Juan Menduiña

Trabajo Práctico N° 1:
Creación, Consulta y Mantenimiento de Archivos Secuenciales - Algorítmica Básica.

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.

```
program TP1_E1;
{$codepage UTF8}
uses crt;
const
  num_salida=30000;
type
  t_archivo_enteros=file of int16;
procedure leer_numero(var num: int16);
var
  i: int8;
begin
  i:=random(100);
  if (i=0) then
    num:=num_salida
  else
    num:=random(high(int16));
end;
procedure cargar_archivo_enteros(var archivo_enteros: t_archivo_enteros);
var
  num: int16;
begin
  rewrite(archivo_enteros);
  textcolor(green); write('Los números ingresados son: ');
  leer_numero(num);
  while (num<>num_salida) do
  begin
    textcolor(yellow); write(num,' ');
    write(archivo_enteros,num);
    leer_numero(num);
  end;
  close(archivo_enteros);
end;
var
  archivo_enteros: t_archivo_enteros;
begin
  randomize;
  assign(archivo_enteros,'enteros');
  cargar_archivo_enteros(archivo_enteros);
end.
```

Ejercicio 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.

```
program TP1_E2;
{$codepage UTF8}
uses crt;
const
  num_corte=1500;
type
  t_archivo_enteros=file of int16;
procedure procesar_archivo_enteros(var archivo_enteros: t_archivo_enteros; var nums_corte:
int16; var prom: real);
var
  num: int16;
  suma: real;
begin
  reset(archivo_enteros);
  suma:=0;
  textcolor(green); write('El contenido del archivo es: ');
  while (not eof(archivo_enteros)) do
  begin
    read(archivo_enteros,num);
    textcolor(yellow); write(num,' ');
    if (num<num_corte) then
      nums_corte:=nums_corte+1;
    suma:=suma+num;
  end;
  if (fileSize(archivo_enteros)>0) then
    prom:=suma/fileSize(archivo_enteros);
  writeln();
  close(archivo_enteros);
end;
var
  archivo_enteros: t_archivo_enteros;
  nums_corte: int16;
  prom: real;
begin
  nums_corte:=0; prom:=0;
  assign(archivo_enteros,'enteros');
  procesar_archivo_enteros(archivo_enteros,nums_corte,prom);
  textcolor(green); write('La cantidad de números menores a '); textcolor(yellow);
write(num_corte); textcolor(green); write(' es '); textcolor(red); writeln(nums_corte);
  textcolor(green); write('El promedio de los números ingresados es '); textcolor(red);
write(prom:0:2);
end.
```

Ejercicio 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, el cual se proporciona desde el teclado.*
> **(ii)** *Listar en pantalla los empleados de a uno por línea.*
> **(iii)** *Listar en pantalla los empleados mayores de 70 años, próximos a jubilarse.*

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

```
program TP1_E3;
{$codepage UTF8}
uses crt;
const
  apellido_salida='fin';
  edad_corte=70;
  opcion_salida=0;
type
  t_string10=string[10];
  t_string20=string[20];
  t_registro_empleado=record
    numero: int16;
    apellido: t_string10;
    nombre: t_string10;
    edad: int8;
    dni: int32;
  end;
  t_archivo_empleados=file of t_registro_empleado;
function random_string(length: int8): t_string10;
var
  i: int8;
  string_aux: string;
begin
  string_aux:='';
  for i:= 1 to length do
    string_aux:=string_aux+chr(ord('A')+random(26));
  random_string:=string_aux;
end;
procedure leer_empleado(var registro_empleado: t_registro_empleado);
var
  i: int8;
begin
  i:=random(100);
  if (i=0) then
    registro_empleado.apellido:=apellido_salida
  else
    registro_empleado.apellido:=random_string(5+random(5));
  if (registro_empleado.apellido<>apellido_salida) then
  begin
    registro_empleado.numero:=1+random(1000);
    registro_empleado.nombre:=random_string(5+random(5));
```

```
    registro_empleado.edad:=18+random(high(int8)-18);
    if (i<=10) then
      registro_empleado.dni:=0
    else
      registro_empleado.dni:=10000000+random(40000001);
  end;
end;
procedure cargar_archivo_empleados(var archivo_empleados: t_archivo_empleados);
var
  registro_empleado: t_registro_empleado;
begin
  rewrite(archivo_empleados);
  leer_empleado(registro_empleado);
  while (registro_empleado.apellido<>apellido_salida) do
  begin
