****

***Laboratorio 8***

***Archivos***

**Objetivos:**

* Conocer el funcionamiento de los archivos.
* Diferenciar los diversos tipos de archivos.
* Aprender el tratamiento de archivos Secuenciales y archivo Aleatorios
* Resolver problemas básicos haciendo uso de archivos.

**Pre Laboratorio**

* **Archivos.**

Un archivo (file) es una colección de información almacenada como una unidad en alguna parte de la computadora. Esta colección de datos sirve para entrada y salida a la computadora y se manejan con un programa. Hay que tener presente que los elementos (datos ) 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.

* **Tipos de archivos.**

Existen varias formas de clasificar los archivos, alguna tipificación de archivos los clasifica en :

* Archivo de entrada ( o solo lectura) : una colección de datos localizados en un dispositivo de entrada.
* Archivos de salida ( o solo escritura) : una colección de información que se presenta en un dispositivo de salida o un dispositivo de escritura.
* Archivos 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 dispositivo de almacenamiento.
* **Acceso a un archivo**

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

***El acceso secuencial,***exige acceder elemento a elemento, en este caso es necesario recorrer secuencialmente el archivo desde el primer elemento, hasta llegar al elemento deseado.

***El acceso directo****,* permite procesar o acceder a un elemento determinado, referenciado directamente su posición en el soporte de almacenamiento., en este caso no es necesario leer los elementos que les anteceden.

Pascal estándar solo admite el acceso secuencial, pero Turbo Pascal admite el acceso directo.

**Manejo de archivos en Turbo Pascal.**

* **Nombre de archivos.**

Un archivo tiene un nombre que lo identifica y está almacenado en un directorio junto a otros archivos de disco. Consta de un nombre y una extensión.

Son ejemplos de nombre de archivos : Demo.Pas , Prueba. Pas , Estudiantes.DAT

Otro ejemplo, de un archivo EMPLEADOS.DAT que está en el directorio TURBO, que a su vez contiene un directorio LIB., es :

C:\TURBO\LIB\EMPLEADOS.DAT

* **Tipos de archivos.**

Existen tres tipos de archivos de datos en Turbo Pascal :

* Archivos texto (txt) o secuenciales ( acceso secuencial) : son archivos que contienen texto ( caracteres ASCII)
* Tipeados (tipificados) o con tipo ( file of) : son archivos de acceso aleatorio o directo : son archivos que contienen datos de cualquier tipo de los permitidos en Pascal, como integer, byte, real, record..es decir datos con estructuras y contenidos conocidos.
* No tipeados (no tipificados) o sin tipo (file). Son archivos en los que no se conoce su estructura ni su contenidos; están concebidos para acceso de bajo nivel a los datos de un disco.
* **Declaración de archivos :**

La declaración de archivos consta de dos pasos :

**Paso 1 :** Declaración del tipo de archivo deseado, que puede ser

**file of char o file of text** para declarar un archivo de texto

**file of <tipo >** para declarar un archivo tipeado o tipificado

**file** para declarar un archivo sin tipo**.**

**Paso 2 :** Declaración de una variable archivo de un tipo de archivo declarado

La declaración de un tipo archivo (file), se hace de igual modo que cualquier otro tipo de dato definidos por el usuario en la sección de declaración de tipos ( type), usando el formato **:**

|  |
| --- |
| **type**  nombre = **file of** tipo de datos |

Donde *nombre* es el identificador que se le asigna al archivo

*tipo de datos* : es el tipo de datos de los elementos del archivo

Ejemplos :

**type**

ArchEstudiantes = **file of** string[20] ; tipo de archivo de cadenas

**type**

ArchEnteros = **file of** integer ; tipo de archivo de enteros

**Variables tipo archivo ( file)**

Para definir un archivo con tipos, simplemente declare una variable archivo.

Ejemplo :

var

Cantidades : ArchEnteros;

var

cantidades, Notas, Salarios : ArcEnteros;

o también se pueden declarar directamente variables del tipo file,.

