Visual C#.Net 应用精彩50例

张龙卿　欧 洋　编著

清华大学出版社

Visual C# .Net 应用精彩 50 例

张龙卿　欧 洋　编著

清华大学出版社

(京)新登字 158 号

内容简介

本书重点讲述了 C# 语言的基础知识及使用 Visual Studio. Net 集成开发环境开发各种 C# 应用程序的技巧,内容主要包括:C# 语言基础知识、集成环境中基本工具的使用、开发控制台应用程序、开发 ASP. Net 应用程序、开发 Web 应用程序、开发数据库应用程序以及建立各种实用程序等。在讲解时,分别从实例说明、实现步骤、执行结果、关键代码及说明等方面进行分析,从而使读者学习这些实例后,可以熟练掌握使用 C# 语言开发各种类型应用程序的技巧与方法。

本书由浅入深、通俗易懂,在实例选择上既有深度,又有广度,非常适合于初、中级程序员学习,同时也可以作为大专院校学生及其他软件开发人员的学习参考书。

图书在版编目(CIP)数据

Visual C# . Net 应用精彩 50 例/张龙卿,欧阳编著. 北京:清华大学出版社,2002

ISBN 7-302-05724-9

Ⅰ. V... Ⅱ. ①张...②欧... Ⅲ. C 语言-程序设计 Ⅳ. TP312

中国版本图书馆 CIP 数据核字(2002)第 054206 号

书　　名:Visual C# . Net 应用精彩 50 例
作　　者:张龙卿　欧　洋　编著
出 版 者:清华大学出版社(北京清华大学学研大厦,邮编 100084)
http://www. tup. tsinghua. eda. cn
责任编辑:宋　韬
封面设计:付剑飞
印 刷 者:北京市清华园胶印厂
发 行 者:新华书店总店北京发行所
开　　本:787×1092　1/16　印张:17.5　字数:404 千字
版　　次:2002 年 8 月第 1 版　2002 年 8 月第 1 次印刷
书　　号:ISBN 7-302-05724-9/TP·3376
印　　数:0001～5000 册
定　　价:26.00 元

前　言

Visual Studio .Net是一套完整的开发工具，用于生成ASP Web应用程序、XML Web services应用程序、桌面应用程序和移动应用程序。Visual Basic. Net、Visual C++ . Net和Visual C# . Net全都使用相同的集成开发环境（IDE），该环境允许它们共享工具并有助于创建混合语言解决方案。另外，这些语言利用了.Net框架的功能提供对简化ASP Web应用程序和XML Web services开发的关键技术的访问。

Visual C#（发音为C sharp）是一种新的面向对象的编程语言，它从C和C++演变而来，在语句、表达式和运算符方面使用了许多C++功能。同时它又是一种类似于Java的语言，使用了与Java非常相似的语法及程序结构，但是C#与Java相比，在许多方面又进行了改进，并删除了一些不易使用的功能。所以，无论是C程序员还是Java程序员，要掌握C#语言都会非常迅速。

C#语言是一种简单和类型安全的语言。C# 语言在类型安全性、版本转换、事件和垃圾回收等方面进行了相当大的改进和创新。C#语言还提供对常用API样式（如.Net框架、COM、自动化和C样式API等）的访问。它还支持unsafe模式，在此模式下可以使用指针操作不受垃圾回收器控制的内存。

C#凭借对集成现有代码提供完全的COM/平台支持；通过提供垃圾回收和类型安全实现系统可靠性；通过提供内部代码信任机制保证系统安全性；完全支持可扩展元数据概念等功能提供生成持久系统级组件的能力。

C#还可以凭借通过COM+ 1.0和.Net框架服务提供具有紧密库访问的完全相互作用支持；对基于Web的组件交互提供XML支持等功能；与其他语言交互操作、跨平台互用并与遗留的数据交互操作，版本转换功能使管理和部署变得容易。

本书由浅入深，循序渐进地分析了50个典型实例，包括了应用程序开发的多个方面。对大家掌握C#语言会起到很大的帮助作用，因为许多有经验的程序员都明白，学习一种语言的捷径就是分析各种实例。

本书主要由张龙卿、欧洋编著，另外，下列人员也参加了本书的编写工作：李平、王谦、宋子营、吴动、孙朋、郭支清、刘雪、张劲、杜海、夏宁、倪湖、钱信、何京、景年冰、粟岭、林庆、顾虹、卞颂、惠鸣、冯敬、隋利、王挺、白群、曾文岸、武知讯、熊军、孙天平、秉易、项星臣、丁林海、霍强、王芝等，在此一并表示感谢。

