UAI Universidad Abierta Interamericana

01 – PROGRAMACIÓN I

MG. NICOLÁS BATTAGLIA

Ti UAI TECNOLOGIA INFORMATICA

UAIOnline ultra

UAI Universidad Abierta Interamericana

UNIDAD 4

CLASE 10 - ARCHIVOS

UAIOnline ultra

INTRODUCCIÓN

- En las aplicaciones que se han desarrollado hasta el momento, los datos únicamente los manejamos en tiempo de ejecución, es decir, una vez cerrada la aplicación los datos que fueron cargados en memoria se pierden.
- La manera de almacenar y recuperar información que perdure en el tiempo se basa en el uso de "memoria secundaria", compuesta esencialmente unidades externas como CD, DVD, memorias USB, SD, entre otros tipos de almacenamiento.
- En cualquiera de estos medios, la unidad de almacenamiento de información se denomina archivo

ARCHIVOS

- Conjunto de bits que son almacenados en un dispositivo.
- Es identificado por un nombre y la descripcion de la carpeta o directorio que lo contiene.
- Se almacenan físicamente en discos y tienen extensiones variadas.

UAIOnline ultra

UAIOnline ultra

VENTAJAS DE USO DE ARCHIVOS

- RESIDENCIA
- INDEPENDENCIA
- PERMANENCIA
- PORTABILIDAD
- CAPACIDAD

UAIOnline ultra

DESVENTAJAS DE USO DE ARCHIVOS

- CONCURRENCIA
- TRANSACCIONES
- DEPENDENCIA CON EL PROGRAMA QUE LO GESTIONA

CLASIFICACIÓN

- Por flujo de datos
  - Entrada
  - Salida
  - Entrada/Salida
- Por los datos cargados
  - Texto
  - Binarios
- Por tipo de acceso
  - Secuencial
  - Aleatorio
- Por uso
  - Permanente
  - Movimiento
  - Temporales

UAIOnline ultra

TIPOS DE ARCHIVOS – FLUJO DE DATOS

- Entrada : Lectura de Datos
  - Se crea un flujo de datos de lectura.
  - Se leen datos con los métodos apropiados.
  - Se cierra el flujo de datos.
- Salida : Escritura de Datos
  - Se crea un objeto flujo de datos de escritura.
  - Se escriben datos utilizando los métodos apropiados
  - Se cierra el flujo de datos.
- Entrada/Salida: Lectura/Escritura de datos

UAIOnline ultra

TIPOS DE ARCHIVOS - USO

- ARCHIVOS PERMANENTES
  - Los datos guardados en ellos cambian poco en el tiempo, son generalmente usados para estadisticas y cumplen una funcion de soporte o cumplemento.
- ARCHIVOS DE MOVIMIENTO
  - Son los tradicionales archivo donde el sistema realiza las actualizaciones constantes
- ARCHIVOS TEMPORALES
  - Se crea solo cuando se usa y luego se destruye

