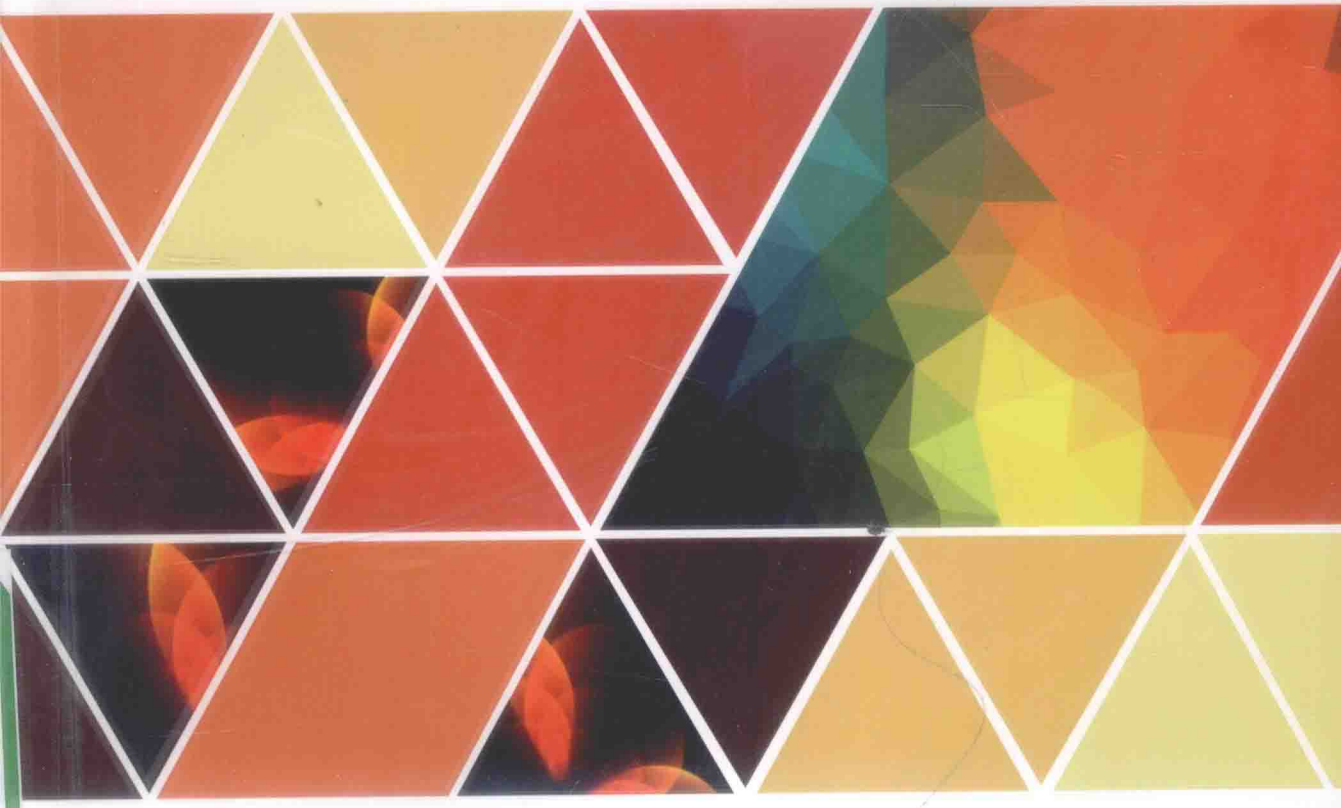
COMPUTER
计算机应用系列
COMPUTER

程序设计基础

主编　杨爱武　董海峰　谢钟扬



西北工业大学出版社

CH

程序设计基础

主　编　杨爱武　董海峰　谢钟扬
主　审　符开耀　王　雷

西北工业大学出版社

【内容简介】 本书C#部分以Microsoft Visual Studio 2010作为操作平台，包含教程、练习、实验部分。教程部分首先熟悉开发环境，然后比较系统地介绍C#语言基础、常用数据类型的用法、流程控制、面向对象编程基础、面向对象的高级编程，在此基础上系统介绍Windows应用程序、文件操作、C#多线程技术。Java部分以Java语言为基础，将软件技术相关专业技能抽查的“程序设计”模块为主线，用实例重点阐述了Java中的变量、常量、数据类型和运算符；分支结构、循环结构；数组与字符串；面向对象的思想，以及Java中对面向对象思想的实现，包括类和对象、封装、继承、多态；Java处理输入输出。教程部分一般节有小综合、章有大综合。实验部分先跟着教程实例做，然后再思考与练习。

