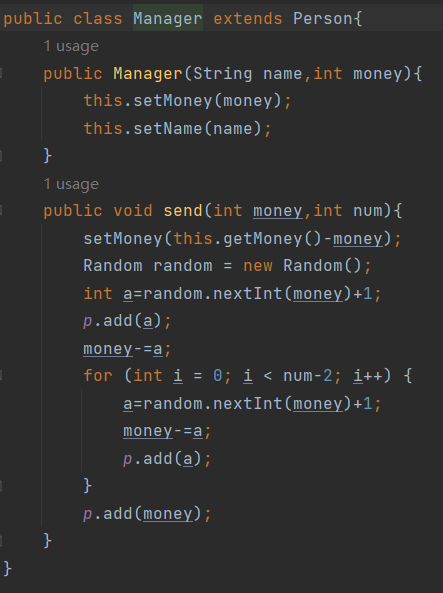
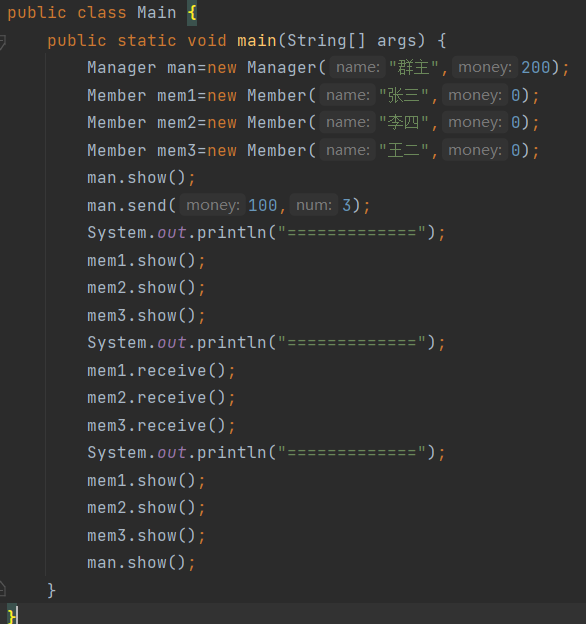
##### Java学习日志 - JAVA接口的调用和抽象方法以及数据库的编写

