



C#程序设计

第9章 文件

杨琦
西安交通大学
计算机教学实验中心
<http://ctec.xjtu.edu.cn>

教学目标

- C#文件系统
- 使用 File 类创建文件，复制文件
- 使用 Directory 类实现文件夹的浏览和创建
- 使用 FileStream 类访问文件
- 读写文本文件
 - 掌握StreamReader
 - 掌握StreamWriter
- 读写二进制文件
 - 掌握BinaryReader
 - 掌握BinaryWriter

9-1 C#的文件系统

- 使用程序访问文件是十分重要的。
- 对于计算机而言，文件往往保存在磁盘之类的外部设备中，对文件的操作常常涉及对相关文件夹的操作，操作文件和操作文件夹是程序访问文件的两个主要方面。
- 一个文件 是一个存储在磁盘中带有指定名称和目录路径的数据集合。
- 文件有很多分类的标准，按照文件的访问方式可将文件分为顺序文件和随机文件两种。

9-1 C#的文件系统

- ①顺序文件
- 顺序存取的文件称为顺序文件。顺序文件没有内部逻辑结构。
- ②随机文件
- 随机存取的文件称为随机文件，它以记录格式保存数据。
- 文件由多个记录组成，每个记录都有相同的大小和格式。
- 只要给出记录号，就可以迅速地找到指定的记录进行读写操作。

9-1 C#的文件系统

- 按照文件的存储方式，可以将文件分为**二进制文件**和**ASCII码文件**两种。
- ①二进制文件
- 数据均以二进制方式存储，存储的基本单位是字节。
- 二进制文件能够任意读写所需要的字节，可以节省存储空间和避免编码转换。
- ② ASCII码文本文件
- ASCII码文本文件中的数据以字符形式表示，因而便于按字符形式逐个处理，也便于打印输出字符。
- ASCII文本文件一般占用存储空间较多，且存在编码转换的运行开销。

9-1 C#的文件系统

- 当打开文件进行读写时，它变成一个流。
- 流是通过通信路径传递的字节序列。
- 有两个主要的流：**输入流**和**输出流**。输入流用于从文件读取数据（读操作），输出流用于向文件写入数据（写操作）。
- .NET框架结构在**System.IO**名称空间中提供了多种类型，用于进行数据文件和数据流的读写操作。这些操作可以同步进行，也可以异步进行。

9-1 C#的文件系统

- .NET用流（Stream）来表示数据的传输操作。
- 流是字节序列的抽象概念，例如文件、输入/输出设备、内部进程通信管道或者TCP/IP 套接字。
- 将数据从内存传输到某个载体或设备中的流称为输出流；
- 将数据从设备或载体传入内存的流称为输入流。
- 流涉及三种基本操作：
 - (1) 读取：从流到数据结构（如字节数组）的数据传输。
 - (2) 写入：从数据结构到流的数据传输。
 - (3) 查找：对流内的当前位置进行查询和修改。

9-1 C#的文件系统

- 对文件的访问可以借助**文件流**来实现，对文件读写时，将文件处理成**字符流**或**二进制流**，对文件的读写其实就是读取字符流或二进制流。
- 在.NET框架中，对文件的读写操作借助于I/O数据的通用模型System.IO，对所有的数据源使用相同的代码进行操作。

I/O 类	描述
Directory	有助于操作目录结构。
DirectoryInfo	用于对目录执行操作。
DriveInfo	提供驱动器的信息。
File	有助于处理文件。
FileInfo	用于对文件执行操作。
FileStream	用于文件中任何位置的读写。
MemoryStream	用于随机访问存储在内存中的数据流。
Path	对路径信息执行操作。
StreamReader	用于从字节流中读取字符。
StreamWriter	用于向一个流中写入字符。
StringReader	用于读取字符串缓冲区。
StringWriter	用于写入字符串缓冲区。
BinaryReader	从二进制流读取原始数据。
BinaryWriter	以二进制格式写入原始数据。
BufferedStream	字节流的临时存储。

