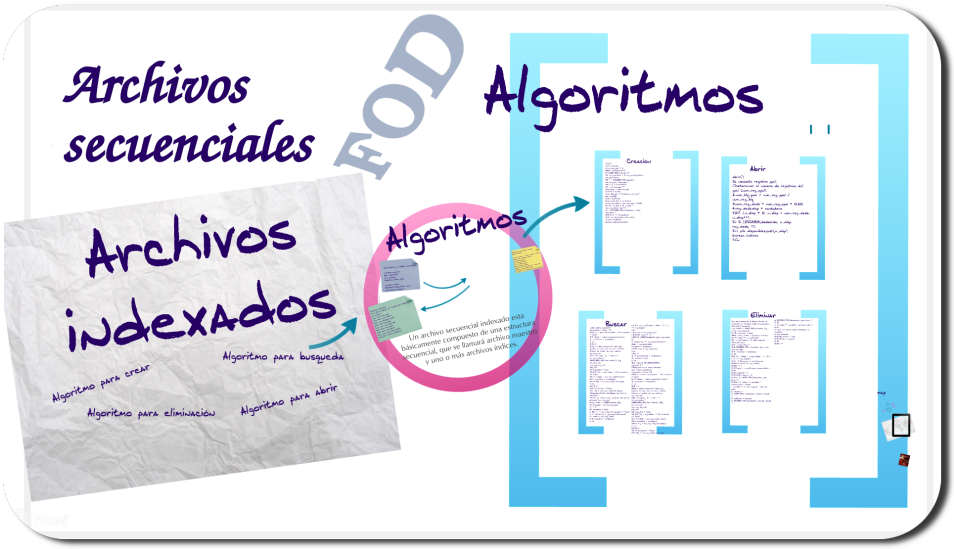
Fundamentos a la Organizacion de Datos
FOD - Cursada 2024

Práctica 1 - Creación, consulta y mant. de archivos secuenciales



Alsina Alejandro Marcos

CC BY SA

FACULTAD DE INFORMÁTICA

PRACTICA 1

Objetivo *Creación, consulta y mantenimiento de archivos secuenciales - Algorítmica Básica.*

FUNCIONES GENERICAS CON ARCHIVOS | Se usará en todos los items del TP (asigna y/o valida archivos)

```
---------------------------- Para evaluar la creacion o si existe archivo ----------------------------
function validaEnlace (var archivo:archivoFisico):boolean;
var ok:boolean; // F->hubo fallo, V->si abrio s/error entonces existe y debe cerrarlo
begin
  {$I-} reset(archivo);{$I+} // bloquea,-intento apertura- desvía errores
  ok:=(ioResult = 0);        // ok=mantener apertura - si existio F, no debe cerrar nada
  if ok then close(archivo); // V->(valido,existio arch y debe cerrarse)
  validaEnlace:= ok ;        //retornar (F->error si lo hay, V->si esta ok EXISTE)
end;

function enlaceLogico (var archivo:archivoFisico; operacion:string):boolean;
var nomArchivo: string; ok:boolean;
begin
  //->si hay que abrirlo verificar que realmente exista el archivo c/nombre esp
  if (operacion ='abrir')then begin
    write(' -INGRESE NOMBRE DE ARCHIVO PARA SU APERTURA: '); readln(nomArchivo);
    assign(archivo,nomArchivo);
    if NOT(validaEnlace(archivo))then begin    // verificar si existe archivo
      writeln('-Error en la apertura del archivo NO EXISTE ¨',nomArchivo,'¨');
      ok:=false;
      end else
        ok:=true; //->OPERACION EXITOSA archivo se pudo alcanzar(EXISTE)
    end else begin
      //->sino operacion<>.ªbreçrear archivo para cargar datos, enlace (es nuevo archivo)
      write(' -Ingrese nombre de archivo a crear: '); readln(nomArchivo);
      assign(archivo,nomArchivo);
      ok:=true;
      end;
  enlaceLogicoArchivo:=ok; //->retorna booleano de la operacion
end;
```

NOTA: Estas 2 funciones se utilizarán en todos los ejercicios del TP donde el tipo 'archivoFisico' será el tipo declarado de archivo en el type según corresponda cada ejercicio.