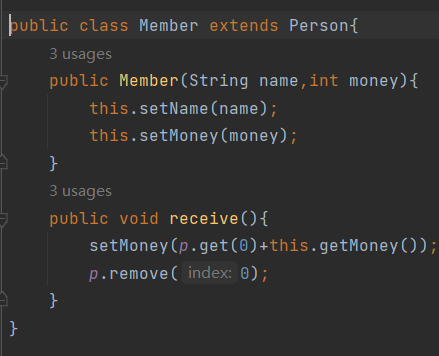
##### 日期：2024年8月19日

##### 学习内容：

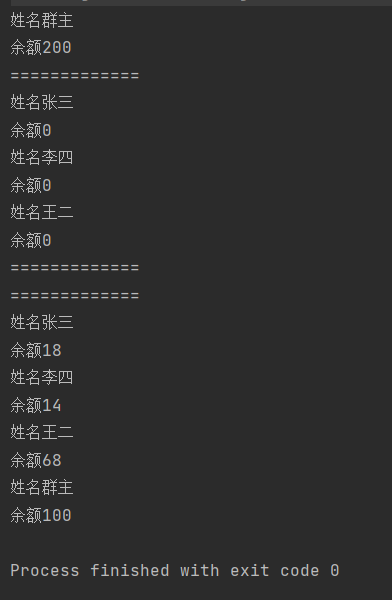
学习了 JAVA接口的调用和抽象方法以及数据库的编写任务一：

任务一：





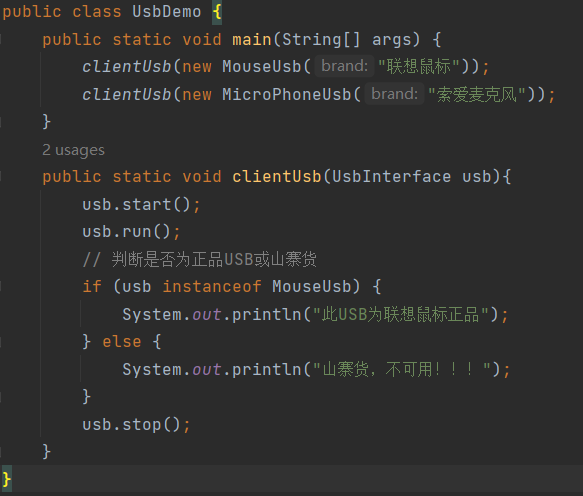
结果：



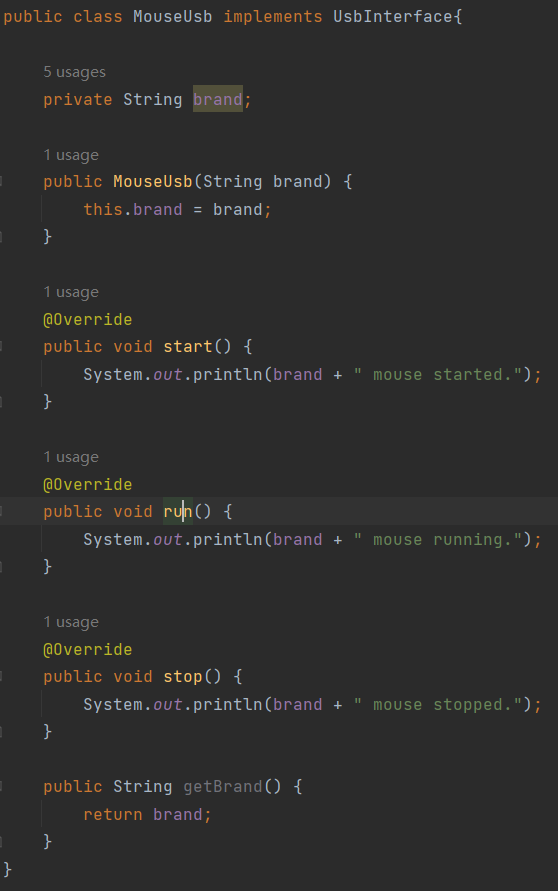
任务二：



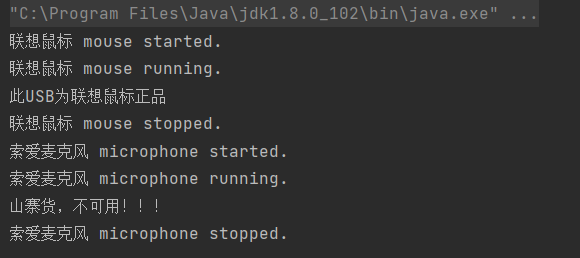
主函数：



MouseUsb的实现：

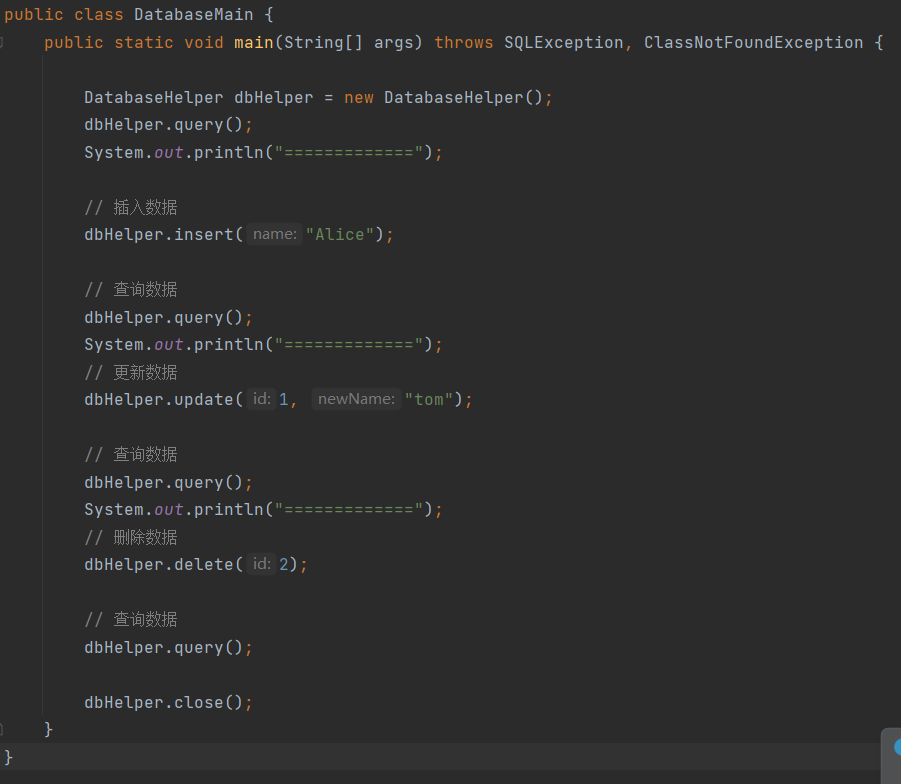


结果

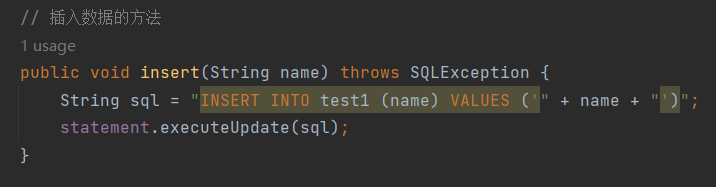


任务三：

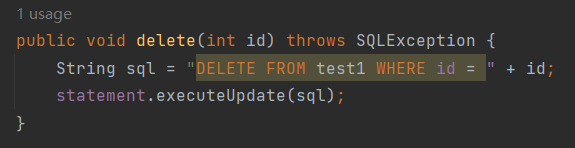
主函数：



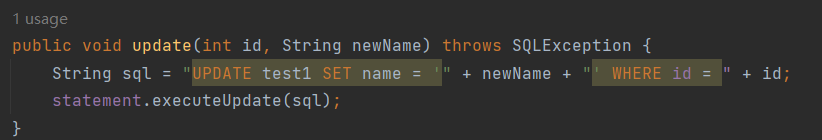
插入：



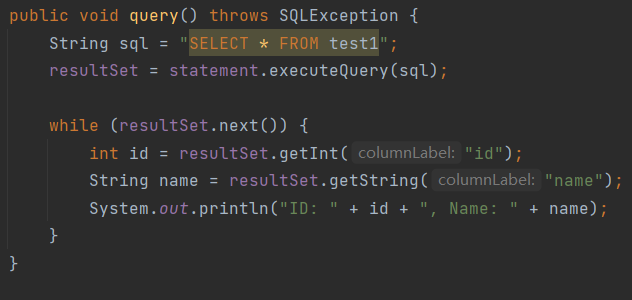
删除：



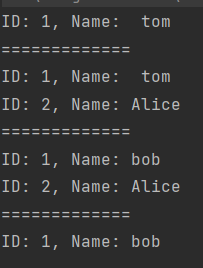
更新：



查询：



结果：



**张世纪**

1. 抢红包

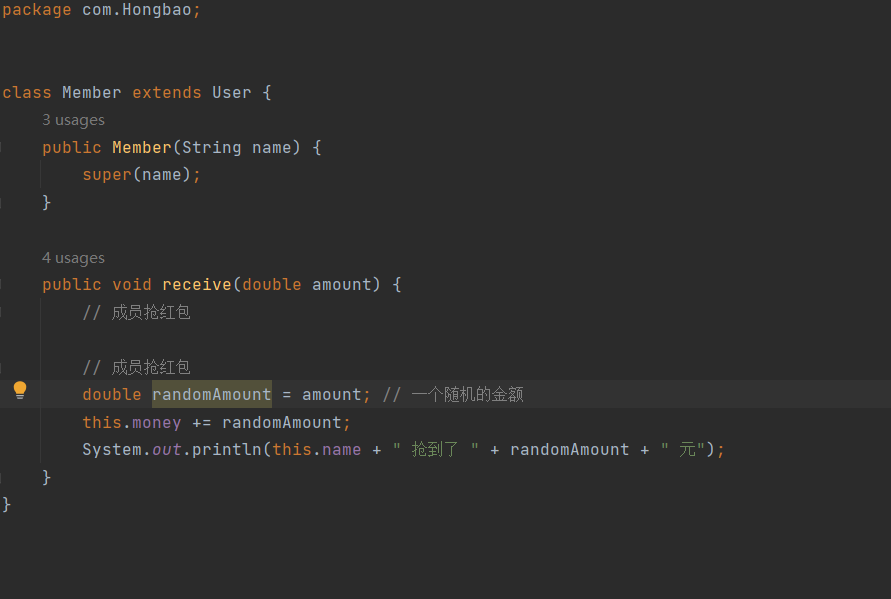
父类



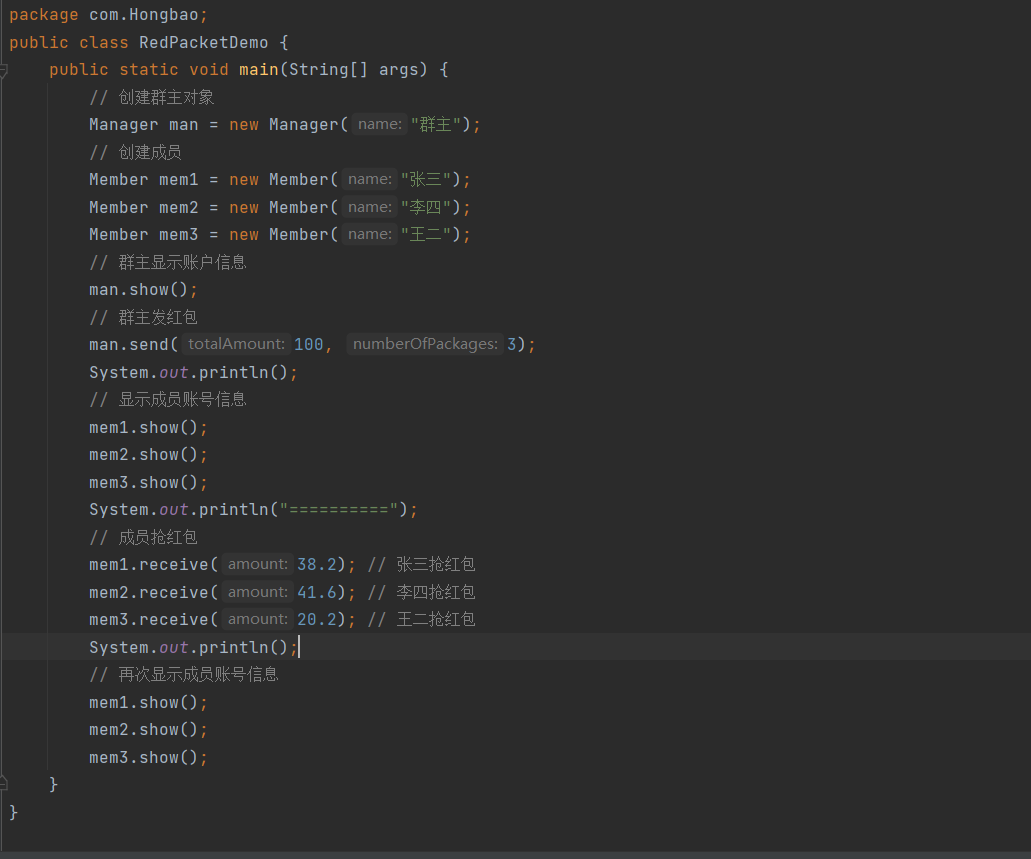
子类：群主



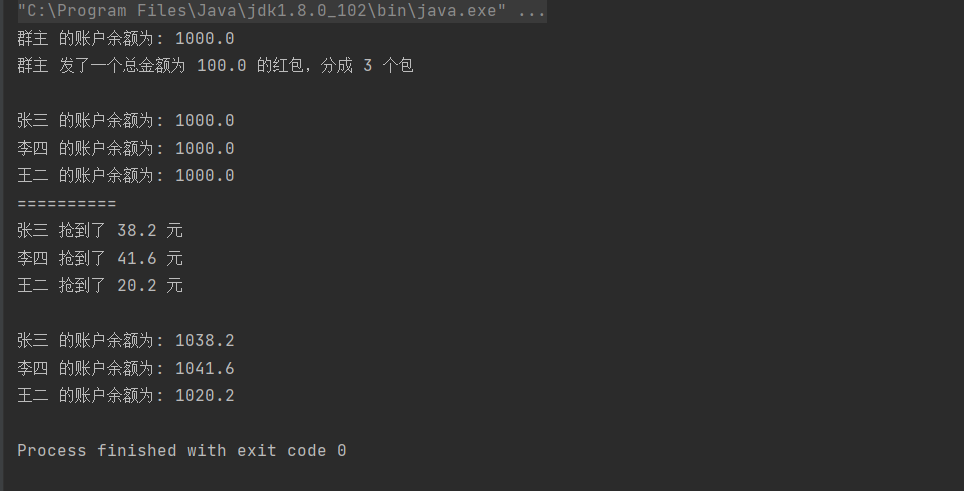
子类：成员



Main：



运行结果：



第二题没理解题意，第三题还没做。

**叶翔昊**

8·19实训日志

总体情况：任务全部完成，问题顺利解决。学习了面向对象构造函数，以及使用ArrayList完成任务。对JAVA代码的编写更加了解和熟悉。

任务一：

发红包

完成情况：

顺利完成，熟悉父类与子类，继承的用法。

代码：

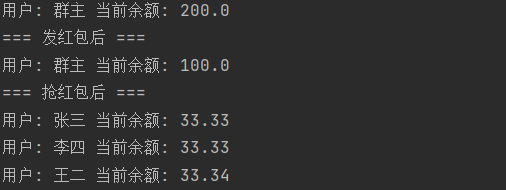
package Hongbao;  
  
import java.util.List;  
  
public class Main {  
 public static void main(String[] args) {  
 // 创建群主对象  
 Manager man = new Manager("群主");  
 man.setBalance(200); // 设置群主余额  
  
 // 创建成员  
 Member mem1 = new Member("张三");  
 Member mem2 = new Member("李四");  
 Member mem3 = new Member("王二");  
  
 // 群主发红包  
 man.show();  
 List<Double> redPackets = man.send(100, 3);  
 System.*out*.println("=== 发红包后 ===");  
 man.show();  
  
 // 成员抢红包  
 mem1.receive(redPackets);  
 mem2.receive(redPackets);  
 mem3.receive(redPackets);  
  
 System.*out*.println("=== 抢红包后 ===");  
 mem1.show();  
