Apunte del curso ALGORITMOS y PROGRAMACIÓN (FI-UBA, Prof. Ing. F. J. LAGE, J. T. P. Ing. Z. CATALDI; A.T.P. Srta. A Pauluzzi, Sr. C Corradino, Sr. F Gómez

ARCHIVOS de TEXTO en PASCAL

Un archivo o fichero (file) es una estructura de datos que reside en memoria secundaria, consistente en un conjunto de informaciones estructuradas en unidades de acceso denominadas registros, todos del mismo tipo y en número indeterminado. En contraste con los arreglos, el tamaño de los archivos no es fijo, y está limitado solamente por la capacidad de almacenamiento disponible. Además cuando uno apaga la máquina los arreglos se pierde su contenido, el contenido de los archivos no se pierde por esta causa.

Los archivos en general, están compuestos por registros. Mediante programas adecuados se pueden manipular los distintos tipos de archivos (texto, tipeados, no tipeados). Cada archivo es referenciado por un identificador (su nombre y su extensión). Cuando hablamos de archivos tipeados, nos referimos a un archivo de registros.

Los registros en un archivo son de dos tipos.

Registro **lógico**: que es cada uno de los componentes del archivo, conteniendo el conjunto de informaciones que se tratan de manera unitaria. Está constituido por uno o más elementos denominados campos, que pueden ser de diferentes tipos y que a su vez pueden estar compuestos por subcampos.

Si un archivo contiene la información de un conjunto de individuos u objetos, sus registros contienen la información de cada uno de ellos y los campos los diferentes dates que lo componen.

Por ejemplo, en el archivo de empleados de una empresa, cada registro contiene la información de un empleado y los campos contienen su nombre, dirección, fecha de ingreso, etc.

Registro **físico**: o *bloque* corresponde a la cantidad de información que se transfiere en cada operación de acceso (lectura o escritura).

Un registro lógico y registro físico, se diferencian en que el *tamaño y formato del registro lógico los define el programador, mientras que el tamaño del registro físico viene dado por las características físicas de la computadora utilizada.*

En general, un bloque puede contener uno o más registros, pero también puede ocurrir que un registro ocupe más de un bloque. En el primer caso se dice que los registros están bloqueados, denominándose factor de bloqueo al número de registros lógicos que contiene cada registro físico.

Para poder seleccionar un registro del conjunto que compone el archivo, se necesita un dato identificativo que lo distinga de los demás. Se denomina campo clave a un campo especial del registro que sirve para identificarlo.

Algunos archivos en sus registros no tienen campo clave, mientras que otros pueden tener varios, denominándose a éstos clave primaria, secundaria, etc. Por ejemplo, en el archivo de empleados antes citado, un campo clave podría ser el número de DNI y una clave secundaria el nombre completo del empleado.

CARACTERISTICAS DE LOS ARCHIVOS

Las principales características que diferencian esta estructura de dates de las restantes son las siguientes:

- **Residencia en soportes de información externos**, también denominados memorias secundarias o auxiliares, como son las cintas y discos (magnéticos u opticos).
- **Independencia respecto de los programas**, lo que significa que la vida del archivo no está limitada por la vida del programa que lo creó y también que en diferentes momentos pueden hacer uso del mismo archivo diferentes programas.
- **Permanencia de las informaciones almacenadas**, es decir, la información contenida en un archivo no desaparece cuando se desconecta la computadora, a diferencia de todas las informaciones almacenadas en la memoria central.
- **Gran capacidad de almacenamiento**, siendo esta capacidad teóricamente ilimitada; por el contrario, las estructuras de datos que residen en la memoria central tiene limitado su tamaño por la capacidad de ésta.

CLASIFICACIÓN DE LOS ARCHIVOS

Los archivos se pueden clasificar por su tipo.

Tipos de archivos

Los elementos de un archivo pueden ser de cualquier tipo, simple o estructurado, excepto un tipo archivo (file) o cualquier tipo estructurado con un componente tipo archivo. Los principales tipos de archivos son:

- Archivo de **datos**: es una colección de datos localizados en un dispositivo de entrada./salida
- Archivo de **programa**: un programa codificado en un lenguaje específico y localizado o almacenado en un dispositivo de almacenamiento.
- Archivo de **texto**: una colección de caracteres almacenados como una unidad en un dispositivo de almacenamiento.