1. Realizar un algoritmo que cree un archivo de números enteros no ordenados y permita incorporar datos al archivo. Los números son ingresados desde teclado. La carga finaliza cuando se ingresa el número 30000, que no debe incorporarse al archivo.
   El nombre del archivo debe ser proporcionado por el usuario desde teclado.

2. Realizar un algoritmo, que utilizando el archivo de números enteros no ordenados creado en el ejercicio 1, informe por pantalla cantidad de números menores a 1500 y el promedio de los números ingresados.
   El nombre del archivo a procesar debe ser proporcionado por el usuario una única vez.
   Además, el algoritmo deberá listar el contenido del archivo en pantalla.

Ejercicios 1 y 2 — FOD - TP01

```
program FOD_2024_TP01_Ej_01y02;

uses crt;

const FIN=30000;

type
   archivoFisico = file of integer;

{**************** TP01-EJ01 ***************}
procedure cargarEnteros(var a:archivoFisico);

  procedure leer( var n:integer);
  begin
    write('Ingrese Numero: ');
    readln(n);
  end;

var n:num;
begin
  if(enlaceLogico(a,'nuevo'))[*] then begin
    rewrite(a) ;
    leer(n);
    while ( n <> FIN ) do begin
      write(a,n);
      leer(n);
      end;
    close(a) ;
   end else
     writeln('Error al crear el archivo');
end;

{**************** TP01-EJ02 ****************}
procedure informe(var a:archivoFisico);
var
 totNum,menor_1500,n:integer;  promedio:real;
begin
 if(enlaceLogico(a,'abrir'))[*] then begin
  totNum:=0; menor_1500:=0;  promedio:=0;
  reset(a) ;
  while NOT EOF(a) do begin
   read(a,n);
   write('|',n);
   promedio:= promedio + n;
   totalNumeros:= totalNumeros +1;
   if(n < 1500)then menor_1500:= menor_1500 +1;
   end;
  writeln('|');
  promedio:= promedio / totalNumeros;
  writeln('El prom de num fue: ',promedio:4:2);
  writeln('Cant num > 1500 fue: ',menor_1500);
  close(a) ;
  end else
   writeln('Error. Informe s/procesar');
end;

{****************************************** MENU ********************************************}
procedure menu();
var    op:byte;  a:archivoFisico
begin
repeat
  clrscr;
  writeln('---------------------------- MENU OPCIONES ----------------------------');
  writeln('|                                                                      |');
  writeln('| 1.- Cargar archivo binario de numeros                                |');
  writeln('|                                                                      |');
  writeln('| 2.- Informe cantidad menores a 1500 y el promedio de los numeros     |');
  writeln('|                                                                      |');
  writeln('| f.- <<-- Para finalizar y salir ....                                 |');
  writeln('|                                                                      |');
  writeln('------------------------------------------------------------------------');
  write(' -Ingrese opcion: ');   readln(op);
   case opcion of
    '1': begin cargarEnteros(numeros); end;
    '2': begin informe(numeros); end;;
    'f': begin writeln('Finalizando y cerrando aplicacion...'); delay(1500); end;
     else begin writeln ('Opcion invalida, Seleccione opcion valida del Menu. '); end;
    end;
  if(opcion<>'f')then begin write('Presione tecla para continuar...'); readkey(); end;
  until (op = 'f');
end;

{****************************************** MAIN ********************************************}
BEGIN
  menu();
  write('Presione una tecla para finalizar...');
  readkey;
END.
```

3. Realizar un programa que presente un menú con opciones para:

   a) Crear un archivo de registros **no ordenados** de empleados y completarlo con datos ingresados desde teclado. De cada empleado se registra: número de empleado, apellido, nombre, edad y DNI. Algunos empleados se ingresan con DNI 00. La carga finaliza cuando se ingresa el String 'fin' como apellido.

   b) Abrir el archivo anteriormente generado y:

      I Listar en pantalla los datos de empleados que tengan un nombre o apellido determinado.