Var

Archiprueba1 : file of real;

Archiprueba2 : fije of integer;

* **Procedimientos y funciones estándar para la gestión de archivos :**

**Procedimientos**

|  |  |  |
| --- | --- | --- |
| **Todos tipos de archivos** | **Archivos de textos** | **Archivos sin tipo** |
| **Assign**  **ChDir**  **Close**  **Erase**  **getDir**  **MkDir**  **Rename**  **Reset**  **Rewrite**  **RmDir** | **Append**  **Flush**  **Read**  **Readln**  **SetTexBuf**  **Write**  **Writeln** | **BlockRead**  **BlockWrite** |

**Funciones**

|  |  |  |
| --- | --- | --- |
| **Todos tipos de archivos** | **Archivos de textos** | **Archivos sin tipo** |
| **Eof**  **IOResult** | **Eoln**  **SeekEof**  **SeekEoln** |  |

**Los archivos de texto ( Archivos Secuenciales)**

Los archivos de textos están constituidos por elementos que son caracteres Pascal , pertenecientes al código ASCII. Este consta de una serie de líneas ( cada una contiene caracteres, palabras y frases) y separadas por una marca de fin de línea ( eoln, end of line)., que constituyen una secuencia de caracteres CR y LF (códigos ASCII 10 y 13).

Los archivos de textos también se **pueden crear con un editor de texto** que proveea el sistema operativo, del mismo modo que se crea un programa fuente en Pascal. Para ello se introducen los caracteres individuales por teclado y la marca fin de línea se inserta en el archivo al pulsar la tecla Enter.

Los archivos de texto son casi similares a los archivos de caracteres (tipo char), la 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 de carácter en carácter, mientras que los archivos de texto se leen línea a línea. Su declaración se hace mediante las palabras : **file of char.**

**Tratamiento de archivos de texto.**

El proceso o tratamiento de archivos de texto exige los siguientes pasos:

1.- Declaración del archivo.

2.- Apertura del archivo.

3.- Procesamiento del archivo ( Leer los datos del archivo o escribir datos en el ).

4.- Cierre del archivo.

* **Declaración de un archivo :**

Esta se realiza a través de dos operaciones:

1. Definir una variable de tipo archivo TEXT. A través del formato general

|  |  |
| --- | --- |
| **var** nombreVar : **text** ; | **var**  NombreVar: **file of** char; |

1. Asociar a esta variable el nombre de un archivo en disco (Asignación de archivos con la sentencia ASSIGN)

La operación de asignar un archivo , establece una correspondencia entre variable tipo archivo con un archivo externo situado en un disco.

Esto se hace a través del formato :

|  |  |
| --- | --- |
| Assign ( var f: nombre; string) | Assign ( f, nombre) |

Donde ” f” es el nombre interno del archivo con el que se conoce el archivo dentro del programa y “nombre”, es el nombre externo con el que se almacena el archivo en el dispositivo físico de almacenamiento. Ejemplo: b:\mis documentos\notas.txt.

Ejemplos :

1) Const

NombreAr = ´Estudiantes.txt´;

Var

Miarch : text;

Begin

Assign ( Miarch, NombreAr);

2) var

Fichtext : text;

…

Assign ( fichtext, ´c:\misdocumentos\turbo\estudiante,pas´;)

3) var

Datos : text;

Nombre : string [20];

…

Write (´escriba el nombre del archivo en el dispositivo físico´);

readLn (Nombre);

assign (Datos, Nombre);

…

Después de haber asignado un identificador a un archivo que se encuentra en un dispositivo físico, se procede a abrir el archivo.

* **Apertura de archivo :**

Esta operación se realiza con REWRITE O RESET

RESET : Abre un nuevo archivo existente para una operación de lectura. Si se intenta utilizar RESET y el archivo especificado no existe, se producirá un error.

|  |  |
| --- | --- |
| Formato | Ejemplo |
| reset ( nombre del archivo) | reset (MiArch);  reset (fichtext); |

REWRITE : crea y abre un nuevo archivo. Si el archivo ya existe, REWRITE borra su contenido; en caso contrario, el archivo queda abierto para una operación de escritura.