Tipos de acceso a un archivo

Existen dos modalidades para acceder a un archivo de datos: acceso secuencial y acceso directo o aleatorio.

- Acceso secuencial exige el tratamiento elemento a elemento, es necesario una exploración secuencial, comenzando desde el primer elemento.
- Acceso directo permite procesar o acceder a un elemento determinado. Si se referencia correctamente por su posición en el soporte de almacenamiento.

OPERACIONES SOBRE ARCHIVOS

Entre las operaciones más usuales que se realizan con archivos destacan la *creación*, *copia*, *consulta*, *clasificación*, *concatenación*, *intersección*, *fusión*, *partición*, *actualización*, *reorganización* y *borrado*.

Estas operaciones se llevan a cabo bien por un programa del usuario, o bien por un programa del propio sistema operativo de la computadora.

Las operaciones más usuales a nivel de registro son: *inserción*, *supresión*, *modificación* y *consulta* del contenido de los mismos. Estas operaciones se hacen generalmente a través de programas de actualización.

Creación

Consiste en la escritura o grabación en un soporte determinado de todos los registros que van a conformar el archivo. Los dates pueden ser introducidos desde un teclado, pueden preceder de otro archivo o ser obtenidos como resultado de algún proceso.

Copia

Es una de las operaciones más usadas y consiste en crear un archivo nuevo como duplicación de otro existente. La copia puede realizarse en el mismo o en diferente soporte de información.

Un caso particular de esta operación es la impresión (copia en impresora) de un archivo.

Consulta

Se realiza para obtener el contenido de uno o varios registros. En muchos casos va precedida de una búsqueda de los mismos.

Por ejemplo, si se desean conocer todos los datos de un alumno, almacenados en el archivo de alumnos, lo haremos mediante un programa al que proporcionamos el número de matrícula como dato de entrada, que será utilizado para realizar la búsqueda y sacar por pantalla o impresora el resto de campos almacenados en el registro correspondiente a ese número de matrícula.

Clasificación u ordenación

Esta operación consiste en reubicar los registros, de tal forma que queden ordenados con respecto a los valores de un campo que denominamos clave de ordenación.

En un archivo clasificado serán mucho más rápidas las consultas que se realicen por medio del campo que rija la ordenación. Por ejemplo, será muy útil clasificar alfabéticamente el archivo de alumnos por el campo nombre, ya que la mayoría de las consultas se harán utilizando este campo.

Concatenación

Dados dos archivos con registros de igual estructura, se trata de obtener uno sólo en que figuren todos los registros del primero, y a continuación todos los del segundo. Esta operación se puede generalizar para más de dos archivos.

Intersección

Dados dos archivos de igual estructura, se trata de obtener otro archivo en el que figuren los registros comunes a ambos.

Por ejemplo, si tenemos el archivo de alumnos matriculados en un centro en primer curso y deseamos saber quienes son repetidores, realizaremos una intersección con el archivo de alumnos matriculados en primer curse el año o años anteriores.

Fusión o mezcla

A partir de dos archivos de igual estructura clasificados por un mismo campo, se obtiene como resultado un archivo que contiene los registros de ambos y que mantiene la ordenación.

La fusión o mezcla de archivos no clasificados consiste en la concatenación de los mismos.

Partición

Consiste en descomponer un archivo en dos, atendiendo a alguna característica de sus registros.

Por ejemplo, podemos realizar una partición del archivo de alumnos en dos, según el valor del campo EDAD; el primero contendrá los mayores y el segundo los menores de una determinada edad.

Actualización

Es la operación de modificar un archivo de situación por medio de un archivo de movimientos, conteniendo altas, bajas y modificaciones que hay que realizar sobre el archivo maestro para ponerlo al día.

Reorganización

La operación de reorganización consiste en reubicar los registros de un archivo que ha sufrido actualizaciones, de tal manera que se ocupen los posibles huecos libres intermedios, resultantes de bajas de registros, para optimizar la ocupación de la memoria, liberando la que no estaba aprovechada.

