CAPITULO 6.- ARCHIVOS

Los archivos de pascal, independientemente de su tipo, tienen características comunes. A priori, todos los archivos se utilizan tanto para entrada como para salida de datos. Se le llama "entrada" a la operación que ejecuta un programa cuando toma datos de un archivo y los utiliza en sus cálculos; se le llama "salida" cuando los resultados del proceso se envían a un archivo.

Los archivos pueden ser grabados en discos flexibles o en discos duros. También hay otros medios de almacenamiento: cintas, discos ópticos, discos en RAM, etc...; pero son menos comunes.

El DOS requiere que los nombres de los archivos tengan de 1 a 8 caracteres; puede incluirse una extensión de 1 a 3 caracteres, que sirve para ayudar a describir el contenido del archivo.

ARCHIVOS DE TEXTO

Los archivos de texto constan de líneas terminadas por un retorno de carro (carriage return), de una línea (linefeed CR/LF) y de los caracteres que contiene (palabras, oraciones, números,etc.).

Nombre del archivo: Texto.dat

Línea 1.- Este es un ejemplo de texto. [CR/LF]
Línea 2.- Cada línea de texto termina con [CR/LF].
Línea 3.- Retorno de carro y un alimentador de línea.[CR/LF]
Línea 4.- [CR/LF]
Línea 5.- Aun las líneas vacías, como la anterior. [CR/LF]
Línea 6.- 50 12.23 40343 332324 [CR/LF]

Antes de empezar a trabajar con archivos de texto, se debe declarar un IDENTIFICADOR DE ARCHIVO DE TEXTO en el programa. Los identificadores para los archivos de texto se declaran como cualquier identificador, utilizando la palabra reservada en Turbo Pascal TEXT. Ejemplo:

```
Program ArchivoTexto;

var

  Arch : Text;
```

El identificador del archivo es Arch (de tipo Text). Antes de utilizar el archivo para Entrada o Salida es necesario asignar la variable a un nombre de archivo en el disco; a esta operación se le llama asignación:

```
assign (Arch, 'TEXTO.DAT');
```

Una vez que el archivo se asignó, no vuelve a ser referenciado por su nomber otra vez.

Después de asignar el nombre del archivo, es necesario prepararlo para la operación que se desa hacer con él : RESET, REWRITE o APPEND.

Reset "abre" el archivo y lo prepara para lectura; solo se pueden utilizar comandos de entrada cuando Reset se utiliza. Cualquier intento de escriturá generará un error de I/O.

El comando Reset posiciona al inicio el apuntador de archivo, (el cual es un contador que mantiene la posición del programa en el archivo). Esto hace que las operaciones se empiecen desde el inicio del archivo.

Intentar aplicar Reset a un archivo que no existe genera un error de I/O.

Rewrite y Append preparan el archivo para salida (escritura), pero funcionan de diferente manera. Cuando se prepara un archivo ya existente con Rewrite, se borra su contenido y el apuntador de archivo se posiiona al inicio de este. Si Rewrite se utiliza con un archivo que no existe, se crea uno nuevo con el nombre utilizado en el comando Assign.

Por otro lado, el comando Append preserva el contenido del archivo y posiciona el apuntador de archivo al final de éste. Como resultado, cualquier dato añadido al archivo se escribe al final, junto con lo que ya había.

Cuando se termina de utilizar el archivo, es necesario "cerrarlo". El comando Close cierra el archivo y se asegura que todos los datos almacenados en el buffer temporal se almacenen en el disco (a esto se le conoce como FLUSH).

Una vez cerrado, un archivo no puede utilizarse ni para entrada, ni para salida, hasta que sea abierto con Reset, Rewrite o Append. Sin embargo, el encadenamiento con el sistema operativo permanece, de tal forma que para volver a abrir el archivo no es necesario utilizar el comando Assign.

LEYENDO UN STRING DE ARCHIVOS DE TEXTO

Una vez que un archivo de texto se abrió para lectura, se puede extraer información de él con los procedimientos Read y Readln;

```
Program Text1;
Var
      ArchTxt      : Text;
      [S]          :string[10];
begin
      assign (ArchTxt,'TEST.DAT');
      reset (ArchTxt);
      readln (ArchTxt,S);
      writeln(S);
      close (ArchTxt);
end.
```

ArchTxt se prepara para lectura con el comando Reset. La primera operación de entrada en el programa es el estatuto.

```
ReadIn(ArchTxt,S);
```

Este le dice a Turbo Pascal que lea los caracteres de la línea actual en el archivo y los coloque en la variable S. Después de que los caracteres sean leídos, el apuntador de archivo brinca los caracteres remanentes de la línea y va hacia el inicio de la siguiente.

Cuando se lee un string de un archivo de texto con el procedimiento ReadIn, pueden ocurrir tres cosas:

1) Que haya en la línea exactamente los caracteres necesarios para llenar la variable en toda su longitud.

2) Que no haya suficientes caracteres en la línea para llenar la variable de lectura.

3) Que haya demasiados caracteres en la línea para la longitud de la variable de lectura.

En los primeros dos casos, Turbo Pascal lee todos los caracteres de la línea, asignándolos a S y moviendo el apuntador al inicio de la siguiente línea. La longitud del string S es igual al número de caracteres leído.

En el tercer caso, Turbo Pascal lee tantos caracteres como sea necesario para llenar la variable de lectura completamente, y después mueve el apuntador del archivo a la siguiente línea. Todos los caracteres sobrantes en la línea son descartados.