Se debe tener en consideración las siguientes precauciones :

* Si al abrir un archivo de texto, con **assign y reset**, ya existe en el disco, la sentencia rewrite lo rescribirá, es decir “ se perderá” el archivo inicial.
* Por el contrario, las sentencias assign y rewrite supone la existencia del archivo llamado en el disco. Si este archivo no existe, las sentencias anteriores producirán un error de ejecución.
* **Escritura de un archivo :**

Una vez que se haya abierto un archivo para escritura, las sentencias WRITE Y WRITELN se utilizan para escribir datos en el nuevo archivo, a través de los siguientes formatos :

|  |  |
| --- | --- |
| Write ( f, v1, v2,….) | Writeln (f1, v1, ve,…vn) |

Donde :

f : es una variable tipo archivo

V1, v2…..variables del tipo de datos.

Ejemplos :

1.- Write (Aarchdemo, ´esto es una prueba de escritura´):

WriteLn ( Archdemo, ´y esta es la siguiente prueba´);

2.- **Var**

NombreArch : string [60];

Archtex : text;

. . .

Write (´nombre de archivo´);

Readln (NombreArch);

Assign ( Archtex, NombreArch);

Reset (Archtex)

NOTA :

El uso del Write y writeln en archivos de textos es exactamente el mismo que en la pantalla. Si el primer argumento de write o writeln no es un archivo, se utiliza output por defecto.

Write (x) es equivalente a Write ( output, x )

* **Lectura de un archivo :**

Para la lectura de los datos contenidos en un archivo tipo texto, se utilizan las sentencias READ y READLN., a través del siguiente formato:

|  |  |
| --- | --- |
| Read (f, v1, v2…..) | Readln (f1, v1, v2,…vn) |

Donde :

f: son variable archive de texto ( si se omite, se supone la variable de archivo input. Ejemplo Read (x) equivale a Read ( input,x)).

v1,v2… son variables de tipo char, integer, real o string.

Ejemplo:

**Var**

Total, Horas : real;

Archivo : text;

Mensaje :string[30];

Begin

Assign (Archivo, c:\mis’documentos \prueba´);;

Reset (Archivo);

. . .

ReadLn (Archivo, Mensaje, horas);

* **Cierre de un archivo :**

El cierre de un archivo texto se efectúa a través de la sentencia CLOSE .

|  |
| --- |
| Close ( nombre del archivo) |

**Otras funciones y procedimientos de interés para el manejo de archivos :**

* **Función eoln ( end of line) y eof ( end of file).**

La función **eol** devuelve el estado de fin d línea de un archivo. Es una función de tipo lógico, cuyo uso se realiza así :

|  |  |
| --- | --- |
| Función | Llamada a la función |
| Eoln (var f: text) | Eoln (f) |

Donde f: es una variable de archivo de texto.

Eoln (f) devuelve “true” si el puntero del archivo está en la fin de línea o de lo contrario , devuelve “false”.

Por su parte la función eof ( end of file), fin de archivo, también es una función del tipo lógica, que devuelve “true” si se llegó al fin de archivo y “false” en caso contrario.

|  |  |
| --- | --- |
| Función | Llamada a la función |
| Eof (var f: text) | Eof (f) |

* **Procedimiento Append ( para añadir datos a un archivo de texto)**

El procedimiento APPEND abre un archivo existente para añadir datos al final del mismo, su formato es :

|  |
| --- |
| Append (f) |

Donde : (f) es una variable de archivo de texto que debe haber sido asociada con un archivo almacenado en disco.

Si el archivo no existe, se produce un error y si ya estaba abierto, primero se cierra y luego se reabre.

Ejemplo :

Programa Agregar;

Var

Fich : text;

Begin

Assign ( Fich, ´texto.txt´);

| Rewrite ( Fich); {crea nuevo archivo}

WriteLn ( Fich, ´linea primitiva´);

Close (Fich): {cierra archivo, guarda cambios}

Append (Fich); {añade texto al final}

Write (Fich, ´linea añanida´);

Close (Fich)

End.