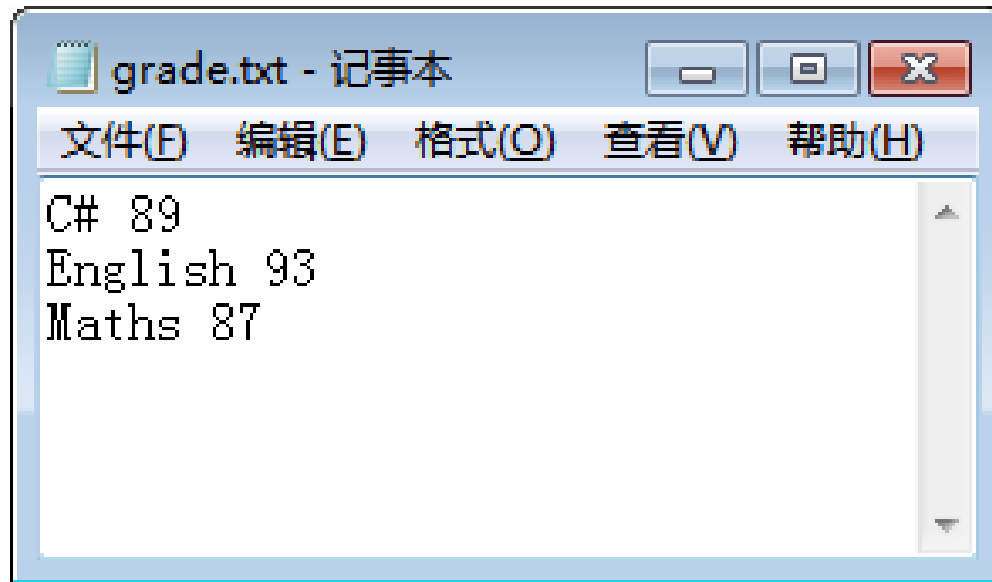
9-2 文本文件的读写

- **StreamReader**类
- **Read**方法：用于读取输入流中的下一个字符，并使当前流的位置提升一个字符。
- 其方法原型为：`public override int Read();`
- **ReadLine**方法：从当前流中读取一行字符并将数据作为字符串返回。
- 其方法原型为：`public override string ReadLine();`
- **ReadToEnd()**

9-2 文本文件的读写

- **StreamWriter**类
-
- **Write**方法： 可用于将字符、字符数组、字符串等写入流，
- **WriteLine**方法： WriteLine方法用于将后跟行结束符的字符、字符数组、字符串等写入文本流。

【例9-1】创建一个名为“grade.txt”的文件



【例9-1】 创建一个名为 “grade.txt”的文件

```
1. using System;
2. using System.IO;
3. class My
4. {
5.     static int Main()
6.     {
7.         StreamWriter sw = new StreamWriter("grade.txt");
8.         sw.WriteLine("C# {0} ", 89);
9.         sw.WriteLine("English {0} ", 93);
10.        sw.WriteLine("Maths {0} ", 87);
11.        sw.Close();
12.        return 0;
13.    }
14. }
```

【例9-2】读取 “grade”文件

- 读取例9-1创建的 “grade”文件，并将文件内容显示在屏幕上。
- 输入和输出
- C# 89
- English 93
- Maths 87

【例9-2】读取 “grade”文件

```
1. class My{
2.     public static void Main()  {
3.         try    {
4.             StreamReader sr = new StreamReader("grade.txt");
5.             string line;
6.             line = sr.ReadToEnd();
7.             Console.WriteLine(line);
8.             sr.Close();
9.         }
10.        catch (Exception e)    {
11.            Console.WriteLine(e.Message);
12.        }
13.    }
14. }
```

【例9-3】 利用StreamReader类

- 利用StreamReader类显示二进制文件的内容，如果是可打印字符，则按原样输出；如果是不可打印字符，则输出“.”。
- 请输入显示文件的名称： A01.exe
- MZ.....@.....!..L!Th
- is program cannot be run in DOS
mode....\$.....PE..L...(U.Q.....
- ..F.....-<Module>.A01.exe.Example.mscorl
- ib.System.Object.Main..ctor.System.Reflection.AssemblyTitleAttribute.AssemblyDes

【例9-3】利用StreamReader类

```
1. using System;
2. using System.IO;
3. class My
4. {
5.     static int Main()
6.     {
7.         string fname, strContent="";
8.         char ch;
9.         Console.Write("请输入显示文件的名称: ");
10.        fname = Console.ReadLine();
```