      II Listar en pantalla los empleados de a uno por línea.

      III Listar en pantalla empleados mayores de 70 años, próximos a jubilarse.

   NOTA: El nombre del archivo a crear o utilizar debe ser proporcionado por el usuario.

4. Agregar al menú del programa del ejercicio 3, opciones para:

   a) Añadir uno o más empleados al final del archivo con sus datos ingresados por teclado. Tener en cuenta que no se debe agregar al archivo un empleado con un número de empleado ya registrado (control de unicidad).

   b) Modificar la edad de un empleado dado.

   c) Exportar el contenido del archivo a un archivo de texto llamado "todos_empleados.txt".

   d) Exportar a un archivo de texto llamado: "faltaDNIEmpleado.txt", los empleados que no tengan cargado el DNI (DNI en 00).

      NOTA: Las búsquedas deben realizarse por número de empleado.

Ejercicios 3 y 4 — FOD - TP01

```
program FOD_2024_TP01_Ej_03y04;
const FIN= 'fin'; // no debe incorporarse al archivo
type
  tPersona = record
    nro : integer;
    nom : string;
    ape : string;
    eda : integer;
    dni : string[8];
    end;
  archivoFisico = file of tPersona;

procedure cargarPersonas(var e:archivoFisico);
var   p:tPersona;
begin
  if (enlaceLogico[*] (e,'crear') then begin
    rewrite(e);
    leer(p);
    while ( p.ape <> FIN ) do begin
      write(e,p);
      leer(p);
      end;
    close(e);
    end else
      writeln('Nada por hacer, s/archivo');
end;

procedure leer(var p:tPersona);
begin
  with p do begin
    write('Apellido: ');      readln(ape);
    if ( ape <> FIN )then begin
      write('Nombre:');       readln(nom);
      write('Nro empleado:'); readln(nro);
      write('DNI: ');         readln(dni);
      write('Edad: ');        readln(eda);
      writeln('.........................');
      end;
    end;
end;
```

Ejercicios 3 y 4 (continuación I)— FOD - TP01

```
procedure listar_busq_NomApe(var a:archivoFisico);
  function encontro(n,a,pal:string):boolean;
  begin
    encontro:=((pos(pal,n)>0)or(pos(pal,a)>0));
  end;
var e:tPersona; buscar:string;
begin
  if ( validaEnlace(a) ) then begin;
    write('Nom o Ape a buscar:'); read(buscar);
    reset(a);
    while NOT EOF(a) do begin
      read(a,e);
      with e do begin
        if ( encontro(nom,ape,buscar) ) then
          writeln(e.nro,e.nom,e.ape,e.eda,e.dni);
        end;
      end;
    close(a);
    writeln(' Finalizo la busqueda);
    end else
      writeln('Nada por hacer, s/archivo');
end;

procedure listarEmpleados(var a:archivoFisico);
var p:tPersona;
begin
  if ( validaEnlace(a) )then begin;
    reset(a);
    while NOT EOF(a) do begin
      read(a,p);
      writeln(p.nro,p.nom,p.ape,p.eda,p.dni);
      end;
    close(a);
    end else
      writeln('Nada por hacer, s/archivo');
end;

procedure listarProxJubilar(var a:archivoFisico);
  function se_jubila(p:tPersona):boolean;
  begin
    se_jubila:= p.eda ¿70;
  end;
var p:tPersona;
begin
  if (validaEnlace(a))then begin;
    reset(a);
    while NOT EOF(a)do begin
      read(a,p);
      if se_jubila(p)then
        writeln(p.nro,p.nom,p.ape,p.eda,p.dni);
      end;;
    close(a);
    end else
      writeln('Nada por hacer, s/archivo');
end;
```

```
procedure agregarEmpleados(var a:archivoFisico);
var p,e:tPersona; continuar:char; noExiste:boolean;
begin
  if ( validaEnlace(a) )then begin;
    reset(a);
    repeat
      seek(a,0);
      leer(p);
      if ( p.ape <> FIN )then begin
        noExiste:=true;
        while(not eof(a))and(noExiste) do begin
          read(a,e);
          noExiste:= p.nro <> e.nro;
          end;
        if noExiste then write(a, e )
          else writeln('Empleado existe');
        write('Cargar otro empleado? (s) o (f) ');
        readln(continuar);
      end else
        continuar:='f';
    until ( (continuar ='f');
    close(a);
    end else
      writeln('Nada por hacer, s/archivo');
end;

procedure modificarEdad(var a:archivoFisico);
var p:tPersona; nro:integer; existe:boolean;
begin
  if (validaEnlace(a))then begin;
    reset(a);
    write('Nro. empleado a modificar edad:');
    readln(nro);
    repeat
      read(a,p);
      existe:=( p.nro = nro );
    until( ( EOF(a) ) or (existe) );
    if (existe) then begin
      write('Modificacion: ');
      readln(p.eda);
      seek(a,filepos(a)-1);
      write(a,p);
      writeln('edad actualizada');
      end else
        writeln('No se encontro empleado');
    close (a);
    end else
      writeln('Nada por hacer, s/archivo');
end;
```

