《C#程序设计》教案

第 1 章

课时内容	.NET 架构				
授课日期		授课时长		课时	
教学目标	思政目标： ☞理解学习的重要性，树立正确的职业道德观 ☞了解.NET 的框架内容，对其发展历史有较清楚的认识，培养探究精神 技能要求： ☞理解什么是.NET、C#语言 ☞掌握.NET 架构的相关知识 实践目标： ☞能够知道.NET 架构的组成 ☞能够制定专业发展目标				
教学设计	教学思路：通过对.NET 发展背景的讲解，掌握.NET 框架内容 教学手段：通过课件展示和软件知识的解析深入学习 C#的背景知识。				
教学环节	教学内容				
讨论问题	1.什么是.NET? 2..NET 和 C#之间的关系是什么？ 3..NET 和微软的关系是什么？				
内容大纲	**1.1 .NET 平台概述** 2000 年 6 月微软公司公布了其下一代基于互联网平台的软件开发构想，即.NET。.NET 的构想在 IT 行业引发了极大的反响，持续至今。 **1.2 .NET 优势** .NET 是第三代因特网的高分布式环境下的应用程序开发，实现不同语言和平台的高度交互，基于开放的互联网标准和协议而构建的新一代计算和通信平台。.NET 是一款跨语言开发平台，它不是一门编程语言。 **1.3 .NET 相关的部分程序设计语言** 1. VB 语言 2. C 语言 3. C++ 4. C# **1.4 .NET 的相关课程**				

	与.NET 开发有关的方向及课程有 C#程序设计、SQL Server 数据库、ASP.NET 应用开发、ASP.NET MVC 高级开发、.NET 开发综合实战、UML 建模与设计模式、软件测试技术等。 **1.5 .NET 的核心组成** .NET 主要由三大部分组成，分别是 CLR、BCL 和编程工具。
课后练习	1. 简述.NET 框架。 2. 如何理解语言无关性？ 3. 如何理解 CLR、BCL 的作用? 4. 如何理解.NET 与 C#的关系?

《C#程序设计》教案

第 2 章

课时内容	开发环境				
授课日期		授课时长		课时	
教学目标	思政目标： ☞通过搭建开发环境练习，正确使用相关开发工具，探究事物规律 ☞了解 C#的大致内容，培养规范化的编程行为，维护网络安全 ☞理解学习的重要性，树立正确的职业道德观 技能要求： ☞掌握开发环境的配置 ☞掌握 Visual Studio 的使用方法 实践目标： ☞能够正确安装并使用 Visual Studio				

	☞能够创建项目，调试程序
教学设计	教学思路：通过搭建开发环境，掌握软件的使用；通过创建项目案例，学习调试程序。 教学手段：通过课件展示和软件操作学习开发环境的搭建和软件的使用。
教学环节	教学内容
讨论问题	1.什么是 C#语言? 2.Visual Studio 是什么？ 3.调试程序的步骤有哪些？
内容大纲	**2.1 Visual Studio 的介绍** Microsoft Visual Studio 是微软公司开发的集成工具。它性能优良，有出色的代码管控工具、集成开发环境，其代码适用于微软支持的所有平台。微软支持的平台有 Microsoft Windows、Windows Phone、Windows Mobile、.NET、Windows CE、.NET Core、.NET Compact Framework 和 Microsoft Silverlight。 **2.2 Visual Studio 安装与配置** Visual Studio 是功能完备的编程工具，是性能优良的集成开发环境。它是众多编程人员的首选开发工具，适用于 Windows、Web、Android 以及 iOS 开发。 **2.3 工具栏与使用技巧** 2.3.1 工具栏 VS 工具栏中有文件、编辑、视图、Git、项目、生成、调试、测试、分析、工具、扩展、窗口和其他等功能。 2.3.2 Visual Studio 使用技巧 1. 设置行号 2. 主题 3. 任务列表 **2.4 创建项目** 通过创建一个简单的项目，学习 Visual Studio 的使用。通过运行程序，掌握正确运行软件的方法。
课后练习	1. 安装 Visual Studio。 2. 将 Visual Studio 主题设置为深色主题。 2. 练习创建一个新项目。 3. 调试初始化程序，保存程序文件。