【例9-3】利用StreamReader类

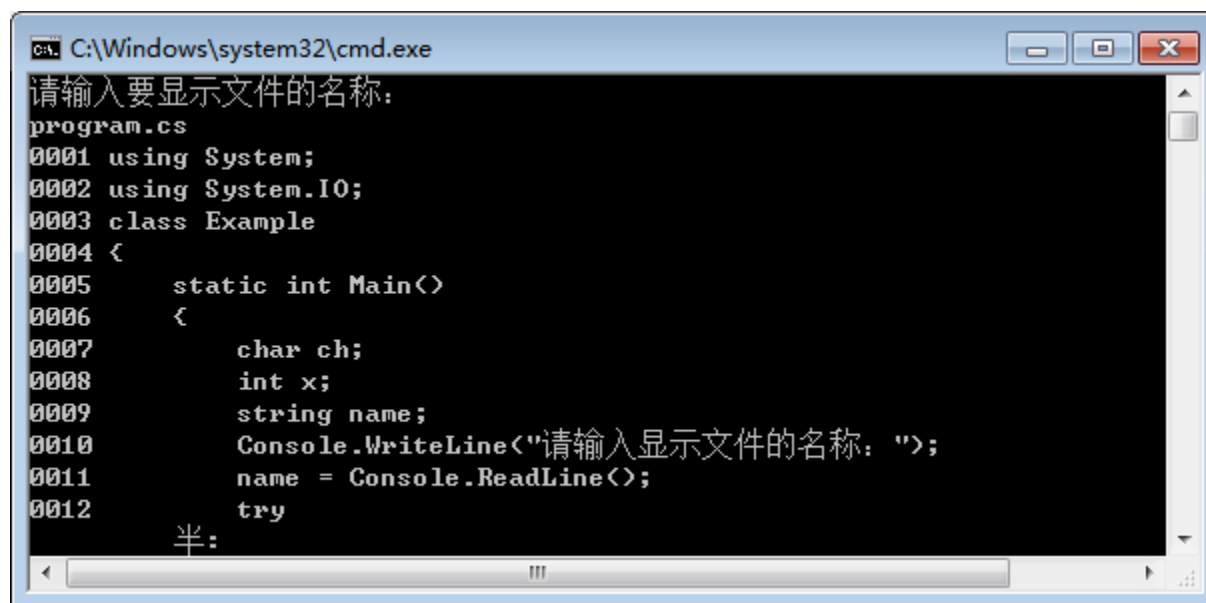
```
1.      try
2.      {
3.          StreamReader sr = new StreamReader(fname);
4.          strContent = sr.ReadToEnd();
5.          for (int i = 0; i < strContent.Length; i++)
6.          {
7.              ch = strContent[i];
8.              if (ch >= 0x20 && ch <= 0x7e)
9.                  Console.Write(ch);
10.             else
11.                 Console.Write('.');
12.         }
```

【例9-3】利用StreamReader类

```
1.         Console.WriteLine();
2.         sr.Close();
3.     }
4.     catch (Exception e)
5.     {
6.         Console.WriteLine(e.Message);
7.     }
8.     return 0;
9. }
10. }
```

【例9-4】利用StreamWriter类

- 从命令行读入一个C#源文件，每一行加上行号后在屏幕上显示出来。



```
C:\Windows\system32\cmd.exe
请输入要显示文件的名称:
program.cs
0001 using System;
0002 using System.IO;
0003 class Example
0004 {
0005     static int Main()
0006     {
0007         char ch;
0008         int x;
0009         string name;
0010         Console.WriteLine("请输入显示文件的名称: ");
0011         name = Console.ReadLine();
0012         try
            半:
```

符号	十进制	含义
\n	10	换行
\r	13	回车

【例9-4】利用StreamWriter类

- 从命令行读入一个C#源文件，每一行加上行号后在屏幕上显示出来。
 - 请输入要显示文件的名称：
 - program.cs
 - 0001 using System;
 - 0002 using System.IO;
 - 0003 class Example
 - 0004 {
 - 0005 static int Main()
 - 0006 {
 - 0007 char ch;
 - 0008 int x;

【例9-4】利用StreamWriter类

```
1. using System;  
2. using System.IO;  
3. class My  
4. {  
5.     static int Main()  
6.     {  
7.         string strName;  
8.         string strDoc;  
9.         int i, ln = 1;  
10.        char ch;
```