Ejercicios 3 y 4 (continuación II)— FOD - TP01

```
procedure listarTodosTxt(var a:archivoFisico);
var p:tPersona; txt:text;
begin
  if ( validaEnlace(a) ) then begin;
    assign(txt,'todos_empleados.txt');
    rewrite (txt);
    reset (a);
    while ( NOT EOF(a) ) do begin
      read(a,p);
      with p do
        writeln(txt,nro,nom,ape,eda,dni);
      end;
    close (txt);
    close (a);
    end else
      writeln('Nada por hacer, s/archivo');
end;
```

```
procedure listarSinDNI(var a:archivoFisico);
var p:tPersona; txt:text;
begin
  if ( validaEnlace(a) ) then begin;
    assign(txt,'faltaDNIEmpleado.txt');
    rewrite (txt);
    reset (a);
    while ( NOT EOF(a) ) do begin
      read(a,p);
      with p do
        if ( dni = '  00' )then
          writeln(txt,nro,nom,ape,eda,dni);
      end;
    close (txt);
  close (a);
  end else
    writeln('Nada por hacer, s/archivo');
end;
```

```
procedure menu();
  procedure mostrarMenu();
  begin
    writeln(' ------------------------------ MENU OPCIONES --------------------------------');
    writeln('| 1.- Asignar y cargar archivo binario de empleados                           |');
    writeln('| 2.- Buscar en archivo empleados coincidencias por Apellido o Nombre         |');
    writeln('| 3.- Listar o mostrar el archivo binario con todos los empleados             |');
    writeln('| 4.- Listar o mostrar prox a jubilarse del binario de empleados (mayor 70)   |');
    writeln('| 5.- Agregar uno o mas empleados al archivo binario de empleados             |');
    writeln('| 6.- Modificar edad a una o mas empleados del binario empleados              |');
    writeln('| 7.- Exportar a un txt archivo binario de empleados (todos sus registros)    |');
    writeln('| 8.- Exportar a un txt empleados que le falta info en su DNI                 |');
    writeln('| f.≪ -- Para finalizar y salir ...                                           |');
    writeln(' -----------------------------------------------------------------------------');
    write(' Ingrese opcion: ');
  end;
var empleados:archivoFisico; opcion:char;
begin
  repeat
    mostrarMenu;
    readln(opcion);
    case opcion of
      '1': begin cargarPersonas(empleados); end;            //Ej03-a)
      '2': begin listar_busq_NomApe(empleados); end;        //Ej03-b-i)
      '3': begin listarEmpleados(empleados); end;           //Ej03-b-ii)
      '4': begin listarProxJubilar(empleados); end;         //Ej03-b-iii)
      '5': begin agregarEmpleados(empleados); end;          //Ej04-a)
      '6': begin modificarEdad(empleados); end;             //Ej04-b)
      '7': begin listarTodosTxt (empleados) end;            //Ej04-c)
      '8': begin listarSinDNI( empleados) end;              //Ej04-d)
      'f': begin writeln('Finalizando y cerrando aplicacion...'); delay(1500); end;
      else begin write ('ERROR- OPCION NO VALIDA... '); end;
      end;
    if( opcion <> 'f') then begin
      write(' =>Presione tecla para continuar... '); readkey();
      end;
  until (opcion = 'f');
end;

------------------------------------------ MAIN ------------------------------------------
BEGIN
  menu;
END.
```