本书可作为高等职业教育软件类专业教材，也可作为C#和Java初学者的自学用书，并可供从事信息系统开发的设计、开发人员参考。

图书在版编目(CIP)数据

程序设计基础/杨爱武，董海峰，谢钟扬主编. —西安：西北工业大学出版社，2016.8
ISBN 978-7-5612-4997-0

Ⅰ. ①程… Ⅱ. ①杨… ②董… ③谢… Ⅲ. ①程序设计 Ⅳ. ①TP311.1

中国版本图书馆CIP数据核字(2016)第187703号

出版发行：西北工业大学出版社
通信地址：西安市友谊西路127号 邮编：710072
电　　话：(029)88493844　88491757
网　　址：www.nwpup.com
印 刷 者：兴平市博闻印务有限公司
开　　本：787 mm×1 092 mm　1/16
印　　张：11.75
字　　数：281千字
版　　次：2016年8月第1版　2016年8月第1次印刷
定　　价：30.00元

前　言

C#是.NET平台一种全新设计的现代编程语言，对于开发基于Windows平台的应用系统简单方便，深受大家欢迎。同时，作为ASRNET脚本语言也已经非常流行。

本书以Microsoft Visual Studio 2010作为平台，包含教程、练习、实验部分。教程部分首先熟悉开发环境，然后比较系统地介绍C#语言基础、面向对象编程基础和面向对象编程进阶；在此基础上系统介绍Windows应用程序、GDI+编程、文件操作、C#多线程技术。教程部分根据需要，节有小综合，章有大综合，这样便于逐步积累、形成系统。实验部分先跟着教程实例做，然后再思考与练习，实验内容具有弹性。

本书不仅适合C#课程教学，也非常适合需要掌握C#语言的用户学习和开发应用系统人员参考。只要阅读本书，结合上机实验进行操作练习，就能在较短的时间内基本掌握C#语言及其应用开发技术。

本书C#部分由杨爱武编写并负责统稿，Java部分由董海峰编写。

在本书的编写过程中，得到了湖南软件职业学院谭长富院长、符开耀副院长、王雷教授等领导和专家们的大力支持与热心帮助，在此表示衷心感谢。

本书的出版还得到湖南软件职业学院教学质量工程项目（TD1501，ZY1402，KC1401，KC1302）、湖南省教育厅科学研究项目（14C0617，14C0618）、湖南省职业院校教育教学改革研究项目（ZJB2013045）、湖南省职业教育与成人教育学会科研规划课题（XHB2015063）、湖南省职业教育名师空间课堂建设项目（课程：《数据库原理与应用》湘教科研通〔2015〕38号文）、2014年度湖南省普通高校青年骨干教师培养对象（湘教办通〔2014〕186号文）等项目的资助，在此一并表示感谢。

由于本书的编写目的定位于C#和Java的基础知识与案例分析相结合，试图让读者在深入了解C#和Java编程的相关概念与关键技术的基础上，能尝试开展C#和Java编程的一些初步编程工作，本书的内容编写与结构组织具有一定的难度，加之笔者水平有限，虽然几经修改，但书中仍然难免存在一些疏漏与不足之处，敬请读者、专家、以及同行朋友们的批评指正，在此先行表示感谢。

