

Fundamentos de Organización de Datos

Archivos

La Cátedra

Profesores:

Mg. Rodolfo Bertone

Viernes 8.30 hs

Mg. Thomas Pablo

Jueves 14.30 hs

Trabajos Prácticos:

JTP: Lic. Castelli Viviana Lunes

JTP: Lic. Sobrado Ariel

JTP: Lic. Nucilli Emanuel Martes

La Cátedra

- **Clases**

- Teóricas
- Explicaciones de Prácticas (donde se presentan ejemplos)
- Prácticas
- Se utilizará la plataforma Ideas – Curso **Fundamentos de Organización de Datos (FOD)**

- **Para aprobar la cursada**

Examen práctico:

- 1º Fecha Martes 01/06
- 2º Fecha Martes 22/06
- 3º Fecha Martes 13/07



La Cátedra

- Explicación de práctica y teoría asíncronas, se dejará url en ideas de acceso a las mismas.
- Las consultas de teoría virtuales sincrónicas son los días jueves a las 15 hs y los viernes a las 9 hs.
- Consulta práctica con ayudantes, martes 18 horas.
- Evaluación del examen es por temas.

Bibliografía

- Introducción a las Bases de Datos. Conceptos Básicos (Bertone, Thomas)
- Estructuras de Archivos (Folk-Zoellick)
- Files & Databases: An Introduction (Smith-Barnes)
- Fundamentos de Bases de Datos (Korth Silvershatz)

Fundamentos de Organización de Datos

Archivos

Tipos de Archivos

Registros de longitud fija (File of <tipo_dato>)

Texto(Text): Caracteres estructurados en líneas.
Lectura/escritura con conversión automática de tipos.
El acceso es exclusivamente secuencial.
Útiles para importar y exportar datos.

Bloques de bytes (File): Se verá más adelante en el curso.

Acceso a la información

Métodos de acceso

Secuencial: El acceso a cada elemento de datos se realiza luego de haber accedido a su inmediato anterior.

Directo: Se recupera un elemento de datos de un archivo en un solo acceso.

Secuencial indizado: El acceso a los elementos de un archivo se realiza mediante una estructura externa. No tiene en cuenta el orden físico.

Operaciones básicas

Definición de Archivos

Dos formas:

- **var** archivo_logico: **file of** tipo_de_dato;
- **type**
 archivo = **file of** tipo_de_datos;
 var archivo_logico: archivo

Ejemplo

10

type

```
persona = record
    dni: string[8];
    apellido: string[25];
    nombre: string[25];
    direccion: string[25];
    sexo: char;
end;

archivo_enteros = file of integer;
archivo_string = file of string;
archivo_personas = file of persona;
```

var

```
enteros: archivo_enteros;
texto: archivo_string;
personas: archivo_personas;
```

Operaciones

```
assign(nombre_logico, nombre_fisico);
```

Realiza una correspondencia entre el archivo lógico y archivo físico.

Ejemplo:

```
assign(enteros, 'c:\archivos\enteros.dat');
```

```
assign(texto, ' c:\archivos\texto.dat');
```

```
assign(personas, 'c:\archivos\personas.dat');
```

Operaciones

Apertura y creación de archivos

rewrite (nombre_logico) ; → Crea un archivo

reset (nombre_logico) ; → Abre un archivo existente

Ejemplo:

rewrite (enteros) ;

reset (personas) ;

Operaciones

Cierre de archivos

```
close (nombre_logico) ;
```

Transfiere la información del buffer al disco.

Ejemplo:

```
close (enteros) ;
```

```
close (personas) ;
```

Operaciones

Lectura y escritura de archivos

read(nombre_logico, var_dato);

write(nombre_logico, var_dato);

El tipo de dato de la variable `var_dato` es igual al tipo de datos de los elementos del archivo.

