# 实验4 面向对象思考

## 实验目的和要求

掌握类之间的关系

掌握包装类为基本类型为基本类型值创建对象

掌握基本类型与包装类型间的自动转化

掌握String类处理不可改变的字符串的方法

掌握StrigBuilder和StringBuffer类处理可以改变的字符串的方法

## 实验题目

### P338 10.28

原题：使用StringBuilder类中的reverse方法编写三条语句，倒置字符串s。

源代码：

**public** **class** T1 {

**public** **static** **void** main(String[] args) {

//P338 10.28

StringBuilder s = **new** StringBuilder();

s.append("java");

s.reverse();

System.***out***.println(s);

}

}

结果及截图：



### P338 10.29

原题：编写三条语句，从包含20个字符的字符串s中删除下标从4到10的字串。使用StringBuilder类中的delete方法。

源代码：

**public** **class** T2 {

**public** **static** **void** main(String[] args) {

// P338 10.29

StringBuilder s = **new** StringBuilder();

s.append("be interestingthings");

s.delete(4,10);

System.***out***.println(s);

}

}

结果及截图：



### P339 10.1

原题：（时间类Time）设计一个名为Time类。这个类包括：

表示时间的数据域hour、minute、和second。

一个以当前时间创建Time对象的无参的构造方法（数据域的值表示当前时间）。

一个构造Time对象的构造方法，这个对象有一个特定的时间值，这个值是以毫秒表示的、从1970年1月1日午夜开始到现在流逝的时间段（数据域的值表示这个时间）。

一个构造带特定的小时、分钟和秒的Time对象的构造方法。

三个数据域hour、minute和second各自的get方法。

一个名为setTime（long elapseTime）的方法使用流逝的时间为对象设置一个新时间。例如，如果流逝的时间为555550000毫秒，则转换为10小时，10分钟，10秒。

画出该类的UML图并实现这个类。编写一个测试程序，创建两个Time对象（使用new Time（）和new Time（555550000）），然后显示它们的小时、分钟、秒。

源代码：

**public** **class** T3 {

**public** **static** **void** main(String[] args) {

// P339 10.1

Time t1 = **new** Time();

t1.showTime();

Time t2 = **new** Time(555550000);

t2.setTime(555550000);

t2.showTime();

}

}

**class** Time

{

**private** **int** hour;

**private** **int** minute;

**private** **int** second;

**private** **long** t;

**public** Time()

{

**long** t = System.*currentTimeMillis*();

t /= 1000;

second = (**int**)t%60;

minute = (**int**)t/60%60;

hour = (**int**)t/60/60%24+8;

}

**public** **void** showTime()

{

System.***out***.println(hour+":"+minute+":"+second);

}

**public** Time(**long** t)

{

**this**.t = t/1000;

}

**public** Time(**int** hour,**int** minute,**int** second)

{

**this**.hour = hour;

**this**.minute = minute;

**this**.second = second;

}

**public** **int** getHour()

{

**return** hour;

}

**public** **int** getMinute()

{

**return** minute;

}

**public** **int** getSecond()

{

**return** second;

}

**public** **void** setTime(**long** elapseTime)

{

**long** t = elapseTime;

t /= 1000;

second = (**int**)t%60;

minute = (**int**)t/60%60;

hour = (**int**)t/60/60%24;

}

}

结果及截图：



### P340 10.4

原题：（MyPoint类）设计一个名为MyPoint的类，表示一个带x坐标和y坐标的点。该类包括：

两个带get方法的数据域x和y分别表示它们的坐标。

一个创建点（0，0）的无参构造方法。

一个创建特定点的构造方法。

一个名为distance的方法，返回从该点到MyPoint类型的指定点之间的距离。

一个名为distance的方法，返回从该点到指定x和y坐标的指点之间的距离。

画出该类的UML图并实现这个类。编写一个测序程序，创建两个点（0，0）和（10，30.5），并显示它们之间的距离。

源代码：

**public** **class** T4 {

**public** **static** **void** main(String[] args) {

//P340 10.4

MyPoint mp1 = **new** MyPoint();

MyPoint mp2 = **new** MyPoint(10,30.5);

System.***out***.println(mp1.distance2(mp2.getX(), mp2.getY()));

}

}

**class** MyPoint

{

**private** **double** x;

**private** **double** y;

**public** MyPoint()

{

**this**.x = 0;

**this**.y = 0;

}

**public** MyPoint(**double** x,**double** y)

{

**this**.x = x;

**this**.y = y;

}

**public** **double** getX()

{

**return** x;

}

**public** **double** getY()

{

**return** y;

}

**public** **double** distance1(**double** x,**double** y)

{

**double** d = 0;

d = Math.*sqrt*((**this**.x-x)\*(**this**.x-x)+(**this**.y-y)\*(**this**.y-y));

**return** d;

}

**public** **double** distance2(**double** x,**double** y)

{

**double** d = 0;

d = Math.*sqrt*((**this**.x-x)\*(**this**.x-x)+(**this**.y-y)\*(**this**.y-y));

**return** d;

}

}

结果及截图：



### P340 10.7

原题：（游戏：ATM机）使用编程练习9.7中创建的Account类来模拟一台ATM机。创建一个有10个账户的数组，其id为0，1，...，9，并初始化为收支为100美元。