UAIOnline ultra

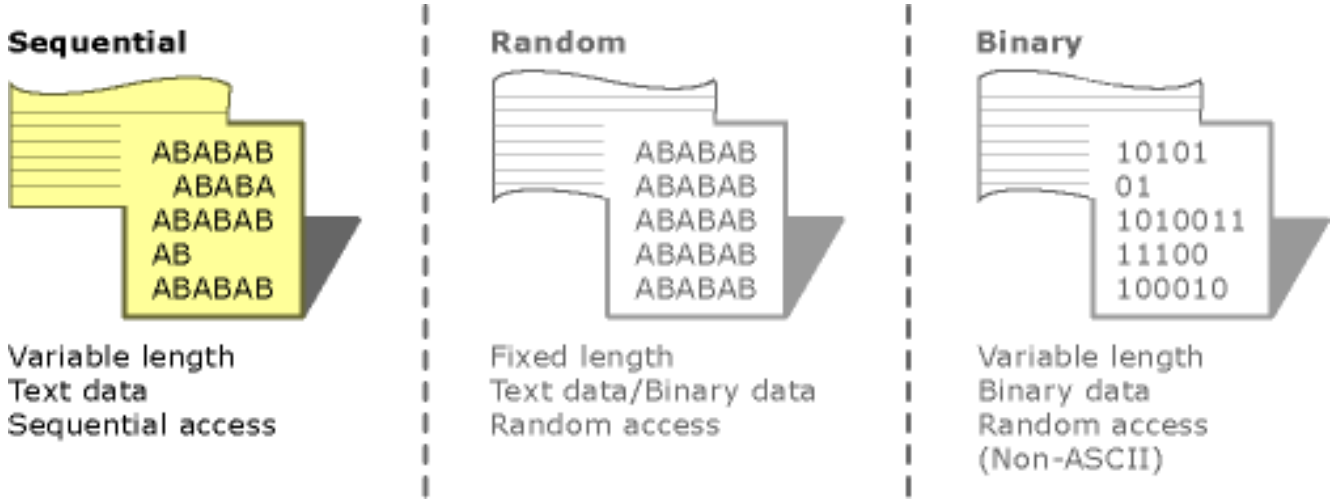
TIPOS DE ARCHIVOS – DATOS CARGADOS

- Los archivos de texto plano son aquellos que están compuestos únicamente por texto sin formato, solo caracteres. Estos caracteres se pueden codificar de distintos modos dependiendo de la lengua usada

- Un archivo binario es una archivo informático que contiene información de cualquier tipo, codificada en forma binaria para el propósito de almacenamiento y procesamiento de ordenadores.

TIPOS DE ARCHIVOS – ACCESO SECUENCIAL

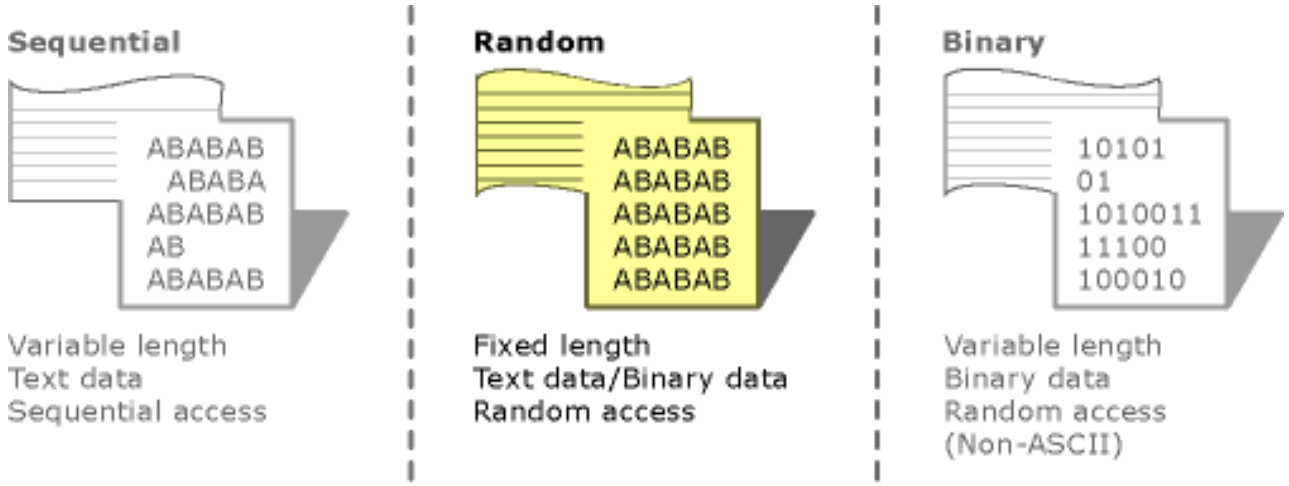
- El *acceso secuencial* está diseñado para utilizarlo con archivos de texto sin formato.
- Se considera que cada carácter de un archivo representa un carácter de texto o una secuencia de formato de texto, como un carácter de nueva línea
- Los datos se almacenan como caracteres ANSI.



UAIOnline ultra

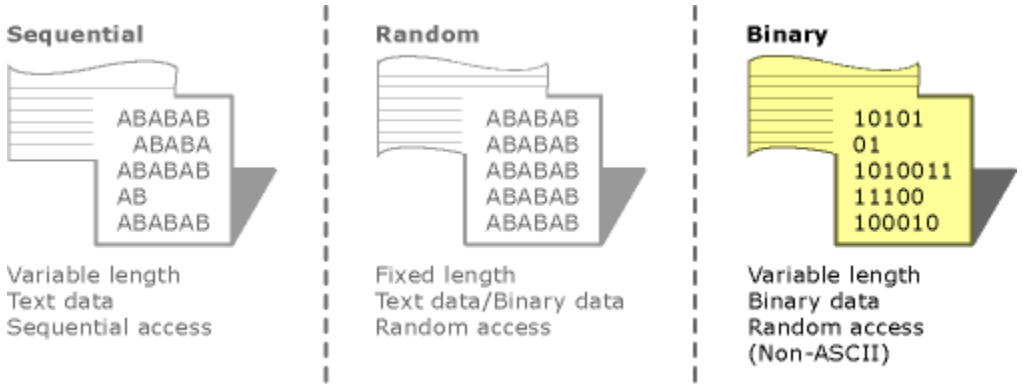
TIPOS DE ARCHIVOS – ACCESO ALEATORIO

- Un archivo abierto para *acceso aleatorio* se considera que está compuesto de un conjunto de registros de longitud idéntica o de conjuntos de campos que contienen información.
- Se pueden utilizar tipos definidos por el usuario para crear registros compuestos por numerosos campos, cada uno de los cuales puede tener diferentes tipos de datos. Los datos se almacenan como información binaria.



