


Ejercicios Sistema de Ficheros

[Descargar estos ejercicios](#)

Índice

1. [Ejercicio 1](#)
2. [Ejercicio 2](#)
3.  [Ejercicio 3](#)

Ejercicio 1

Como se explica en el tema, para trabajar con **las cadenas que representan las rutas de un sistema de archivos**, podemos usar la clase de utilidad **Path** . Esta clase tiene varios métodos, algunos se nombran en el tema pero otros no. Crea un programa para probar, al menos, los métodos que se relacionan a continuación. Indica al lado de cada método y con comentarios la salida después de la ejecución.

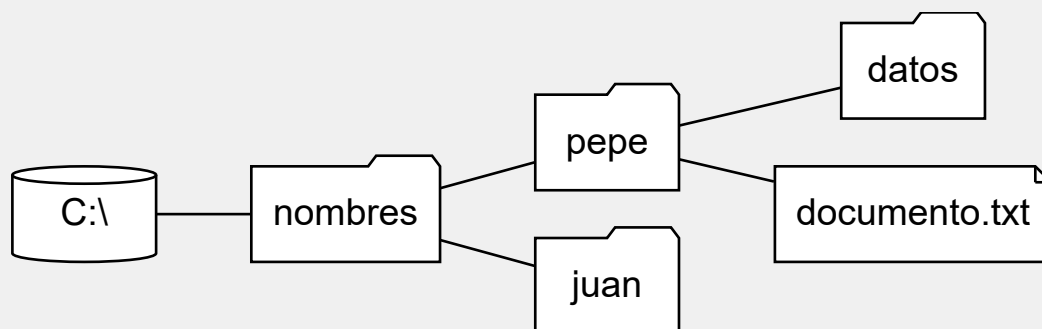
```
string GetExtension(string ruta)
string GetFileName(string ruta)
string GetFileNameWithoutExtension(string ruta)
string GetDirectoryName(string ruta)
string GetPathRoot(string ruta)
string ChangeExtension(string ruta, string nuevaExtensión)
string GetFullPath(string rutaRelativa)
string Combine(string ruta1, string ruta2)
```

Nota: Busca en la documentación de Microsoft información y ejemplos sobre los mismos.

Ejercicio 2

Crear un programa que tenga los siguientes métodos, que podrán llamarse mediante un menú con opción de salir con ESC. Usa las clases de utilidad **File** y **Directory** para resolver el ejercicio.

1. **CreaArbolDeDirectorios** : Que creará el siguiente árbol con directorios y ficheros incluidos, si ya existen, avisará mediante un mensaje.



2. **EliminaDirectorio** : Al que le pasarás la ruta de uno de los directorios del árbol y te lo eliminará, siempre y cuando exista, avisando en caso contrario.
3. **EliminaFichero** : Idem al anterior pero con ficheros.
4. **MuestraInformación** : Que mostrará todos los miembros de un directorio, siempre y cuando exista mostrando un mensaje en caso contrario.
5. **MuestraAtributos** : Que mostrará el estado de los atributos del fichero que le indiques.

✓ Ejercicio 3

Vamos a crear algunos **comandos similares a los del S.O.**, que nos ayuden a entender mejor el funcionamiento de las clases **DirectoryInfo**, **FileInfo** y sus homólogas de utilidad. Para ello deberemos crear distintos programas, de forma que cada uno haga una de las siguientes funciones. El nombre de cada proyecto será el nombre del comando y todos irán en una solución llamada **ejercicio3** :

1. Comando **listaCarpeta**: Se encargará de mostrar el contenido del directorio actual si no se le indica ruta, o del directorio que se indique en la ruta. Sino existe el directorio, se capturará la excepción mostrando un mensaje de error.
Puedes usar la funcionalidad de las enumeraciones excluyentes para conocer el tipo de elemento:

```
//recuerda, podemos saber que elemento es un archivo si hacemos algo como lo siguiente  
a.Attributes & FileAttributes.Archive==FileAttributes.Archive
```

Una posible salida si listamos la carpeta de usuario `listaCarpeta c:\users` , podría ser:

All Users	Directory 07/12/2019 10:30:39
Default	Directory 07/12/2019 10:03:44
Default User	Directory 07/12/2019 10:30:39
desktop.ini	Archive 07/12/2019 10:14:54
Public	Directory 07/12/2019 10:14:52
xusa	Directory 07/10/2020 21:58:18


2. Comando **creaDirectorio**: Se le pasa una ruta y te crea un directorio y tendrás que controlar si se indica más de una entrada o capturar las excepciones de ruta inválida.
3. Comando **eliminaDirectorio**: Se le pasa una ruta y te elimina un directorio, tendrás que controlar si se indica más de una entrada o capturar las excepciones de ruta inválida.
4. Comando **eliminaFichero**: Se le pasa una ruta y elimina un fichero y tendrás que controlar si se indica más de una entrada o capturar las excepciones de ruta inválida.
5. Comando **copiaFicheros**: Se le pasa dos rutas:
 1. La primera de un directorio válido.
 2. la segunda de un directorio que no tiene porque existir.

El comando se encargará de copiar todos los ficheros que hay en la primera ruta a la segunda, creando la carpeta si fuera necesario. Además, deberás controlar las posibles excepciones.

Este comando podrá, además, ejecutarse con la opción de añadir un filtro a la copia (solo se copiarán los archivos que cumplan el filtro, eso lo podemos hacer con la sobrecarga del método `EnumerateFile`).

Ejemplo del comando para copiar los ficheros que contengan en su nombre *windows*, desde la carpeta logs a la carpeta copia:

```
copiaFicheros c:\logs c:\copia *windows*
```

 **Nota:** La ejecución y el paso de información al programa se realizará **a través de la línea de comandos**.

La explicación está en el primer ejercicio del tema anterior de excepciones.