第1章 习题答案

一、选择题

(1)D (2)A (3)C (4)D (5)D

二、简答题

(1) 参考第1.1.3节内容。

(2) C# 集成开发环境中主要有起始页、设计器视图与代码视图、解决方案资源管理器与

类视图、工具箱与服务器资源管理器、属性与动态帮助等。

(3) 命名空间既是Visual Studio.NET提供系统资源的分层组织方式，也是分层组织程序的方式。命名空间有两种：一种是系统命名空间，一种是用户自定义命名空间。

第2章 习题答案

一、选择题

(1) AC (2) B (3) BC (4) D (5) A (6) B (7) ABCD (8) B (9) A (10) D

二、简答题

(1) 答：C#中变量命名规则如下：① 变量名的第一个字符必须是字母或下划线，其余字符必须是字母、数字或下划线。②变量名不能是C#的关键字或库函数名。

(2) 答：值类型变量直接包含其本身的数据，因此每个值类型变量本身就包含有赋给它的值。与值类型不同，引用类型变量本身不包含数据，只是存储对数据的引用，数据保存在内存的其他位置。

(3) 答：结构与枚举都属于值类型。结构类型可以用来处理一组类型不同、内容相关的数据。枚举类型是由用户定义的一组整型符号常量，使用枚举可以将一组相关的有限常量组织在一起，保证变量只能具有预定的值。

第3章 习题答案

一、选择题

(1) A (2) D (3) C (4) D (5)A (6)A (7)B (8)B

二、简答题

(1) 描述一下C#中索引器的实现过程，是否只能根据数字进行索引？

答：可以用任意类型实现索引器。

(2) 解释类的静态成员和非静态成员的区别，说明如何引用静态成员和非静态成员。

答：类的静态成员是类的所有实例公有的，使用类名.成员名进行引用；

类的非静态成员是类的某个实例所有的，使用对象名名.成员名进行引用。

(3)说明String类和String Builder类的区别。

答：String类在进行字符串运算时（如赋值、字符串连接等）会产生一个新的字符串实例，需要为新的字符串实例分配内存空间，相关的系统开销可能会非常昂贵。如果要修改字符串而不创建新的对象，且操作次数非常多，则可以使用StringBuilder 类，例如，当在一个循环中将许多字符串连接在一起时。StringBuilder 类在原有字符串的内存空间上进行操作，使用StringBuilder 类可以提升性能。

第4章 习题答案

一、选择题

(1) D (2) A (3) A (4) A (5)D (6)B (7)D (8)B

二、简答题

(1) C#中的委托是什么？事件是不是一种委托？

答：委托可以把一个方法作为参数代入另一个方法，委托可以理解为指向一个函数的引用。是，事件是一种特殊的委托。

(2) override和重载的区别是什么？

答：重载是方法的名称相同，参数或参数类型不同，进行多次重载以适应不同的需要。

 override 是在派生类进行基类中函数的重写，为了适应派生类的需要。

(3) 概述序列化的原理和作用。

序列化：将对象转换为另一种媒介传输的格式过程。如，序列化一个对象，用Http协议通过internet在客户端和服务器之间传递该对象，在另一端用反序列化从该流中重新得到对象。

第5章 习题答案

**1. 简述为什么要进行异常处理。**

答：在程序运行过程中，经常可能发生各种不可预测的意外情况，如被零除情况或内存不足警告等意外情况，也就是异常。异常处理是为了识别和捕获运行时的错误。当程序引发异常时，如果没有适当的异常处理机制，程序将会终止，并使所有已分配的资源保持不变，这样会导致资源泄露。

**2. 简述逐语句和逐过程两种调试方式的区别。**

答：逐语句和逐过程是两种不同的单步调试的方式。

“逐语句”和“逐过程”的差异仅在于它们处理函数调用的方式不同。这两个命令都指示调试器执行下一行的代码。如果某一行包含函数调用，“逐语句”仅执行调用本身，然后在函数内的第一个代码行处停止。而“逐过程”执行整个函数，然后在函数外的第一行处停止。如果要查看函数调用的内容，请使用“逐语句”。若要避免单步执行函数，请使用“逐过程”。