 mem2.show();  
 mem3.show();  
 }  
}

package Hongbao;  
  
import java.util.ArrayList;  
import java.util.List;  
  
public class Manager extends User {  
 public Manager(String name) {  
 super(name);  
 }  
  
 public List<Double> send(double totalAmount, int count) {  
 List<Double> redPackets = new ArrayList<>();  
  
 int totalAmountInCents = (int) Math.*round*(totalAmount \* 100);  
 if (totalAmountInCents > getBalance() \* 100) {  
 System.*out*.println("余额不足，无法发红包！");  
 return redPackets;  
 }  
  
 setBalance(getBalance() - totalAmount);  
  
 int avgInCents = totalAmountInCents / count;  
 int remainderInCents = totalAmountInCents % count;  
  
 for (int i = 0; i < count; i++) {  
 if (i == count - 1) {  
 redPackets.add((avgInCents + remainderInCents) / 100.0);  
 } else {  
 redPackets.add(avgInCents / 100.0);  
 }  
 }  
  
 return redPackets;  
 }  
}

package hongbao;  
  
import java.util.List;  
  
public class Member extends User {  
 public Member(String name) {  
 super(name);  
 }  
  
 public void receive(List<Double> redPackets) {  
 if (redPackets.size() > 0) {  
 double amount = redPackets.remove(0);  
 setBalance(getBalance() + amount);  
 }  
 }  
}

package hongbao;  
  
  
  
public class User {  
 private String name;  
 private double balance;  
  
 public User(String name) {  
 this.name = name;  
 this.balance = 0;  
 }  
  
 public void show() {  
 System.*out*.println("用户: " + name + " 当前余额: " + balance);  
 }  
  
 public String getName() {  
 return name;  
 }  
  
 public double getBalance() {  
 return balance;  
 }  
  