系统提示用户输入一个id。如果用户输入的id不正确，就要求用户输入正确的id。一旦接收一个id，就显示如运行实例所示的主菜单。可以选择1来查看当前的

收支，选择2表示收钱，选择3表示存钱，选择4表示退出主菜单。一旦退出，系统就会提示再次输入id。所以，系统一旦启动就不会停止。

源代码：

import java.util.Date;

import java.util.Scanner;

public class T5 {

public static void menuPrint()

{

System.out.println("Main menu");

System.out.println("1: check balance");

System.out.println("2: withdraw");

System.out.println("3: deposit");

System.out.println("4: exit");

}

public static void main(String[] args) {

//P340 10.7

Scanner in = new Scanner(System.in);

ATM atm = new ATM();

while(true)

{

System.out.println("Enter an id:");

int id = in.nextInt();

if(id<0||id>9) {continue;}

else

while(true)

{

menuPrint();

System.out.println("Enter a choice:");

int n = in.nextInt();

switch(n)

{

case 1:System.out.println("The balance is:"+atm.getA(id).getBalance());continue;

case 2:

{

System.out.println("Enter an amount to withdraw:");

double m = in.nextDouble();

atm.getA(id).withDraw(m);

continue;

}

case 3:

{

System.out.println("Enter an amount to deposit:");

double m = in.nextDouble();

atm.getA(id).deposit(m);

continue;

}

case 4: break;

}

}

}

}

}

class ATM

{

private Account[] a = new Account[10];

public ATM()

{

for(int i=0; i<a.length; i++)

{

a[i] = new Account();

a[i].setId(i);

a[i].setBalance(100);

}

}

public Account getA(int i)

{

return a[i];

}

}

class Account

{

private int id;

private double balance;

private double annualInterestRate;

private Date dateCreated = null;

public Account(){}

public Account(int id,double balance)

{

this.id = id;

this.balance = balance;

}

public int getId()

{

return id;

}

public double getBalance()

{

return balance;

}

public double getAnnualInterestRate()

{

return annualInterestRate;

}

public void setId(int id)

{

this.id = id;

}

public void setBalance(double balance)

{

this.balance = balance;

}

public void setAnnualInterestRate(double annualInterestRate)

{

this.annualInterestRate = annualInterestRate;

}

public Date getdateCreated()

{

return dateCreated;

}

public double getMonthlyInteresrRate()

{

double rate = annualInterestRate/12;

return rate;

}

public void withDraw(double money)

{

balance -= money;

}

public void deposit(double money)

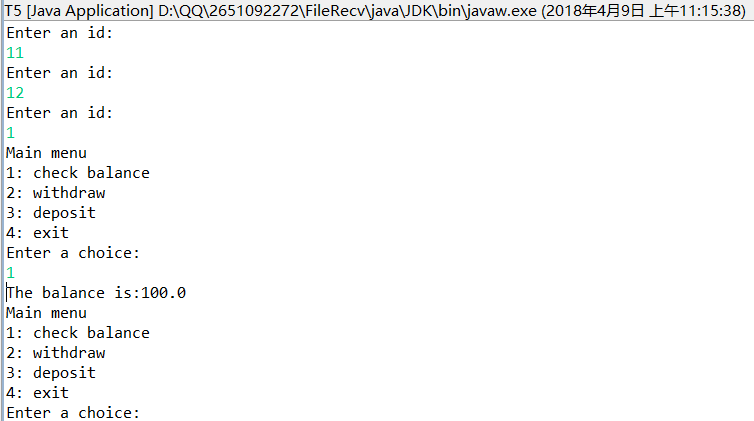
{

balance += money;

}

}

结果及截图：



### P342 10.9

原题：（课程类Course）如下改写Course类：

程序清单10-6中数组的大小是固定的。对它进行改进，通过创建一个新的更大的数组并复制当前的数组的内容来实现数组大小的自动增长。

实现dropStdutent方法。

添加一个名为clear（）的新方法，然后删掉选某门课程的学生。

编写一个测试程序，创建一门课程，添加三个学生，然后显示这门课程的学生。

源代码：

**import** java.util.ArrayList;

**import** java.util.Arrays;

**import** java.util.Collections;

**import** java.util.List;

**public** **class** Course {

**private** String courseName;

**private** String[] students = **new** String[100];

**private** **int** numberOfStudents;

**public** Course(String courseName)

{

**this**.courseName = courseName;

}

**public** **void** addStudent(String student)

{

students[numberOfStudents] = student;

numberOfStudents++;

}

**public** String[] getStudents()

{

**return** students;

}

**public** **int** getNumberOfStudents()

{

**return** numberOfStudents;

}

**public** String getCourseName()

{

**return** courseName;

}

**public** **void** dropStudent(String student)

{

List<String> student2 = **new** ArrayList<String>();

Collections.*addAll*(student2, students);

**for**(**int** i=0,len=student2.size();i<len;i++)

{

**if**(student2.get(i)==student)

{

student2.remove(i);

--len;

}

}

students = (String[])student2.toArray(**new** String[student2.size()]);

}

**public** **void** clear()

{

students = **null**;

}

**public** **static** **void** main(String[] args) {

// P342 10.9

Course c = **new** Course("English");

c.addStudent("张三");

c.addStudent("李四");

c.addStudent("王二");

c.dropStudent("王二");

System.***out***.println("课程"+c.getCourseName()+"剩余学生：");

**for**(**int** i=0;i<c.getNumberOfStudents()-1;i++)

{

System.***out***.println(c.getStudents()[i]);

}

}

}

结果及截图：