【例9-4】利用StreamWriter类

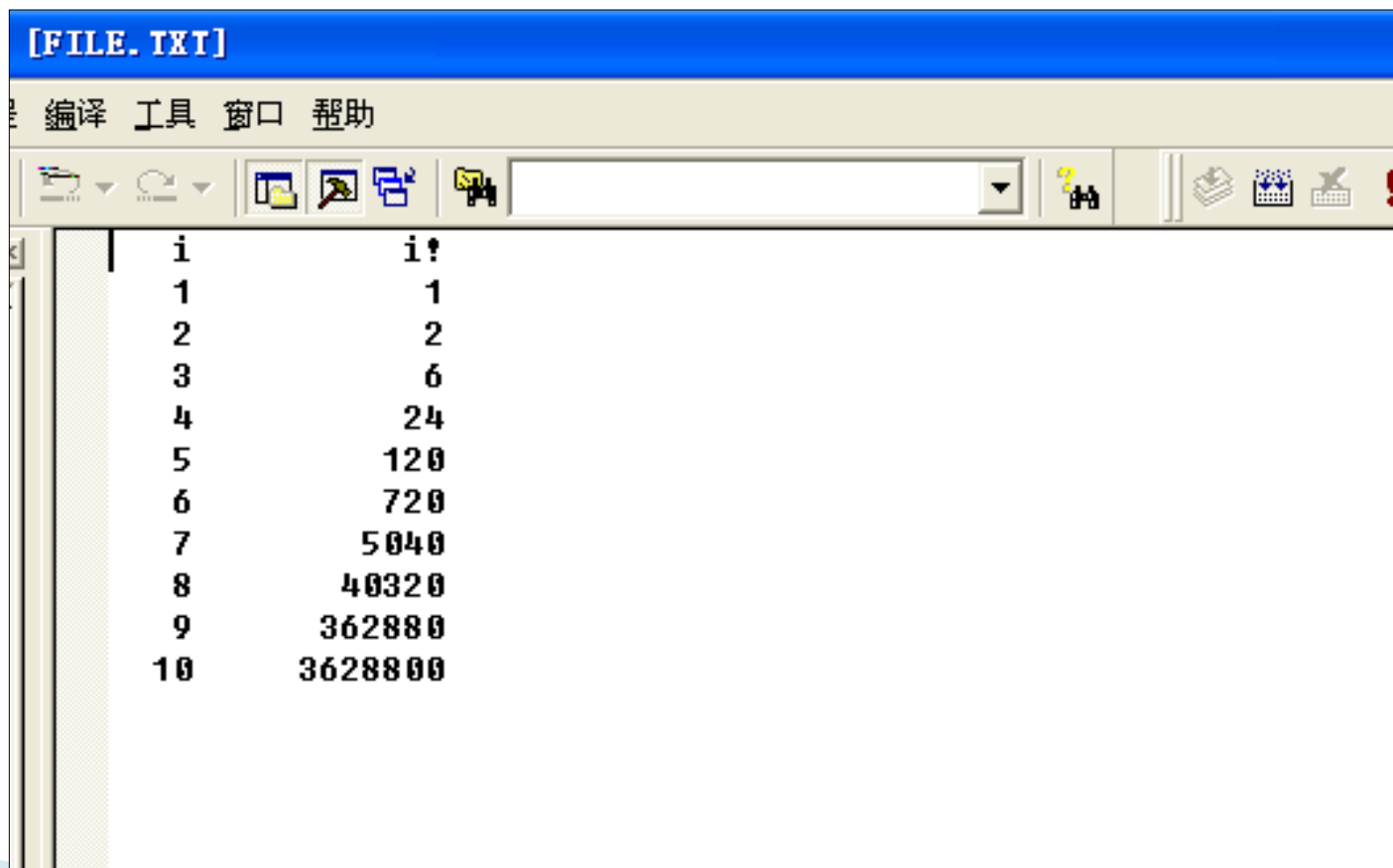
```
1.      try
2.      {
3.          Console.WriteLine("请输入要显示文件的名称: ");
4.          strName = Console.ReadLine();
5.          StreamReader sr = new StreamReader(strName);
6.          strDoc = sr.ReadToEnd();
7.          sr.Close();
8.          Console.Write("{0:D4} ", ln++);
9.          for (i = 0; i < strDoc.Length; i++)
10.         {
11.             ch = strDoc[i];
12.             Console.Write(ch);
```

【例9-4】利用StreamWriter类

```
1.         if (ch == '\n')
2.         {
3.             Console.Write("{0:D4} ", ln++);
4.         }
5.     }
6. }
7. catch (Exception e) {
8.     Console.WriteLine(e.Message);
9. }
10. return 0;
11. }
12. }
```


【例9-5】利用StreamWriter类

- 例：将1-10的阶乘值输出到文件FILE.TXT中



```
[FILE.TXT]
编译 工具 窗口 帮助
1 1
2 2
3 6
4 24
5 120
6 720
7 5040
8 40320
9 362880
10 3628800
```

【例9-5】利用StreamWriter类

```
1. using System;
2. using System.IO;
3. class My
4. {
5.     static int Main()
6.     {
7.         string strName = "File.txt";
8.         try
9.         {
10.             Console.WriteLine("请输入要写入文件的名称: ");
11.             strName = Console.ReadLine();
```

【例9-5】利用StreamWriter类

```
1.      StreamWriter sw = new StreamWriter(strName);
2.      int i, u = 1;
3.      sw.WriteLine("\ti\ti!");
4.      for (i = 1; i <= 10; i++)
5.      {
6.          u = u * i;
7.          sw.WriteLine(" {0,8} {1,8}", i, u);
8.      }
9.      sw.Close();
10.     }
```