 public void setBalance(double balance) {  
 this.balance = balance;  
 }  
}

运行情况：



任务二：

判断usb是否为正版或山寨。

完成情况：

顺利完成，熟悉掌握接口以及重写相关代码写法。

代码：

package USB;  
  
// 定义USB接口  
interface UsbInterface {  
 // 开启USB  
 void start();  
 // 运行USB  
 void run();  
 // 停止USB  
 void stop();  
}  
  
// 鼠标USB产品  
class MouseUsb implements UsbInterface {  
 private String brand;  
  
 public MouseUsb(String brand) {  
 this.brand = brand;  
 }  
  
 @Override  
 public void start() {  
 System.*out*.println(brand + " 鼠标USB开始工作");  
 }  
  
 @Override  
 public void run() {  
 System.*out*.println(brand + " 鼠标USB正在运行");  
 }  
  
 @Override  
 public void stop() {  
 System.*out*.println(brand + " 鼠标USB停止工作");  
 }  
  
 public String getBrand() {  
 return brand;  
 }  
}  
  
// 麦克风USB产品  
class MicroPhoneUsb implements UsbInterface {  
 private String brand;  
  
 public MicroPhoneUsb(String brand) {  
 this.brand = brand;  
 }  
  
 @Override  
 public void start() {  
 System.*out*.println(brand + " 麦克风USB开始工作");  
 }  
  
 @Override  
 public void run() {  
 System.*out*.println(brand + " 麦克风USB正在运行");  
 }  
  
 @Override  
 public void stop() {  
 System.*out*.println(brand + " 麦克风USB停止工作");  
 }  
  
 public String getBrand() {  
 return brand;  
 }  
}  
  
// 键盘USB产品  
class KeyBoardUsb implements UsbInterface {  
 private String brand;  
  
 public KeyBoardUsb(String brand) {  
 this.brand = brand;  
 }  
  
 @Override  
 public void start() {  
 System.*out*.println(brand + " 键盘USB开始工作");  
 }  
  
 @Override  
 public void run() {  
 System.*out*.println(brand + " 键盘USB正在运行");  
 }  
  