Borrado

Eliminación total del archivo cuando ya no se necesite, dejando libre el espacio de memoria que ocupaba en el soporte utilizado.

ARCHIVOS EN TURBO PASCAL

Todos los archivos de Turbo Pascal, con independencia de su tipo, comparten características comunes.

- Todos los archivos se utilizan como entrada y/o salida
- El sistema operativo almacena y accede a archivos, genera directorios, copia, renombra y transfiere archivos entre discos, memoria y dispositivos de E/S (modem, impresora...).
- Los archivos se pueden almacenar en disquetes y unidades de discos duros. Existen otros dispositivos de almacenamiento (cinta, discos ópticos, discos RAM, etc.), aunque son menos frecuentes.

Los archivos de texto estándar de Turbo Pascal tienen la extensión **.TXT**; por consiguiente, es casi norma el uso de la extensión **.DAT** para aquellos archivos creados por Ud. O sus programas.

Tipos de archivos

Existen tres tipos de archivos de datos en Turbo Pascal:

- **texto** (text) o **secuenciales** (acceso secuencial), son archivos que contienen texto (carácter ASCII)
- **tipeados** (tipificados) o con tipo (file of) (acceso aleatorio), **aleatorios**, archivos que contienen datos de cualquier tipo como integer, byte, real, record... (datos con estructuras y contenidos conocidos)
- **no tipeados** (no tipificados) o sin tipo (file). archivos en los que no se conoce su estructura ni su contenido; están concebidos para acceso de bajo nivel a los datos de un disco (E/S de bytes).

LOS ARCHIVOS DE TEXTO (SECUENCIALES)

Un archivo de texto es un tipo estándar, están constituidos por caracteres del código ASCII. Un archivo de texto consta de una serie de líneas separadas por una marca fin de línea (eoln, ”end of line”). La marca fin de línea es una secuencia de caracteres CR (carriage return) y LF (line feed), que se conoce como retorno de carro y avance de línea. La combinación CR/LF (códigos ASCII 10 y 13) se conoce como delimitador y se obtiene pulsando la tecla Intro (Enter o Return), o bien las combinaciones de teclas CTRL-M, CTRL-J.

Declaración de archivos

Para declarar un archivo como en otros casos se declara primero el tipo y luego una variable asociado al tipo.

```
Type
    T_Archivo = file of char;  {* archivo de texto *}
    T_Texto = file of text;

Var
    Archivo : T_Texto;
```

PROCEDIMIENTOS Y FUNCIONES DE TRATAMIENTO DE ARCHIVOS.

Para todos los tipos de archivos	**Solo para archivos de texto**
Procedimientos	
Assign	Append
ChDir	Flush
Close Erase	Read Readln
GetDir	SetTexBuf
MkDir	Write
Rename	Writeln
Reset	
Rewrite	
RmDir	
Funciones	
Eof	Eoln
IOResult	SeekEof
	SeekEoln

Ejemplo de cómo crear un archivo de texto

Un archivo de texto está constituido por una serie de líneas de caracteres separados por CR/LF. Esta combinación se obtiene, cuando se realiza una pulsación de la tecla Intro. (↵)

Esto es una prueba de un archivo de texto ↵
Cada línea en un archivo de texto finaliza con CR/LF, ↵
es decir un Retorno de Carro y un Avance de Línea ↵
(línea vacía) ↵
45671.45 es la cantidad total ↵

Los archivos de texto se terminan con una marca final de archivo Ctrl-Z (eof, end of file). Cuando trabajamos con ellos en Pascal es obligatorio declararlos.
Los archivos de texto se pueden crear con el editor del sistema operativo WordPad o con un programa de edición de texto(Word, WordStar, etc.) en estos casos en el momento de grabarlos (Guardar Como) se elegirá la opción *Sólo Texto.* También se puede escribir un archivo utilizando el editor de Pascal.

Funciones eoln//eof

La función eoln devuelve el estado de fin de línea de un archivo. Es una función de tipo lógico.