【例9-5】利用StreamWriter类

```
1.      catch (Exception e)
2.      {
3.          Console.WriteLine(e.Message);
4.      }
5.      return 0;
6.  }
7. }
```

9.3 文件和目录的管理

9.3 文件和目录的管理

文件管理 File类

静态方法	描述
File.Create	在指定路径中创建文件
File.Copy	将现有文件复制到新文件
File.Delete	删除指定的文件
File.Move	将指定文件移到新位置，并提供指定新文件名的选项
File.Open	打开指定路径上的 FileStream
File.Exists	确定指定的文件是否存在。
File.ReadAllText	

FileMode 值

FileMode	描述
FileMode.Append	打开现有文件并查找到文件尾，或创建新文件。
FileMode.Create	指定操作系统应创建新文件。如果文件已存在，它将被改写。
FileMode.CreateNew	指定操作系统应创建新文件。
FileMode.Open	指定操作系统应打开现有文件。
FileMode.OpenOrCreate	指定操作系统应打开文件（如果文件存在）；否则，应创建新文件。
FileMode.Truncate	指定操作系统应打开现有文件。文件一旦打开，就将被截断为零字节大小。

【例9-7】 利用File类显示文件的内容

请输入显示文件的名称: grade.txt

C# 89

English 93

Maths 87

【例9-7】利用File类显示文件的内容

```
1. class My {  
2.     static int Main() {  
3.         string fname, strContent="";  
4.         Console.Write("请输入显示文件的名称: ");  
5.         fname = Console.ReadLine();  
6.         try {  
7.             if(File.Exists(fname))  
8.                 strContent = File.ReadAllText(fname);  
9.             Console.WriteLine(strContent);  
10.        }  
11.        catch (Exception e) {  
12.            Console.WriteLine(e.Message);  
13.        }  
14.        return 0;  
15.    }  
16. }
```

目录管理Directory

静态方法	描述
Directory.CreateDirectory	创建指定路径中的所有目录。
Directory.Delete	删除指定的目录。
Directory.Exists	确定给定路径是否引用磁盘上的现有目录。
Directory.GetFiles	返回指定目录中的文件的名称。
Directory.Move	将文件或目录及其内容移到新位置。
GetCurrentDirectory()	获取应用程序的当前工作目录

【例9-8】利用Directory 类创建和删除目录

```
1. class My {  
2.     static int Main() {  
3.         string DirName = "Student";  
4.         Console.Write("请输入目录的名称: ");  
5.         DirName = Console.ReadLine();  
6.         try {  
7.             if (Directory.Exists(DirName))  
8.             {  
9.                 Directory.Delete(DirName);  
10.                Console.WriteLine(DirName + "目录删除成功! ");  
11.            }  
12.            else  
13.            {  
14.                Directory.CreateDirectory(DirName);
```

【例9-8】利用Directory 类创建和删除目录

```
1.         Console.WriteLine(DirName + "目录创建成功！");
2.     }
3. }
4. catch (Exception e)
5. {
6.     Console.WriteLine(e.Message);
7. }
8. return 0;
9. }
10. }
```

【例9-9】 利用DriveInfo类

- 来显示计算机中驱动器的信息。
 - 盘符:C:\,大小:209711706112,空闲:130647855104,文件系统:NTFS,类型:Fixed
 - 盘符:D:\,大小:209715195904,空闲:78445092864,文件系统:NTFS,类型:Fixed
 - 盘符:E:\,大小:209711706112,空闲:179210338304,文件系统:NTFS,类型:Fixed

【例9-9】

```
1. using System;
2. using System.IO;
3. class My
4. {
5.     static void Main()
6.     {
7.         var localList = DriveInfo.GetDrives();
8.         for (int i = 1; i < localList.Length-1; i++ )
9.         {
10.            var item = localList[i];
11.            Console.WriteLine("盘符:{0},大小:{1},空闲:{2},文件系统:{3},类型:{4}",
12. item.Name, item.TotalSize, item.AvailableFreeSpace, item.DriveFormat,
13. item.DriveType);
14.        }
15.    }
16. }
```

9.4 按字节读写文件FileStream

- 使用**FileStream**类读写文件把所有数据都看作字节流，需要将数据先转换成字节。
- 其常用**属性**包括：
 - (1) CanRead: 决定当前文件流是否支持文件读取操作；
 - (2) CanSeek: 决定当前文件流是否支持文件移动操作；
 - (3) CanWrite: 决定当前文件流是否支持文件写入操作；
 - (4) Length: 获取用字节表示的文件流的长度；
 - (5) Position: 获取或设置文件流的当前位置。