5. Realizar un programa para una tienda de celulares, que presente un menú con opciones para:

   a) Crear un archivo de registros no ordenados de celulares y cargarlo con datos ingresados desde un archivo de texto denominado "celulares.txt". Los registros correspondientes a los celulares deben contener: código de celular, nombre, descripción, marca, precio, stock mínimo y stock disponible.

   b) Listar en pantalla los datos de aquellos celulares que tengan un stock menor al stock mínimo.

   c) Listar en pantalla los celulares del archivo cuya descripción contenga una cadena de caracteres proporcionada por el usuario.

   d) Exportar el archivo creado en el inciso a) a un archivo de texto denominado "celulares.txt" con todos los celulares del mismo. El archivo de texto generado podría ser utilizado en un futuro como archivo de carga (ver inciso a), por lo que debería respetar el formato dado para este tipo de archivos en la NOTA 2.

   NOTA 1: El nombre del archivo binario de celulares debe ser proporcionado por el usuario.

   NOTA 2: El archivo de carga debe editarse de manera que cada celular se especifique en tres líneas consecutivas. En la primera se especifica: código de celular, el precio y marca, en la segunda el stock disponible, stock mínimo y la descripción y en la tercera nombre en ese orden. Cada celular se carga leyendo tres líneas del archivo

6. Agregar al menú del programa del ejercicio 5, opciones para:

   a) Añadir uno o más celulares al final del archivo con sus datos ingresados por teclado.

   b) Modificar el stock de un celular dado.

   c) Exportar el contenido del archivo binario a un archivo de texto denominado: "SinStock.txt", con aquellos celulares que tengan stock 0.

   NOTA: Las búsquedas deben realizarse por nombre de celular.

Ejercicios 5 y 6 — FOD - TP01

```
program FOD_2024_TP01_Ej_05y06;
uses crt;
type
  tCelular = record
    cod:integer;  nom:string;  des:string;  mar:string;  act:integer;  min:integer;  pre:real;
    end;

  archivoFisico = file of tCelular;

procedure cragarCelulares(var c:archivoFisico);
var data:text; celu:tCelular; bco:char;
begin
  assign(data,'celulares.txt');
  if ( validaTxt(data) ) then begin
    reset(data);
    if enlaceLogico(c,'crear') then begin
      rewrite(c);
      while NOT EOF(data) do begin
        with celu do begin
          readln(data,cod,bco,pre,mar);
          readln(data,act,bco,min,bco,des);
          readln(data,nom);
          end;
        write(c,celu);
        end;
      close(c);
      close(data);
    end else
      writeln('Error en crear archivo);
  end else
    writeln('NO existe 'celulares.txt');
end;
```

```
function validaTxt (var txt:text) : boolean;
var  ok:boolean;
begin
  {$I-} reset(txt);{$I+}
  ok:=(ioResult = 0);
  if ok then close(txt);
  validaEnlace:= ok ;
end;
procedure listarStkBajo(var c:archivoFisico);
var celu:tCelular;
begin
  if ( validaEnlace(c) )then begin
    reset(c);
    while NOT EOF(c) do begin
      read(c,celu);
      if (celu.act < celu.min) then
        with celu do
          writeln(cod,nom,mar,pre,act,min);
      end;
    close(c);
    end else
      writeln('Nada por hacer, s/archivo');
end;
```

