Archivos

Introducción

Los archivos son estructuras de almacenamiento que se asocian con un dispositivo de memoria auxiliar y permanente. Son colecciones de registros semejantes, o del mismo tipo, guardados en dispositivos de almacenamiento secundarios, como podría ser un disco duro, un cd, un pendrive.

Son un medio de almacenamiento persistente, esto quiere decir, que a diferencia de cuando almacenamos información en la memoria ram, estos datos persisten al finalizar la ejecución del programa y pueden volver a utilizarse en otro momento, ademas de permitirnos manipularlos de manera fraccionada.

Existen muchos tipos de archivos, los que estudiamos son los archivos de datos homogeneos, que consisten en una secuencia de elementos del mismo tipo, que pueden ser de tipo simple o compuesto (por lo general son compuestos).

El manejo de los archivos esta muy ligado a cada lenguaje de programación. En el año nosotros analizanos mayormente este tema desde el punto de vista de Pascal y un poco sobre c.

Manejo de archivos

Declarar variable archivo

Para declarar una variable archivo es necesario definir el tipo de datos que se van a almacenar en el.

Se puede hacer de varias maneras:

- Archivo de texto, que contienen texto plano.
- Archivos con tipo, que contienen datos de cualquier tipo, como enteros, reales, registros, etc.
- Archivos sin topo, que permiten acceso de bajo nivel a los datos de un disco duro.

Tipos de acceso

Hay dos modos de acceso en archivos: el acceso secuencial y acceso directo o aleatorio.

El acceso secuencial exige una exploracion secuencial de los elementos, comenzando desde el primero. En camio el acceso directo permite acceder a un elemento determinado haciendo referencia directamente por su posición en el soporte de almacenamiento. Similar a los indices de los arreglos.

Los archivos de texto solo pueden tratarse mediante acceso secuencial. Serial algo similar a una secuencia de caracteres, con todas las ventajas de un archivo.

Los archivos con tipo se pueden tratar de cualquiera de las dosmaneras.

Opreaciones para acceso secuencial:

- Asignar
- Abrir
- Cerrar
- Leer
- Escribir
- EOLn
- EOF

Opreaciones para acceso directo:

- PosiciónActual
- TamañoArchivo
- irPos

Archivos y secuencias

Teniendo en cuenta lo anterior podemos ver que hay varias maneras de tratar archivos. si quisieramos manejarlos usando un tratamiento secuencial usando las operaciones adecuadas y analizar que modelo responden, para no cometer errores en el tratamiento.

Entonces si, por ejemplo, queremos trabajar con un archivo de texto primero deberíamos declarar la variable archivo, y tambien una variable del mismo tipo que el contenido para manipular la información del archivo.

```
arch ∈ Text
cad ∈ Cadena
```

los cambios que se hagan sobre la variable arch deberían verse reflejados en un archivo en el medio donde está almacenado, pero para que esto suceda lo primero que necesitamos es el nombre de este archivo, su ubicación en el medio. El nombre con el que lo reconoce el SO se llama **nombre externo** y la variable "arch" sería el **nombre interno**.

Estos dos nombres por lo general son distintos, y para que no haya ambiguedades se deben vincular mediante la primitiva "Asignar".

```
Asignar(arch,'datos.txt')
```

siendo arch el nombre interno y "datos.txt" el nombre externo.

Lo siguiente que debe hacerse es Abrir el archivo usando la primitiva abrir, ya que otro programa podria intentar acceder al archivo y esto no se deberia poder permitir ya que nuestro programa lo esta utilizando.

```
Abrir(arch)
```

En pascal hay 3 acciones para abrir un archivo

- Rewrite(x) descarta el valor de x para que pueda ser generado de forma secuencial.
- Reset(x) deja la posición al comienzo del archivo.
- Append(x) Abre un archivo existente para añadir datos el final del mismo. nos posiciona al final del archivo.

En pascal cuando intentamos abrir un archivo que no existe, o que esta bloqueado, se produce un error, para evitar esto se usa la directiva {$I} para capturar el error en la variable IOResult y evitar que el programa falle. Si el valor de IOResult es 0 significa que la apertura fue exitosa.

```
ejemplo de ioresult
```

Una vez que se ha terminado de trabajar con el archivo se lo debe dejar preparado para que otro programa pueda utilizarlo. para eso se usa la primitiva Cerrar.

```
Cerrar(arch)
```

En los archivos de texto, podemos saber cuando estamos en el final de una linea con la función **EOLn(arch)**, que devuelve verdadero si estamos en el fin de linea o falso en caso contrario.

Tambien podemos saber si estamos en el final del archivo usando **EOF(arch)**.

Si quisieramos leer EOF debería ser falso, ya que si fuera verdadero estaríamos en el final del archivo y no habria nada que leer.

Una vez que leemos o escribimos el cabezal se posiciona en el registro siguiente pero sin leer la informacion alojada en el. Esto sucede por la manera en la que funcionan las primitivas leer y escribir.

Entonces por ejemplo podriamos leer toda la informacion del archivo de texto de la siguiente manera

```
Mientras No EOF(arch) hacer
  Leer(arch,cad);
  escribir(cad)
fmientras
```

Acceso directo

Tambien podemos acceder a los elementos del archivo de forma directa usando las operaciones

- PosicionActual(arch): devuelve la posicion actual del dispositivo en el archivo. Los registros comienzan desde la posición 0.
- TamañoArchivo(arch): cantidad de elementos en el archivo.
- irPos(arch,indice): Situa el cabezal sobre el archivo en la posicion indicada.

Comentar sobre borrado virtual y fisico.