    write(archivo_empleados,registro_empleado);
    leer_empleado(registro_empleado);
  end;
  close(archivo_empleados);
  textcolor(green); writeln('El archivo de empleados fue creado y cargado con éxito');
end;
procedure imprimir_registro_empleado(registro_empleado: t_registro_empleado);
begin
  textcolor(green); write('Número: '); textcolor(yellow); write(registro_empleado.numero);
  textcolor(green); write('; Apellido: '); textcolor(yellow);
write(registro_empleado.apellido);
  textcolor(green); write('; Nombre: '); textcolor(yellow); write(registro_empleado.nombre);
  textcolor(green); write('; Edad: '); textcolor(yellow); write(registro_empleado.edad);
  textcolor(green); write('; DNI: '); textcolor(yellow); writeln(registro_empleado.dni);
end;
procedure imprimir1_archivo_empleados(var archivo_empleados: t_archivo_empleados);
var
  registro_empleado: t_registro_empleado;
  texto: t_string10;
begin
  texto:=random_string(5+random(5));
  reset(archivo_empleados);
  textcolor(green); write('Los datos de los empleados con nombre o apellido ');
textcolor(yellow); write(texto); textcolor(green); writeln(' son: ');
  while (not eof(archivo_empleados)) do
  begin
    read(archivo_empleados,registro_empleado);
    if ((registro_empleado.nombre=texto) or (registro_empleado.apellido=texto)) then
      imprimir_registro_empleado(registro_empleado);
  end;
  close(archivo_empleados);
end;
procedure imprimir2_archivo_empleados(var archivo_empleados: t_archivo_empleados);
var
  registro_empleado: t_registro_empleado;
begin
  reset(archivo_empleados);
  textcolor(green); writeln(' Los empleados del archivo son: ');
  while (not eof(archivo_empleados)) do
  begin
    read(archivo_empleados,registro_empleado);
    imprimir_registro_empleado(registro_empleado);
  end;
  close(archivo_empleados);
end;
procedure imprimir3_archivo_empleados(var archivo_empleados: t_archivo_empleados);
var
  registro_empleado: t_registro_empleado;
begin
  reset(archivo_empleados);
```

```
  textcolor(green); write('Los empleados mayores a '); textcolor(yellow); write(edad_corte);
textcolor(green); writeln(' años son: ');
  while (not eof(archivo_empleados)) do
  begin
    read(archivo_empleados,registro_empleado);
    if (registro_empleado.edad>edad_corte) then
      imprimir_registro_empleado(registro_empleado);
  end;
  close(archivo_empleados);
end;
procedure leer_opcion(var opcion: int8);
begin
  textcolor(red); writeln('MENÚ DE OPCIONES');
  textcolor(yellow); write('OPCIÓN 1: '); textcolor(green); writeln('Crear un archivo de
registros no ordenados de empleados y completarlo con datos ingresados desde teclado');
  textcolor(yellow); write('OPCIÓN 2: '); textcolor(green); writeln('Listar en pantalla los
datos de empleados que tengan un nombre o apellido determinado');
  textcolor(yellow); write('OPCIÓN 3: '); textcolor(green); writeln('Listar en pantalla los
empleados de a uno por línea');
  textcolor(yellow); write('OPCIÓN 4: '); textcolor(green); writeln('Listar en pantalla los
empleados mayores a 70 años, próximos a jubilarse');
  textcolor(yellow); write('OPCIÓN 0: '); textcolor(green); writeln('Salir del menú de
opciones');
  textcolor(green); write('Introducir opción elegida: '); textcolor(yellow); readln(opcion);
  writeln();
end;
procedure menu_opciones(var archivo_empleados: t_archivo_empleados);
var
  opcion: int8;
begin
  leer_opcion(opcion);
  while(opcion<>opcion_salida) do
  begin
    case opcion of
      1: cargar_archivo_empleados(archivo_empleados);
      2: imprimir1_archivo_empleados(archivo_empleados);
      3: imprimir2_archivo_empleados(archivo_empleados);
      4: imprimir3_archivo_empleados(archivo_empleados);
    else
        textcolor(green); writeln('La opción ingresada no corresponde a ninguna de las
mostradas en el menú de opciones');
    end;
    writeln();
    leer_opcion(opcion);
  end;
end;
var
  archivo_empleados: t_archivo_empleados;
begin
  randomize;
  assign(archivo_empleados,'empleados');
  menu_opciones(archivo_empleados);
end.