 @Override  
 public void stop() {  
 System.*out*.println(brand + " 键盘USB停止工作");  
 }  
  
 public String getBrand() {  
 return brand;  
 }  
}  
  
class UsbDemo {  
 public static void main(String[] args) {  
 // 插入鼠标USB  
 *clientUsb*(new MouseUsb("联想鼠标"));  
 // 插入麦克风USB  
 *clientUsb*(new MicroPhoneUsb("索爱麦克风"));  
 // 插入键盘USB  
 *clientUsb*(new KeyBoardUsb("罗技键盘"));  
 }  
  
 // 定义静态方法判断是否为正品货山寨货  
 public static void clientUsb(UsbInterface usb) {  
 if (usb instanceof MouseUsb) {  
 MouseUsb mouseUsb = (MouseUsb) usb;  
 if ("联想鼠标".equals(mouseUsb.getBrand())) {  
 System.*out*.println("此USB为联想鼠标正品");  
 } else {  
 System.*out*.println("此产品为山寨货，不可用!!!");  
 }  
 } else if (usb instanceof MicroPhoneUsb) {  
 MicroPhoneUsb microPhoneUsb = (MicroPhoneUsb) usb;  
 if ("索爱麦克风".equals(microPhoneUsb.getBrand())) {  
 System.*out*.println("此USB为索爱麦克风正品");  
 } else {  
 System.*out*.println("此产品为山寨货，不可用!!!");  
 }  
 } else if (usb instanceof KeyBoardUsb) {  
 KeyBoardUsb keyBoardUsb = (KeyBoardUsb) usb;  
 if ("罗技键盘".equals(keyBoardUsb.getBrand())) {  
 System.*out*.println("此USB为罗技键盘正品");  
 } else {  
 System.*out*.println("此产品为山寨货，不可用!!!");  
 }  
 }  
 }  
}

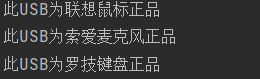
package USB;  
  
// 定义USB接口  
interface UsbInterface {  
 // 开启USB  
 void start();  
 // 运行USB  
 void run();  
 // 停止USB  
 void stop();  
}  
  
// 鼠标USB产品  
class MouseUsb implements UsbInterface {  
 private String brand;  
  
 public MouseUsb(String brand) {  
 this.brand = brand;  
 }  
  
 @Override  
 public void start() {  
 System.*out*.println(brand + " 鼠标USB开始工作");  
 }  
  
 @Override  
 public void run() {  
 System.*out*.println(brand + " 鼠标USB正在运行");  
 }  
  
 @Override  
 public void stop() {  
 System.*out*.println(brand + " 鼠标USB停止工作");  
 }  
  
 public String getBrand() {  
 return brand;  
 }a  
}  
  
// 麦克风USB产品  
class MicroPhoneUsb implements UsbInterface {  
 private String brand;  
  
 public MicroPhoneUsb(String brand) {  
 this.brand = brand;  
 }  
  
 @Override  
 public void start() {  
 System.*out*.println(brand + " 麦克风USB开始工作");  
 }  
  
 @Override  
 public void run() {  
 System.*out*.println(brand + " 麦克风USB正在运行");  
 }  
  
 @Override  
 public void stop() {  
 System.*out*.println(brand + " 麦克风USB停止工作");  
 }  
  
 public String getBrand() {  
 return brand;  
 }  
}  
  
// 键盘USB产品  
class KeyBoardUsb implements UsbInterface {  
 private String brand;  
  
 public KeyBoardUsb(String brand) {  
 this.brand = brand;  
 }  
  
 @Override  
 public void start() {  
 System.*out*.println(brand + " 键盘USB开始工作");  
 }  
  
 @Override  
 public void run() {  
 System.*out*.println(brand + " 键盘USB正在运行");  
 }  
  
 @Override  
 public void stop() {  
 System.*out*.println(brand + " 键盘USB停止工作");  
 }  
  
 public String getBrand() {  
 return brand;  
 }  
}  
  
class UsbDemo {  
 public static void main(String[] args) {  
 // 插入鼠标USB  
 *clientUsb*(new MouseUsb("联想鼠标"));  
 // 插入麦克风USB  
 *clientUsb*(new MicroPhoneUsb("索爱麦克风"));  
 // 插入键盘USB  
 *clientUsb*(new KeyBoardUsb("罗技键盘"));  
 }  
  
 // 定义静态方法判断是否为正品货山寨货  
 public static void clientUsb(UsbInterface usb) {  
 if (usb instanceof MouseUsb) {  
 MouseUsb mouseUsb = (MouseUsb) usb;  
 if ("联想鼠标".equals(mouseUsb.getBrand())) {  
 System.*out*.println("此USB为联想鼠标正品");  
 } else {  
 System.*out*.println("此产品为山寨货，不可用!!!");  
 }  
 } else if (usb instanceof MicroPhoneUsb) {  
 MicroPhoneUsb microPhoneUsb = (MicroPhoneUsb) usb;  
 if ("索爱麦克风".equals(microPhoneUsb.getBrand())) {  
 System.*out*.println("此USB为索爱麦克风正品");  
 } else {  
 System.*out*.println("此产品为山寨货，不可用!!!");  
 }  
 } else if (usb instanceof KeyBoardUsb) {  
 KeyBoardUsb keyBoardUsb = (KeyBoardUsb) usb;  
 if ("罗技键盘".equals(keyBoardUsb.getBrand())) {  
 System.*out*.println("此USB为罗技键盘正品");  
 } else {  
 System.*out*.println("此产品为山寨货，不可用!!!");  
 }  
 }  
 }  
}

package USB;  
  
// 定义USB接口  
interface UsbInterface {  
 // 开启USB  
 void start();  
 // 运行USB  
 void run();  
 // 停止USB  
 void stop();  
}  
  