使用逐语句的方式进行单步调试，可以从“调试”菜单中选择“逐语句”命令，进行调试。使用逐过程的方式进行单步调试，可以从“调试”菜单中选择“逐过程”命令，进行调试。

**3. 编写一个程序，使用两种不同类型数据进行加法运算，并使用异常处理语句捕获由于数据类型转换和算数运算错误而引发的异常。**

答：设计程序界面如下图。



双击button1，在button1的Click事件中添加button1\_Click方法，该方法的代码如下。

try

{

double num1 = double.Parse(textBox1.Text);

double num2 = double.Parse(textBox2.Text);

double result = num1 + num2;

textBox3.Text = result.ToString();

}

catch

{

textBox3.Text = "运算错误，请输入正确的运算数字！";

}

使用如下下表中的Testcase测试程序。

|  |  |
| --- | --- |
| **Testcase1** | **测试程序是否能正确进行数字运算。** |
| 步骤1 | 在第一个运算数中输入“0.001”，在第二个运算数中输入“-0.00001”。 |
| 步骤2 | 单击“=”按钮，计算运算结果。 |
| 步骤3 | 检查程序的运算结果是否等于两个运算数的和“0.00099”。 |
| **Testcase2** | **测试程序是否能处理错误的运算数。** |
| 步骤1 | 在第一个运算数中输入“0.0-01”，在第二个运算数中输入“-0.00001”。 |
| 步骤2 | 单击“=”按钮，计算运算结果。 |
| 步骤3 | 检查程序的运算结果是否为“运算错误，请输入正确的运算数字！”。 |

可以再添加一些更多的Testcase测试程序。

第6章 习题答案

一、选择题

(1) B (2) A (3) B (4) D (5) C

二、简答题

(1) 简述下拉菜单的基本结构。

答：有文字的单个命令称菜单项，顶层菜单项是横着排列的，构成了菜单栏。单击某个顶级菜单项（或菜单项）后弹出的称为一级菜单（或子菜单），它们均包含若干个菜单项。菜单项有的是变灰显示的，表示该菜单项当前是被禁止使用的。有的菜单项的提示文字中有带下划线的字母，该字母称为热键（或访问键），若是顶层菜单，可通过按“ALT+热键”打开该菜单，若是某个子菜单中的一个选项，则在打开子菜单后直接按热键就会执行相应的菜单命令。有的菜单项后面有一个按键或组合键，称快捷键，在不打开菜单的情况下按快捷键，将执行相应的命令。有的菜单项之间有一个灰色的线条，该线条称为分隔线或分隔符。

(2) 对话框按显示方式可分为哪几类，它们之间有何区别？

答：对话框按显示方式分为模态对话框和非模态对话框。它们之间的区别是：

模态对话框就是指当对话框弹出、显示的时候，用户不能单击这个对话框之外的界面区域。用户要访问界面上的其它对象，必须先关闭模态对话框。模态对话框通常用来限制用户必须完成指定的操作任务。例如，Microsoft Word的“字体”对话框。模态对话框使用ShowDialog方法显示。

非模态对话框通常用于显示用户需要经常访问的控件和数据，并且在使用这个对话框的过程中需要访问其它用户界面对象的情况。用户要访问界面上的其它对象，不必关闭非模态对话框。例如Microsoft Word的“查找和替换”对话框。非模态对话框使用Show方法显示。

三、操作题

见文件夹

第7章 习题答案

一、选择题

（1）D（2）C（3）B（4）A（5）B

二、简答题

**（1）什么是GDI+？GDI+有什么功能？**

GDI+（Graphics Device Interface Plus）是微软在Windows 2000以后操作系统中提供的新的图形设备接口，其通过一套部署为托管代码的类来展现，这套类被称为GDI+的“托管类接口”。

