```
  {
    StreamReader fichero;
    string nombre;

    while (true) {
      Console.Write( "Dime el nombre del fichero (\"fin\" para terminar): ");
      nombre = Console.ReadLine();
      if (nombre == "fin")
        break;
      if ( File.Exists(nombre) )
      {
        fichero = File.OpenText( nombre );
        Console.WriteLine("Su primera linea es: {0}",
          fichero.ReadLine() );
        fichero.Close();
      }
      else
        Console.WriteLine( "No existe!" );
    }
  }

}
```

Ejercicios propuestos:

- **(8.6.1)** Mejorar el ejercicio 8.3.2 para que compruebe antes si el fichero existe, y muestre un mensaje de aviso en caso de que no sea así.

8.7. Más comprobaciones de errores: excepciones

El uso de "File.Exists" nos permite saber si el fichero existe, pero ese no es el único problema que podemos tener al acceder a un fichero. Puede ocurrir que no tengamos permiso para acceder al fichero, a pesar de que exista, o que intentemos escribir en un dispositivo que sea sólo de lectura (como un CD-ROM, por ejemplo).

Por ello, una forma más eficaz de comprobar si ha existido algún tipo de error es comprobar las posibles "excepciones", con las que ya tuvimos un contacto al final del tema 2.

Típicamente, los pasos que puedan ser problemáticos irán dentro del bloque "try" y los mensajes de error y/o acciones correctoras estarán en el bloque "catch". Así, un primer ejemplo, que mostrara todo el contenido de un fichero de texto y que en caso de error se limitara a mostrar un mensaje de error y a abandonar el programa, podría ser:

```
/*--------------------------*/
/*  Ejemplo en C# nº 75:    */
/*  ejemplo75.cs            */
/*                          */
/*  Excepciones y ficheros  */
/*  (1)                     */
/*                          */
/*  Introduccion a C#,      */
/*    Nacho Cabanes         */
/*--------------------------*/

using System;
using System.IO;
```

```
public class Ejemplo75
{

  public static void Main()
  {
    StreamReader fichero;
    string nombre;
    string linea;

    Console.WriteLine("Introduzca el nombre del fichero");
    nombre = Console.ReadLine();

    try
    {
      fichero = File.OpenText(nombre);
      do
      {
        linea = fichero.ReadLine();
        if (linea != null)
          Console.WriteLine( linea );
      }
      while (linea != null);

      fichero.Close();
    }
    catch (Exception exp)
    {
      Console.WriteLine("Ha habido un error: {0}", exp.Message);
      return;
    }
  }

}
```

El resultado, si ese fichero no existe, sería

```
Introduzca el nombre del fichero
prueba
Ha habido un error: No se pudo encontrar el archivo 'C:\Fuentes\nacho\prueba'.
```

Pero, como ya vimos, generalmente lo razonable no es interceptar "todas las excepciones a la vez", sino crear un análisis para cada caso, que permita recuperarse del error y seguir adelante, para lo que se suelen crear varios bloques "catch". Por ejemplo, en el caso de que queramos crear un fichero, podemos tener excepciones como éstas:

- El fichero existe y es de sólo lectura (se lanzará una excepción del tipo "IOException").
- La ruta del fichero es demasiado larga (excepción de tipo "PathTooLongException").
- El disco puede estar lleno (IOException).

Así, dentro de cada bloque "catch" podríamos indicar una excepción más concreta, de forma que el mensaje de aviso sea más concreto, o que podamos dar pasos más adecuados para solucionar el problema:

```
/*--------------------------*/
/*  Ejemplo en C# nº 76:    */
/*  ejemplo76.cs            */
/*                          */
/*  Excepciones y ficheros  */
/*  (2)                     */
/*                          */
/*  Introduccion a C#,      */
/*    Nacho Cabanes         */
/*--------------------------*/

using System;
using System.IO;

public class Ejemplo76
{

  public static void Main()
  {
    StreamWriter fichero;
    string nombre;
    string linea;

    Console.Write("Introduzca el nombre del fichero: ");
    nombre = Console.ReadLine();
    Console.Write("Introduzca la frase a guardar: ");
    linea = Console.ReadLine();

    try
    {
      fichero = File.CreateText(nombre);
      fichero.WriteLine( linea );
      fichero.Close();
    }
    catch (PathTooLongException e)
    {
      Console.WriteLine("Nombre demasiado largo!");
    }
    catch (IOException e)
    {
      Console.WriteLine("No se ha podido escribir!");
      Console.WriteLine("El error exacto es: {0}", e.Message);
    }
  }

}
```

Hay que recordar que como la consola se comporta como un fichero de texto (realmente, como un fichero de entrada y otro de salida), se puede usar "try…catch" para comprobar ciertos errores relacionados con la entrada de datos, como cuando no se puede convertir un dato a un cierto tipo (por ejemplo, si queremos convertir a Int32, pero el usuario ha tecleado sólo texto).

Ejercicios propuestos:

- **(8.7.1)** Un programa que pida al usuario el nombre de un fichero de origen y el de un fichero de destino, y que vuelque al segundo fichero el contenido del primero,