由于笔者水平有限，不当之处，恳请读者批评指正。

编　者

2016年04月

目　录

第一章　C#语言基础

本章介绍C#语言的基础知识，希望具有C语言基础的读者能够基本掌握C#语言，并以此为基础，能够进一步学习用C#语言编写Windows应用程序和Web应用程序。当然仅靠一章的内容就完全掌握C#语言是不可能的，如需进一步学习C#语言，还需要认真阅读有关C#语言的专著。

1.1　C#语言特点

Microsoft.NET(以下简称.NET)框架是微软提出的新一代Web软件开发模型，C#语言是.NET框架中新一代的开发工具。C#语言是一种现代、面向对象的语言，它简化了C++语言在类、命名空间、方法重载和异常处理等方面的操作，它摒弃了C++的复杂性，更易使用，更少出错。它使用组件编程，和VB一样容易使用。C#语法和C++和Java语法非常相似，如果读者用过C++和Java，学习C#语言应是比较轻松的。

用C#语言编写的源程序，必须用C#语言编译器将C#源程序编译为中间语言(MicroSoft Intermediate Language，MSIL)代码，形成扩展名为exe或dll文件。中间语言代码不是CPU可执行的机器码，在程序运行时，必须由通用语言运行环境(Common Language Runtime，CLR)中的即时编译器(JUST IN Time，JIT)将中间语言代码翻译为CPU可执行的机器码，由CPU执行。CLR为C#语言中间语言代码运行提供了一种运行时环境，C#语言的CLR和Java语言的虚拟机类似。这种执行方法使运行速度变慢，但带来其他一些好处，主要有：

(1)通用语言规范(Common Language Specification，CLS)：.NET系统包括如下语言：C#，C++，VB，J#，他们都遵守通用语言规范。任何遵守通用语言规范的语言源程序，都可编译为相同的中间语言代码，由CLR负责执行。只要为其他操作系统编制相应的CLR，中间语言代码也可在其他系统中运行。

(2)自动内存管理：CLR内建垃圾收集器，当变量实例的生命周期结束时，垃圾收集器负责收回不被使用的实例占用的内存空间。不必象C和C++语言，用语句在堆中建立的实例，必须用语句释放实例占用的内存空间。也就是说，CLR具有自动内存管理功能。

(3)交叉语言处理：由于任何遵守通用语言规范的语言源程序，都可编译为相同的中间语言代码，不同语言设计的组件，可以互相通用，可以从其他语言定义的类派生出本语言的新类。由于中间语言代码由CLR负责执行，因此异常处理方法是一致的，这在调试一种语言调用另一种语言的子程序时，显得特别方便。

(4)增加安全：C#语言不支持指针，一切对内存的访问都必须通过对象的引用变量来实现，只允许访问内存中允许访问的部分，这就防止病毒程序使用非法指针访问私有成员。也避

免指针的误操作产生的错误。CLR 执行中间语言代码前，要对中间语言代码的安全性，完整性进行验证，防止病毒对中间语言代码的修改。

(5)版本支持：系统中的组件或动态联接库可能要升级，由于这些组件或动态联接库都要在注册表中注册，由此可能带来一系列问题，例如，安装新程序时自动安装新组件替换旧组件，有可能使某些必须使用旧组件才可以运行的程序，使用新组件运行不了。在.NET 中这些组件或动态联接库不必在注册表中注册，每个程序都可以使用自带的组件或动态联接库，只要把这些组件或动态联接库放到运行程序所在文件夹的子文件夹 bin 中，运行程序就自动使用在 bin 文件夹中的组件或动态联接库。由于不需要在注册表中注册，软件的安装也变得容易了，一般将运行程序及库文件拷贝到指定文件夹中就可以了。

(6)完全面向对象：不象 C++语言，即支持面向过程程序设计，又支持面向对象程序设计，C#语言是完全面向对象的，在 C#中不再存在全局函数、全区变量，所有的函数、变量和常量都必须定义在类中，避免了命名冲突。C#语言不支持多重继承。