由于时间紧张，作者水平有限，尽管付出了自己最大的努力，也难免存在这样或那样的错误，欢迎大家提出宝贵意见。E-mail:sdzlq123@sohu.com。

作　者

目　录

实例 1　Windows 程序 Hello World

1. 实例说明

本实例通过在窗口中显示一条“Hello world!”信息，说明在 Visual Studio. Net 集成环境中，如何使用 C＃语言建立一个 Windows 应用程序。

该应用程序的实现过程同样应用于使用 Visual Basic. Net 及 Visual C++ . Net 语言开发的应用程序中。

2. 实现步骤

（1）打开集成环境。运行 Visual Studio. Net 应用程序，则会打开其集成开发环境。

（2）新建一个项目。选择“文件”|“项目”菜单，则会打开“新建项目”对话框，在“项目类型”列表框中选择“Visual C＃项目”；在“模板”列表框中选择“Windows 应用程序”；在“名称”编辑框中为该项目输入一个名字为“Sample1 _ WHello”；在“位置”下拉列表框中输入保存项目的目录为“D: \ C＃实例”，以便将新建立的项目保存到该目录下，也可通过单击“浏览”按钮选择该目录。此时“新建项目”对话框如图 1－1 所示。

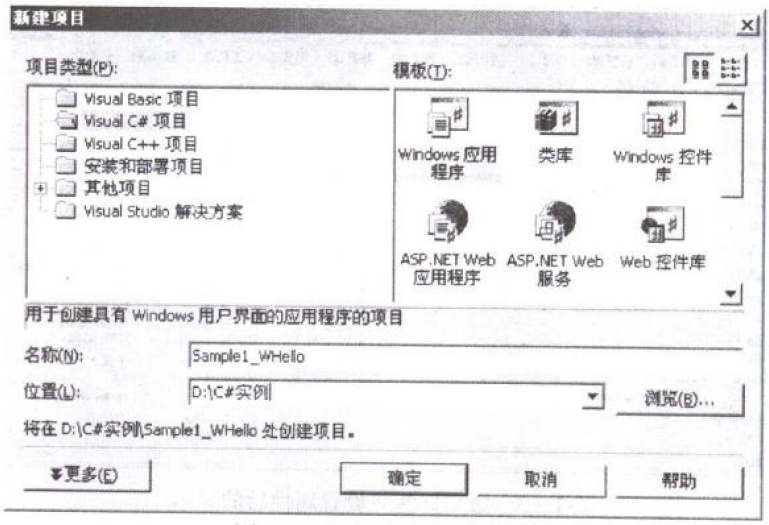

图 1－1　“新建项目”对话框

单击“确定”按钮则建立了一个新的 Windows 应用程序项目。此时 Visual Studio. Net 开发环境的显示，如图 1－2 所示。

（3）添加控件。在集成环境左边的工具箱中选中“Label”工具，然后在中间的表单中按下鼠标左键并拖动鼠标，则会在表单中添加一个标签。以后可随时用鼠标拖动该控件的边框来改变其大小。

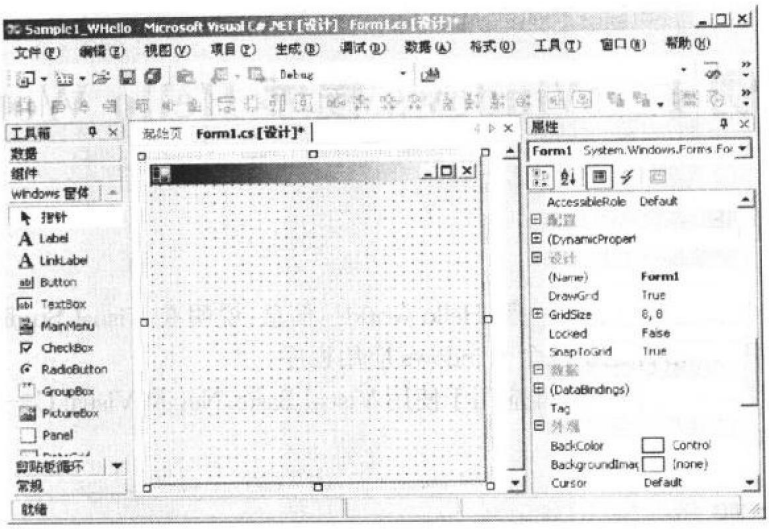

图 1-2 新建项目的显示

(4) 设置标签的属性。选择“视图”|“属性窗口”菜单或按 F4 键打开“属性窗口”,为该控件设置如下属性:在“Text”栏中,将默认属性“label1”修改为“Hello World!”;将“Font”属性,设置为“宋体”,“粗体”,“二号”;将“ForeColor”属性,设置为“HotTrack”,即字体颜色设置为蓝色。