**Tratamiento de archivos de Acceso Aleatorio**

El proceso de operaciones con archivos con tipo de de acceso ALEATORIO es similar al ya tratado en los archivos de textos ( o secuenciales), con algunas diferencia que se describirán a continuación.

Los pasos necesarias para la creación, lectura y escritura de una archivo aleatorio son :

1.- Definir el tipo de componentes del archivo y declarar una variable de archivo como estructura de tipo file (declaraciones usando TYPE y VAR).

2.- Establecer un enlace entre las variables tipo archivo y el nombre físico del archivo en disco. ( a través del procedimiento ASSING).

3.- Abrir un archivo nuevo o uno ya existente (usando el procedimiento REWRITE o RESET). En cualquier caso, el archivo se abre tanto para lectura como para escritura.

4.- Marcar una posición específica en el archivo para la siguiente operación de lectura o escritura (SEEK). Leer el registro (READ) o escribir un nuevo registro en esa posición ( WRITE).

5.- Cerrar el archivo al terminar las operaciones de lectura/escritura (CLOSE).

* **Declaración de un tipo de archivo**

La declaración de un archivo de acceso directo se efectua con la ayuda de las palabras reservadas FILE OF., como por ejemplo :

**type** cliente = **record**

Código : string [10];

Nombre : sting[40];

Dirección : string[30];

Teléfono : string[10];

Saldo : real;

Edad : integer

**End;**

**Var**

Fich\_cliente **: file of** cliente**;**

Un\_cliente : cliente;

Car : char;

* **Asignación de archivo**

El procedimiento de asignación de archivo es idéntico para el secuencial, ya conocido

|  |
| --- |
| Assign ( nombre\_archivo\_interno, nombre\_archivo\_externo) |

Ejemplo : assign ( fich\_cliente, ´c:\misdocumentos\archivos\cliente.dat´)

* **Apertura del archivo**

Esta operación se efectúa por uno de los procedimientos ya conocidos : RESET o REWRITE

|  |  |
| --- | --- |
| REWRITE (arch\_cliente)  Acá el archivo puede existir | Reset (f)  Acá el archivo debe existir |

Es importante saber que con RESET se necesita que el archivo exista y con REWRITE, se crea y se abre un archivo, si este ya existía su contenido se destruye., por lo tanto debe tener la precaución al usar REWRITE para evitar pérdidas indeseadas de la información.

* **Operaciones de lectura, escritura y fin de archivo.**

Los archivos binarios ( con tipo) no están constituídos por líneas de datos, en consecuencia , los procedimientos estándar READLN, WRITELN y EOLN no se pueden utilizar en archivos de este tipo.

Así, los únicos procedimientos que se pueden utilizar para lectura y escritura en estos archivos, son :

|  |  |
| --- | --- |
| Procedimiento | Formato de uso |
| read | Read ( nombre del archivo, lista de elementos) |
| write | Write (nombre del archivo, lista de elementos) |
| eof | Eof (nombre del archivo) |

En cada operación de lectura y escritura, el puntero asociado al archivo en uso, se incrementa automáticamente de modo que se posiciona en el registro siguiente.

* **Cierre del archivo:**

El cierre de un archivo con tipo ( aleatorio) se efectúa de manera similar para un archivo de texto., a través del siguiente formato :

|  |
| --- |
| Close ( nombre del archivo) |

* **Mantenimiento de archivos aleatorios.**

Las operaciones básicas en el mantenimiento de archivo son :

* Creación
* Acceso
* Consulta
* Actualización
* **Ejemplo de un programa de creación de un archivo aleatorio**

Un programa para crear un archivo de registro de direcciones

Program CrearArch;

Uses

Crt;

Type

Direc = Record

Nombre : string [30];

Calle : string[25];

Ciudad : string[15];

Telefono : string[10]

End;