Eoln (f) devuelve true si en la posición actual del archivo (puntero) está la marca fin de línea, o bien si eof (f) es true; en caso contrario devuelve false.

Ejemplo de uso

```
While not eoln(f)
```

Donde f es el alias del archivo.

```
WriteLn  (eoln);
```

La función eof (end of file), fin de archivo, devuelve el estado de un archivo de texto. Es una función de tipo lógico que indica si el fin del archivo se ha almacenado; devuelve true si se ha almacenado, false en caso contrario.

Ejemplo de uso

```
While not eof(f)
```

Archivos tipo char

Hay muy pocas diferencias entre un archivo de texto y un archivo de caracteres (char). La única diferencia es que un **archivo de texto se divide en líneas** y **un archivo de caracteres no**. Los archivos de caracteres se leen y escriben carácter a carácter, mientras que los archivos de texto se leen línea a línea.

La declaración es mediante las palabras:

```
file of char;
```

PROCESAMIENTO DE ARCHIVOS DE TEXTO

El tratamiento de archivos de cualquier archivo incluyendo los de texto, nos exige de los siguientes pasos:

- *Declarar el archivo.*
- *Abrir el archivo.*
- *Leer o escribir datos de él, o en él.*
- *Cerrar el archivo.*

Declaración de un archivo

Para declarar un archivo debe realizarse los siguientes pasos:

- ✓ *Declara una variable de tipo archivo TEXT.*
- ✓ *Asociar a esta variable el nombre de un archivo en disco (sentencia assign).*

Ejemplo de uso

```
Type
    Tipo_Archivo = file of char;

Var
    Mi_Archivo : Tipo_Archivo;
    Archivito : text;
```

Como cualquier otra variable, una variable de tipo TEXT se puede definir local o globalmente; pero al contrario que otras estructuras, la longitud de una variable archivo no se precisa.

Asignación de archivos

La operación de asignar un archivo establece una correspondencia entre variable tipo archivo con un archivo externo situado en un disco. Mirándolo desde otro punto de vista, será asignarle un alias por el cual llamaremos al archivo en cuestión.

Ejemplos

```
Begin
    ..............
    ..............
    assign (Mi_Archivo, 'Archivo1.txt');
```

o

```
Write ('ingrese el nombre del archivo a usar');
ReadLn (Nombre);
assign (Archivito, Nombre);
```

o

```
assign (Archivito, 'C:\Algoritmos\practica\Archivo1.txt');
```

Después que se ha asignado una referencia a un archivo, el mismo esta listo en el caso de los de texto para alguno de estos tres procedimientos:

reset: abrir para leer, si se llama a un archivo no existente el programa aborta por error E/S.

rewrite: crear un archivo nuevo, si existiera otro con ese nombre lo borra

append: agrega información a un archivo existente, si no existe produce un error de E/S.

La forma de en que se utilizan estos procedimientos es la siguiente. **Siempre van después de la asignación**

Ejemplos

```
reset (Mi_Archivo);
```

o

```
rewrite (Archivito);
```

o

```
append (Archivito);
```

Pero cómo hacemos para que la sentencias reset no produzca un error si el archivo al cual hago referencia no existe. O cómo evito sobreescribir sobre un archivo existente (el cual guarda datos que aún necesito, cuando ejecuto rewrite. Para esto casos se debe dar la directiva al compilador **{ $I+ }**. Con esta se desactiva la detección de errores de Entrada/Salida y así se previene una parada en la ejecución del programa ante la eventualidad de la ausencia de un archivo. Una posible solución es la siguiente:

```
{ $I- )
    reset (Archivito)
{ $I+ )
if IOResult = 0 then
        {* el archivo existe  *)
    else
        { * el archivo NO existe  *}
```

La función IOResult devuelve el valor 0 si la sentencia reset se ha ejecutado correctamente. El correcto uso de la misma evitará los problemas ya descriptos.

Escritura de un archivo

Una vez que se ha abierto un archivo para escritura, los procedimientos a utilizar son: write y writeln sirven para escribir datos en el nuevo archivo.