Ejercicios 5 y 6 (continuación I)— FOD - TP01

```
procedure buscaDescripcion(var c:archivoFisico);
var celu:tCelular; buscar:string;
begin
  if ( validaEnlace(c) )then begin
    write('Ingrese texto:');
    readln(buscar);
    reset(c);
    while (not eof(c))do begin
      read(c,celu);
      if( encontro(buscar, celu.des) )then
        with celular do
          writeln(cod,nom,mar,pre,act,min);
      end;
    close(c);
    end else
      writeln('Nada por hacer, s/archivo');
end;

procedure agregarCelular(var c:archivoFisico);
  end;
var celu:tCelular; fin:char;
begin
  reset(c);
  seek(c,fileSize(c));
  repeat
    leer(celu);
    write(c,celu);
    writeln('Agregar otro? [n] para finalizar');
    write('Ingrese caracter:');
    readln(fin);
  until ( fin = 'n' );
  close(c);
end;

procedure exportaCelularTxt(var c:archivoFisico);
var celu:tCelular;
    txt:text;
begin
  if ( validaEnlace(c) )then begin
    assign(txt,'celulares1.txt');
    rewrite(txt);
    reset(c);
    while NOT EOF(c) do begin
      read(c,celu);
      with celu do begin
        writeln(txt,cod,' ',pre:6:2,' ',mar);
        writeln(txt,act,' ',min,' ',des);
        writeln(txt,nom);
        end;
      end;
    close(c);
    close(txt);
    end else
      writeln('Nada por hacer, s/archivo');
end;
```

```
procedure leer(var c:tCelular);
begin
  with c do begin
    write('Codigo: ');          readln(cod);
    write('Marca: ');           readln(mar);
    write('Nnombre: ');         readln(nom);
    write('Precio $: ');        readln(pre);
    write('Stock actual: ');    readln(act);
    write('Stock minimo: ');    readln(min);
    write('Descripcion: ');     readln(des);
    end;
end;

function encontro(n,a,pal:string):boolean;
begin
  encontro:=((pos(pal,n)>0)or(pos(pal,a)>0));
end;

procedure modificaStock(var c:archivoFisico);
var celu:tCelular; nombre:string;
begin
  if ( validaEnlace(c) ) then begin
    writeln('Nombre celular a modificar stock:');
    readln(nombre);
    reset(c);
    read(c,celu);
    while ((NOT EOF(c))and(nombre<>celu.nom)) do
      read(c,celu);
      if (nombre = celu.nom) then begin
        seek(c,filePos(c) -1);
        readln(celu.act);
        write(c,celu);
      end;
    close(c);
    end else
      writeln('Nada por hacer, s/archivo');
end;

procedure sinStock(var c:archivoFisico);
var celu:tCelular; txt:text;
begin
  if (validaEnlace(c))then begin
    assign(txt,'SinStock.txt');
    rewrite(txt);
    reset(cs);
    while NOT EOF(c)do begin
      read(c,celu);
      if (celu.act = 0 ) then
        with celu do
          writeln(txt,'cod,nom,mar,pre,act,min');
      end;
      close(c);
      close(txt);
    end else
      writeln('Nada por hacer, s/archivo');
end;
```