```

Ejercicio 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.

```
program TP1_E4;
{$codepage UTF8}
uses crt;
const
  apellido_salida='fin';
  edad_corte=70;
  dni_corte=0;
  opcion_salida=0;
type
  t_string10=string[10];
  t_string20=string[20];
  t_registro_empleado=record
    numero: int16;
    apellido: t_string10;
    nombre: t_string10;
    edad: int8;
    dni: int32;
  end;
  t_archivo_empleados=file of t_registro_empleado;
function random_string(length: int8): t_string10;
var
  i: int8;
  string_aux: string;
begin
  string_aux:='';
  for i:= 1 to length do
    string_aux:=string_aux+chr(ord('A')+random(26));
  random_string:=string_aux;
end;
procedure leer_empleado(var registro_empleado: t_registro_empleado);
var
  i: int8;
begin
  i:=random(100);
  if (i=0) then
    registro_empleado.apellido:=apellido_salida
  else
    registro_empleado.apellido:=random_string(5+random(5));
  if (registro_empleado.apellido<>apellido_salida) then
  begin
    registro_empleado.numero:=1+random(1000);
```

```
    registro_empleado.nombre:=random_string(5+random(5));
    registro_empleado.edad:=18+random(high(int8)-18);
    if (i<=10) then
      registro_empleado.dni:=0
    else
      registro_empleado.dni:=10000000+random(40000001);
  end;
end;
procedure cargar_archivo_empleados(var archivo_empleados: t_archivo_empleados);
var
  registro_empleado: t_registro_empleado;
begin
  rewrite(archivo_empleados);
  leer_empleado(registro_empleado);
  while (registro_empleado.apellido<>apellido_salida) do
  begin
    write(archivo_empleados,registro_empleado);
    leer_empleado(registro_empleado);
  end;
  close(archivo_empleados);
  textcolor(green); writeln('El archivo binario de empleados fue creado y cargado con éxito');
end;
procedure imprimir_registro_empleado(registro_empleado: t_registro_empleado);
begin
  textcolor(green); write('Número: '); textcolor(yellow); write(registro_empleado.numero);
  textcolor(green); write('; Apellido: '); textcolor(yellow);
write(registro_empleado.apellido);
  textcolor(green); write('; Nombre: '); textcolor(yellow); write(registro_empleado.nombre);
  textcolor(green); write('; Edad: '); textcolor(yellow); write(registro_empleado.edad);
  textcolor(green); write('; DNI: '); textcolor(yellow); writeln(registro_empleado.dni);
end;
procedure imprimir1_archivo_empleados(var archivo_empleados: t_archivo_empleados);
var
  registro_empleado: t_registro_empleado;
  texto: t_string10;
begin
  texto:=random_string(5+random(5));
  reset(archivo_empleados);
  textcolor(green); write('Los datos de los empleados con nombre o apellido ');
textcolor(yellow); write(texto); textcolor(green); writeln(' son: ');
  while (not eof(archivo_empleados)) do
  begin
    read(archivo_empleados,registro_empleado);
    if ((registro_empleado.nombre=texto) or (registro_empleado.apellido=texto)) then
      imprimir_registro_empleado(registro_empleado);
  end;
  close(archivo_empleados);
end;
procedure imprimir2_archivo_empleados(var archivo_empleados: t_archivo_empleados);
var
  registro_empleado: t_registro_empleado;
begin
  reset(archivo_empleados);
  textcolor(green); writeln(' Los empleados del archivo son: ');
  while (not eof(archivo_empleados)) do
  begin
    read(archivo_empleados,registro_empleado);
    imprimir_registro_empleado(registro_empleado);
  end;
  close(archivo_empleados);
end;
procedure imprimir3_archivo_empleados(var archivo_empleados: t_archivo_empleados);
var
  registro_empleado: t_registro_empleado;
begin
  reset(archivo_empleados);
```

```
  textcolor(green); write('Los empleados mayores a '); textcolor(yellow); write(edad_corte);
textcolor(green); writeln(' años son: ');
  while (not eof(archivo_empleados)) do
  begin
    read(archivo_empleados,registro_empleado);
    if (registro_empleado.edad>edad_corte) then
      imprimir_registro_empleado(registro_empleado);
  end;
  close(archivo_empleados);
