第一章

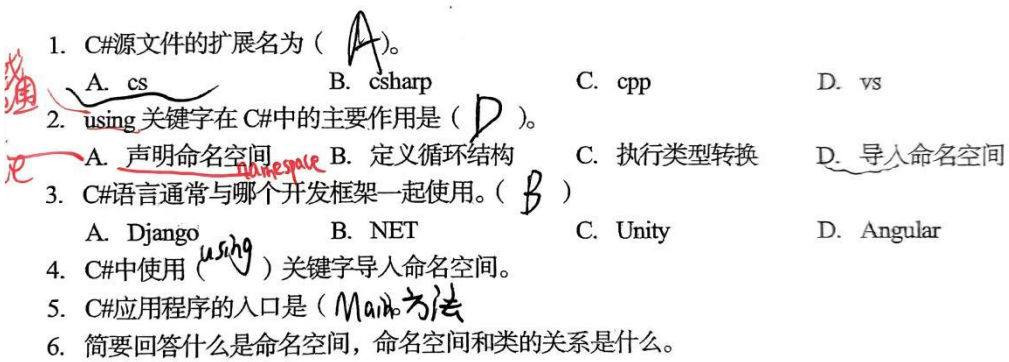
1. .NET Framework 和.NET Framework Core 区别。
.NET 支持跨平台。
.NET Framework 是一个完整的库，**它不能跨平台**，只能在 windows 系统上运行；
.NET Framework Core 是核心库，**它可以跨平台**。

2. 三种项目类型。
控制台应用程序；windows 窗体应用程序（form 窗体）；web 类型程序（web 应用）。
一个解决方案下可以有多个项目。

3. 命名空间和类
类如果较多是通过命名空间来**组织**的。命名空间是**逻辑意义**上的概念。
引用(导入)命名空间：通过 using 关键字。
定义(声明)命名空间：namespace
Main 方法是程序的入口点。
代码的注释：//　　/* */　　///
快速键入 C#代码段：连续按两次 Tab 键。

4. 断点调试
双击设置断点。
逐过程 F10：系统把一个过程当做一条语句，**不转入过程内部**。而逐语句 F11 会进入函数内部。

5. C/S 模式和 B/S 模式
C/S 模式：客户端/服务端。Eg：QQ。如果应用功能复杂，选择 CS
B/S 模式：浏览器/服务器。特殊的 CS。门户类网站，不复杂，选择 BS


1. C#源文件的扩展名为（ A ）。
A. cs　　B. csharp　　C. cpp　　D. vs
2. using 关键字在 C#中的主要作用是（ D ）。
A. 声明命名空间　　B. 定义循环结构　　C. 执行类型转换　　D. 导入命名空间
3. C#语言通常与哪个开发框架一起使用。（ B ）
A. Django　　B. NET　　C. Unity　　D. Angular
4. C#中使用（ using ）关键字导入命名空间。
5. C#应用程序的入口是（ Main 方法 ）
6. 简要回答什么是命名空间，命名空间和类的关系是什么。

第二章

1. 显示：Show()和 ShowDialog()
Show()**将窗体显示出来后立即返回**，接着执行 Show 方法后面的代码，而不是等待窗体关闭。
ShowDialog()**仅在窗体关闭后才能执行后面的代码，“模式”窗体**。

2. 其他方法
Hide()隐藏
Close()关闭
Application.Run(new Form1()**);** 启动应用程序消息循环，显示 Form1 窗体。

1. 利用 Label 控件的（A）属性可设置文字的字体大小。
   A. FontSize　　B. Foreground　　C. Font　　D. FontFamily
2. 在 WinForms 应用程序中，如果复选框控件的 Checked 属性值为 False，表示该复选框（B）。
   A. 被选中　　B. 未被选中　　C. 显示信息　　D. 不显示信息
3. System.Console 类提供了一个（Read）方法从标准输入流依次读取字符。
4. 在 Windows 窗体应用程序的 Main 方法中，（Application.Run()）方法用于在当前线程上启动应用程序消息循环，并显示窗体。
5. Windows 窗体编程模型一般用于 C/S 模式的（客户端）开发。
6. 简要回答窗体的显示方式有哪些，并说明其特点。

第六章

1. 定义端口是为了**区分进程**。取值范围 0 到 65535，不能复用。
2. IP 有两种 v4 和 v6，Parse 方法可以检查 IP 合法性。AddressFamily 属性可以判断地址是 IPv4 还是 v6
3. 域名解析：
域名解析：利用互联网 DNS(域名系统）将域名转换位对应的 IP 地址的过程。

名　称	说　明
GetHostAddresses	返回指定主机的Internet协议IP地址与该方法对应的还有异步方法
GetHostEntry	将主机名或IP地址解析为IPHostEntry实例
GetHostName	获取本地计算机的主机名

4. 进程和线程。
进程：正在运行的程序。**进程是资源调度和分配的基本单位。**P134
线程：**线程是 CPU 调度和分配的基本单位**
主线程：自动运行，不需要建立。属于前台线程。

前台线程：会影响进程的终止。
后台线程：不会。
前台线程都终止后，后台线程都终止。
5. **Lock 语句**的作用：**实现线程同步。锁定代码块。被锁定的代码块称为临界区。**
6. Invoke 方法是同步调用。将操作交给控件的创建线程来处理。
7. 编码用什么，解码就用什么，防止出现乱码。
编码：字符序列通过某种编码格式转换为字节序列的过程。
解码：将字节序列转换为字符序列。

