# 实验二 类与对象编程练习

## 【实验目的】

1. 理解C#语言是如何体现面向对象编程基本思想；
2. 掌握类对象的定义；
3. 了解类的封装方法，以及如何创建类和对象；
4. 了解成员变量和成员方法的特性；
5. 掌握静态成员的用法；
6. 掌握构造函数和析构函数的含义与作用、定义方式和实现，能够根据要求正确定义和重载构造函数。能够根据给定的要求定义类并实现类的成员函数；
7. 理解类的成员的访问控制的含义，公有、私有和保护成员的区别；
8. 掌握参数传递的用法；
9. 掌握属性的作用和使用。

## 【实验要求】

1. 编写一个包含类和类方法的程序；
2. 编写一个创建对象和使用对象的方法程序；
3. 编写不同成员变量、方法修饰方法的程序；。
4. 编写含有构造方法的类的程序；
5. 编写重载构造方法的程序；
6. 编写类含有静态变量的程序；
7. 编写通过ref、out修饰符传递参数的类程序；
8. 编写含有属性的类程序。

## 【实验内容】

### 一、定义一个圆类，计算圆的面积和周长

public class circle

{

public static void Main()

{

double radium, delimeter, square;

const double pai = 3.1415926;

radium = Convert.ToInt32(Console.ReadLine());

delimeter = 2 \* pai \* radium;

square = pai \* pai \* radium;

Console.WriteLine("delimeter={0},square={1}", delimeter, square);

Console.ReadLine();

}

}

或者：

public class circle

{

double delimeter, square;

const double pai = 3.1415926;

public void calculate(double rad)

{

delimeter = 2 \* pai \* rad;

square = pai \* pai \* rad;

Console.WriteLine("delimeter={0},square={1}",delimeter,square);

}

public static void Main()

{

double radium;

circle cir = new circle();

radium = Convert.ToInt32(Console.ReadLine());

cir.calculate(radium);

Console.ReadLine();

}

}

请比较以上两个程序，看起来后一个程序把问题复杂化了，是不是不如第一个程序好，它从设计思想上有什么优势么？

答：这个问题其实是面向对象编程和面向过程编程的比较。就本例中程序的性能而言，第二个程序确实把问题复杂了；不过它的设计思想上有一定的优势，那就是将数据和对数据的可用方法封装在一起，易于维护，安全性更高。

### 二、实现一个学生类

程序要求如下：

其中有3个数据成员有学号、姓名、年龄，以及若干成员函数。同时编写主函数使用这个类，实现对学生数据的赋值和输出。要求：使用成员函数实现对数据的输出；使用构造函数实现对数据的输入。

参考代码如下：

public class students

{

string id,name;

int age;

public students(string id,string name,int age )

{

this.id = id;

this.name = name;

this.age = age;

}

public void Display()

{

Console.WriteLine("id={0},name={1},age={2}",id,name,age);

}

public static void Main()

{

//string id, name;

//int age;

students stu = new students("0001","zhangsan",16);

stu.Display();

Console.ReadLine();

}

}

以上程序使用了构造方法，请回答关键字this有何作用，你能将成员函数Display修改成别的代码也实现响应的功能么？

答：关键词this其实是相当于指向类自身的一个指针，可以区别类成员变量和形式参数；

我可以定义各个数据成员的get函数，用get函数对数据成员进行访问。如图：

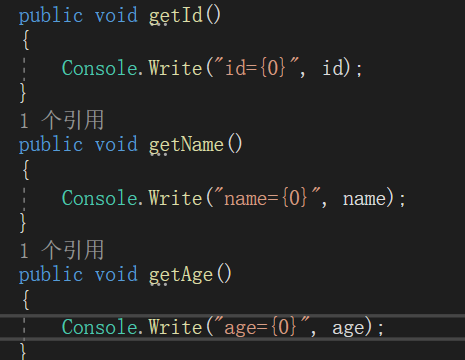


图1

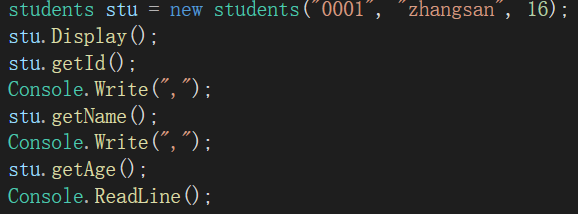


图2

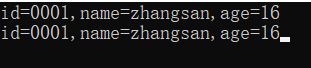


图3

### 三、实现一个日期类型

程序要求如下：

定义日期类型Date。要求有以下面成员：年、月、日变量，重载的构造方法，一个实现年、月、日单独的输入，一个实现从系统时间里读出年月日，并实现打印方法成员，该方法要按照“XXXX年XX月XX日”格式输出日期。

参考代码如下：

public class Date

{

private int Year, Month, Day;

public Date(int Year, int Month,int Day)

{

this.Year=Year;

this.Month=Month;

this.Day=Day;

}

public Date(System.DateTime dt)

{

Year = dt.Year;

Month = dt.Month;

Day = dt.Day;