1.2 编写控制台应用程序

第一个程序总是非常简单的，程序首先让用户通过键盘输入自己的名字，然后程序在屏幕上打印一条欢迎信息。程序的代码是这样的：

```
using System;//导入命名空间//为 C#语言新增解释方法，解释到本行结束
class Welcome//类定义，类的概念见下一节
{ /*解释开始，和 C 语言解释用法相同
  解释结束*/
  static voidMain()//主程序，程序入口函数，必须在一个类中定义
  { Console.WriteLine("请键入你的姓名:");//控制台输出字符串
    Console.ReadLine();//从键盘读入数据，输入回车结束
    Console.WriteLine("欢迎!");
  }
}
```

可以用任意一种文本编辑软件完成上述代码的编写，然后把文件存盘，假设文件名叫做 welcome.cs，C#源文件是以 cs 作为文件的扩展名。和 C 语言相同，C#语言是区分大小写的。高级语言总是依赖于许多在程序外部预定义的变量和函数。在 C 或 C++中这些定义一般放到头文件中，用#include 语句来导入这个头文件。而在 C#语言中使用 using 语句导入名字空间，using System 语句意义是导入 System 名字空间，C#中的 using 语句的用途与 C++中#include 语句的用途基本类似，用于导入预定义的变量和函数，这样在自己的程序中就可以自由地使用这些变量和函数。如果没有导入名字空间的话我们该怎么办呢？程序还能保持正确吗？答案是肯定的，那样的话我们就必须把代码改写成下面的样子：

```
class Welcome
{ static void Main()
  { System.Console.WriteLine("请键入你的姓名:");
    System.Console.ReadLine();
    System.Console.WriteLine("欢迎!");
```

```
    }
}
```

也就是在每个 Console 前加上一个前缀 System.，这个小原点表示 Console 是作为 System 的成员而存在的。C#中抛弃了 C 和 C++中繁杂且极易出错的操作符象::和->等，C#中的复合名字一律通过.来连接。System 是.Net 平台框架提供的最基本的名字空间之一，有关名字空间的详细使用方法将在以后详细介绍，这里只要学会怎样导入名字空间就足够了。

程序的第二行 class Welcome 声明了一个类，类的名字叫做 Welcome。C#程序中每个变量或函数都必须属于一个类，包括主函数 Main()，不能象 C 或 C++那样建立全局变量。C#语言程序总是从 Main()方法开始执行，一个程序中不允许出现两个或两个以上的 Main()方法。请牢记 C#中 Main()方法必须被包含在一个类中，Main 第一个字母必须大写，必须是一个静态方法，也就是 Main()方法必须使用 static 修饰。static void Main()是类 Welcome 中定义的主函数。静态方法意义见以后章节。

程序所完成的输入输出功能是通过 Console 类来完成的，Console 是在名字空间 System 中已经定义好的一个类。Console 类有两个最基本的方法 WriteLine 和 ReadLine。ReadLine 表示从输入设备输入数据，WriteLine 则用于在输出设备上输出数据。

如果在电脑上安装了 Visual Studio.Net，则可以在集成开发环境中直接选择快捷键或菜单命令编译并执行源文件。如果您不具备这个条件，那么至少需要安装 Microsoft.Net Framework SDK，这样才能够运行 C#语言程序。Microsoft.Net Framework SDK 中内置了 C#的编译器 csc.exe，下面让我们使用这个微软提供的命令行编译器对程序 welcome.cs 进行编译。假设已经将 welcome.cs 文件保存在 d:\Charp 目录下，启动命令行提示符，在屏幕上输入一行命令：d:回车，cd Charp 回车，键入命令：

C:\WINNT\Microsoft.NET\Framework\v1.0.3705\csc welcome.cs

如果一切正常 welcome.cs 文件将被编译，编译后生成可执行文件 Welcome.exe。可以在命令提示符窗口运行可执行文件 Welcome.exe，屏幕上出现一行字符提示您输入姓名：请键入你的姓名：输入任意字符并按下回车键，屏幕将打印出欢迎信息：欢迎！如图 1-1 所示。