《C#程序设计》教案

第 3 章

课时内容	C#程序与数据类型				
授课日期		授课时长		课时	
教学目标	思政目标： ☞通过学习数据类型及转换，培养学生的认真、细致的编程态度 ☞通过编程练习，培养学生分析问题和解决问题的能力 技能要求： ☞掌握 C#基本语法要求 ☞掌握 C#中的基本数据类型及转换 ☞正确理解变量的命名规则，学会使用值类型和引用类型 实践目标： ☞通过认识 C#基本语法，为深入学习打基础 ☞认识变量与数据类型，能够在编程中正确运用 ☞熟悉变量的使用，能够在适当场景中使用				
教学设计	教学思路：通过案例讲解学习数据类型及转换；通过编程练习掌握 C#基本语法要求。 教学手段：通过课件展示、编程练习、课后习题学习 C#中的基本数据类型及转换和变量的命名规则。				
教学环节	教学内容				
讨论问题	1.C#中的数据类型有哪些? 2.数据类型之间如何进行转换？ 3.C#中变量的命名规则有哪些？ 4.变量和常量之间有什么不同？				
内容大纲	**3.1　编写首个 C#程序** 3.1.1　编写程序 我们从一个简单的 C#程序开始，去认识 C#语言的基本构成。输入如图 3 - 1 所示代码，运行结果如图 3 - 2 所示。				

```
using System;

namespace ConsoleApp2
{
    0 个引用
    class Program
    {
        0 个引用
        static void Main(string[] args)
        {
            Console.WriteLine("欢迎来到C#的世界");
            Console.ReadKey();
            /*这是第一个程序*/
        }
    }
}
```

图 3-1　第一个 C#程序

图 3-2　运行结果

观察上述程序，分析程序的组成部分。从运行结果可知，代码中的语句被输出到屏幕。整个程序包括 using System、namespace、class、Main 方法、语句或表达式，以及注释。这些内容构成了一个完整的程序。

3.1.2　程序详解

通过分析程序的组成，了解程序的基础内容。程序的组成如图 3 - 3 所示。

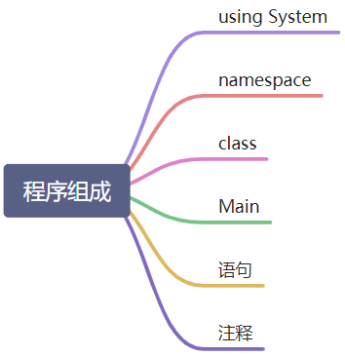

图 3-3　程序组成

3.1.3　注释方法

在程序中，经常用到注释。注释是一种备注手段，主要为了方便程序编写和维护人员修改、维护代码。注释内容并不会被程序编译器运行，即注释不会影响程序运行的结果。在 C#中，主要有 3 种注释方式。

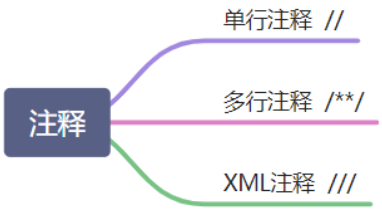

图 3-4　注释方法

3.1.4　标识符与关键字

1. 标识符

标识符是用来命名变量、方法、参数等的一种字符串。程序中标识符的位置有规定：数字不能放在首位；字母和下划线随意使用；@字符只能放在标识符的首位；标识符区分大小写，如 MyCar 与 myCar 是不同的标识符。需要注意的是，标识符不能与 C#的类库名称相同。

2. 关键字

关键字是 C#编译器预定的保留字，用于定义固定内容。关键字不能用作变量名，也不能用作标识符，关键字应全部小写。

3.2　常量与变量

3.2.1　常量

常量是指在程序运行过程中，值不改变的量。通常，程序中多次出现且固定不变的值定义为常量。

3.2.2　变量

变量是指在程序运行过程中，值可以改变的量。变量具备两个要点：变量名和数据类型。变量名可以方便访问变量中所存储的数据，而数据类型决定了变量的存储方式。变量被分配存储空间后才能正常使用，而变量名其实就是为其所分配内存空间的命名。通过变量名可以访问相应存储空间中所存储的数据。

1. 变量的命名
2. 变量的命名法
3. 变量的赋值

3.3　数据类型

3.3.1　整型

表　整型数据类型

类型	说明	范围

sbyte	8 位有符号整数	-128～127
byte	8 位无符号整数	0～255
short	16 位有符号整数	-32 768～32 767
ushort	16 位无符号整数	0～65 535
int	32 位有符号整数	-2 147 483 648～2 147 483 647
uint	32 位无符号整数	0～4 294 967 295
long	64 位有符号整数	-9 223 372 036 854 775 808～9 223 372 036 854 775 807
ulong	64 位无符号整数	0～18 446 744 073 709 551 615