Ejemplo:

read(enteros, e); ➡ e : integer;

write(personas, p); ➡ p : persona;

Operaciones adicionales

EOF (nombre_logico) ;

Controla el fin de archivo.

fileSize (nombre_logico) ;

Devuelve el tamaño de un archivo.

filePos (nombre_logico) ;

Devuelve la posición actual del puntero en el archivo.
En longitud fija, los registros se numeran de 0..N-1.

seek (nombre_logico, pos) ;

Establece la posición del puntero en el archivo.

```
program creacion_archivo;  
type  
    persona = record  
        dni: string[8]  
        apellidoyNombre: string[30];  
        direccion: string[40];  
        sexo      : char;  
        salario   : real;  
    end;  
    archivo_personas = file of  
persona;  
  
var  
    personas: archivo_personas;  
    nombre_fisico: string[12];  
    per: persona;
```


begin

```
write('Ingrese el nombre del archivo: ');  
readln(nombre_fisico)
```

{enlace entre el nombre lógico y el nombre físico}

```
assign(personas, nombre_fisico);
```

{apertura del archivo para creación}

```
rewrite(personas);
```

```
        {lectura del DNI una persona}  
        readln(per.dni);  
  
        while (per.dni <> '') do begin  
        {lectura del resto de los datos de la persona}  
        readln(per.apellidoyNombre);  
        readln(per.direccion);  
        readln(per.sexo);  
        readln(per.salarario);  
  
        {escritura del registro de la persona en el archivo}  
        write(personas, per);  
  
        {lectura del DNI de una nueva persona}  
        readln(per.dni);  
        end;  
  
        {cierre del archivo}  
        close(personas);  
    end.
```

Ejemplo-archivo de texto- binario

19

Type

```
tRegistroVotos=Record  
  codProv: integer;  
  codLoc: integer;  
  nroMesa: integer;  
  cantVotos: integer;  
  desc:String;
```

End;

```
tArchVotos=File of tRegistroVotos;
```

Var

```
  opc: Byte;  
  nomArch, nomArch2: String;  
  arch: tArchVotos; carga: Text { archivo de texto con  
datos de los votos, se lee de el y se genera archivo  
binario.  
  votos: tRegistroVotos;
```

Ejemplo-archivo de texto- binario

20

{Opción 1 crea el archivo binario desde un texto}

```
Case opc of
  1: begin
    Write('Nombre del archivo de carga: ');
    ReadLn(nomArch2);
    Assign(carga, nomArch2);
    Reset(carga); {abre archivo de texto con datos}
    Rewrite(arch); {crea nuevo archivo binario}
    while (not eof(carga)) do begin
      With votos do ReadLn(carga, codProv, codLoc,
nroMesa, cantVotos, desc); {lectura de archivo de texto}
      Write(arch, votos); {escribe binario}
    end;
    Write('Archivo cargado. ');
    ReadLn;
    Close(arch);    Close(carga)    {cierra los dos
archivos} end;
```

Ejemplo-archivo de texto- binario

21

```
Begin
WriteLn('VOTOS');
WriteLn;
WriteLn('0. Terminar el Programa');
WriteLn('1. Crear un archivo binario desde un arch
texto');
WriteLn('2. Abrir un archivo binario y exportar a texto');
Repeat
Write('Ingresa el nro. de opcion: '); ReadLn(opc);
If (opc=1) or (opc=2) then begin
    WriteLn;
    Write('Nombre del archivo de votos: ');
    ReadLn(nomArch);
    Assign(arch, nomArch);
end;
```

Ejemplo-archivo de texto- binario

22

```
{Opcion 2 exporta el contenido del binario a un texto}
2: begin
    Reset (arch); {abre archivo binario}
    Rewrite (carga); {crea archivo de texto, se
utiliza el mismo de opcion 1 a modo ejemplo}
    while (not eof (arch)) do begin
        Read (arch, votos); {lee votos del arch binario}
        With votos do
            WriteLn (codProv:5,          codLoc:5,          nroMesa:5,
cantVotos:5,      desc:5); {escribe en pantalla el
registro}
            With votos do
                WriteLn (carga, '      ', codProv, '      ', codLoc, '      ',
nroMesa, '      ', cantVotos, '      ', desc); {escribe en el
archivo texto los campos separados por el carácter
blanco}
        end;
    Close (arch); Close (carga)
```



¿Dudas?