end;
function control_unicidad(var archivo_empleados: t_archivo_empleados; numero: int16): boolean;
var
  registro_empleado: t_registro_empleado;
  ok: boolean;
begin
  ok:=false;
  while ((not eof(archivo_empleados)) and (ok=false)) do
  begin
    read(archivo_empleados,registro_empleado);
    if (registro_empleado.numero=numero) then
      ok:=true;
  end;
  control_unicidad:=ok;
end;
procedure agregar_empleado(var archivo_empleados: t_archivo_empleados);
var
  registro_empleado: t_registro_empleado;
  empleados: int16;
begin
  empleados:=0;
  reset(archivo_empleados);
  leer_empleado(registro_empleado);
  while (registro_empleado.apellido<>apellido_salida) do
  begin
    if (control_unicidad(archivo_empleados,registro_empleado.numero)=false) then
    begin
      seek(archivo_empleados,filesize(archivo_empleados));
      write(archivo_empleados,registro_empleado);
      empleados:=empleados+1;
    end;
    leer_empleado(registro_empleado);
  end;
  close(archivo_empleados);
  textcolor(green); write('Se han agregado '); textcolor(yellow); write(empleados);
textcolor(green); writeln(' empleados al final del archivo');
end;
procedure modificar_edad_empleado(var archivo_empleados: t_archivo_empleados);
var
  registro_empleado: t_registro_empleado;
  numero: int16;
  ok: boolean;
begin
  numero:=1+random(1000);
  ok:=false;
  reset(archivo_empleados);
  while ((not eof(archivo_empleados)) and (ok=false)) do
  begin
    read(archivo_empleados,registro_empleado);
    if (registro_empleado.numero=numero) then
    begin
      registro_empleado.edad:=18+random(high(int8)-18);
      seek(archivo_empleados,filepos(archivo_empleados)-1);
      write(archivo_empleados,registro_empleado);
      ok:=true;
    end;
  end;
```

```
  close(archivo_empleados);
  if (ok=true) then
  begin
    textcolor(green); write('Se ha modificado la edad del empleado con número ');
textcolor(yellow); write(numero); textcolor(green); writeln(' en el archivo');
  end
  else
  begin
    textcolor(green); write('No se ha encontrado el empleado con número '); textcolor(yellow);
write(numero); textcolor(green); writeln(' en el archivo');
  end;
end;
procedure exportar_archivo_txt1(var archivo_empleados: t_archivo_empleados);
var
  registro_empleado: t_registro_empleado;
  archivo_txt: text;
begin
  reset(archivo_empleados);
  assign(archivo_txt,'todos_empleados.txt');
  rewrite(archivo_txt);
  while (not eof(archivo_empleados)) do
  begin
    read(archivo_empleados,registro_empleado);
    with registro_empleado do
      writeln(archivo_txt,'Número: ',numero,'; Apellido: ',apellido,'; Nombre: ',nombre,';
Edad: ',edad,'; DNI: ',dni);
  end;
  close(archivo_empleados);
  close(archivo_txt);
  textcolor(green); write('Se ha exportado el contenido del archivo al archivo de texto
llamado '); textcolor(yellow); writeln('"todos_empleados.txt"');
end;
procedure exportar_archivo_txt2(var archivo_empleados: t_archivo_empleados);
var
  registro_empleado: t_registro_empleado;
  archivo_txt: text;
begin
  reset(archivo_empleados);
  assign(archivo_txt,'faltaDNIEmpleado.txt');
  rewrite(archivo_txt);
  while (not eof(archivo_empleados)) do
  begin
    read(archivo_empleados,registro_empleado);
    if (registro_empleado.dni=dni_corte) then
      with registro_empleado do
        writeln(archivo_txt,'Número: ',numero,'; Apellido: ',apellido,'; Nombre: ',nombre,';
Edad: ',edad,'; DNI: ',dni);
  end;
  close(archivo_empleados);
  close(archivo_txt);
  textcolor(green); write('Se ha exportado a un archivo de texto llamado ');
textcolor(yellow); write('"faltaDNIEmpleado.txt"'); textcolor(green); writeln(' los empleados
que no tienen cargado el DNI (DNI en 00)');
end;
procedure leer_opcion(var opcion: int8);
begin
  textcolor(red); writeln('MENÚ DE OPCIONES');
  textcolor(yellow); write('OPCIÓN 1: '); textcolor(green); writeln('Crear un archivo de