}

public void DisplayDate()

{

Console.WriteLine("{0}年{1}月{2}日",Year,Month,Day);

}

}

public class Tester

{

public static void Main()

{

System.DateTime currentTime=System.DateTime.Now;

Date dt=new Date(2008,7,18);

dt.DisplayDate();

Date dt2 = new Date(currentTime);

dt2.DisplayDate();

Console.ReadLine();

}

}

请练习System.Datetime类型及其各属性的应用。列出System.Datetime主要包含哪些属性，并思考重载构造方法的作用。

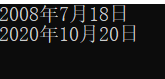


图4

答：包含属性有：Date,Day,DayOfWeek,DayOfYear,Hour,Kind,Milisecond,Minute,Month,Now,Second,Ticks,TimeOfDay,Today,UtcNow,Year.

重载有参数的构造方法可以更方便的在类实例化的同时进行成员变量的赋值。

### 四、实现一个包含类属性方法的简单加法程序

程序要求如下：

建立一个类，使其可以进行简单的加法运算。该程序要包含类、属性、方法等面向对象的基本元素。

public class PropertyTest

{

double xVal, yVal;

double zVal,sVal;

public PropertyTest(double x)

{

xVal = x;

}

public double YVAL

{

get

{

return yVal;

}

set

{

yVal = value;

}

}

public void Display()

{

//Console.WriteLine("xVal={0},yVal={1}",xVal,yVal);

zVal = Math.Sqrt(Math.Pow(xVal,2)-Math.Pow(yVal,3));

sVal=Math.Log(yVal);

// zVal = xVal \* xVal - yVal;

// sVal = yVal + 5;

Console.WriteLine("xVal={0},yVal={1},zVal={2},sVal={3}",xVal,yVal,zVal,sVal);

}

}

public class Tester

{

public static void Main()

{

PropertyTest pt=new PropertyTest(4);

pt.YVAL=2;

pt.Display();

Console.ReadLine();

}

}

作以下修改分析程序输出的原因：

将"xVal={0},yVal={1},zVal={2},sVal={3}"

改成"xVal={0},yVal={1},zVal={0},sVal={1}"查看区别，写出原因；

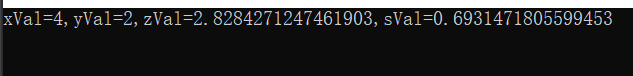


图5 修改前



图6 修改后

原因：修改后，zVal输出的是xVal的值，sVal输出的是yVal的值。

了解Math类下主要的静态数学方法，掌握求幂，求平方和对数等静态方法的使用规则。

Math类是个静态类，常用的静态数学方法有：Abs->返回绝对值；Exp->返回e的指定次幂;pow->返回指定数字的指定次幂;sqrt->返回指定数字的平方根;Log->返回指定数字的对数;Max和Min返回最大最小值;

### 五、通过类程序说明静态变量/方法与实例变量/方法的区别

static 声明的成员变量/方法被视为类的成员变量/方法，而不把它当作实例对象的成员变量/方法。换句话说，静态变量/方法是类固有的，可以直接引用，其它成员变量/方法仅仅被声明，生成实例对象后才存在，才可以被引用。基于这样的事实，也把静态变量/方法称为类变量/方法，非静态变量称为实例变量/方法。

**1.程序功能要求**

编写帐户类，对每一账号赋值,并设置初始化存款为0.00元，设计一变量统计账号生成的数目。

public class BankAccount

{

static int totalAccountNumber=0;

string BankAccountId;

double initialDepositAmount = 0.00;

public BankAccount(string myId)

{

this.BankAccountId = myId;

this.initialDepositAmount = 0.00;

totalAccountNumber++;

}

public void displayid()

{

Console.WriteLine("mbaid={0},initialDepositAmount={1}",this.BankAccountId,this.initialDepositAmount);

}

public static void display()

{

Console.WriteLine("totalAccountNumber={0}", totalAccountNumber);

}

}

public class Tester

{

public static void Main()

{

BankAccount mba = new BankAccount("37000001");

BankAccount mba2 = new BankAccount("3700002");

BankAccount mba3 = new BankAccount("");

BankAccount mba4 = new BankAccount("3700004");

// Console.WriteLine("mba2ID={0}", mba2.BankAccountId);

mba2.displayid();

BankAccount.display();

Console.ReadLine();