3.3.2　布尔类型

bool 类型的变量仅有两个值：true 和 false。true 和 false 两种状态的转换可以通过“!”运算符来实现。

3.3.3　char 类型

无论中文字符、英文字符或者数字都归属于 char 类型，char 类型占两个字节，故最多可以容纳 65 536 个符号，其取值范围为 0～65 535。char 类型的赋值需要以成对单引号标记。

3.3.4　枚举类型

枚举类型被用来表达若干固定值，该类型使用 enum 定义，定义方式如下：

enum 枚举类型名称{枚举元素 1[=数值 1]， 枚举元素 2[=数值 2]，…}

3.3.5　隐式类型

隐式类型用 var 声明，var 关键字是“万能类型”的定义方式，可以用来声明任何类型的变量，但并不意味着声明之后其类型仍不确定。

3.3.6　浮点类型

浮点类型，分别是 float、double 和 decimal，如表 3 -

	4 所示。float 和 double 都为浮点类型，分别称为单精度浮点型和双精度浮点型；而 decimal，一般用于精度要求高的场合。小数类型（decimal）是高精度的数据类型，占用 16 个字节（128 位），主要为了满足需要高精度的财务和金融计算机领域。 3.3.7　数据类型转换 1. 隐式转换 2. 显式转换 3. Type.Parse 4. Convert 5. 装箱与拆箱
课后练习	1. 练习本章所有示例代码。 2. 简述数据类型之间的区别。 3. 编写程序并运行，使其输出结果为如图所示的金字塔。  金字塔

《C#程序设计》教案

第 4 章

课时内容	运算符与语句				
授课日期		授课时长		课时	
教学目标	思政目标： ☞加强动手能力，培养实事求是、严谨的学习态度 ☞认真修改并测试程序代码，树立新时期下的程序员精神 技能要求： ☞熟练掌握程序各种语句 ☞掌握 C#中的运算符 实践目标： ☞根据所学的示例，能够自己实现一个简单程序				

	☞熟练掌握语句，能够在不同条件下正确使用
教学设计	教学思路：

<table>
<tr><td></td><td>通过对比不同运算符的用法掌握运算法；通过案例讲解运算符的使用；通过表格记忆运算符的优先级；通过案例讲解流程控制语句。
教学手段：通过课件展示、编程练习、课后习题学习运算符和流程控制语句。</td></tr>
<tr><td>教学环节</td><td>教学内容</td></tr>
<tr><td>讨论问题</td><td>1.运算符是什么？
2.流程控制语句是什么？
3.运算符的优先级是什么意思？</td></tr>
<tr><td>内容大纲</td><td>4.1　运算符
运算符是用来标明数值或表达式运算规则的一种符号标记，它所操作的数值称为操作数。

表达式就是由运算符和操作数组合而成。根据操作数的个数，运算符可以分为一元运算符、二元运算符和三元运算符。例如，取负（－）、取反（~）、自增（++）和自减（－－）都是典型的一元运算符；而四则运算符（+、—、*、/）则是典型的二元运算符；三元运算符仅有一个，那就是条件运算符（？:），可以用来改写简单的 if 结构语句。

根据运算的类型，运算符又可以分为算术运算符、赋值运算符、关系运算符、逻辑运算符和条件运算符等。

4.1.1　算术运算符
最常见的运算符就是算术运算符，算术运算符的具体内容如表 4 - 1 所示。

表 4-1　算术运算符
<table><tr><th>运算符</th><th>说明</th><th>示例</th></tr><tr><td>＋</td><td>加</td><td>int s=2012+2015；
int i=0，j=1；int k=i+j；</td></tr><tr><td>—</td><td>减</td><td>int s=2012-2015；
int i=0，j=1；int k=i-j；</td></tr><tr><td>*</td><td>乘</td><td>int s=2012*2015；
int i=0，j=1；int k=i*j；</td></tr><tr><td>/</td><td>除</td><td>int s=2012/15；
int i=2020，j=10；int k=i/j；</td></tr><tr><td>%</td><td>取模</td><td>int s=2012%15；
int i=2020 ， j=10 ； int</td></tr></table></td></tr>
</table>

			k=i%j;