registros no ordenados de empleados y completarlo con datos ingresados desde teclado');
  textcolor(yellow); write('OPCIÓN 2: '); textcolor(green); writeln('Listar en pantalla los
datos de empleados que tengan un nombre o apellido determinado');
  textcolor(yellow); write('OPCIÓN 3: '); textcolor(green); writeln('Listar en pantalla los
empleados de a uno por línea');
  textcolor(yellow); write('OPCIÓN 4: '); textcolor(green); writeln('Listar en pantalla los
empleados mayores a 70 años, próximos a jubilarse');
```

```
  textcolor(yellow); write('OPCIÓN 5: '); textcolor(green); writeln('Añadir uno o más
empleados al final del archivo con sus datos ingresados por teclado');
  textcolor(yellow); write('OPCIÓN 6: '); textcolor(green); writeln('Modificar la edad de un
empleado dado');
  textcolor(yellow); write('OPCIÓN 7: '); textcolor(green); writeln('Exportar el contenido del
archivo a un archivo de texto llamado “todos_empleados.txt”');
  textcolor(yellow); write('OPCIÓN 8: '); textcolor(green); writeln('Exportar a un archivo de
texto llamado “faltaDNIEmpleado.txt” los empleados que no tengan cargado el DNI (DNI en 00)');
  textcolor(yellow); write('OPCIÓN 0: '); textcolor(green); writeln('Salir del menú de
opciones');
  textcolor(green); write('Introducir opción elegida: '); textcolor(yellow); readln(opcion);
  writeln();
end;
procedure menu_opciones(var archivo_empleados: t_archivo_empleados);
var
  opcion: int8;
begin
  leer_opcion(opcion);
  while(opcion<>opcion_salida) do
  begin
    case opcion of
      1: cargar_archivo_empleados(archivo_empleados);
      2: imprimir1_archivo_empleados(archivo_empleados);
      3: imprimir2_archivo_empleados(archivo_empleados);
      4: imprimir3_archivo_empleados(archivo_empleados);
      5: agregar_empleado(archivo_empleados);
      6: modificar_edad_empleado(archivo_empleados);
      7: exportar_archivo_txt1(archivo_empleados);
      8: exportar_archivo_txt2(archivo_empleados);
    else
        textcolor(green); writeln('La opción ingresada no corresponde a ninguna de las
mostradas en el menú de opciones');
    end;
    writeln();
    leer_opcion(opcion);
  end;
end;
var
  archivo_empleados: t_archivo_empleados;
begin
  randomize;
  assign(archivo_empleados,'empleados');
  menu_opciones(archivo_empleados);
end.
```

Ejercicio 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, precio y marca; en la segunda, stock disponible, stock mínimo y descripción; y, en la tercera, nombre en ese orden. Cada celular se carga leyendo tres líneas del archivo "celulares.txt".

```
program TP1_E5;
{$codepage UTF8}
uses crt, sysutils;
const
  codigo_salida=0;
  opcion_salida=0;
type
  t_string10=string[10];
  t_string20=string[20];
  t_registro_celular=record
    codigo: int16;
    nombre: t_string10;
    descripcion: t_string20;
    marca: t_string10;
    precio: real;
    stock_minimo: int16;
    stock_disponible: int16;
  end;
  t_archivo_celulares=file of t_registro_celular;
function random_string(length: int8): t_string10;
var
  i: int8;
  string_aux: string;
begin
  string_aux:='';
  for i:= 1 to length do
    string_aux:=string_aux+chr(ord('A')+random(26));
```

```
  random_string:=string_aux;
end;
procedure leer_celular(var registro_celular: t_registro_celular);
var
  vector_marcas: array[1..10] of t_string10=('Alcatel', 'Apple', 'Huawei', 'Lenovo', 'LG',
'Motorola', 'Nokia', 'Samsung', 'Sony', 'Xiaomi');
  vector_descripciones: array[1..5] of t_string20=('Gama baja', 'Gama media baja', 'Gama
media', 'Gama media alta', 'Gama alta');
  i: int8;
begin
  i:=random(100);
  if (i=0) then
    registro_celular.codigo:=codigo_salida
  else
    registro_celular.codigo:=1+random(1000);
  if (registro_celular.codigo<>codigo_salida) then
  begin
    registro_celular.nombre:=random_string(5+random(5));
    registro_celular.descripcion:=vector_descripciones[1+random(5)];
    registro_celular.marca:=vector_marcas[1+random(10)];