Ejemplos

```
Write (Archivito, 'Esto es lo que quedará grabado en el archivo');
WriteLn (Archivito, 'y esto también quedará grabado');
```

o

```
var
    Texto : string [45];
    Archivito : text;

    Write (Archivito,Texto);
```

Lectura de un archivo

Los procedimientos **read** y **readln** se utilizan para la lectura de los datos situados en un archivo de tipo texto.

Ejemplo

```
var
    Total, Horas : real;
    Nombre : string [30];
    ..................................
    ReadLn (Archivito, Nombre, Horas,Total);
```

Las variables que usan los procedimientos read y readln pueden ser char, integer, real o string

Añadir datos a un archivo de textos

El procedimiento Append abre un archivo existente para añadir datos al final del mismo. Como ya se ha dicho, si el archivo no existe, se produce un error; y si ya estaba abierto, primero se cierra y luego se reabre.

Ejemplo

```
program Agregar;

uses
    Crt, dos;

var
    Fichero : Text;

begin
    Assign (Fichero,'Archivo.TXT');
    Rewrite (Fichero);              {* crea nuevo Fichero *)
    WriteLn (Fichero, 'Primera línea');
    Close (Fichero);        {* cierra archivo, guarda cambios *}
    Append (Fichero);    (* Abre para añadir texto *)
    WriteLn (Fichero, ' esta línea es agregada al texto original');
    Close (Fichero)
end.
```

REDIRECCIONAR LAS ENTRADAS/SALIDAS

Redireccionar es cambiar la dirección de salida o entrada del dato. La dirección normal de salida es la pantalla, cuando redirecciono puedo hacer que en lugar de salir el archivo por pantalla salga por impresora o se agregue a otro archivo. Con el sistema operativo D.O.S. esto era muy común, pero con la aparición de Windows en sus distintas versiones ha tendido a desaparecer su uso. En Turbo Pascal

se puede efectuar una redirección de la entrada estándar (el teclado) y de la salida estándar (la pantalla), y lograr con esto que nuestros programas sean más flexibles.

Ejemplo

(Este ejemplo ha sido sacado del Libro de Joyanes Aguilar (Pag 506).

Lee el contenido de un archivo y escribirlo en otro segundo archivo, carácter a carácter.

```
program Copia;

Uses
    crt;

var
    Fichero1, Fichero2 : Text ;
    Caracter : char;

begin

    ClrScr;
    Assign (Fichero1, '');
    Rewrite (Fichero1);
    Assign (Fichero2, ' ');
    Reset (Fichero2);

    while not Eof (Fichero2) do
        begin
            Read (Fichero2, Car);
            Write (Fichero1, Car)
        end;

    Close (Fichero1);
    Close (Fichero2)

end.
```

Los pasos a dar para la ejecución de este programa son los siguientes.

a) Compilemos este programa con el nombre Copia y obtendremos Copia.EXE.

b) Desde el prompt (>) del DOS ejecute la orden **Copia ¿**

Ahora la pantalla esta limpia. Ingrese un texto (un renglón o menos) por teclado, cuando presione Enter, el texto volverá a ser escrito en la pantalla. Hasta que presione <Ctrl – Z>. Algunas aplicaciones de la redirección (símbolo >) de este programa son:

copia > lpt1	es igual que el anterior en lugar de escribir en pantalla lo hace por impresora
copia < muestro.txt	se ve por pantalla el contenido del archivo
copia entrada.txt < > salida.txt	copia el contenido del archivo entrada.txt en el archivo salida.txt
copia entrada.txt < > lpt1	copia el contenido del archivo entrada.txt en la impresora, en otras palabras imprime el archivo entrada.txt.

*Este programa funciona bien con DOS y Windows 3.1. **NO** se aconseja su uso con Windows 95 o posteriores por que se puede "Colgar" el sistema operativo.*

Algunas funciones no vistas

GetDir (i,S): Busca el directorio actual. Si i es 0 la unidad es la actual, para 1 es la unidad A, 2 es B, 3 es C. S es la variable de tipo string, que contiene el directorio actual.