图　1-1

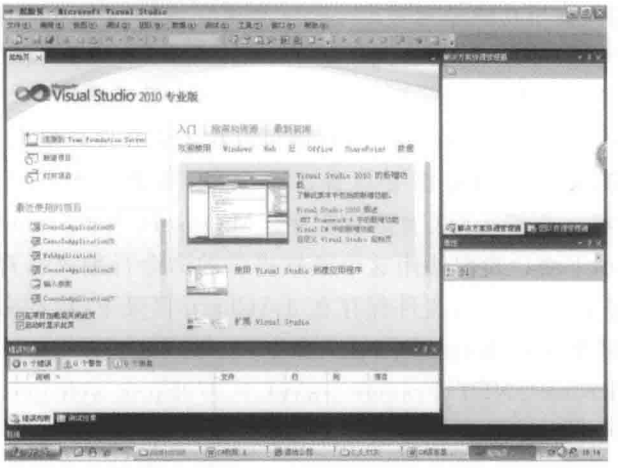
注意，与我们使用过的绝大多数编译器不同，在 C#中编译器只执行编译这个过程，而在 C 和 C++中要经过编译和链接两个阶段。换而言之 C#源文件并不被编译为目标文件.obj，而是直接生成可执行文件.exe 或动态链接库.dll，C#编译器中不需要包含链接器。

使用 Visual Studio2010 建立控制台程序的步骤：

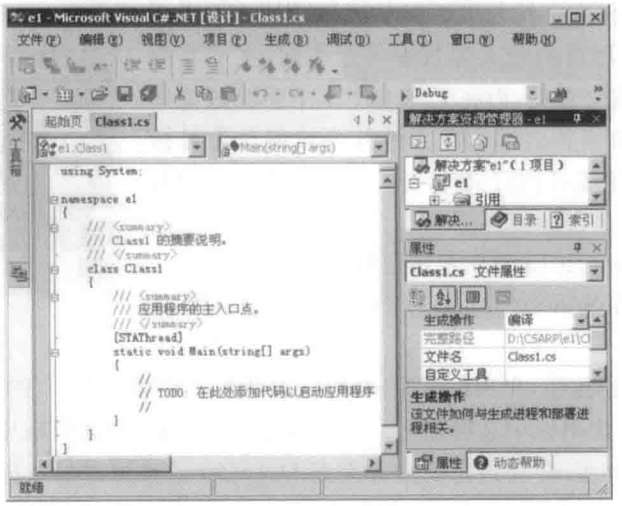
(1)运行 Visual Studio2010 程序，出现如图 1-2 所示界面。

(2)单击新建项目按钮，出现如图 1-3 所示对话框。在项目类型(P)编辑框中选择 Visual C#项目，在模板(T)编辑框中选择控制台应用程序，在名称(N)编辑框中键入 e1，在位置(L)编辑框中键入 D:\csarp，必须预先创建文件夹 D:\csarp。也可以单击浏览按钮，在打开文件对话框中选择文件夹。单击确定按钮，创建项目。出现如图 1-3 所示界面。编写一个应用程序，可能包含多个文件，才能生成可执行文件，所有这些文件的集合叫做一个项目。

图 1-2



(a)



(b)

图 1-3

(3)修改 class1.cs 文件如下,有阴影部分是新增加的语句,其余是集成环境自动生成的。

```
using System;
namespace e1
{
    /// <summary>
    /// Class1 的摘要说明
    /// </summary>
    class Class1
    {
        /// <summary>
        /// 应用程序的主入口点
        /// </summary>
        [STAThread]
        static void Main(string[] args)
        {
            //
            // TODO: 在此处添加代码以启动应用程序
            //
            Console.WriteLine("请键入你的姓名:");
            Console.ReadLine();
            Console.WriteLine("欢迎!");
        }
    }
}
```

按 Ctrl+F5 键,运行程序,如图 1-1 所示,屏幕上出现一行字符,提示您输入姓名:请键入你的姓名:输入任意字符并按下回车键,屏幕将打印出欢迎信息:欢迎! 输入回车退出程序。