GDI+主要提供了绘制二维矢量图形的类，这包括存储图形基元自身信息的类(或结构体)、存储图形基元绘制方式信息的类以及实际进行绘制的类。此外，还提供了文字显示功能，可以使用各种字体、字号和样式来显示文本。

**（2）简述绘图的一般步骤。**

一般来说，GDI+绘图程序包含下面四步：

第一步，创建Graphics对象。

第二步，创建绘图工具并设置相应的属性。

第三步，使用Graphics对象的方法进行图像绘制、文本显示等。

第四步，释放Graphics对象、绘图工具等资源。

**（3）什么是画图工具？画图工具一般包含什么？**

画图工具是System.Drawing命名空间中的类，用来绘制线条和填充区域。

画图工一般包含画笔和画刷。画笔的功能是用来画线，C#中用Pen类来实现画笔。画刷的功能是用来填充图形的内部，C#中用Brush类来实现画刷。

三、编程绘制下面的图形

（1）绘制直线、矩形、椭圆和圆弧，并设置线条颜色分别为红色、绿色、蓝色和黄色，线条宽度分别为1、2、3和4，线条类型分别为实线、点线、划线和划线点。***（答案参见本章例7.2、例7.3、例7.5和例7.6）***

（2）用单色画刷填充矩形和多边形，并设置填充颜色分别为红色和绿色。***（答案参见本章例7.11和例7.12）***

用渐变画刷填充椭圆和饼形，并设置颜色分别从红色水平过渡到蓝色和从绿色垂直过渡到蓝色。***（答案参见本章例7.1）***

（3）绘制三串不同的文字，这三串文字分别为宋体、楷体和隶书，字体颜色分别为红色、绿色和蓝色，字体大小分别为10，20和30，字体样式分别为粗体、斜体和下划线。***（答案参见本章例7.16）***

第8章 习题答案

一、选择题

(1) B (2) C (3) D (4) A (5)D

二、简答题

(1) 简述System.IO模型及其作用。

答：System.IO模型提供了一个面向对象的方法来访问文件系统，提供了很多针对文件、文件夹的操作功能，特别是以流（Stream）的方式对各种数据进行访问，这种访问方式不但灵活，而且可以保证编程接口的统一。

System.IO模型实现包含在System.IO命名空间中，该命名空间包含允许在数据流和文件上进行同步和异步读取及写入、提供基本文件和文件夹操作的各种类，即：System.IO模型是一个文件操作类库，包含的类可用于文件的创建、读/写、复制、移动和删除等操作。

(2) 简述System.IO模型中借助文件流进行文件操作的常用步骤。

答：System.IO模型中，借助文件流进行文件操作的常用步骤是：

a.用File类打开操作系统文件；

b.建立对应的文件流即FileStream对象；

c.用StreamReader/StreamWriter类提供的方法对文件流（文本文件）进行读写或用BinaryReader/BinaryWriter类提供的方法对文件流（二进制文件）进行读写。

三、操作题

见文件夹

第9章 习题答案

**1. 为什么要使用多线程？多线程适用于那些应用？**

答：在运行一个比较耗时的任务时，如果采用采用单线程机制的话，那么主线程会不断地处理该任务，从而无暇处理用户交互的要求。。一般情况下，需要用户交互的软件都必须尽可能快地对用户的活动做出反应，以便提供更好的用户体验。有了线程的概念，操作系统在执行长时间运行的任务时，也能随时响应其他的应用程序。另外，线程允许用户使用一个应用程序（例如：任务管理器）强行终止已经“冻结”的那个应用程序。

多线程适用于以下任务：通过网络进行通信的任务、执行占用长时间的操作的任务、区分具有不同优先级的任务。

**2. 简述线程是如何创建的？怎样设置线程的优先级？**

答：创建线程分两步完成：首先创建线程实例，然后调用Start方法创建线程，使线程可以被CPU调度执行。

* 创建线程实例

调用Thread类的构造方法，就会实例化一个线程实例，但尚未创建实际的线程。此时，线程实例处于Unstarted状态。创建一个线程实例的常用形式为：

Thread t = new Thread(线程方法);