用鼠标适当拖动 Label1 的位置,使其居于表单中央;再向上拖动表单窗口的下边框,使表单窗口变小一些。此时集成开发环境的显示,如图 1-3 所示。

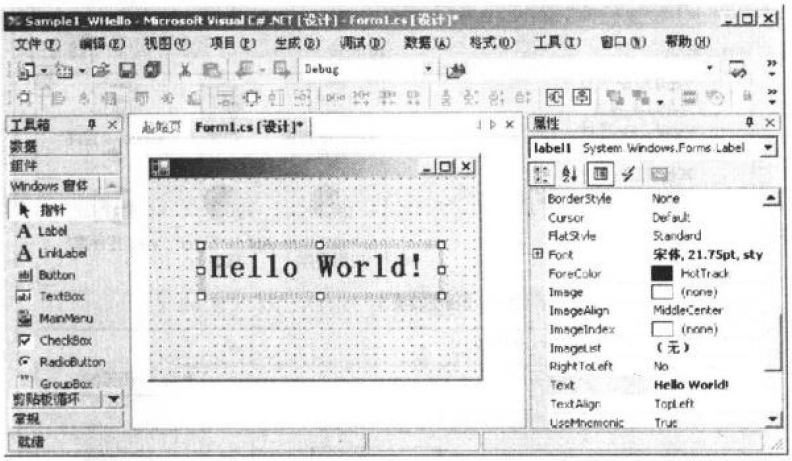

图 1-3 添加标签并设置属性后的显示

(5) 运行应用程序。选择“调试”|“开始执行(不调试)”菜单,直接运行该应用程序,此时会显示编译结果的输出窗口,如图 1-4 所示。如果应用程序没有任何错误,则会显示最终执行的结果。

(6) 保存项目。选择“文件”|“全部保存”菜单,则会保存整个项目。

(7) 关闭项目。选择“文件”|“关闭解决方案”菜单,则会关闭整个项目,并可以重新建立一个新的项目。

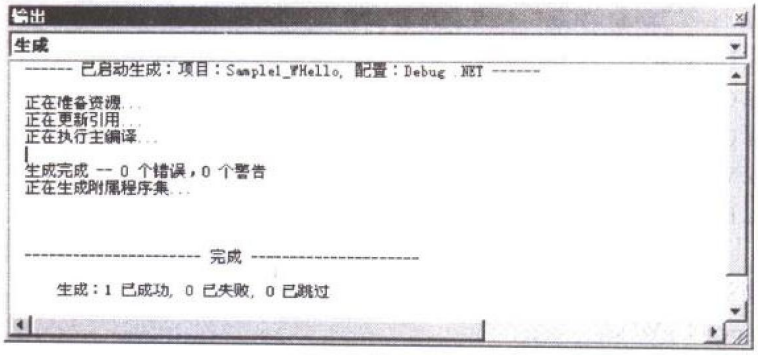

图 1-4　编译结果的输出窗口

3. 执行结果

该应用程序的执行结果如图 1-5 所示。

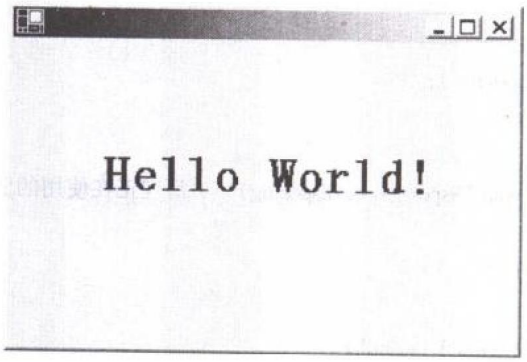

图 1-5　应用程序的执行结果

4. 关键代码及说明

该实例所有代码都是自动生成的。可以选择“视图”|“代码”菜单，或按 F7 键来查看该实例表单 Form1 对应的代码。可以到 D 盘“C# 实例 \ Sample1 _ WHello”目录下查看该项目包括的程序。