9.4 按字节读写文件FileStream

- 文件管理 FileStream类，其常用方法包括：
 - (1) Close方法：用于关闭文件流
 - (2) Read方法：实现文件流的读取
 - (3) ReadByte方法：用于从文件流中读取一个字节的数据
 - (4) Seek方法：将该流的当前位置设置为给定值
 - (5) Write方法：负责将数据写入到文件中
 - (6) WriteByte方法：用于向文件流中写入一个字节的数据
 - (7) Flush方法：负责将保存在缓冲区中的所有数据真正写入到文件中

【例9-6】利用FileStream类

```
1. using System;
2. using System.IO;
3. using System.Text;
4. class My
5. {
6.     static void Main() {
7.         Encoding e1 = Encoding.GetEncoding("GB2312");
8.         FileStream fs = new FileStream("file.txt", FileMode.Create);
9.         if (!File.Exists("file.txt"))
10.        {
11.            Console.WriteLine("文件创建失败！ ");
12.            return;
13.        }
```

【例9-6】利用FileStream类

```
1.     string str = "西安交通大学";  
2.     Byte[] bytes = e1.GetBytes(str);  
3.     fs.Write(bytes, 0, bytes.Length);  
4.     fs.Close();  
5. }  
6. }
```

9.5 二进制文件读写

【例9-9】 BinaryWriter和BinaryReader类

- 例：利用二进制文件存取前10项斐波那契数列。
- 输入和输出
- 0 1 1 2 3 5 8 13 21 34 55

【例9-9】

```
1. using System;
2. using System.IO;
3. using System.Text;
4. class My
5. {
6.     static int Main() {
7.         int[] x = { 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55 };
8.         int[] y = new int[20];
9.         int i, count=10;
10.        try    {
```

【例9-9】

```
1.      FileStream fs = new FileStream("my.dat", FileMode.Create);
2.      BinaryWriter bw = new BinaryWriter(fs);
3.      for ( i = 0; i <= count; i++)
4.          bw.Write(x[i]);
5.      bw.Close();
6.      fs.Close();
7.      FileStream fs1 = new FileStream("my.dat", FileMode.Open);
8.      BinaryReader br = new BinaryReader(fs1);
9.      for ( i = 0; i <= count; i++)
10.         y[i] = br.ReadInt32();
```

【例9-9】

```
1.         br.Close();
2.         fs1.Close();
3.     }
4.     catch (Exception e)
5.     {
6.         Console.WriteLine(e.Message);
7.     }
8.     for (i = 0; i <= count; i++)
9.         Console.Write("{0,4}", y[i]);
10.    Console.WriteLine();
11.    return 0;
12. }
13. }
```


【例9-10】

- 将3门课程的名字和成绩以二进制的形式存放在磁盘中，然后读出该文件，并将内容显示在屏幕上。

- 输入和输出

- C# 89

- English 93

- Maths 87

-

文件grade.dat:

C# 89

English 93

Maths 87

【例9-10】

```
1. using System;
2. using System.IO;
3. class My
4. {
5.     static int Main()
6.     {
7.         string strCourse;
8.         int iScore;
9.         FileStream fs = new FileStream("grade.dat", FileMode.Create);
10.        BinaryWriter bw = new BinaryWriter(fs);
11.        int i;
```

【例9-10】

```
1.     for (i = 0; i < 3; i++)
2.     {
3.         strCourse = Console.ReadLine();
4.         iScore = Convert.ToInt32(Console.ReadLine());
5.         bw.Write(strCourse);
6.         bw.Write(iScore);
7.     }
8.     bw.Close();
9.     fs.Close();
10.    FileStream fs1 = new FileStream("grade.dat", FileMode.Open);
11.    BinaryReader br = new BinaryReader(fs1);
```

【例9-10】

```
1.     Console.WriteLine("File grade.dat:");
2.     for (i = 0; i < 3; i++)
3.     {
4.         strCourse = br.ReadString();
5.         iScore = br.ReadInt32();
6.         Console.WriteLine(strCourse + " " + iScore);
7.     }
8.     br.Close();
9.     fs1.Close();
10.    return 0;
11. }
12. }
```

【例9-11】

- 模拟电视频道的存储和选择。创建一个文件，通过键盘输入电视频道序号和名称，并写入文件；文件中的内容可以输出到屏幕上。

【例9-11】

```
1. class TVChannel {  
2.     public int channelNum;  
3.     public string channelName;  
4.     public TVChannel()  
5.     {  
6.         channelNum = 0;  
7.         channelName = "中央电视台";  
8.     }  
9. };  
10. class My {  
11.     static void Main(string[] args)  
12.     {  
13.         int i;
```