线程方法必须具有如下签名：

void 线程方法(); //无返回值，无参数

或 void 线程方法(Object obj); //无返回值，带有一个Object类型的参数

如果在启动线程时不需要传递参数，该线程方法声明为第一种形式；如果希望在启动线程时传递参数，那么该线程方法声明为第二种形式。

通常情况下，线程方法包含一个循环，循环内部实现用户希望完成的功能。当检测到逻辑表达式为false时，则退出循环并结束线程。线程方法一般定义为如下形式：

void 线程方法()

{

while (逻辑表达式)

{

//执行线程工作

}

}

* 创建线程

调用已经创建的线程对象的Start方法，就会创建一个线程，线程实例状态转换为Running。

创建一个线程的常用形式为：

t.Start (); //无参数的重载形式

或 t.Start (Object obj); //带参数的重载形式

**3. 简述前台线程和后台线程有什么区别？如何将一个线程设置为后台线程。**

答：一个线程或者是后台线程或者是前台线程。后台线程与前台线程类似，区别是后台线程不会防止进程终止。属于某个进程的所有前台线程都终止后，公共语言运行库就会结束该进程，同时所属于该进程的后台线程都会立即停止，无论后台工作是否完成。

使用IsBackground属性，可以设置或判断一个线程是后台线程还是前台线程。如果IsBackground为true，那么此线程是后台线程；否则为前台线程。

第10章 数据库编程基础

一、选择题

（1）B（2）A（3）C（4）B（5）D

二、简答题

**（1）什么是关系数据库？关系数据库包含哪些常用的数据对象？**

关系数据库采用关系模型作为数据的组织方式，它是由许多这样的数据表（Table）所组成。

关系数据库中常用的数据对象有表（Table）、索引（Index）、视图（View）、图表（Diagram）、默认值（Default）、规则（Rule）、触发器（Trigger）和存储过程（Stored Procedure）。详述如下：

① 表（Table）

数据库中的表与我们日常生活中使用的表格类似，它也是由行（Row）和列（Column）组成的。行包括了若干列信息项，一行数据称为一条“记录（Record）”，它表达有一定意义的信息组合。列由同类的信息组成，每列又称为一个“字段（Field）”，每列的标题称为列名。一个数据库表由一条或多条记录组成，没有记录的表称为空表。每个表中通常都有一个主关键字，用于唯一地确定一条记录。

② 索引（Index）

索引是根据指定的数据库表列建立起来的顺序，它提供了快速访问数据的途径，并且可以监督表的数据，使其索引所指向的列中的数据不重复。

③ 视图（View）

视图看上去同表一样，具有一组命名的列和数据项，但它其实是一个虚拟的表，在数据库中并不实际存在。视图是由查询数据库表产生的，它限制了用户能看到和修改的数据。由此可见，视图可以用来控制用户对数据的访问，并能简化数据的显示，即通过视图只显示那些需要的数据信息。