TIPOS DE ARCHIVOS – ACCESO BINARIO

- El *acceso binario* permite utilizar archivos para almacenar datos de cualquier modo que se ajuste a las necesidades, ya sean numéricos, de cadena o ambos.
- El acceso binario es similar al aleatorio, excepto que no se hacen suposiciones acerca del tipo de datos o de la longitud de registro.
- Es necesario conocer cómo se escribieron exactamente los datos en el archivo para poder recuperarlos correctamente.
- **Por ejemplo**, si almacena una serie de nombres y números de teléfono, debe recordar que el primer campo (el nombre) es texto y el segundo (el número de teléfono) es numérico.



UAIOnline ultra

CAMPOS DE UN REGISTRO

- SON LA UNIDAD LOGICA DONDE SE ALMACENAN LOS DATOS DE UN REGISTRO.
- TIENEN COMO CARACTERISTICA NOMBRE, TIPO Y TAMAÑO
- LOS TIPOS DE DATOS Y SU TAMAÑO SE VERAN RESTRINGIDO POR EL LENGUAJE QUE SE UTILICE.
- UN CONJUNTO DE CAMPOS COMPONEN UN REGISTRO.
- UN CONJUNTO DE REGISTROS COMPONEN UN ARCHIVO.

NOMBRE	TIPO	TAMAÑO
NroCta	Numero	5
RazonSocial	Alfabetico	25
FechaDeposito	Fecha	6
Comentarios	Memo	200

SYSTEM.IO

- **System.IO.File**
  - Proporciona métodos estáticos para crear, copiar, eliminar, mover y abrir un solo archivos y contribuye a la creacion de objetos **FileStream**
- **FileStream** Es una secuencia de bytes que representa el contenido del archivo.
- **File.Open(string, fileMode) as FileStream** Modo de Apertura de archivos – Ejemplos
  - **Append** Abre el archivo si existe y busca hasta el final del archivo o crea un nuevo archivo
  - **Create** pecifica que el sistema operativo debe crear un nuevo archivo. Si el archivo ya existe, se sobrescribirá
  - **CreateNew** Especifica que el sistema operativo debe crear un nuevo archivo
  - **Open** Especifica que el sistema operativo debe abrir un archivo existente
  - **OpenOrCreate** Especifica que el sistema operativo debe abrir un archivo si existe; de lo contrario, debe crearse un nuevo archivo
  - **Truncate** Especifica que el sistema operativo debe abrir un archivo existente. Cuando se abre el archivo, debe truncarse para que su tamaño es cero bytes

UAIOnline ultra

SYSTEM.IO - RESUMEN

Clase	Descripción
FileStream	Permite crear objetos para leer o escribir datos en archivos. Para ello es necesario definir el nombre completo del archivo (incluyendo la ruta de ubicación de sus carpetas)
StreamWriter	Permite crear objetos que implementan un sistema de escritura de datos basado en secuencias de caracteres.
StreamReader	Permite crear objetos que implementan un sistema de lectura de datos basado en secuencias de caracteres
File	Esta clase contiene métodos estáticos para manipular archivos como su creación, copiarlos, eliminarlos, moverlos o detectar su existencia.

FILESTREAM

Los archivos pertenecen a la clase FileStream y podemos reducir sus operaciones como:

- Crear un archivo o abrir uno ya existente
- Leer o escribir datos en un archivo
- Cerrar el archivo

- Lo primero que debemos hacer es crear un objeto de tipo FileStream asociado a un archivo en particular , el código podría ser :
  - FileStream miArchivo (ruta, modo apertura)
  - FileStream miArchivo(ruta, modo apertura , acceso)
  - FileStream miArchivo(ruta, modo apertura , acceso , sharing)
- Donde:
  - Ruta: directorio donde se desea guardar el archivo
  - Modo apertura (FileMode): crearse, abrirse ,crearse si no existe, abrirse para agregar datos , etc
  - Acceso (FileAccess): leer, escribir o ambas
  - Sharing: modo de bloqueo