Archivo = file of Direc;

Var

Dir : Archivo;

Elemento : Direc;

Respuesta : Char;

Begin

Assing ( Dir, ´Agenda.dat´); { asigna Dir a Agenda.Dat}

Rewrite (Dir); {abre archivo Dir}

with Elemento do

Repeat

Write (´ Introduzca el Nombre:´);

Readln (Nombre);

Write (´ Introduzca la Calle:´);

Readln (Calle);

Write (´ Introduzca la Ciudad:´);

Readln (Ciudad);

Write (´ Introduzca el Telefono:´);

Readln (Telefono);

Write (Dir, Elemento); { escritura de un registro en el archivo}

Write (´Desea introducir otro elemento s/n? ´);

Respuesta := Upcase(Readkey)

Until respuesta = ´N´;

Close (Dir);

End.

**LABORATORIO**

1.- Usando archivos de texto (secuenciales), crear las siguientes estructuras de almacenamiento.

Carreras ( CódigoCarrera, DescripciónCarrera) y

Estudiantes (Cédula, Nombre, Sexo, CodCarrera, FechaIngreso)

Para ello, elabore los programas en Pascal que :

* Capture por pantalla los Códigos y descripción de las carreras que se imparten en la UCAB y las almacene en un archivo.
* Capture por pantalla la cédula, el nombre, el sexo, el código de carrera y la fecha de ingreso de los estudiantes y los almacene en un archivo.
* A partir de los archivos creados , Carreras y Estudiantes, generar un listado (archivo de salida) de los estudiantes de una carrera dada por pantalla.

El archivo de salida (reporte) debe mostrar el titulo “Estudiantes por carreras “ y luego por la carrera dada mostrar su código y su descripción y debajo debe mostrar la cédula, nombre , sexo y fecha de ingreso de los estudiantes, asociados a esa carrera.

2.- Hacer el mismo programa usando archivos Aleatorio.

**POST LABORATORIO**

1. Escriba un programa que reciba el nombre físico de un archivo de texto (caracteres) existente y genere un nuevo archivo, pero con los caracteres de cada registro invertidos. Se debe recibir como parámetro el nombre físico del archivo a generar. **(3 ptos)**

2. Escriba una rutina que reciba el nombre físico de dos archivos de números enteros y genere uno nuevo con la concatenación de ambos. Se debe recibir como parámetro el nombre físico del archivo a generar**. (2 ptos)**

3. Se dispone de un archivo de registro, cada uno de los cuales contiene los campos nombre, dirección, edad, fecha de nacimiento, sexo y ciudad, y se pide:

* Crear un nuevo archivo de registro que contenga solo los nombres y edad de los hombres.
* Generar un reporte que contenga el nombre y dirección de las personas cuyos nombres comiencen con las letras A,B y C y hayan nacido en el mes de abril. **( 5 ptos)**

4. Elaborar un programa en Pascal que cree un archivo VACIO de registro de estudiantes de la UCAB con la siguiente información : cédula, nombre, dirección, código postal , promedio de notas, edad y estatus ( A: activo, R: retirado, E.Egresado ). El programa también, debe permitir añadir (insertar), borrar (eliminar), modificar (actualizar) y buscar (consultar) un registro en el archivo creado. Para ello utilice un menú de opciones y procedimientos para cada operación. Considere la cédula del estudiante como campo único, asi como en que caso es conveniente abrir el archivo con REWRITE o con RESET. Utilizar archivos de acceso directo. (**10 ptos).**

**Instrucciones para el envío de la tarea :**

Crear una carpeta comprimida que incluya:

* Un archivo .doc con los algoritmos de los problemas propuesto en el Postlaboratorio y con las capturas de pantalla de las corridas de los programas respectivamente.
* Los archivos .pas de los problemas resuelto (un archivo .pas por cada programa)

Además debe enviar la HOJA DE CONTROL DE PRACTICAS con la autoevaluación de la práctica de la semana.

Adjuntar archivos a la tarea correspondiente en Módulo 7