【例9-11】

```
1.   TVChannel[] tv = new TVChannel[101];
2.   for (i = 0; i <= 100; i++)
3.       tv[i] = new TVChannel();
4.   do
5.   {
6.       Console.Write("Enter channel number (1 to 100, 0 to end input)? ");
7.       i = Convert.ToInt32(Console.ReadLine());
8.       if (i <= 0 || i > 100)
9.           break;
10.      tv[i].channelNum = i;
11.      Console.Write("Enter Channel Name? ");
12.      tv[i].channelName = Console.ReadLine();
13.  } while (true);
14.  try
15.  {
```

【例9-11】

```
1.      FileStream fs = new FileStream("tv.dat", FileMode.Create);
2.      BinaryWriter outTV = new BinaryWriter(fs);
3.      for (i = 0; i <= 100; i++)
4.      {
5.          outTV.Write(tv[i].channelNum);
6.          outTV.Write(tv[i].channelName);
7.      }
8.      outTV.Close();
9.      fs.Close();
10.     }
11.     catch (Exception e)
12.     {
13.         Console.WriteLine(e.Message);
14.     }
```


【例9-11】

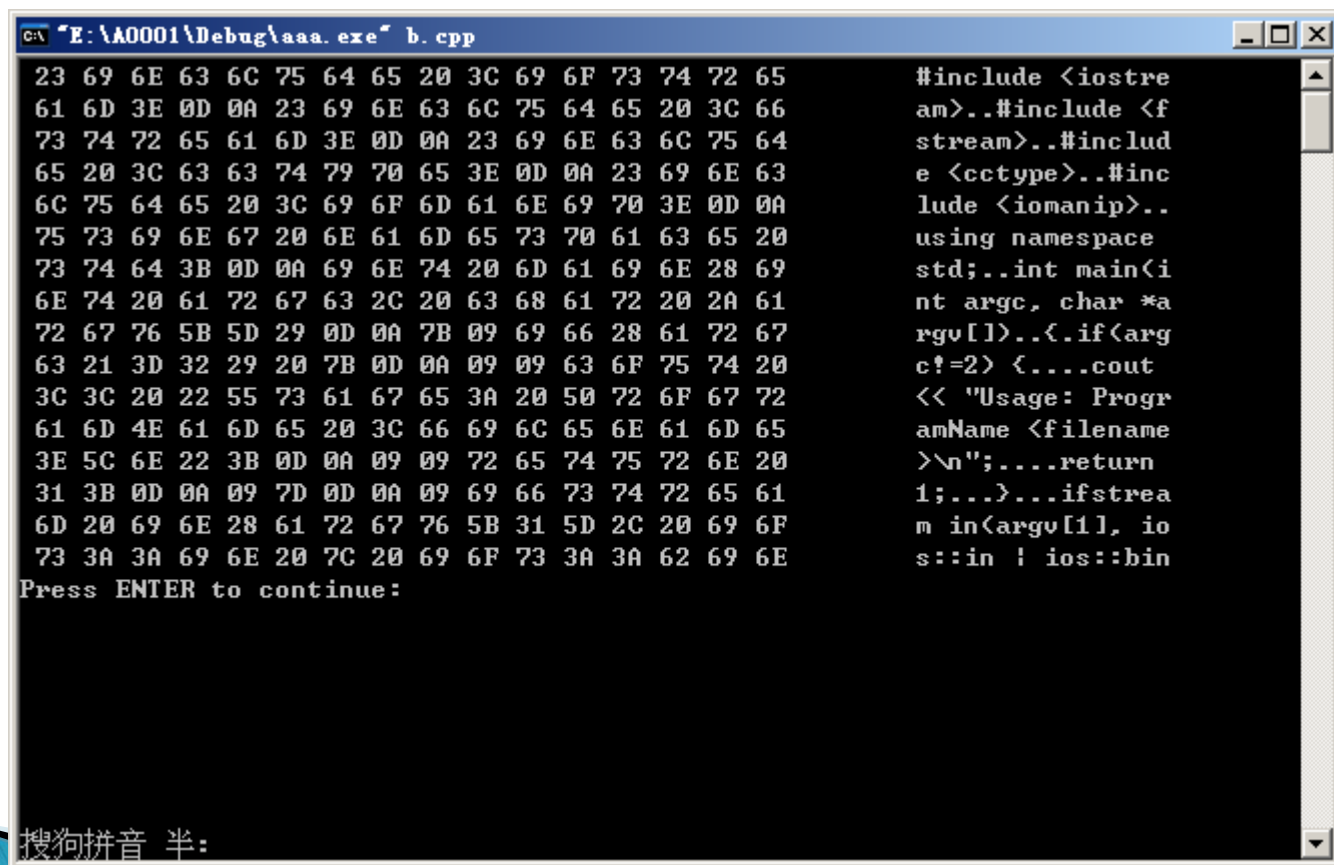
```
1.  try
2.      {
3.          FileStream fs = new FileStream("tv.dat", FileMode.Open);
4.          BinaryReader inTV = new BinaryReader(fs);
5.          for (i = 0; i <= 100; i++)
6.          {
7.              tv[i].channelNum = inTV.ReadInt32();
8.              tv[i].channelName = inTV.ReadString();
9.              if (tv[i].channelNum > 0)
10.             {
11.                 Console.WriteLine("{0}--> {1}", tv[i].channelNum, tv[i].channelName);
12.             }
13.         }
14.         inTV.Close();
15.         fs.Close();
```