1.3　类的基本概念

C#语言是一种现代的、面向对象的语言。面向对象程序设计方法提出了一个全新的概念:类,它的主要思想是将数据(数据成员)及处理这些数据的相应方法(函数成员)封装到类中,类的实例则称为对象。这就是我们常说的封装性。

1.3.1　什么是类

类可以认为是对结构的扩充,它和 C 中的结构最大的不同是:类中不但可以包括数据,还包括处理这些数据的函数。类是对数据和处理数据的方法(函数)的封装。类是对某一类具有相同特性和行为的事物的描述。例如,定义一个描述个人情况的类 Person 如下:

```
using System;
classPerson//类的定义,class 是保留字,表示定义一个类,Person 是类名
{   private string name="张三";//类的数据成员声明
    private int age=12;//private 表示私有数据成员
```

```
    public void Display()//类的方法(函数)声明,显示姓名和年龄
    {   Console. WriteLine("姓名:{0},年龄:{1}",name,age);
    }
    publicvoid SetName(string PersonName)//修改姓名的方法(函数)
    {   name=PersonName;
    }
    publicvoid SetAge(int PersonAge)
    {   age=PersonAge;
    }
}
```

Console. WriteLine("姓名:{0},年龄:{1}",name,age)的意义是将第二个参数变量 name 变为字符串填到{0}位置,将第三个参数变量 age 变为字符串填到{1}位置,将第一个参数表示的字符串在显示器上输出。

大家注意,这里我们实际定义了一个新的数据类型,为用户自己定义的数据类型,是对个人的特性和行为的描述,它的类型名为 Person,和 int,char 等一样为一种数据类型。用定义新数据类型 Person 类的方法把数据和处理数据的函数封装起来。类的声明格式如下:

属性 类修饰符 class 类名{类体}

其中,关键字 class、类名和类体是必须的,其他项是可选项。类修饰符包括 new,public,protected,internal,private,abstract 和 sealed,这些类修饰符以后介绍。类体用于定义类的成员。

1.3.2 类成员的存取控制

一般希望类中一些数据不被随意修改,只能按指定方法修改,即隐蔽一些数据。同样一些函数也不希望被其他类程序调用,只能在类内部使用。如何解决这个问题呢?可用访问权限控制字,常用的访问权限控制字如下:private(私有),public(公有)。在数据成员或函数成员前增加访问权限控制字,可以指定该数据成员或函数成员的访问权限。

私有数据成员只能被类内部的函数使用和修改,私有函数成员只能被类内部的其他函数调用。类的公有函数成员可以被类的外部程序调用,类的公有数据成员可以被类的外部程序直接使用修改。公有函数实际是一个类和外部通讯的接口,外部函数通过调用公有函数,按照预先设定好的方法修改类的私有成员。对于上述例子,name 和 age 是私有数据成员,只能通过公有函数 SetName()和 SetAge()修改,即它们只能按指定方法修改。

这里再一次解释一下封装,它有两个意义,第一是把数据和处理数据的方法同时定义在类中。第二是用访问权限控制字使数据隐蔽。

1.3.3 类的对象

Person 类仅是一个用户新定义的数据类型,由它可以生成 Person 类的实例,C#语言叫对象。用如下方法声明类的对象:Person OnePerson=new Person();此语句的意义是建立 Person 类对象,返回对象地址赋值给 Person 类变量 OnePerson。也可以分两步创建 Person 类的对象:Person OnePerson;OnePerson=new Person();OnePerson 虽然存储的是 Person 类对象地址,但不是 C 中的指针,不能象指针那样可以进行加减运算,也不能转换为其他类型

地址，它是引用型变量，只能引用(代表)Person 对象，具体意义参见以后章节。和 C，C++不同，C#只能用此种方法生成类对象。

在程序中，可以用 OnePerson. 方法名或 OnePerson. 数据成员名访问对象的成员。例如：OnePerson. Display()，公用数据成员也可以这样访问。注意，C#语言中不包括 C++语言中的－＞符号。