Ejercicios 5 y 6 (continuación II)— FOD - TP01

```
procedure mostrarMenu();
begin
  writeln(' ------------------------------ MENU OPCIONES --------------------------------');
  writeln('| 1.- Asignar y cargar archivo binario de celulares c/çelulares.txt"          |');
  writeln('| 2.- Listar o mostrar celulares con sotck por debajo del minimo              |');
  writeln('| 3.- Buscar en celulares coincidencias en decripcion con una cadena dada     |');
  writeln('| 4.- Exportar el binario celulares al archivo texto çelulares.txt"           |');
  writeln('| 5.- Agregar uno o mas celulares al archivo binario de celulares             |');
  writeln('| 6.- Modificar stock act de 1 celular dado un nombre en archivo celulares    |');
  writeln('| 7.- Exportar a un txt "SinStock.txt" el archivo binario de celulares        |');
  writeln('| f. ≪-- Para finalizar y salir ...                                           |');
  writeln('-------------------------------------------------------------------------------');
  writeln();
  write(' Ingrese opcion: ');
end;

procedure menu();
var celulares:archivoFisico; opcion:char;
begin
  repeat
    mostrarMenu;
    readln(opcion);
    case opcion of
      '1': begin cragarCelulares(celulares); end; //Ej03-a)
      '2': begin listarStoksBajos(celulares); end; //Ej03-b-i)
      '3': begin buscaDescripcion(celulares); end; //Ej03-b-ii)
      '4': begin exportaCelularesTexto(celulares); end; //Ej03-b-iii)
      '5': begin agregarCelular(celulares); end; //Ej04-a)
      '6': begin modificaStock(celulares); end; //Ej04-b)
      '7': begin sinStock(celulares) end; //Ej04-c)
      'f': begin writeln('Finalizando y cerrando aplicacion...'); delay(1500); end;
        else begin write ('ERROR- OPCION NO VALIDA... '); end;
      end;
    if ( opcion <> 'f') then write(' =>Presione tecla para continuar... ');
  until (opcion = 'f');
end;
------------------------------------------ MAIN ------------------------------------------
BEGIN
  menu;
END.
```

7. Realizar un programa que permita:

   a) Crear un archivo binario a partir de la información almacenada en un archivo de texto. El nombre del archivo de texto es: "novelas.txt". La información en el archivo de texto consiste en: código de novela, nombre, género y precio de diferentes novelas argentinas. Los datos de cada novela se almacenan en dos líneas en el archivo de texto. La primera línea contendrá la siguiente información: código novela, precio y género, y la segunda línea almacenará el nombre de la novela.

   b) Abrir el archivo binario y permitir la actualización del mismo. Se debe poder agregar una novela y modificar una existente. Las búsquedas se realizan por código de novela.

   NOTA: El nombre del archivo binario es proporcionado por el usuario desde el teclado.

Ejercicios 7 — FOD - TP01

```
program FOD_2024_TP1_Ej_07;

uses crt;

type
  tNovela = record
    cod : integer;
    pre : real;
    gen : string;
    nom : string;
    end;

  archivoFisico = file of tNovela;

procedure cragarNovelas(var n:archivoFisico);
  function validaEnlace (var a:text:boolean;
  var ok:boolean;
  begin
    {$I-}reset(a);{$I+} ok:=(ioResult = 0);
    if ok then close(a);
    validaEnlace:= ok ;
  end;
var data:text; n:tNovela; bco:char;
begin
  assign(data,'novelas.txt');
  if (validaEnlace(data))then begin
    reset(data);
    if enlaceLogico(n,'crear') then begin
      rewrite(n);
      while NOT EOF(data)do begin
        with novela do begin
          readln(data,cod,bco,pre,gen);
          readln(data,nom);
          end;
        write(novelas,novela);
        end;
      close(n);
      close(data);
      writeln('Finalizo carga c/Novelas.');
    end else
      writeln('Error en crear archivo binario');
    end else begin
      writeln('NO existe "novelas.txt"...');
end;

procedure mostrarMenu1();
begin
  writeln('----- ACTUALIZAR BIN NOVELAS ------');
  writeln(' 1- Cargar 1 novela al archivo bin ');
  writeln(' 2- Listar datos binario c/novelas ');
  writeln(' 3- Modificar datos de una novela  ');
  writeln(' f- Volver menu principal ...      ');
  writeln('-----------------------------------');
end;

procedure mostrarMenu2;
begin
  writeln('------- MODIFICA UNA NOVELA -------');
  writeln(' a.-Modificar todo                 ');
  writeln(' b.-Modifica solo codigo           ');
  writeln(' c.-Modifica solo genero           ');
  writeln(' d.-Modifica solo precio           ');
  writeln(' e.-Modifica solo nombre           ');
  writeln('-----------------------------------');
end;

procedure leer(var n:tNovela; c,p,g,t:boolean);
begin
  if (t) then begin
    write('Nombre: '); readln(n.nom);
    end;
  if (g) then begin
    write('Genero: '); readln(n.gen);
    end;
  if (p) then begin
    write('Precio: '); readln(n.pre)
    end;
  if (c) then begin
    write('Codigo de novela: '); readln(n.cod);
    end;
end;

procedure modifica(var n:archivoFisico;
                   var nov:tNovela;
                   var ok:boolean);
var busca:string; opcion:char;
begin
  write('Novela a modificar: ');
  readln(busca);
  ok:=false;
  while ( NOT EOF(n) ) and (not ok) do begin
    read(n,nov);
    ok:= ( nov.nom = busca );
    end;
  if ( ok ) then begin
   repeat
    mostrarMenu2;
    write('Ingrese opcion: ');
    readln(opcion);
    if (ord(opcion)<97)or(ord(opcion)>101) then
      write ('ERROR- OPCION NO VALIDA... ');
   until (ord(opcion)>=97)and(ord(opcion)<=101);
   write('Modifica ');
   case opcion of
    'a':begin leer(nov,true,true,true,true);   end;
    'b':begin leer(nov,true,false,false,false);end;
    'c':begin leer(nov,false,false,true,false);end;
    'd':begin leer(nov,false,true,false,false);end;
    'e':begin leer(nov,false,false,false,true);end;
    end;
   seek(novelas,filepos(novelas)-1);
   write(novelas,novela);
   end else
    novela.nom:=busca;
end;
```