    registro_celular.precio:=100+random(9001)/10;
    registro_celular.stock_minimo:=1+random(10);
    registro_celular.stock_disponible:=random(101);
  end;
end;
procedure cargar_archivo_carga(var archivo_carga: text);
var
  registro_celular: t_registro_celular;
begin
  rewrite(archivo_carga);
  leer_celular(registro_celular);
  while (registro_celular.codigo<>codigo_salida) do
  begin
    with registro_celular do
    begin
      writeln(archivo_carga,codigo,' ',precio:0:2,' ',marca);
      writeln(archivo_carga,stock_disponible,' ',stock_minimo,' ',descripcion);
      writeln(archivo_carga,nombre);
    end;
    leer_celular(registro_celular);
  end;
  close(archivo_carga);
end;
procedure cargar_archivo_celulares(var archivo_celulares: t_archivo_celulares; var
archivo_carga: text);
var
  registro_celular: t_registro_celular;
begin
  rewrite(archivo_celulares);
  reset(archivo_carga);
  while (not eof(archivo_carga)) do
    with registro_celular do
    begin
      readln(archivo_carga,codigo,precio,marca); marca:=trim(marca);
      readln(archivo_carga,stock_disponible,stock_minimo,descripcion);
descripcion:=trim(descripcion);
      readln(archivo_carga,nombre);
      write(archivo_celulares,registro_celular);
    end;
  close(archivo_celulares);
  close(archivo_carga);
  textcolor(green); writeln('El archivo binario de celulares fue creado y cargado con éxito');
end;
procedure imprimir_registro_celular(registro_celular: t_registro_celular);
begin
  textcolor(green); write('Código: '); textcolor(yellow); write(registro_celular.codigo);
```

```
  textcolor(green); write('; Nombre: '); textcolor(yellow); write(registro_celular.nombre);
  textcolor(green); write('; Descripción: '); textcolor(yellow);
write(registro_celular.descripcion);
  textcolor(green); write('; Marca: '); textcolor(yellow); write(registro_celular.marca);
  textcolor(green); write('; Precio: '); textcolor(yellow);
write(registro_celular.precio:0:2);
  textcolor(green); write('; Stock mínimo: '); textcolor(yellow);
write(registro_celular.stock_minimo);
  textcolor(green); write('; Stock disponible: '); textcolor(yellow);
writeln(registro_celular.stock_disponible);
end;
procedure imprimir1_archivo_celulares(var archivo_celulares: t_archivo_celulares);
var
  registro_celular: t_registro_celular;
begin
  reset(archivo_celulares);
  textcolor(green); writeln('Los datos de aquellos celulares que tienen un stock menor al
stock mínimo son: ');
  while (not eof(archivo_celulares)) do
  begin
    read(archivo_celulares,registro_celular);
    if (registro_celular.stock_disponible<registro_celular.stock_minimo) then
      imprimir_registro_celular(registro_celular);
  end;
  close(archivo_celulares);
end;
procedure imprimir2_archivo_celulares(var archivo_celulares: t_archivo_celulares);
var
  registro_celular: t_registro_celular;
  vector_descripciones: array[1..5] of t_string20=('Gama baja', 'Gama media baja', 'Gama
media', 'Gama media alta', 'Gama alta');
  descripcion: t_string20;
begin
  reset(archivo_celulares);
  descripcion:=vector_descripciones[1+random(5)];
  textcolor(green); write('Los celulares del archivo cuya descripción contiene la cadena de
caracteres '); textcolor(yellow); write(descripcion); textcolor(green); writeln(' son: ');
  while (not eof(archivo_celulares)) do
  begin
    read(archivo_celulares,registro_celular);
    if (registro_celular.descripcion=descripcion) then
      imprimir_registro_celular(registro_celular);
  end;
  close(archivo_celulares);
end;
procedure exportar_archivo_txt(var archivo_celulares: t_archivo_celulares);
var
  registro_celular: t_registro_celular;
  archivo_txt: text;
begin
  reset(archivo_celulares);
  assign(archivo_txt,'celulares2.txt');
  rewrite(archivo_txt);
  while (not eof(archivo_celulares)) do
  begin
    read(archivo_celulares,registro_celular);
    with registro_celular do