＋＋	自增 1	int i=2012； int j=i++，k=++i；
－－	自减 1	int i=2015； int j=i - -，k= - -i；

4.1.2　关系运算符

关系运算符的运算结果是布尔值，要么为 true，要么为 false。关系运算符用于比较两个操作数的大小关系，值是比较的结果，如表 4 - 2 所示。

表 4-2　关系运算符

运算符	说明	运算符	说明
==	等于	<	小于
!=	不等于	<=	小于或等于
>	大于	>=	大于或等于

注意：==与=表示含义不相同，前者表示相等，后者表示赋值。

4.1.3　赋值运算符

表 4-3　赋值运算符

运算符	说明	运算符	说明
=	赋值	<<=	左移赋值
+=	加法赋值	>>=	右移赋值
－=	减法赋值	&=	and 位操作赋值
*=	乘法赋值	|=	or 位操作赋值
/=	除法赋值	^=	xor 位操作赋值
%=	取模赋值		

4.1.4　逻辑运算符

逻辑运算符有&、|、!、^、~、&&、||。其中，&和|执行按位的“与”和“或”，而~和^执行按位的“非”和“异或”。&&和||执行布尔的“与”和“或”，而！执行布尔的“非”。另外，要注意区分位运算和布尔运算。

表 4-4　二进制数位运算的结果

运算类型	值
x&y	x 和 y 同时为 1 时，结果为 1，其他情况结果均为 0
x|y	x 和 y 任一个为 1 时，结果为 1，同时为 0 时结果为 0
x^y	x 和 y 同为 0 或 1 时，结果为 0，x 和 y 的取值不同时结果为 1
~X	x 为 0 时结果为 1，x 为 1 时结果为 0

4.1.5　条件运算符

条件运算符是一个三元运算符，由“?”和“：”组成，以条件运算符构成的表达式称为条件表达式。其一般格式如下：

操作数 1？操作数 2：操作数 3

4.1.6　位运算符

表 4-6　位运算符

运算符	说明	运算符	说明
&	and（与）	|	or（或）
～	取反	^	xor（异或）
>>	右移位	<<	左移位

4.1.7　自增自减

自增运算符（++）和自减运算符（－－）的使用频率相对较高，它们都具有两种形式：前缀和后缀。此处主要介绍自增运算。

前缀自增便是++在操作符前面，如++i；后缀自增便是++在操作符后面，如 i++。两者都是实现 i 递增 1，即 i=i+1。

4.2　运算符的优先级

常见运算符的优先级如表 4 - 11 所示，从上到下优先级逐渐降低。

表 4-11　运算符的优先级

运算符类型	运算符
初级运算符	（），[ ]，x.y，++（后缀），－－（后缀），new，sizeof，typeof，checked/unchecked
一元运算符	!，~，++（前缀），－－（前缀），（T）x
乘除、取模运算符	*，/，%
增量运算符	+，－
移位运算符	<<，>>
关系运算符	<，>，<=，>=，is，as
等式运算符	==，!=
逻辑与运算符	&
逻辑异或运算符	^
逻辑或运算符	|
条件与运算符	&&
条件或运算符	||
条件运算符	?：
赋值运算符	=，*=，/=，+=，-=，<<=，>>=，&=，^=，|=，%=

4.3　流程控制语句

4.3.1　if-else 语句

if 语句是最常见的程序流程控制语句，它可以配合 else 或者 else if 来无限扩展选择执行的分支。if 语句的使用形式有如下四种，但无论采用哪种方式，无论产生多少分支，最终，也只有一个分支能够得以运行。

4.3.2　switch 语句

switch 语句与 if 语句类似，也是在众多分支中选择一个匹配的分支来执行。其执行机制是：根据表达式的值，在各个 case 中寻找相匹配的，若找到，则执行相应的语句序列直到遇到 break，若没有，则在 default