1.3.4　类的构造函数和析构函数

在建立类的对象时，需做一些初始化工作，例如对数据成员初始化。这些可以用构造函数来完成。每当用 new 生成类的对象时，自动调用类的构造函数。因此，可以把初始化的工作放到构造函数中完成。构造函数和类名相同，没有返回值。例如可以定义 Person 类的构造函数如下：

```
publicPerson(string Name,int Age)//类的构造函数，函数名和类同名，无返回值
{   name=Name;
    age=Age;
}
```

当用 Person OnePerson=new Person("张五",20)语句生成 Person 类对象时，将自动调用以上构造函数。请注意如何把参数传递给构造函数。

变量和类的对象都有生命周期，生命周期结束，这些变量和对象就要被撤销。类的对象被撤销时，将自动调用析构函数。一些善后工作可放在析构函数中完成。析构函数的名字为～类名，无返回类型，也无参数。Person 类的析构函数为～ Person()。C#中类析构函数不能显示地被调用，它是被垃圾收集器撤销不被使用的对象时自动调用的。

1.3.5　类的构造函数的重载

在 C#语言中，同一个类中的函数，如果函数名相同，而参数类型或个数不同，认为是不同的函数，这叫函数重载。仅返回值不同，不能看作不同的函数。这样，可以在类定义中，定义多个构造函数，名字相同，参数类型或个数不同。根据生成类的对象方法不同，调用不同的构造函数。例如可以定义 Person 类没有参数的构造函数如下：

```
publicPerson()//类的构造函数，函数名和类同名，无返回值。
{   name="张三";
    age=12;
}
```

用语句 Person One Person=new Person("李四",30)生成对象时，将调用有参数的构造函数，而用语句 Person OnePerson=new Person()生成对象时，调用无参数的构造函数。由于析构函数无参数，因此，析构函数不能重载。

1.3.6　使用 Person 类的完整的例子

下边用一个完整的例子说明 Person 类的使用：(VisualStudio. Net 编译通过)

```
using System;
namespace e1//定义以下代码所属命名空间，意义见以后章节
{   class Person
```

```
{   private String name="张三";//类的数据成员声明
    private int age=12;
    public void Display()//类的方法(函数)声明,显示姓名和年龄
    {   Console.WriteLine("姓名:{0},年龄:{1}",name,age);
    }
    public void SetName(string PersonName)//指定修改姓名的方法(函数)
    {   name=PersonName;
    }
    public void SetAge(int PersonAge)//指定修改年龄的方法(函数)
    {   age=PersonAge;
    }
    public Person(string Name,int Age)//构造函数,函数名和类同名,无返回值
    {   name=Name;
        age=Age;
    }
    publicPerson()//类的构造函数重载
    {   name="田七";
        age=12;
    }
}
class Class1
{   static void Main(string[] args)
    {   Person OnePerson=new Person("李四",30);//生成类的对象
        OnePerson.Display();
//下句错误,在其他类(Class1 类)中,不能直接修改 Person 类中的私有成员
        //OnePerson.name="王五";
//只能通过 Person 类中公有方法 SetName 修改 Person 类中的私有成员 name
        OnePerson.SetName("王五");
        OnePerson.SetAge(40);
        OnePerson.Display();
        OnePerson=new Person();
        OnePerson.Display();
    }
}
```