8. 序列化：将对象状态转换为可存储或传输的格式的过程。

怎么声明类和成员可以被序列化？
类之前必须有 [**DataContract**] 特性
字段（类中成员）之前必须有 [**DataMember**] 特性

9. 文件拷贝。

```
string sourcePath = @"C:\source\file.txt"; // 源文件路径
string destinationPath = @"C:\destination\file.txt"; // 目标文件路径
try
{
    // 拷贝文件
    File.Copy(sourcePath, destinationPath, true); // true 表示如果目标文件已存在，则覆盖
}
catch (IOException ioEx)
{
    Console.WriteLine("文件拷贝出错：" + ioEx.Message);
}
```

方法二：

```
string path = @"d:\EM1.txt";
string path2 = @"d:\云轩\EM2.txt";
FileInfo fi1 = new FileInfo(path);
FileInfo fi2 = new FileInfo(path2);

fi1.CopyTo(path2);
```

10. StreamReader 类和 StreamWriter 类的用法。
如果数据来源是文件流、内存流或者网络流，可以利用 StreamReader 和 StreamWriter 对象的构造函数得到读写流。

```
NetworkStream networkStream = client.GetStream( );
StreamReader sr = new StreamReader (networkStream);
StreamWriter sw = new StreamWriter (networkStream);
```

如果需要处理的是文件流，还可以直接利用文件路径创建 StreamWriter 对象。

```
StreamWriter sw= new StreamWriter ("C:\\file1.txt");
```

第七章

1. TCP 和 UDP 比较。
TCP 是一种面向连接的传输层协议。一对一；字节流收发数据；**数据无消息边界。**
UDP 是一种无连接的运输层协议。**有消息边界。**
侧重更可靠服务的应用选择 TCP；侧重传输速度和多点传输的应用选择 UDP；
TCP 消息的无边界问题：**TCP 不能保证单次发送的消息被单次接收。**

2. TCP client 搭配 TCP listener 结合来实现

3. 简单的消息发送，简单的文件传递。

TCP client 和 listener 做一个简单的聊天程序。并且能够在这里面去做文件的发送。文件的发送知道文件发送的思路是什么，在文件发送的过程中怎么来去解决它的 TCP 消息通信无边界的问题.

4. 异步方法。ASYNC 和 await

如果用异步方法来实现，必须用 async 和 Task 共同表示没有返回值的任务，用 async 和 Task<TResult>共同表示返回值为 TResult 类型的任务。

```
没有返回值的异步方法 1 定义
public async Task Method1Async()
{
   await Task.Delay(500);
   Console.WriteLine("Thread ID: {0}", Thread.CurrentThread.ManagedThreadId);
}
返回 int 的异步方法 2 定义
public async Task<int> Method2Async()
{
   var range = Enumerable.Range(1, 1000);
   int n = range.Sum();
   await Task.Delay(0);
   Console.WriteLine("Thread ID: {0}", Thread.CurrentThread.ManagedThreadId);
   return n;
}
```

await 运算符和同步编程的最大区别是:异步等待任务完成时，既不会继续执行其后面的代码，也不会影响用户对 UI 界面的操作。

```
private async void ButtonOK_Click(...)
{
   Task a = Method1Async(); //创建任务a
   await a; //等待任务a完成，任务完成前不会执行该语句后面的代码，但也不会影响界面操作
   Task<int> b = Method2Async(); //创建任务b
   int x = await b; //等待任务b完成，任务完成前不会执行该语句后面的代码，也不会影响界面操作
}
private async Task Method1Async() {.....}
private async Task<int> Method2Async{.....}
```

给普通方法加上 Task.Run 外壳就可以执行异步任务

```
await Task.Run(() => Method1());
```

仅包含 async 和 await 关键字的异步方法与用 TaskRun 调用的异步方法有何不同？

不同：使用了 async 和 await 关键字的可以不阻塞 UI 进行操作

5. 简要回答仅包含 async 和 await 关键字的异步方法与用 Task.Run 调用的异步方法有哪些不同。

6. 利用 TCP 进行通信时，发送方先发送字符串“1234”，然后发送字符串“abcd”，接收方不可能出现的情况是（ ）。

A. 第一次接收 1234，第二次接收 abcd

B. 第一次接收 123，第二次接收 4abcd

C. 第一次接收 1234ab，第二次接收 cd

D. 第一次接收 123a，第二次接收 4bcd

7. TcpListener 类提供的（ ）方法用于异步接收客户端连接请求。

A. AcceptTcpClient()　　B. AcceptAsync()

C. ConnectAsync()　　D. AcceptTcpClientAsync()

第八章

UdpClient 的用法，利用 UDP 同步、异步进行收发数据。广播。

P174　P178

1. UDP 和 TCP 的主要区别有哪些?
2. 什么是广播？什么是组播？两者有什么区别?
3. 简要回答利用 UdpClient 加入组播组和退出组播组的步骤。
4. 下列有关 UDP 的说法不正确的是（ ）。

A. UDP 是面向数据报的无连接协议

B. UDP 可以实现一对一和一对多的传输

C. UDP 传送速度比 TCP 快

D. UDP 实现时需要考虑消息边界问题，实现起来要比 TCP 困难

5. 有关广播和组播的说法不正确的是（ ）。

A. 广播和组播都能实现一对多的通信需要

B. 广播可以向子网内的所有计算机发送消息

C. 组播组都是永久的，加入组播组的计算机可以收到发到该组播组的任何消息

D. 组播使用的 224.0.0.0 到 239.255.255.255 的 D 类 IP 地址进行广播

6. TCP 和 UDP 均是传输层的协议。当实现向多个用户同时发送消息时（比如即时新闻发布、网络会议等应用），应该选择（ ）比较合适。若需要实现可靠性要求比较高的应用时，选择（ ）比较合适。

7. 编写 UDP 应用程序时，对基础套接字进行封装的类是（ ），使用该类提供的（ ）方法可以加入到一个组播组，使用（ ）方法能够退出一个组播组。