④ 图表（Diagram）

即数据库表之间的关系示意图，利用它可以编辑表与表之间的关系。

⑤ 默认值（Default）

它是在表中创建列或插入数据时，对没有指定其具体值的列或列数据项赋予事先设定好的值。

⑥ 规则（Rule）

它是对数据库表中数据信息的约束命令，并且限定的是表中的列。  
 ⑦ 触发器（Trigger）

它是一个用户定义的SQL事务命令的集合。当对一个表进行插入、更改、删除时，这组命令就会自动执行。

⑧ 存储过程（Stored Procedure）

存储过程是为完成特定的功能而汇集在一起的一组SQL语句，是经编译后存储在数据库中供用户调用的SQL程序。

**（2）什么是SQL语言？SQL语言可以分为几类？**

SQL语言是结构化查询语言（Structured Query Language，SQL），它是标准的关系数据库查询语言，[IBM](http://zh.wikipedia.org/zh-cn/IBM)公司最早使用在其开发的数据库系统中。[1986年](http://zh.wikipedia.org/zh-cn/1986%E5%B9%B4" \o "1986年)，美国国家标准局（[ANSI](http://zh.wikipedia.org/zh-cn/ANSI)）对SQL进行规范后，以此作为[关系式数据库管理系统](http://zh.wikipedia.org/zh-cn/%E5%85%B3%E7%B3%BB%E5%BC%8F%E6%95%B0%E6%8D%AE%E5%BA%93%E7%AE%A1%E7%90%86%E7%B3%BB%E7%BB%9F" \o "关系式数据库管理系统)的标准语言 (ANSI X3. 135-1986)，[1987年](http://zh.wikipedia.org/zh-cn/1987%E5%B9%B4" \o "1987年)得到[国际标准组织](http://zh.wikipedia.org/zh-cn/%E5%9B%BD%E9%99%85%E6%A0%87%E5%87%86%E7%BB%84%E7%BB%87)（ISO）的支持下成为国际[标准](http://zh.wikipedia.org/zh-cn/%E6%A0%87%E5%87%86" \o "标准)。

SQL语言包括数据定义语言（Data Definition Language，DDL），数据查询语言（Data Query Language，DQL），数据操纵语言（Data Manipulation Language，DML）和数据控制语言（Data Control Language, DCL）四大类。数据定义语言DDL用于建立数据库的结构，提供对数据表、索引和触发器等用户自定义对象的创建、修改和删除等操作；数据查询语言DQL用于数据查询而不会对数据本身进行修改；数据操纵语言DML用于访问数据库中的数据，包括插入、更新和删除等操作；数据控制语言DCL用于处理用户对某个特定对象的允许。

**（3）简述ADO.NET模型的组成。**

ADO.NET是Microsoft的新一代数据处理技术，是ADO组件的后继者，具有与ADO相似的编程方式。ADO.NET数据提供程序的四个核心对象：数据连接（Connection）、数据命令（Command）、数据阅读器（DataReader）和数据适配器（DataAdapter）。其中数据连接对象用于建立到指定资源的连接；数据命令对象用于执行各种查询命令；数据阅读器对象主要用于从数据源获取一个只读数据流；数据适配器对象用于填充一个数据集，解析数据源的更新等操作。

**（4）什么是数据绑定？利用数据绑定技术浏览干部基本信息表TB\_Commoninf中所有记录。**

数据绑定就是把控件链接到数据源的过程，即把已经打开的数据集中某个或某些字段绑定到控件的某些属性上的一种技术，比如可以把已经打开数据的某个或者某些字段绑定到Text、ListBox或ComBox等控件能够显示数据的属性上面。根据所绑定控件的不同，数据绑定可以分为两种类型：简单数据绑定和复杂数据绑定。当对控件完成数据绑定后，其显示字段的内容将随着数据记录指针的变化而变化。

如何利用数据绑定技术浏览干部基本信息表TB\_Commoninf中所有记录，参见本章例10.5和例10.6。

**三、编程实现ADO.NET连接SQL Server数据库，并实现以下功能：**

**（1）查询干部基本信息表TB\_Commoninf中所有男性干部的编号、姓名、民族、身份证号、职务和籍贯，并按照编号升序排列。**

核心代码如下：

private void Form1\_Load(object sender, EventArgs e)

{ // 利用SqlConnection对象，连接数据库

string mystr;

mystr = "Data Source =ZGC-20130515AMW\\SQLEXPRESS; Initial Catalog = gbxxdb;User Id = sa;Password = "; //OK

SqlConnection myconn = new SqlConnection();

myconn.ConnectionString = mystr;

myconn.Open();

if (myconn.State == ConnectionState.Open)

MessageBox.Show(this, "恭喜，成功连接!", "连接SQL Server数据库测试程序", MessageBoxButtons.OKCancel);

else

MessageBox.Show(this, "连接失败，程序还有问题!", "连接SQL Server数据库测试程序", MessageBoxButtons.OKCancel);