	分支存在的情况下，执行 default 分支。 4.3.3　for 语句 for 语句是最常使用的循环语句，特点是使用方式的灵活。其执行机制是：首先，执行初始化语句；其次，执行条件测试语句，当条件测试语句返回 true 时，接着执行循环语句序列；最后，执行迭代语句，这是第一次循环的过程，除第一次循环，其他时刻不再执行初始化语句。从第二次循环开始，每次首先执行条件测试语句，成立则执行循环语句序列，再执行迭代语句，然后又进入下一轮循环的条件测试语句判断，直至该语句不成立时，整个循环方才结束。 4.3.4　while 语句 其执行机制是：首先执行条件表达式，若为真则执行循环语句序列，接着再执行条件表达式，直到条件表达式不成立退出循环为止，继而执行循环体之外的语句。当条件表达式第一次就不成立时，此时循环语句序列不会获得任何执行机会。 4.3.5　do-while 语句 其执行机制是：首先执行循环语句序列，然后执行条件表达式，若为真则接着执行循环语句序列，接着再执行条件表达式，直到条件表达式不成立退出循环而执行循环之外的语句。 从其执行机制可以看出，do-while 与 while 的区别在于，do-while 语句中的循环语句序列至少会获得一次执行机会。 4.3.6　break 语句 break 语句可强行退出循环，即将程序的执行流程从循环内转到循环外。
课后练习	1. 练习本章所有示例代码。 2. 简述各语句的特点。 3. 假设在程序中 a、b、c 均被定义成整型，所赋的值都大于 1，则下列能正确表示代数式 1/abc 的表达式的是（　　）。 A. 1.0/a*b*c　　B. 1/(a*b*c) C. 1/a/b/(float)c　　D. 1.0/a/b/c 4. 操作题。 （1）从键盘上输入三个整数，由用户回答它们的和、差、积、商和取余运算结果，然后统计出正确答案的个数。 （2）请利用本章所学内容，找出所有介于 10~1 000 之间的偶数（使用 for、while、do…while 3 种循环分别实现）。

《C#程序设计》教案

第 5 章

课时内容	数组				
授课日期		授课时长		课时	
教学目标	思政目标： ☞逻辑是代码设计中的关键，通过学习数组，加强逻辑思维能力 ☞通过编程训练，培养动手能力及团队协作能力 技能要求： ☞掌握数组的基本概念、声明及初始化 ☞了解数组的访问与遍历 ☞了解数组的 Array 类 ☞理解并掌握索引器的定义和使用 实践目标： ☞能够在编程中熟练使用数组 ☞能够对数组进行基本的操作，为以后的编程做铺垫				
教学设计	教学思路：通过讲解，学习数组的声明、数组的初始化、Array 类和索引器等内容。 教学手段：通过课件展示、编程练习、课后习题学习数组的声明和初始化、访问数组、数组的基本操作。				
教学环节	教学内容				
讨论问题	1.数组是怎么声明的？ 2.怎样访问数组中的元素？ 3.什么是索引器？				

内容大纲	**5.1　数组的声明和初始化** 数组是一个数据结构，用来存储相同类型元素。其中，需要说明的有以下两点。 （1）数组分为一维和多维，其访问方式由相同数组名称和不同索引来实现。 （2）元素的类型可以是引用类型或者整型这些基本类型。 5.1.1　数组的声明 使用数组前，需要对数组先进行声明。 在数组的分类中，一维数组的使用最普遍，其一般声明方式如下： 类型[ ]数组名称= new 类型[数组大小]； 5.1.2　数组的初始化 数组的初始化一般采用如下方式： 类型[ ]数组名称= new 类型[数组大小]{与数组大小相等个数的元素值列表}； **5.2　访问元素** 数组的访问是通过数组名和索引来进行的，其中，索引从 0 开始 **5.3　数组的基本操作** 5.3.1　Array 类 在 C#中，System.Array 类是基类，所有数组都继承自它。该类提供了一系列的属性和方法，用于各种数组。 1. Array 的常用属性 2. Array 的常用方法 5.3.2　索引器 索引器是一个与属性很类似的类成员，可以具有 get 和 set 两个访问器，分别用于实现读和写的功能。索引的主要不同之处在于：定义索引器时一定要使用 this 关键字，而不需要像定义属性一样要程序编写人员定义一个属性名字；索引器一定需要参数；索引器不能定义为 static。
课后练习	1. 练习本章所有示例代码。 2. 简述数组和索引器的区别。 3. 随机生成 20 个整数，并且这 20 个随机数的正负性也随机处理，然后将这 20 个随机数存入数组，最后将这 20 个随机数中的正数存入另外一个数组。

《C#程序设计》教案

第 6 章

课时内容	类和方法				
授课日期		授课时长		课时	
教学目标	思政目标： ☞理论来源于实践，通过实践操作，激发对编程的兴趣 ☞通过面向对象程序设计方法的学习，提高抽象思维能力 技能要求： ☞掌握面向对象程序设计的基本概念 ☞了解类与对象、字段、属性、方法的概念及用法 ☞理解并掌握 Main( )函数和 static 方法的用法 实践目标： ☞根据示例，对所学的概念进行验证 ☞熟练应用面向对象方法实现编程设计 ☞能够通过练习进一步认识面向对象设计方法				
教学设计	教学思路：通过讲解，学习类与对象、字段、属性、方法、main（）函数、static、构造函数与析构函数、继承、多态、类的密封等内容。 教学手段：通过课件展示、编程练习、课后习题学习类与成员、类方法、类的派生。				
教学环节	教学内容				
讨论问题	1.什么是面向对象？ 2.面向对象的三大特征是什么？ 3.什么是类？ 4.类的派生有哪些？				