注意：

如果您现在还无法搞清楚下面代码的意义，也不要着急，可以先跳过该部分内容。随着学习的深入，自然会明白其意义。

另外应记住，C# 代码都使用 .cs 作为文件扩展名。

该实例主表单对应的源代码如下（多余的注释已经删除，并添加了新的注释），所有代码都会自动生成，在设置了表单控件属性后，这些属性的设置代码也会自动添加到源代码中：

```
//代码中使用的命名空间
```

```
using System;
using System.Drawing;
using System.Collections;
using System.ComponentModel;
using System.Windows.Forms;
using System.Data;

//定义一个新的代码空间,该命名空间在集成环境中会自动生成
namespace Sample1 _ WHello
{
public class Form1 : System.Windows.Forms.Form   //定义一个类
{
    private System.Windows.Forms.Label label1;                    //声明一个标签
    private System.ComponentModel.Container components = null;    //生命控件模型容器

    public Form1()
    {
        InitializeComponent();
    }

    protected override void Dispose(bool disposing)    //清理正在使用的资源
    {
        if( disposing )
        {
            if (components != null)
            {
                components.Dispose();
            }
        }
        base.Dispose( disposing );
    }

        #region Windows Form Designer generated code    //设计表单产生的代码
        private void InitializeComponent()          //对各个控件或变量进行初始化
        {
            this.label1 = new System.Windows.Forms.Label();
            this.SuspendLayout();
            //设置 Label1 的属性
            this.label1.Font = new System.Drawing.Font("宋体", 21.75F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((System.Byte)(134)));
            this.label1.Location = new System.Drawing.Point(48, 64);
            this.label1.Name = "label1";
            this.label1.Size = new System.Drawing.Size(208, 40);
```

```
            this.label1.TabIndex = 0;
            this.label1.Text = "Hello World!";
            // 设置 Form1 的属性
            this.AutoScaleBaseSize = new System.Drawing.Size(6, 14);
            this.ClientSize = new System.Drawing.Size(292, 181);
            this.Controls.AddRange(new System.Windows.Forms.Control[] {this.label1});
            this.ForeColor = System.Drawing.SystemColors.HotTrack;
            this.Name = "form1";
            this.Text = "";
            this.ResumeLayout(false);
        }
        #endregion                    //设计表单产生的代码段结束

        // 应用程序的主入口点
        [STAThread]                   //表示单线程的应用程序
        static void Main()            //应用程序的主方法,必须使用前面的两个限定符
        {
            Application.Run(new Form1());  //运行应用程序
        }
    }
}
```

实例 2　控制台程序 Hello World

1. 实例说明

该实例说明如何在控制台中显示一条“Hello World!”的信息。该实例与实例 1 属于两种类型的应用程序，该实例只能在命令行提示符下执行，无法使用表单。

该实例需要修改应用程序的代码，要添加一条在控制台中显示信息的语句：

Console. WriteLine("Hello World!");

Console 是 System 命名空间中的一个类，它提供了 WriteLine()方法，用于在控制台中显示字符串信息并回车换行。

2. 实现步骤

(1) 打开集成环境。运行 Visual Studio. Net 应用程序，则会打开其集成开发环境。

(2) 新建一个项目。选择“文件”|“项目”菜单，则会打开“新建项目”对话框，在“项目类型”列表框中选择“Visual C# 项目”；在“模板”列表框中选择“控制台应用程序”；在“名称”编辑框中为该项目输入一个名字为“Sample _ CHello”；在“位置”下拉列表框中，输入保存项目的目录“D: \ C# 实例”。此时“新建项目”对话框如图 2-1 所示。