    begin
      writeln(archivo_txt,codigo,' ',precio:0:2,' ',marca);
      writeln(archivo_txt,stock_disponible,' ',stock_minimo,' ',descripcion);
      writeln(archivo_txt,nombre);
    end;
  end;
  close(archivo_celulares);
  close(archivo_txt);
```

```
  textcolor(green); write('Se ha exportado el archivo creado en el inciso (a) a un archivo de
texto denominado '); textcolor(yellow); write('"celulares2.txt"'); textcolor(green); writeln('
con todos los celulares del mismo');
end;
procedure leer_opcion(var opcion: int8);
begin
  textcolor(red); writeln('MENÚ DE OPCIONES');
  textcolor(yellow); write('OPCIÓN 1: '); textcolor(green); writeln('Crear un archivo de
registros no ordenados de celulares y cargarlo con datos ingresados desde un archivo de texto
denominado "celulares1.txt"');
  textcolor(yellow); write('OPCIÓN 2: '); textcolor(green); writeln('Listar en pantalla los
datos de aquellos celulares que tengan un stock menor al stock mínimo');
  textcolor(yellow); write('OPCIÓN 3: '); textcolor(green); writeln('Listar en pantalla los
celulares del archivo cuya descripción contenga una cadena de caracteres proporcionada por el
usuario');
  textcolor(yellow); write('OPCIÓN 4: '); textcolor(green); writeln('Exportar el archivo
creado en el inciso (a) a un archivo de texto denominado "celulares2.txt" con todos los
celulares del mismo');
  textcolor(yellow); write('OPCIÓN 0: '); textcolor(green); writeln('Salir del menú de
opciones');
  textcolor(green); write('Introducir opción elegida: '); textcolor(yellow); readln(opcion);
  writeln();
end;
procedure menu_opciones(var archivo_celulares: t_archivo_celulares; var archivo_carga: text);
var
  opcion: int8;
begin
  leer_opcion(opcion);
  while(opcion<>opcion_salida) do
  begin
    case opcion of
      1: cargar_archivo_celulares(archivo_celulares,archivo_carga);
      2: imprimir1_archivo_celulares(archivo_celulares);
      3: imprimir2_archivo_celulares(archivo_celulares);
      4: exportar_archivo_txt(archivo_celulares);
    else
        textcolor(green); writeln('La opción ingresada no corresponde a ninguna de las
mostradas en el menú de opciones');
    end;
    writeln();
    leer_opcion(opcion);
  end;
end;
var
  archivo_celulares: t_archivo_celulares;
  archivo_carga: text;
begin
  randomize;
  assign(archivo_carga,'celulares1.txt');
  cargar_archivo_carga(archivo_carga);
  assign(archivo_celulares,'celulares2.txt');
  menu_opciones(archivo_celulares,archivo_carga);
end.
```

Ejercicio 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.*

```
program TP1_E6;
{$codepage UTF8}
uses crt, sysutils;
const
  codigo_salida=0;
  stock_disponible_corte=0;
  opcion_salida=0;
type
  t_string10=string[10];
  t_string20=string[20];
  t_registro_celular=record
    codigo: int16;
    nombre: t_string10;
    descripcion: t_string20;
    marca: t_string10;
    precio: real;
    stock_minimo: int16;
    stock_disponible: int16;
  end;
  t_archivo_celulares=file of t_registro_celular;
function random_string(length: int8): t_string10;
var
  i: int8;
  string_aux: string;
begin
  string_aux:='';
  for i:= 1 to length do
    string_aux:=string_aux+chr(ord('A')+random(26));
  random_string:=string_aux;
end;
procedure leer_celular(var registro_celular: t_registro_celular);
var
  vector_marcas: array[1..10] of t_string10=('Alcatel', 'Apple', 'Huawei', 'Lenovo', 'LG',
'Motorola', 'Nokia', 'Samsung', 'Sony', 'Xiaomi');
  vector_descripciones: array[1..5] of t_string20=('Gama baja', 'Gama media baja', 'Gama
media', 'Gama media alta', 'Gama alta');
  i: int8;
begin
  i:=random(100);
  if (i=0) then
    registro_celular.codigo:=codigo_salida
  else
    registro_celular.codigo:=1+random(1000);
  if (registro_celular.codigo<>codigo_salida) then
  begin
    registro_celular.nombre:=random_string(5+random(5));
    registro_celular.descripcion:=vector_descripciones[1+random(5)];