按 Ctrl+F5 键运行后,显示的效果是:

```
姓名:李四,年龄:30
姓名:王五,年龄:40
姓名:田七,年龄:12
```

1.4 C#的数据类型

从大的方面来分,C#语言的数据类型可以分为值类型、引用类型、指针类型 3 种。指针

类型仅用于非安全代码中。本节重点讨论值类型和引用类型。

1.4.1　值类型和引用类型区别

在C#语言中，值类型变量存储的是数据类型所代表的实际数据，值类型变量的值（或实例）存储在栈（Stack）中，赋值语句是传递变量的值。引用类型（例如类就是引用类型）的实例，也叫对象，不存在栈中，而存储在可管理堆（Managed Heap）中，堆实际上是计算机系统中的空闲内存。引用类型变量的值存储在栈（Stack）中，但存储的不是引用类型对象，而是存储引用类型对象的引用，即地址，和指针所代表的地址不同，引用所代表的地址不能被修改，也不能转换为其他类型地址，它是引用型变量，只能引用指定类对象，引用类型变量赋值语句是传递对象的地址。如：

```
using System;
class MyClass//类为引用类型
{   public int a=0;
}
class Test
{   static void Main()
    {   f1();
    }
    static public void f1()
    {   int v1=1;//值类型变量 v1,其值 1 存储在栈(Stack)中
        int v2=v1;//将 v1 的值(为 1)传递给 v2,v2=1,v1 值不变
        v2=2;//v2=2,v1 值不变。
        MyClass r1=new MyClass();//引用变量 r1 存储 MyClass 类对象的地址
        MyClass r2=r1;//r1 和 r2 都代表是同一个 MyClass 类对象
        r2.a=2;//和语句 r1.a=2 等价
    }
}
```

存储在栈中的变量，当其生命周期结束，自动被撤销，例如，v1 存储在栈中，v1 和函数 f1 同生命周期，退出函数 f1，v1 不存在了。但在堆中的对象不能自动被撤销。因此 C 和 C++ 语言，在堆中建立的对象，不使用时必须用语句释放对象占用的存储空间。.NET 系统 CLR 内建垃圾收集器，当对象的引用变量被撤销，表示对象的生命周期结束，垃圾收集器负责收回不被使用的对象占用的存储空间。例如，上例中引用变量 r1 及 r2 是 MyClass 类对象的引用，存储在栈中，退出函数 f1，r1 和 r2 都不存在了，在堆中的 MyClass 类对象也就被垃圾收集器撤销。也就是说，CLR 具有自动内存管理功能。

1.4.2　值类型变量分类

C#语言值类型可以分为以下几种。

（1）简单类型（Simple types）。简单类型中包括：数值类型和布尔类型（bool）。数值类型又细分为：整数类型、字符类型（char）、浮点数类型和十进制类型（decimal）。

（2）结构类型（Struct types）。

(3)枚举类型(Enumeration types)。C#语言值类型变量无论如何定义,总是值类型变量,不会变为引用类型变量。

1.4.3 结构类型

结构类型和类一样,可以声明构造函数、数据成员、方法、属性等。结构和类的最根本的区别是结构是值类型,类是引用类型。和类不同,结构不能从另外一个结构或者类派生,本身也不能被继承,因此不能定义抽象结构,结构成员也不能被访问权限控制字 protected 修饰,也不能用 virtual 和 abstract 修饰结构方法。在结构中不能定义析构函数。虽然结构不能从类和结构派生,可是结构能够继承接口,结构继承接口的方法和类继承接口的方法基本一致。下面例子定义一个点结构 point:

```
using System;
struct point//结构定义
{   public int x,y;//结构中也可以声明构造函数和方法,变量不能赋初值
}
class Test
{   static void Main()
    {   point P1;
        P1.x=166;
        P1.y=111;
        point P2;
        P2=P1;//值传递,使 P2.x=166,P2.y=111
        point P3=new point();//用 new 生成结构变量 P3,P3 仍为值类型变量
    }//用 new 生成结构变量 P3 仅表示调用默认构造函数,使 x=y==0
}
```

1.4.4 简单类型

简单类型也是结构类型,因此有构造函数、数据成员、方法、属性等,因此下列语句 inti=int.MaxValue;string s=i.ToString()是正确的。即使一个常量,C#也会生成结构类型的实例,因此也可以使用结构类型的方法,例如:string s=13.ToString()是正确的。简单类型包括:整数类型、字符类型、布尔类型、浮点数类型、十进制类型。见表 1-1。

表 1-1

保留字	System 命名空间中的名字	字节数	取值范围
sbyte	System.Sbyte	1	−128～127
byte	System.Byte	1	0～255
short	System.Int16	2	−32768～32767
ushort	System.UInt16	2	0～65535
int	System.Int32	4	−2147483648～2147483647
uint	System.UInt32	4	0～4292967295