}

}

请回答问题：

1. 按你自己的算法修改以上程序，比如可只输出生成的账户数。

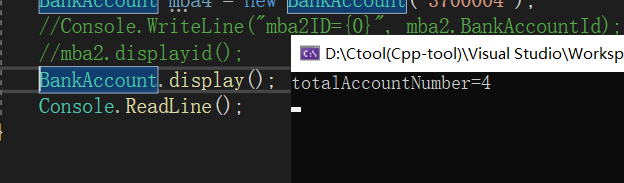


图7

1. 把注释去掉后会怎样，为什么？

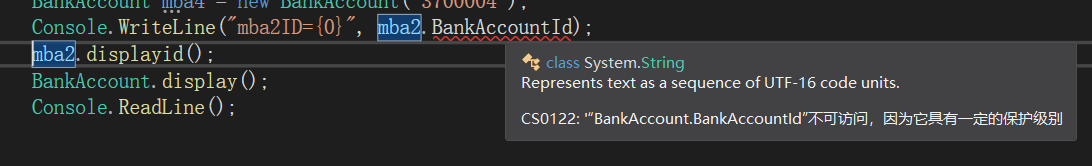


图8

会报错，因为BankAccountId是private，不能直接在主函数里访问。

1. 为什么display用类名直接引用，可以用对象来引用么？尝试输出结果。



图9

（4）类的静态变量和非静态变量的引用区别。判断一下语句的正确性：

静态方法只能使用静态变量，不能使用实例变量。因为对象实例化之前，实例变量不可用。这个观点真确么？（对）

类的静态变量只有一个版本，所有实例对象引用的都是同一个版本。（对）

对象实例化后，每个实例变量都被制作了一个副本，它们之间互不影响。（对）

### 六、方法中参数传递的练习

在其它语言中，函数调用或过程调用时参数有传值调用和传地址调用之分。在C# 中，方法中的参数传递可以分为传值调用或对象方法调用等方式。传值调用即传递的参数是基本数据类型，调用方法时在方法中将不能改变参数的值，这意味着只能使用它们。对象调用是指先调用对象，再调用对象的方法，这种方式可以修改允许存取的成员变量。所以，如果不想改变参数的值，可以采用传值调用的方法。如果想改变参数的值，可采用对象调用的方法，间接修改参数的值。

**1.编写一个传值调用的程序**

程序功能要求：程序首先给整型变量x 和y 赋初值3，5，然后使用传值调用方式调用方法对x 和y 做乘方并及输出x 和y 的乘方值，最后输出x和y得值。再将此方法给为对象调用加ref修饰查看输出结果差异。参考代码如下：

public class Power

{

// public void MyPower(ref int x, ref int y)

public void MyPower(int x, int y)

{

x = 1; y = 2;

Console.WriteLine("x={0},y={1}", x, y);

Console.WriteLine("x\*x={0},y\*y={1}",x\*x,y\*y);

}

}

public class Tester

{

public static void Main()

{

int x, y;

x = 3; y = 5;

Power mp = new Power();

// mp.MyPower(ref x,ref y);

mp.MyPower(x,y);

Console.WriteLine("x={0},y={1}",x,y);

Console.ReadLine();

}

思考：（1）将响应的注释修改再调试查看结果，分析原因。

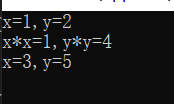


图10 修改前

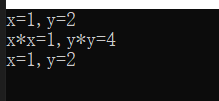


图11 修改后

原因：对象调用是指先调用对象，再调用对象的方法，这种方式可以修改允许存取的成员变量，所以第二张图里，函数中赋的值覆盖了原本的x和y的值。

1. 将Main中x和y赋初值去掉，结果会怎样？如果Main中加ref，类Power的方法中参数前不加ref又会有何变化？说明了什么？

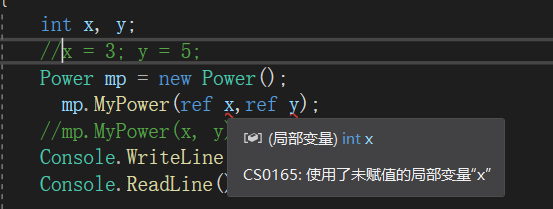


图12 去掉赋值会报错

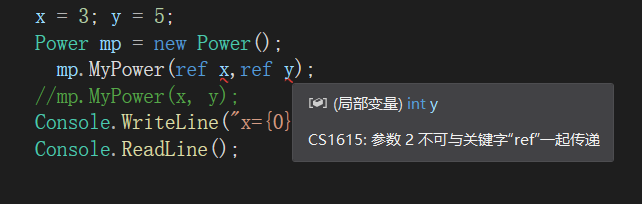


图13 会报错

说明传值类型不同的时候，虽然执行逻辑一致，但实际上这是两个函数，所以Main中无法调用。

1. 如果不想对x作无用的初始化，直接作参数传递，怎么实现？

可以使用out进行第三种传递——输出传递，ref和out使用效果上面是等效的，它们的区别在于：参数标记为ref，那么必须在调用函数之前初始化参数的值；参数标记为out，调用函数之前不需要初始化对象，但调用的函数必要在函数返回之前为对象赋值。

【思考题】

1. 方法的参数传递有哪些方式？区别时什么？

有值传递，引用传递，输出传递；

值传递不会改变变量本身的值；引用传递可以改变变量的值；输出传递效果和引用传递一样，不过不需要在调用参数前对其进行初始化。

1. 什么是构造方法。

构造方法又叫构造函数，它是一种特殊的方法，因为构造方法本身没有返回值，并且通常是public访问类型，方法的名称必须与类名相同。构造方法在类实例创建之前执行，用来初始化对象。所有的类都有构造方法，如果没有定义的构造方法，系统默认生成空的构造方法；如果自己定义了默认的构造方法就会失效了。