// 利用SqlDataReader对象，查询TB\_Commoninf表中的内容，返回表中前六个字段的所有记录

string sql\_str;

sql\_str = "select cno, name, sex, nation, CID, position, native from TB\_Commoninf where sex='男' order by cno asc";

SqlCommand myCommand = new SqlCommand();

myCommand.CommandText = sql\_str;

myCommand.Connection = myconn;

SqlDataReader myDataReader = myCommand.ExecuteReader();

listBox1.Items.Add("编号\t姓名\t性别\t民族\t身份证号\t\t职务\t籍贯");

listBox1.Items.Add("================================================

=====================");

while (myDataReader.Read())

{ listBox1.Items.Add(String.Format("{0}\t{1}\t{2}\t{3}\t{4}\t{5}\t{6}",

myDataReader[0].ToString().TrimEnd(), myDataReader[1].ToString().TrimEnd(),

myDataReader[2].ToString().TrimEnd(), myDataReader[3].ToString().TrimEnd(),

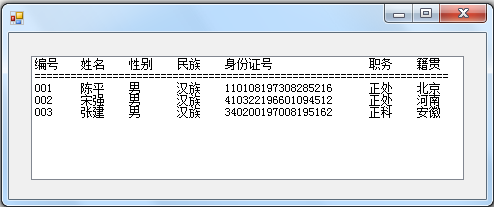
myDataReader[4].ToString().TrimEnd(), myDataReader[5].ToString().TrimEnd(),

myDataReader[6].ToString().TrimEnd())); }

myconn.Close();

}

结果如下图所示：



**（2）查询每名干部的编号、姓名及其所掌握的外语语种和熟练程度。**

核心代码如下：

将第（1）小题16行之后的代码替换为下面的代码：

// 利用SqlDataReader对象，查询TB\_Commoninf表中的干部编号、干部姓名和

TB\_FamiliarForeign表中外语语种和熟练程度。

string sql\_str;

sql\_str = "select TB\_Commoninf.cno, TB\_Commoninf.name, TB\_FamiliarForeign.foreignKind, TB\_FamiliarForeign.level from TB\_Commoninf, TB\_FamiliarForeign where TB\_Commoninf.cno=TB\_FamiliarForeign.cno";

SqlCommand myCommand = new SqlCommand();

myCommand.CommandText = sql\_str;

myCommand.Connection = myconn;

SqlDataReader myDataReader = myCommand.ExecuteReader();

listBox1.Items.Add("编号\t姓名\t外语语种\t熟练程度");

listBox1.Items.Add("=========================================");

while (myDataReader.Read())

{listBox1.Items.Add(String.Format("{0}\t{1}\t{2}\t\t{3}",

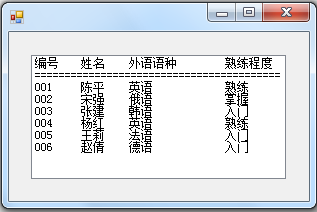
myDataReader[0].ToString().TrimEnd(),myDataReader[1].ToString().TrimEnd(),

myDataReader[2].ToString().TrimEnd(), myDataReader[3].ToString().TrimEnd()));

}

myconn.Close();

其余代码不变，运行结果如下图所示：



**（3）将干部基本信息表TB\_Commoninf中的副处级干部的干部职务更改为正处级。**

核心代码如下：

将第（1）小题16行之后的代码替换为下面的代码：

// 修改TB\_Commoninf表中的内容

string sql\_str;

sql\_str = "update TB\_Commoninf set position='正处' where position='副处'";

SqlCommand myCommand = new SqlCommand();

myCommand.CommandText = sql\_str;

myCommand.Connection = myconn;

myCommand.ExecuteNonQuery();

myconn.Close();

**（4）删除干部基本信息表TB\_Commoninf中籍贯是安徽的干部。**

核心代码如下：

将第（1）小题16行之后的代码替换为下面的代码：

// 修改TB\_Commoninf表中的内容

string sql\_str;

sql\_str = "delete from TB\_Commoninf where native='安徽'";