El procedimiento Read opera como el Readln, pero después de leer al string, Read coloca el apuntador del archivo enseguida del último caracter leído; no mueve el apuntador al inicio de la siguiente línea. Si el procedimiento Read encuentra un [CR/LF], indicando que se encontró al final de la línea, detiene la lectura de caracteres y no avanza el apuntador de archivo hasta que se utilice un procedimiento Readln.

LEYENDO VARIOS STRINGS POR LINEA

Un procedimiento Readln sencillo puede leer varios strings a la vez. Por ejemplo, el estatuto readln (ArchTxt,S1,S2,S3) lee caracteres de la línea actual, llenando las variables S1, S2 y S3, en ese orden. Si la línea a leer es:

" Esta es una línea de caracteres "

y las variables srtings son todas del tipo string[5], los strings asignados serían:

[Esta]	[es un]	[a lín]	[ea de caracteres]
S1	S2	S3	sin usar
S1	S2	S3	sin usar

LEYENDO NUMEROS DE LOS ARCHIVOS DE TEXTO

Los archivos de texto no solamente pueden almacenar strings o sentencias, sino también datos numéricos. Los números, sin embargo, no se almacenan en el archivo en forma binaria, sino como caracteres. Por ejemplo, en RAM el valor entero 20,545 es almacenado en 2 bytes con el valor binario 0101 0000 0100 0001. Pero en un archivo de texto, el número se almacena con los caracteres "2", "0", "5", "4", "5", requiriendo un total de 5 bytes. Cuando se lee un número de un archivo de texto, Turbo Pascal convierte el número de un string de caracteres a formato binario de enteros.

Asimismo, cuando se lee un número de un archivo de texto, Turbo Pascal brinca los caracteres de espacio de la línea hasta que encuentra un caracter no-espacio. Entonces lee los caracteres siguientes hasta que encuentra un caracter de espacio otra vez. De esta manera sigue leyendo hasta que encuentra un [CR/LF]. Turbo Pascal combina los caracteres leídos en un string que convierte a un valor entero o real, dependiendo del tipo de variable que se use. Si la conversión tiene éxito, el número resultante se asigna a la variable; si no, se genera un error de I/O.

Para observar en detalle cómo leer datos numéricos de un archivo de texto, examinemos el archivo de texto TEST.DAT.

11	27.53	6.4144900000E+02
21	50.83	1.1843390000E+03
31	74.13	1.2547890000E+03
41	97.43	2.1259870000E+03
51	120.73	2.8139002400E+03
61	144.03	3.8654741700E+03
71	167.33	4.4412450000E+03
81	190.63	4.9852140000E+03
91	213.93	5.0214578825E+03
101	273.23	5.5274590000E+03

Este archivo contiene 3 columnas de números. La primera columna es de enteros, la segunda de reales en formato decimal y la tercera de reales en notación científica. Se pueden leer 3 números por línea, empleando los siguientes estatutos:

```
Read  (ArchTxt,I2);
Read  (ArchTxt,R1);
Read  (ArchTxt,R2);
```

o bien usando el estatuto equivalente:

```
Readln (ArchTxt,I2,R1,R2);
```

Turbo Pascal asigna el primer número encontrado en la línea a la variable entera I2; y los siguientes 2 números a las variables reales R1 y R2.

El siguiente programa contiene una rutina que lee los datos numéricos contenidos en el archivo de texto TEST.DAT, y calcula el promedio de cada columna.

```
program CalculaPromedios;
var:
      f              :Text;
      i, count       : integer;
      imean,
      r1, r2,
      r1mean,
      r2mean         : real;
begin
      assign (f,'TEST.DAT');
      reset (f);
      count          := 0;
      imean          := 0;
      r1mean         := 0;
      r2mean         := 0;
      while not eof(f) dp
      begin
            readln (f,i, r1, r2);
            writeln (i:10,' ',r1:10:3,' ',r2:10:3);
            count          := count + 1;
```

```
        imean      := imean + i;
        r1mean     := r1mean + r1;
        r2mean     := r2mean + r2;
    end;
    imean := imean / count;
    r1mean := r1mean / count;
    r2mean := r2mean / count;
    writeln;
    writeln (imean:10:3,' ', r1mean:10:3,' ',  r2mean:10:3);
    close (f);
end.
```

Si el formato de un número leído de un archivo de texto es incorrecto, el programa produce un error de I/O.

Por ejemplo, leer el número 50,000 con una variable del tipo INTEGER causa error, ya que el número más grande permitido para ese tipo es de 32,767. Similarmente, leer el número 32.1 en una variable de tipo INTEGER causa error por el punto decimal (es ilegal para el tipo Integer). El tipo REAL tiene menos restricciones, ya que estos números pueden ser leídos con o sin punto decimal y en notación normal o científica.

APENDICE A.-REFERENCIA COMPLETA DE FUNCIONES

La definición de ISO Standar Pascal incluye un número de "funciones estándar" construidas dentro del lenguaje, y no necesitan ser declaradas ni codificadas dentro del programa fuente. Esas funciones caen dentro de dos grupos básicos: 1) Las funciones matemáticas, las cuales proveen operaciones fundamentales, como la raíz cuadrada, el valor absoluto, los logaritmos naturales, las funciones trigonométricas; y 2) las funciones de transferencia, las cuales permiten la relación entre tipos incompatibles, como enteros y de caracter.

Muchos libros refieren los valores pasados a una función estándar como "argumentos". Este término es común dentro del argot matemático, por lo que puede ser confundido con el término parámetro. No hay diferencia mas que de término.

Turbo Pascal implementa todas las funciones del ISO Standar Pascal. A continuación se muestra un glosario completo. También se han añadido las propias del IBM PC.