内容大纲	**6.1 类与成员** 6.1.1 类与对象 面向对象是一种软件开发方法，它的三大基本特征如下。 （1）封装。它将数据及对数据的操作封装为一个有机不可分割的整体，对外隐藏具体实现细节，从而实现可重用、易维护等优秀特性。 （2）继承。子类从父类获得父类特征，同时也可以扩展自己新的特征。继承也能实现代码重用。 （3）多态。多态性意味着有多重形式，往往表现为“一个接口，多个功能”。 6.1.2 字段 字段即类中的常量或者变量，它使类具备封装数据的能力。 6.1.3 属性 1. 常规属性 2. 自动属性 6.1.4 方法 1. 方法的定义与使用 2. 方法的重载 3. 参数的个数不定问题——params 4. this **6.2 类方法** 6.2.1 Main（）函数 Main( )函数也称 Main( )方法，它是一种特殊的方法。其特殊之处如下： （1）Main( )函数是可执行程序的入口点，且是唯一的入口点。 （2）Main( )函数可以带一个字符串数组参数，也可以不带参数。 （3）Main( )函数一般为 void 类型的，也可以声明为 int 类型。 （4）Main( )函数可以声明为 static，也可以声明为非 static 类型。 （5）应用程序的执行在 Main( )中开始，也在 Main( )中结束，对应的线程为主线程。 6.2.2 Static 类的成员类型分为静态的和非静态的，因而方法也分为静态方法和非静态方法（即实例方法），使用 static 的方法是静态方法，没有使用的就是非静态方法。静态方法和非静态方法的区别是：静态方法属于类所有，非静态方法属于用该类定义的对象（实例）所有。 6.2.3 构造函数与析构函数

	1. 构造函数 2. 析构函数 **6.3 类的派生** 6.3.1 继承 6.3.2 多态 6.3.3 类的密封
课后练习	1. 练习本章所有示例代码。 2. 请简述类和对象的关系。 3. 请简述方法和属性各自的作用。 4. 如何理解继承和多态？

《C#程序设计》教案

第 7 章

课时内容	委托与事件				
授课日期		授课时长		课时	
教学目标	思政目标： ☞科学观念对学习有决定性作用，通过编程培养科学观念 ☞通过委托与事件的学习，提高抽象思维能力 技能要求： ☞掌握委托与事件的基本概念 ☞了解委托与事件的使用 ☞了解匿名方法与普通方法的区别、匿名类的创建 Lambda				

	表达式的使用 实践目标： ☞根据示例，对所学的概念进行验证 ☞熟练应用面向对象方法实现编程设计
教学设计	教学思路：通过讲解，学习委托的概念、委托的使用、多播委托、事件的概念、事件的使用、三类事件等内容。 教学手段：通过课件展示、编程练习、课后习题学习委托、事件、匿名方法、Lambda 表达式。
教学环节	教学内容
讨论问题	1.什么是委托？ 2.怎样声明委托？ 3.什么是事件？ 4.怎样声明事件？
内容大纲	**7.1 委托** 7.1.1 委托 委托的作用在于：可以将方法引用封装在委托对象内，然后可以将该委托对象传递给可调用所引用方法的代码，而不必在编译时知道将调用哪个方法。通俗的说就是：通过委托，我们可以把方法当成参数传递。 7.1.2 委托的使用 1. 声明委托 2. 实例化 3. 调用 7.1.3 多播委托 **7.2 事件** 7.2.1 事件 7.2.2 事件的使用 1. 定义事件委托 2. 定义事件触发类（用于产生事件） 3. 定义事件接收类（用于处理事件） 7.2.3 三类事件 1. 无参类型 2. 带参类型 3. 自定义参数类型 **7.3 匿名方法** 匿名方法就是没有名称的方法，除此之外，它和普通方法没有什么区别。匿名方法使用时是通过 delegate 修饰。匿名方法声明的一般形式如下： 委托名 委托实例对象= delegate(参数 1，参数 2， …) {方法体；}； **7.4 Lambda 表达式**