ChDir(S(i)) :Cambia de directorio al indicado en S. i es la unidad y debe indicarse si el directorio pertenece a una unidad distinta de la actual.

MkDir (S(i)): Crea el directorio S en la posición actualen ese caso i no va, sino debe ser indicada la unidad y el camino.

RmDir (S): Elimina el directorio S.

Erase (F): Elimina el archivo F

Rename (F1,F2): cambia el nombre del archivo F1 por F2.

SetTexBuf (Archivo, Buf): asigna un buffer a un archivo de texto.

Flush (Archivo): vacía el buffer al archivo en el disco.

Todos estos comandos pueden dar errores de E/S por lo tanto es conveniente utilizar la directiva I para determinar la existencia o no de alguno de estos elementos.

El siguiente programa es un programa que copia carácter a carácter.

```
program Copia;

uses
    crt, dos;

var
    Fichero1, Fichero2: Text;
    Caracter : char;

begin
    ClrScr;
    Assign (Fichero1, 'c:\salida.txt');
    Rewrite (Fichero1);
    Assign (Fichero2, 'c:\entrada.txt');
    Reset (Fichero2);

    while not Eoln (Fichero2) do
        begin

            while not Eof (Fichero2) do
                begin
                    Read (Fichero2, Caracter);
                    Write (Fichero1, Caracter)
                end;

            readln(Fichero2);   {*  se usa para saltar el carácter de fin de línea  *}
            writeln(Fichero1)   {*  se usa para insertar el carácter de fin de línea  *}
        end;

    Close (Fichero1);
    Close (Fichero2)
```

end.

Ejercitación

1. Crear un programa que maneje un archivo donde se almacena el Apellido de una persona, su edad, su estatura en metros (Ej: 1,75 m). El programa deberá almacenar y listar los datos almacenados.
2. Modifique el programa anterior, de forma tal, que pueda listar los datos de una persona solicitada por apellido. Si no existiera, debe mandar un mensaje de que no existe, y listar los apellidos existentes ordenados alfabéticamente.
3. Realizar las modificaciones necesarias para que se pueda realizar modificaciones en los datos grabados en el archivo.
4. Crear un programa que solo permita generar un archivo de texto, donde haya como máximo 80 caracteres por renglón. Pero deberá evitarse cortar palabras.
5. Crear un programa que permita concatenar dos archivos de texto.
6. Crear un programa que dados dos archivos de texto, que contienen números enteros, logre realozar la intersección entre ellos.
7. Dado un archivo que contienen una serie de números, obtener un archivo ordenado.
8. Dado un archivo que contienen una serie de números, obtener a partir de él dos archivos. En el primero solo habrá números pares, y en el segundo solo impares.
9. Dado un archivo de texto, hacer un programa que logre reemplazar cada letra minúscula por otra minúscula pero cuyo código ASCII sea igual al de la primitiva más 6. Cuando el código de un valor que no corresponda a una minúscula, el programa restará al código 26, para obtener el nuevo código. Cada letra mayúscula será reemplazada por otra mayúscula siguiendo la misma regla que en el caso anterior. Los demás signos no serán reemplazados.
10. Dado un archivo de texto, de varios renglones escribir un programa que tenga como salida otro archivo donde los renglones estén escritos en orden inverso:

    Por ejemplo:

    Entrada

    Los días de octubre son hermosos.
    Los niños juegan en a plaza.
    Las madres disfrutan de esos momentos.

    Entrada

    Las madres disfrutan de esos momentos.
    Los niños juegan en a plaza.
    Los días de octubre son hermosos

11. Dado un archivo de texto, de un renglón. Escribir un programa que tenga como salida otro archivo donde el renglón esté escrito en orden inverso:

    Por ejemplo:

    Entrada

    Los días de octubre son hermosos

    Entrada

    sosomreh nos erbutco ed saíd soL

12. Dado un archivo de texto, de varios renglones escribir un programa que tenga como salida otro archivo donde los renglones estén escritos en orden inverso y que cada renglón esta escrito en sentido inverso.

13. Modificar el programa anterior para que no solo haga lo antes expresado. Sino que además, pueda cambiar las letras como se pide en el problema 9.