【例9-11】

```
1.  }  
2.      catch (Exception e)  
3.      {  
4.          Console.WriteLine(e.Message);  
5.      }  
6.  }  
7.  }
```

【例9-12】

- 以十六进制的形式和ASCII码的形式显示文件的内容。



The screenshot shows a Windows command prompt window with the title bar "C:\ "E:\A0001\Debug\aaa.exe" b.cpp". The window displays the output of a program that reads a file and prints its contents in two columns: the left column shows the hexadecimal representation of the file's bytes, and the right column shows the corresponding ASCII characters. The output is as follows:

```
23 69 6E 63 6C 75 64 65 20 3C 69 6F 73 74 72 65      #include <iostre
61 6D 3E 0D 0A 23 69 6E 63 6C 75 64 65 20 3C 66      am>..#include <f
73 74 72 65 61 6D 3E 0D 0A 23 69 6E 63 6C 75 64      stream>..#includ
65 20 3C 63 63 74 79 70 65 3E 0D 0A 23 69 6E 63      e <cctype>..#inc
6C 75 64 65 20 3C 69 6F 6D 61 6E 69 70 3E 0D 0A      lude <iomanip>..
75 73 69 6E 67 20 6E 61 6D 65 73 70 61 63 65 20      using namespace
73 74 64 3B 0D 0A 69 6E 74 20 6D 61 69 6E 28 69      std;..int main(i
6E 74 20 61 72 67 63 2C 20 63 68 61 72 20 2A 61      nt argc, char *a
72 67 76 5B 5D 29 0D 0A 7B 09 69 66 28 61 72 67      rgv[])..<.if<arg
63 21 3D 32 29 20 7B 0D 0A 09 09 63 6F 75 74 20      c!=2> <....cout
3C 3C 20 22 55 73 61 67 65 3A 20 50 72 6F 67 72      << "Usage: Progr
61 6D 4E 61 6D 65 20 3C 66 69 6C 65 6E 61 6D 65      amName <filename
3E 5C 6E 22 3B 0D 0A 09 09 72 65 74 75 72 6E 20      >\n";....return
31 3B 0D 0A 09 7D 0D 0A 09 69 66 73 74 72 65 61      1;...>...ifstrea
6D 20 69 6E 28 61 72 67 76 5B 31 5D 2C 20 69 6F      m in<argv[1], io
73 3A 3A 69 6E 20 7C 20 69 6F 73 3A 3A 62 69 6E      s::in ! ios::bin
```

Below the hexadecimal and ASCII output, the program prompts the user with "Press ENTER to continue:". At the bottom left of the window, there is a watermark text "搜狗拼音 半:".

【例9-12】

```
1. using System;
2. using System.IO;
3. class My
4. {
5.     static int Main()
6.     {
7.         string strName = "program.cs";
8.         string strContent="";
9.         int i, j,k,n;
10.        char[] c;
11.        try
12.        {
13.            Console.WriteLine("Please input filename:");
```

【例9-12】

```
1.      strName = Console.ReadLine();
2.      strContent = File.ReadAllText(strName);
3.  }
4.  catch (Exception e)
5.  {
6.      Console.WriteLine(e.Message);
7.  }
8.  c = strContent.ToCharArray();
9.  n = c.Length;
10. for(k=0;k<n;k=k+16)
11. {
12.     i=(n-k>=16?16:n-k);
```

【例9-12】

```
1.      for (j = 0; j < i; j++)
2.          Console.Write(" {0:X2}", (int)c[k+j]);
3.      for (; j < 16; j++)
4.          Console.Write(" ");
5.      Console.Write("\t");
6.      for (j = 0; j < i; j++)
7.          if (c[k+j] >= 0x20 && c[k+j] <= 0x7F)
8.              Console.Write(c[k+j]);
9.          else
10.             Console.Write(".");
11.      Console.WriteLine();
12.  }
13.  return 0;
14.  }
15. }
```

结 束 语

- 学好程序设计语言的唯一途径是

上机练习

- 你的编程能力与你在计算机上投入的时间成

正比