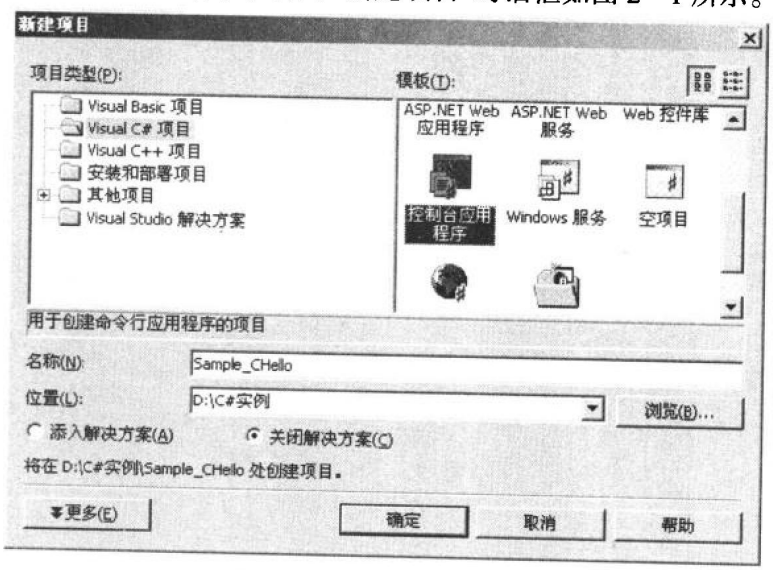

图 2-1　“新建项目”对话框

单击“确定”按钮则会建立一个新的控制台项目。此时 Visual Studio. Net 开发环境的显示，如图 2-2 所示。

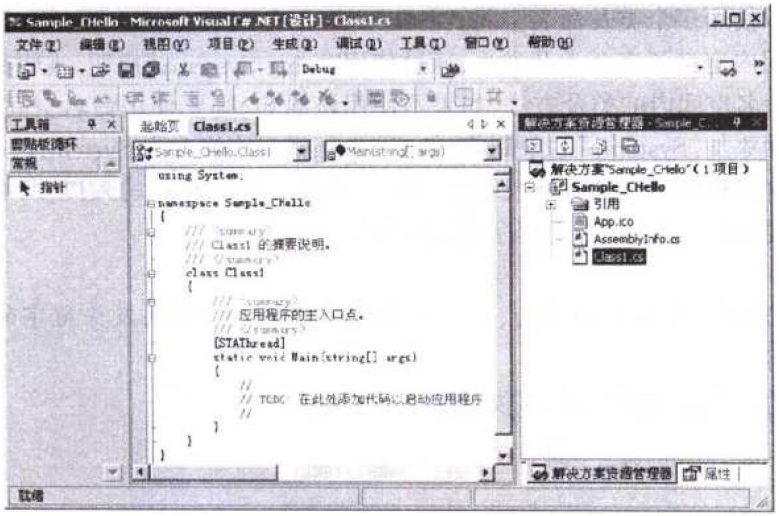

图 2－2　集成环境中新建的控制台应用程序

(3) 添加代码。要在控制台中显示一条信息,应在 Main 方法中添加如下代码:

Console.WriteLine("Hello World!");

(4) 运行应用程序。选择"调试"|"开始执行(不调试)"菜单,直接运行该应用程序。如果应用程序没有任何错误,则会显示最终执行的结果。

(5) 保存并关闭整个项目。

3. 执行结果

应用程序的执行结果如图 2－3 所示,在控制台中显示了一条"Hello World!"的信息。按任意键,可退出控制台并返回到集成开发环境中。

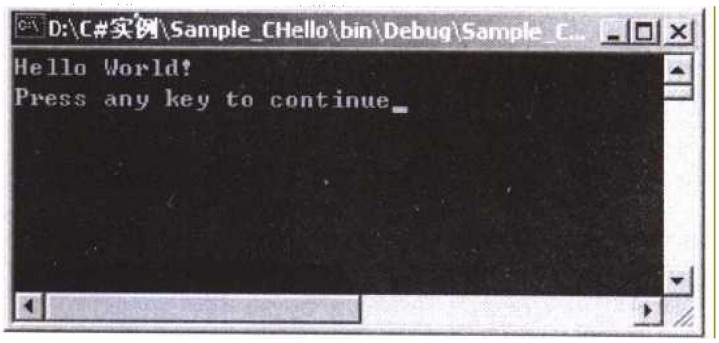

图 2－3　控制台应用程序的执行结果

4. 关键代码及说明

该应用程序的代码比较简单,只在 Main 方法中添加了一条语句。

该程序代码包括 4 个基本元素:命名空间的声明、类、Main 方法和基本语句。一段程

序代码中可以使用多个类及命名空间。

命名空间内定义了多个类及方法，通过使用“using System;”声明，表明程序可以引用该“System”命名空间内的类及方法，如果不使用该语句，则要使用命名空间中的类时，就要在类的前面添加“System.”的限制符，如 Console 就应该修改为 System.Console。