// 鼠标USB产品  
class MouseUsb implements UsbInterface {  
 private String brand;  
  
 public MouseUsb(String brand) {  
 this.brand = brand;  
 }  
  
 @Override  
 public void start() {  
 System.*out*.println(brand + " 鼠标USB开始工作");  
 }  
  
 @Override  
 public void run() {  
 System.*out*.println(brand + " 鼠标USB正在运行");  
 }  
  
 @Override  
 public void stop() {  
 System.*out*.println(brand + " 鼠标USB停止工作");  
 }  
  
 public String getBrand() {  
 return brand;  
 }  
}  
  
// 麦克风USB产品  
class MicroPhoneUsb implements UsbInterface {  
 private String brand;  
  
 public MicroPhoneUsb(String brand) {  
 this.brand = brand;  
 }  
  
 @Override  
 public void start() {  
 System.*out*.println(brand + " 麦克风USB开始工作");  
 }  
  
 @Override  
 public void run() {  
 System.*out*.println(brand + " 麦克风USB正在运行");  
 }  
  
 @Override  
 public void stop() {  
 System.*out*.println(brand + " 麦克风USB停止工作");  
 }  
  
 public String getBrand() {  
 return brand;  
 }  
}  
  
// 键盘USB产品  
class KeyBoardUsb implements UsbInterface {  
 private String brand;  
  
 public KeyBoardUsb(String brand) {  
 this.brand = brand;  
 }  
  
 @Override  
 public void start() {  
 System.*out*.println(brand + " 键盘USB开始工作");  
 }  
  
 @Override  
 public void run() {  
 System.*out*.println(brand + " 键盘USB正在运行");  
 }  
  
 @Override  
 public void stop() {  
 System.*out*.println(brand + " 键盘USB停止工作");  
 }  
  
 public String getBrand() {  
 return brand;  
 }  
}  
  
class UsbDemo {  
 public static void main(String[] args) {  
 // 插入鼠标USB  
 *clientUsb*(new MouseUsb("联想鼠标"));  
 // 插入麦克风USB  
 *clientUsb*(new MicroPhoneUsb("索爱麦克风"));  
 // 插入键盘USB  
 *clientUsb*(new KeyBoardUsb("罗技键盘"));  
 }  
  
 // 定义静态方法判断是否为正品货山寨货  
 public static void clientUsb(UsbInterface usb) {  
 if (usb instanceof MouseUsb) {  
 MouseUsb mouseUsb = (MouseUsb) usb;  
 if ("联想鼠标".equals(mouseUsb.getBrand())) {  
 System.*out*.println("此USB为联想鼠标正品");  
 } else {  
 System.*out*.println("此产品为山寨货，不可用!!!");  
 }  
 } else if (usb instanceof MicroPhoneUsb) {  
 MicroPhoneUsb microPhoneUsb = (MicroPhoneUsb) usb;  
 if ("索爱麦克风".equals(microPhoneUsb.getBrand())) {  
 System.*out*.println("此USB为索爱麦克风正品");  
 } else {  
 System.*out*.println("此产品为山寨货，不可用!!!");  
 }  
 } else if (usb instanceof KeyBoardUsb) {  
 KeyBoardUsb keyBoardUsb = (KeyBoardUsb) usb;  
 if ("罗技键盘".equals(keyBoardUsb.getBrand())) {  
 System.*out*.println("此USB为罗技键盘正品");  
 } else {  
 System.*out*.println("此产品为山寨货，不可用!!!");  
 }  
 }  
 }  
}

package USB;  
  
// 定义USB接口  
interface UsbInterface {  
 // 开启USB  
 void start();  
 // 运行USB  
 void run();  
 // 停止USB  
 void stop();  
}  
  
// 鼠标USB产品  
class MouseUsb implements UsbInterface {  
 private String brand;  
  
 public MouseUsb(String brand) {  
 this.brand = brand;  
 }  
  
 @Override  
 public void start() {  
 System.*out*.println(brand + " 鼠标USB开始工作");  
 }  
  
 @Override  
 public void run() {  
 System.*out*.println(brand + " 鼠标USB正在运行");  
 }  
  
 @Override  
 public void stop() {  
 System.*out*.println(brand + " 鼠标USB停止工作");  
 }  
  
 public String getBrand() {  
 return brand;  
 }  
}  
  
// 麦克风USB产品  
class MicroPhoneUsb implements UsbInterface {  
 private String brand;  
  
 public MicroPhoneUsb(String brand) {  
 this.brand = brand;  
 }  
  
 @Override  
 public void start() {  
 System.*out*.println(brand + " 麦克风USB开始工作");  
 }  
  
 @Override  
 public void run() {  
 System.*out*.println(brand + " 麦克风USB正在运行");  
 }  
  
 @Override  
 public void stop() {  
 System.*out*.println(brand + " 麦克风USB停止工作");  
 }  
  
 public String getBrand() {  
 return brand;  
 }  
}  
  
// 键盘USB产品  
class KeyBoardUsb implements UsbInterface {  
 private String brand;  
  
 public KeyBoardUsb(String brand) {  
 this.brand = brand;  
 }  
  
 @Override  
 public void start() {  
 System.*out*.println(brand + " 键盘USB开始工作");  
 }  
  
 @Override  
 public void run() {  
 System.*out*.println(brand + " 键盘USB正在运行");  
 }  
  
 @Override  
 public void stop() {  
 System.*out*.println(brand + " 键盘USB停止工作");  
 }  
  
 public String getBrand() {  
 return brand;  
 }  
}  
  
class UsbDemo {  
 public static void main(String[] args) {  
 // 插入鼠标USB  
 *clientUsb*(new MouseUsb("联想鼠标"));  
 // 插入麦克风USB  
 *clientUsb*(new MicroPhoneUsb("索爱麦克风"));  
 // 插入键盘USB  
 *clientUsb*(new KeyBoardUsb("罗技键盘"));  
 }  
  