FILEMODE

FileMode	Uso
CreateNew	Crea un nuevo archivo. Si el archivo existe dispara una IOException
Truncate	Abrir un archivo existente. Una vez abierto, el archivo será truncado a cero bytes de longitud.
Create	Crea un nuevo archivo. Si el archivo existe será sobrescrito.
Open	Abrir un archivo existente. Si no existe dispara una FileNotFoundException.
OpenOrCreate	Abrir un archivo existente, si no existe, lo crea
Append	Abrir un archivo para agregar datos al final en caso de existir; de lo contrario crea un archivo nuevo

UAIOnline ultra

FILEACCESS

FileAccess	Uso
Read	Acceso al archivo en modo de solo lectura
ReadWrite	Acceso al archivo en modo de lectura y escritura
Write	Acceso al archivo en modo de solo escritura

STREAMREADER

- Para leer datos de un archivo, es necesario abrirlo en modo lectura estableciendo un flujo al crear un objeto de la clase StreamReader

```
//Cuando queremos leer un archivo deberíamos
//crear el filestream
FileStream archivo=new FileStream("a1.dat",filemode.open)

//creamos el streamreader para leerlo
StreamReader Lector = new StreamReader(archivo);

//Podriamos abreviar mediante
StreamRader Lector= new streamreader("a1.dat");
```

STREAMREADER

Métodos	Descripción
Read	lee el próximo carácter disponible a partir de la posición actual y avanza
ReadBlock	lee un bloque de caracteres y lo almacena en un vector de chars
ReadLine	lee una línea del archivo
ReadToEnd	lee el contenido de todo el archivo y lo guarda en un string
Peek	obtiene el valor del próximo carácter disponible pero no avanza
Close	cierra el stream liberando recursos y referencias al archivo

LECTURA POR LÍNEA - EJEMPLO

```
FileStream miArchivo = new FileStream("a1.txt",FileMode.OpenOrCreate);
        StreamReader lector = new StreamReader(miArchivo);
        string linea;
        linea = lector.ReadLine();
        while (linea!=null)
        {  this.listBox1.Items.Add(linea);
            linea = lector.ReadLine();
        }
        lector.Close();
        miArchivo.Close();
```

LECTURA - EJEMPLO

Utilizando el while Peek

```
StreamReader archivo = new StreamReader("archi.dat");
   string línea;
   línea = archivo.readline();
   while (!archivo.peek() == -1)
       línea = archivo.readline();
   Archivo.close();
```

Si queremos leer el archivo de una sola vez

```
StreamReader archivo = new StreamReader("archi.dat");
   string contenido;
   contenido = archivo.ReadToEnd();
   Archivo.close();)
```

STREAM WRITER

UAIOnline ultra

- Para escribir datos en un archivo, es necesario abrirlo en modo escritura estableciendo un flujo al crear un objeto de la clase StreamWriter

```
StreamWriter escritor= new StreamWriter(NombreArchivo)
```

Métodos	Descripción
Write	escribe una cadena de texto en el archivo
WriteLine	escribe una cadena y le agrega un salto de fin de línea
Flush	devuelve el contenido del buffer del stream al archivo
Close	realiza cualquier escritura pendiente sobre el archivo (flush) y lo cierra

STREAM WRITER

UAIOnline ultra

- Ejemplo para escribir 100 líneas

```
StreamWriter escritor = new StreamWriter("a1.txt");
        int i;
        for (i = 1; i <= 100; i++)
            escritor.WriteLine(i);
            escritor.Close();
```

FILE

- **File.Create(path, bufferSize, FileOptions, FileSecurity)**
  - **path** Nombre del archivo
  - **bufferSize** Número de bytes almacenados en el búfer para leer y escribir en el archivo.
  - **FileOptions** Describe como crear o sobreescribir el archivo.
    - Async, DeleteOnClose, Encrypt, None,
    - **RandomAccess** Indica que el acceso al archivo se realiza aleatoriamente. El sistema puede considerar que esto es una sugerencia para optimizar el almacenamiento en caché del archivo.
    - **SequentialScan** Indica que el acceso al archivo debe ser secuencial de principio a fin.
  - **FileSecurity** Determin ael Control de acceso y la seguridad de auditoria del archivo
- Comprobación
  - **File.Exists(string)**
  - Pueden suceded fallas, archivos inexistentes, dañados o sin permisos de lectura.

MANEJO DEL SISTEMA DE ARCHIVOS

- **¿Cómo saber si un archivo existe?**

```
if (File.Exists(NombreArchivo))
{ ……


 }
```