类中包含了程序执行的方法及数据。Main 方法是整个程序的入口，所有应用程序的主代码中都要包括该方法，否则程序就无法执行。

注意：

C# 应用程序是区分大小写的，所以各种命名空间、类、方法及保留字的名称一定要注意其大小写。

该项目的关键代码(Class1.cs)如下：

```
using System; //使用的命名空间

namespace Sample_ CHello //新定义的命名空间
{
    //定义类 Class1
    class Class1
    {
        // 应用程序的主入口点
        [STAThread]
        static void Main(string[] args)
        {
            Console.WriteLine("Hello World!"); //新添加的代码，在控制台中显示信息
        }
    }
}
```

实例 3　控制台程序的交互

1. 实例说明

在控制台应用程序中，经常需要与用户进行交互性操作，即根据用户输入的不同信息，应用程序执行不同的操作。

应用程序的交互操作一般是通过 Read()或 ReadLine()方法读取用户的输入信息来实现的。本实例将根据用户选择的内容不同而显示不同的信息。

2. 实现步骤

(1) 新建立一个控制台项目。

(2) 为项目命名。在“新建项目”对话框中将项目的名称设置为“Sample3 _ alternation”，并保存到相应目录下。

(3) 添加代码。向主方法 Main 中添加适当的代码，使用 do - while 循环语句控制程序进行多次选择，使用 switch - case 语句来根据用户的不同选择执行不同的操作。

(4) 运行程序并保存结果。最好运行整个应用程序，如果出现错误，应及时修改。最后将整个应用程序存盘。

3. 执行结果

应用程序的执行结果如图 3 - 1 所示。

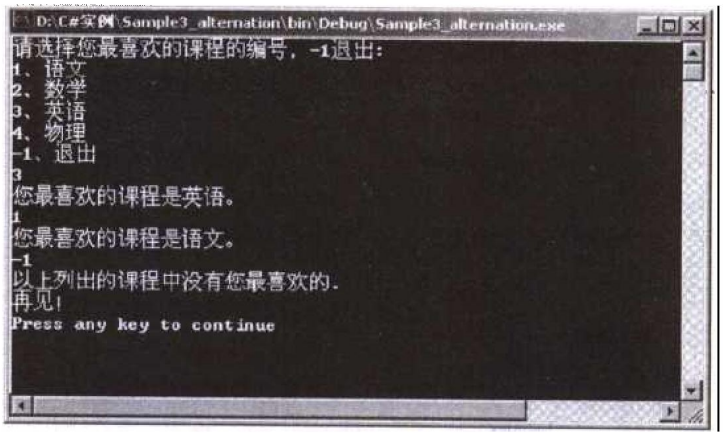

图 3 - 1　应用程序的执行结果

4. 关键代码及说明

该应用程序使用多条 Console.WriteLine()语句来显示输出内容;使用 Console.ReadLine()方法读取用户输入的内容;使用 Int32.Parse()方法将一个字符串类型的变量转换为一个整型变量。

该实例主要代码如下:

```
using System;

namespace Sample3_alternation
{
    class Class1                    //定义一个类
    {
        [STAThread]
        static void Main(string[] args)
        {
            int i;
            string str;
            Console.WriteLine("请选择您最喜欢的课程的编号,-1退出:"); //显示提示信息
            Console.WriteLine("1、语文");
            Console.WriteLine("2、数学");
            Console.WriteLine("3、英语");
            Console.WriteLine("4、物理");
            Console.WriteLine("-1、退出");

            do //控制循环语句
            {
                str=Console.ReadLine(); //读取用户输入信息
                i=Int32.Parse(str); //完成字符串到整型变量的转换

                switch(i) //多重选择
                {
                    case 1:Console.WriteLine("您最喜欢的课程是语文。");
                           break;
                    case 2:Console.WriteLine("您最喜欢的课程是数学。");
                           break;
                    case 3:Console.WriteLine("您最喜欢的课程是英语。");
                           break;
                    case 4:Console.WriteLine("您最喜欢的课程是物理。");
                           break;
                    default:Console.WriteLine("以上列出的课程中没有您最喜欢的.");
                           break;
```

```
            }
        }while(i! = -1);

        Console.WriteLine("再见!"); //退出前显示的信息
    }
}
```