 // 定义静态方法判断是否为正品货山寨货  
 public static void clientUsb(UsbInterface usb) {  
 if (usb instanceof MouseUsb) {  
 MouseUsb mouseUsb = (MouseUsb) usb;  
 if ("联想鼠标".equals(mouseUsb.getBrand())) {  
 System.*out*.println("此USB为联想鼠标正品");  
 } else {  
 System.*out*.println("此产品为山寨货，不可用!!!");  
 }  
 } else if (usb instanceof MicroPhoneUsb) {  
 MicroPhoneUsb microPhoneUsb = (MicroPhoneUsb) usb;  
 if ("索爱麦克风".equals(microPhoneUsb.getBrand())) {  
 System.*out*.println("此USB为索爱麦克风正品");  
 } else {  
 System.*out*.println("此产品为山寨货，不可用!!!");  
 }  
 } else if (usb instanceof KeyBoardUsb) {  
 KeyBoardUsb keyBoardUsb = (KeyBoardUsb) usb;  
 if ("罗技键盘".equals(keyBoardUsb.getBrand())) {  
 System.*out*.println("此USB为罗技键盘正品");  
 } else {  
 System.*out*.println("此产品为山寨货，不可用!!!");  
 }  
 }  
 }  
}

package USB;  
  
// 定义USB接口  
interface UsbInterface {  
 // 开启USB  
 void start();  
 // 运行USB  
 void run();  
 // 停止USB  
 void stop();  
}  
  
// 鼠标USB产品  
class MouseUsb implements UsbInterface {  
 private String brand;  
  
 public MouseUsb(String brand) {  
 this.brand = brand;  
 }  
  
 @Override  
 public void start() {  
 System.*out*.println(brand + " 鼠标USB开始工作");  
 }  
  
 @Override  
 public void run() {  
 System.*out*.println(brand + " 鼠标USB正在运行");  
 }  
  
 @Override  
 public void stop() {  
 System.*out*.println(brand + " 鼠标USB停止工作");  
 }  
  
 public String getBrand() {  
 return brand;  
 }  
}  
  
// 麦克风USB产品  
class MicroPhoneUsb implements UsbInterface {  
 private String brand;  
  
 public MicroPhoneUsb(String brand) {  
 this.brand = brand;  
 }  
  
 @Override  
 public void start() {  
 System.*out*.println(brand + " 麦克风USB开始工作");  
 }  
  
 @Override  
 public void run() {  
 System.*out*.println(brand + " 麦克风USB正在运行");  
 }  
  
 @Override  
 public void stop() {  
 System.*out*.println(brand + " 麦克风USB停止工作");  
 }  
  
 public String getBrand() {  
 return brand;  
 }  
}  
  
// 键盘USB产品  
class KeyBoardUsb implements UsbInterface {  
 private String brand;  
  
 public KeyBoardUsb(String brand) {  
 this.brand = brand;  
 }  
  
 @Override  
 public void start() {  
 System.*out*.println(brand + " 键盘USB开始工作");  
 }  
  
 @Override  
 public void run() {  
 System.*out*.println(brand + " 键盘USB正在运行");  
 }  
  
 @Override  
 public void stop() {  
 System.*out*.println(brand + " 键盘USB停止工作");  
 }  
  
 public String getBrand() {  
 return brand;  
 }  
}  
  
class UsbDemo {  
 public static void main(String[] args) {  
 // 插入鼠标USB  
 *clientUsb*(new MouseUsb("联想鼠标"));  
 // 插入麦克风USB  
 *clientUsb*(new MicroPhoneUsb("索爱麦克风"));  
 // 插入键盘USB  
 *clientUsb*(new KeyBoardUsb("罗技键盘"));  
 }  
  
 // 定义静态方法判断是否为正品货山寨货  
 public static void clientUsb(UsbInterface usb) {  
 if (usb instanceof MouseUsb) {  
 MouseUsb mouseUsb = (MouseUsb) usb;  
 if ("联想鼠标".equals(mouseUsb.getBrand())) {  
 System.*out*.println("此USB为联想鼠标正品");  
 } else {  
 System.*out*.println("此产品为山寨货，不可用!!!");  
 }  
 } else if (usb instanceof MicroPhoneUsb) {  
 MicroPhoneUsb microPhoneUsb = (MicroPhoneUsb) usb;  
 if ("索爱麦克风".equals(microPhoneUsb.getBrand())) {  
 System.*out*.println("此USB为索爱麦克风正品");  
 } else {  
 System.*out*.println("此产品为山寨货，不可用!!!");  
 }  
 } else if (usb instanceof KeyBoardUsb) {  
 KeyBoardUsb keyBoardUsb = (KeyBoardUsb) usb;  
 if ("罗技键盘".equals(keyBoardUsb.getBrand())) {  
 System.*out*.println("此USB为罗技键盘正品");  
 } else {  
 System.*out*.println("此产品为山寨货，不可用!!!");  
 }  
 }  
 }  
}

运行情况：



任务三：

任务:

1. 对数据库的增加删除修改查询方法的封装
2. 用数据库实现用户的增删查操作

3、使用终端操作数据库

完成情况：没有完成，问题较多尚未解决，成功安装MySQL和建立数据库，但是具体的Java与数据库的连接还未建立，jar文件导入也存在问题。

**兰天阳**

2024.8.19

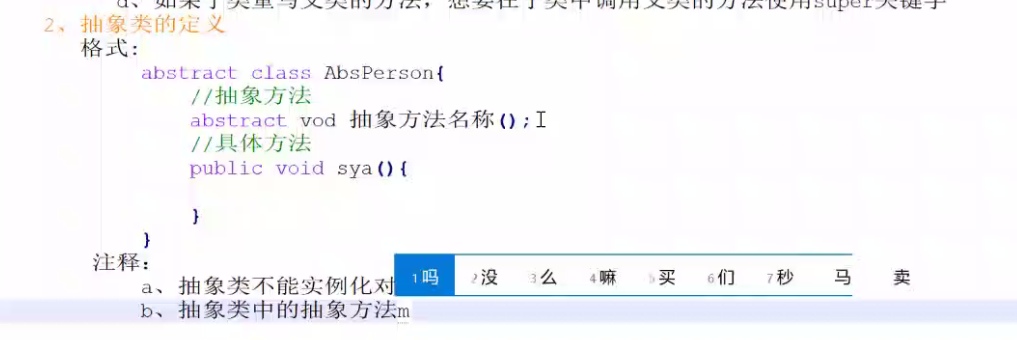
继承：

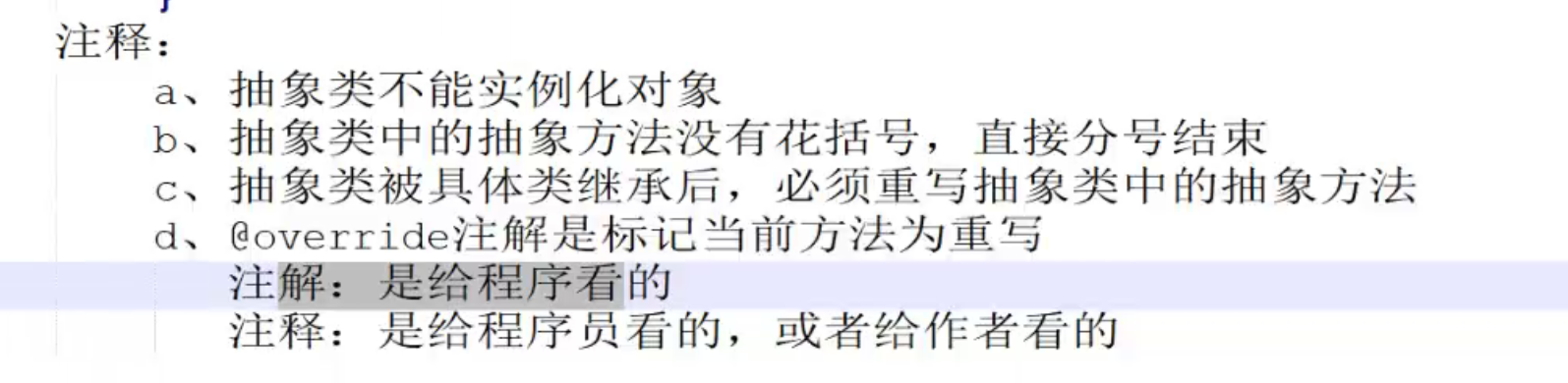
父、子类方法名字相同——重写

名字相同，参数不同——重载

子类重写父类方法，但要使用父类方法，用super

抽象类





抢红包：

package cd;

import java.util.ArrayList;

import java.util.Iterator;

import java.util.Random;

public class Worker {

RedPack r;

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String name;

public int getBalance() {

return balance;

}

public void setBalance(int balance) {

this.balance = balance;

}

public int balance;

public void show(){

System.out.println(name + "的余额为：" + getBalance());

}

public void send(int price, int num){

if (num <= 0 || price <= 0) {

throw new IllegalArgumentException("数量和金额必须为正数");

}

Random rand = new Random();

int[] numbers = new int[num];

int remaining = price;

setBalance(getBalance() - remaining);

for (int i = 0; i < num - 1; i++) {

numbers[i] = rand.nextInt(remaining - (num - 1 - i)) + 1;

remaining -= numbers[i];

}

numbers[num - 1] = remaining;

RedPack r1 = new RedPack(numbers);

r = r1;

}

public void receive(Worker sender){

ArrayList<Integer> a = new ArrayList<>();

Random rand = new Random();

int size = sender.r.each\_price.size();

int choice = rand.nextInt(size);

Iterator<Integer> it = sender.r.each\_price.iterator();

while(it.hasNext()){

int p = it.next();

if(sender.r.each\_price.get(choice) == p){

setBalance(getBalance() + p);

sender.r.each\_price.remove(choice);

System.out.println("成功抢到" + p + "元");

System.out.println(getBalance());

break;

}

}

}

}

package cd;

import java.util.ArrayList;

import java.util.Iterator;

public class RedPack {

ArrayList<Integer> each\_price = new ArrayList<>();

public RedPack(int[] price){

for(int value : price){

each\_price.add(value);

}

Iterator<Integer> it = each\_price.iterator();

while(it.hasNext()){

int n = it.next();

}

}

}

package cd;

public class Manager extends Worker {

public Manager(String name,int balance){

this.name = name;

this.balance = balance;

}

@Override

public void show(){

System.out.println("亲爱的" + name + ",你的余额为：" + getBalance());

}

}

public class Day5\_1 {

public static void main(String[] args) {

//创建群主对象

Manager man = new Manager("群主", 10000);

//创建成员

cd.Member mem1 = new cd.Member("张三");

cd.Member mem2 = new cd.Member("李四");

cd.Member mem3 = new cd.Member("王二");

man.show();

man.send(100,6);

mem1.show();

mem2.show();

mem3.show();

System.out.println("===========");

mem1.receive(man);

mem2.receive(man);

mem3.receive(man);

System.out.println("===========");

mem1.show();

mem2.show();

mem3.show();

}

}