Ejercicios 7 (continuación I)— FOD - TP01

```
procedure listar(var n:archivoFisico);
var n:tNovela;
begin
  while not eof(n) do begin read(novelas,n); writeln(n.cod,n.nom,n.gen,n.pre; end;
  write('Fin de lista. Presione tecla para continuar...'); readkey();
end;

procedure actualizarNovelas(var novelas:archivoFisico);
var novela:tNovela; opcion:char; encontro:boolean;
begin
  if enlaceLogico(novelas,'abrir') then begin
    repeat
      reset(novelas);
      mostrarMenu1;
      write('Ingrese opcion: '); readln(opcion);
        case opcion of
          '1': begin
                leer(novela,true,true,true,true);  seek(novelas,fileSize(novelas));
                write(novelas,novela);
               end;
          '2': begin listar(novelas); end;
          '3': begin
                modifica(novelas,novela,encontro);
                if (encontro) then begin
                  writeln('Novela ',novela.nom,' modificada exitosamente'); delay(1000);
                end else begin writeln('Novela ',novela.nom,' inexistente'); delay(1000);  end;
               end;
          'f':begin write ('Volviendo al menu principal'); end;
             else begin write ('ERROR- OPCION NO VALIDA... '); end;
          end;
        close(novelas);
      until(opcion='f');
    end else begin writeln('Nada por hacer, sin archivo asignado...'); end;
end;

procedure menu();
  procedure mostrarMenu();
  begin
    writeln(' ------------------------------ MENU OPCIONES ---------------------------------');
    writeln(' | 1.- Asignar y cargar archivo binario de Novelas c/"novelas.txt"            |');
    writeln(' | 2.- Realizar modificaciones a un archivo binario c/novelas                 |');
    writeln(' | f. << -- Para finalizar y salir ...                                        |');
    writeln(' ------------------------------------------------------------------------------');
    write(' Ingrese opcion: ');
  end;
var novelas : archivoNovelas; opcion:char;
begin
  repeat
    mostrarMenu;
    readln(opcion);
    case opcion of
      '1': begin cragarNovelas(novelas); end;
      '2': begin actualizarNovelas(novelas); end;
      'f': begin writeln('Finalizando y cerrando aplicacion...'); delay(1500); end;
        else begin write ('ERROR- OPCION NO VALIDA... '); end;
      end;
    if ( opcion <> 'f') then begin write(' =>Presione tecla para continuar... '); readkey(); end;
  until ( opcion = 'f' );
end;
------------------------------------------ MAIN ------------------------------------------
BEGIN
  menu;
END.
```