SqlCommand myCommand = new SqlCommand();

myCommand.CommandText = sql\_str;

myCommand.Connection = myconn;

myCommand.ExecuteNonQuery();

myconn.Close();

第11章 组件技术

一、选择题

(1) C (2) B

二、简答题

(1) 组件是指实现System.ComponentModel.IComponent接口的类，或从实现IComponent的类中直接或间接派生的类。在软件开发中，组件是指可重复使用并且可以和其他对象进行交互的对象。控件是提供或实现用户界面功能的组件，只有提供操作或显示界面的组件才称为控件。

(2) 在工具箱中新建一个选项卡，用鼠标右键单击该选项卡，从快捷菜单中选择“选择项”→“.NET Framework组件”→“浏览”命令，找到所要添加的控件点击“确定”即可。

(3) 略

(4) 略

三、操作题

略

第12章 干部信息管理系统

一、选择题

(1) D (2) B

二、简答题

(1) 一个软件项目开发的主要阶段为：软件需求分析阶段、软件设计测试阶段、软件使用和维护阶段。而软件需求分析是其他几个阶段的基础。如果在软件需求分析阶段出现失误，就很有可能导致整个项目的返工进而使整个项目无法按时完成达到预期结果，甚至最终开发不出用户所需要的产品。这些后果会直接导致损害开发者经济利益。所以，软件开发者在整个项目开发过程中的初级阶段，首先要从分析软件应该为用户解决的业务问题，通过对用户业务的分析，规划出用户所要求的软件产品。

(2) 公共的代码，可以方便开发人员调用，减少代码重复，在后期修改过程中也相对比较灵活。

(3) 首先用connection和command这些对象打开数据库，然后打开连接，进行相关的数据操纵。

(4) ReadIni类的作用是读写配置文件信息。

三、操作题

略

**第十四章 参考答案**

第13章 快餐POS系统

一、简答题

1. C#在连接SQL Server 2000时需要引用“System.Data.SqlClient”命名空间。
2. C#与SQL Server 2000建立连接时，一般需一下几个步骤：

(1)正确的连接字符串conStr，连接字符串格式为："server=服务器IP或计算机名（如果是本地可以就写个点 ．就可以了）;uid=数据库帐号;pwd=数据库密码;database=你要连接的数据库名字"。

(2)新建一个“SqlConnection”类型的连接对象conn，并用连接字符串初始化该对象。

(3)然后调用连接对象conn的Open()方法与数据库建立连接。

1. 快餐POS系统在与SQL Server 2000建立连接和数据操作过程中都用到了Sqlconnection、Sqlcommand、SqlDateReader。

(1)Sqlconnection用来与数据库建立连接.

(2) Sqlcommand用来执行Sql语句。

(3) SqlDateReader用来读取Sqlcommand的执行结果。

1. 见本章的第13.4.3小节。

二、操作题

略。

第14章 部署Windows应用程序

一、简答题

(1) 对于开发的应用程序，通常都需要经过打包部署后才可以交付给用户使用，用户得到应用程序后，通过交互式的安装部署向导将应用程序安装到本地环境中，然后才能正常使用其提供的服务。所以在程序开发完成之后要对应用程序进行打包。

(2) 一个应用程序可以按两种方式进行编译：Debug与Release。Debug模式的优点是便于调试，生成的EXE文件中包含许多调试信息，因而尺寸较大，运行速度慢；而Release模式删除了这些调试信息，运行速度较快。

(3) 首先打开“解决方案管理器”，右击解决方案名称，选择“属性”，在打开的属性页中选择“系统必备”，然后在“系统必备”窗体中选择所要一并打包到程序中的组件，进而在 “指定系统必备安装组件的位置”中选中“从与我的应用程序相同的位置下载系统必备组件”，这样就可以将系统必备组件一并打包到安装程序中。用户在安装程序的时候，如果机器上之前没有安装程序运行所需的组件环境，此时安装程序会将所必备的组件安装到客户机上以便程序可以正常运行。

二、操作题

略