    registro_celular.marca:=vector_marcas[1+random(10)];
    registro_celular.precio:=100+random(9001)/10;
    registro_celular.stock_minimo:=1+random(10);
```

```
    registro_celular.stock_disponible:=random(101);
  end;
end;
procedure cargar_archivo_carga(var archivo_carga: text);
var
  registro_celular: t_registro_celular;
begin
  rewrite(archivo_carga);
  leer_celular(registro_celular);
  while (registro_celular.codigo<>codigo_salida) do
  begin
    with registro_celular do
    begin
      writeln(archivo_carga,codigo,' ',precio:0:2,' ',marca);
      writeln(archivo_carga,stock_disponible,' ',stock_minimo,' ',descripcion);
      writeln(archivo_carga,nombre);
    end;
    leer_celular(registro_celular);
  end;
  close(archivo_carga);
end;
procedure cargar_archivo_celulares(var archivo_celulares: t_archivo_celulares; var
archivo_carga: text);
var
  registro_celular: t_registro_celular;
begin
  rewrite(archivo_celulares);
  reset(archivo_carga);
  while (not eof(archivo_carga)) do
    with registro_celular do
    begin
      readln(archivo_carga,codigo,precio,marca); marca:=trim(marca);
      readln(archivo_carga,stock_disponible,stock_minimo,descripcion);
descripcion:=trim(descripcion);
      readln(archivo_carga,nombre);
      write(archivo_celulares,registro_celular);
    end;
  close(archivo_celulares);
  close(archivo_carga);
  textcolor(green); writeln('El archivo binario de celulares fue creado y cargado con éxito');
end;
procedure imprimir_registro_celular(registro_celular: t_registro_celular);
begin
  textcolor(green); write('Código: '); textcolor(yellow); write(registro_celular.codigo);
  textcolor(green); write('; Nombre: '); textcolor(yellow); write(registro_celular.nombre);
  textcolor(green); write('; Descripción: '); textcolor(yellow);
write(registro_celular.descripcion);
  textcolor(green); write('; Marca: '); textcolor(yellow); write(registro_celular.marca);
  textcolor(green); write('; Precio: '); textcolor(yellow);
write(registro_celular.precio:0:2);
  textcolor(green); write('; Stock mínimo: '); textcolor(yellow);
write(registro_celular.stock_minimo);
  textcolor(green); write('; Stock disponible: '); textcolor(yellow);
writeln(registro_celular.stock_disponible);
end;
procedure imprimir1_archivo_celulares(var archivo_celulares: t_archivo_celulares);
var
  registro_celular: t_registro_celular;
begin
  reset(archivo_celulares);
  textcolor(green); writeln('Los datos de aquellos celulares que tienen un stock menor al
stock mínimo son: ');
  while (not eof(archivo_celulares)) do
  begin
    read(archivo_celulares,registro_celular);
    if (registro_celular.stock_disponible<registro_celular.stock_minimo) then
```

```
      imprimir_registro_celular(registro_celular);
  end;
  close(archivo_celulares);
end;
procedure imprimir2_archivo_celulares(var archivo_celulares: t_archivo_celulares);
var
  registro_celular: t_registro_celular;
  vector_descripciones: array[1..5] of t_string20=('Gama baja', 'Gama media baja', 'Gama
media', 'Gama media alta', 'Gama alta');
  descripcion: t_string20;
begin
  reset(archivo_celulares);
  descripcion:=vector_descripciones[1+random(5)];
  textcolor(green); write('Los celulares del archivo cuya descripción contiene la cadena de
caracteres '); textcolor(yellow); write(descripcion); textcolor(green); writeln(' son: ');
  while (not eof(archivo_celulares)) do
  begin
    read(archivo_celulares,registro_celular);
    if (registro_celular.descripcion=descripcion) then
      imprimir_registro_celular(registro_celular);
  end;
  close(archivo_celulares);
end;
procedure exportar_archivo_txt1(var archivo_celulares: t_archivo_celulares);
var
  registro_celular: t_registro_celular;
  archivo_txt: text;
begin
  reset(archivo_celulares);
  assign(archivo_txt,'celulares2.txt');
  rewrite(archivo_txt);
  while (not eof(archivo_celulares)) do
  begin
    read(archivo_celulares,registro_celular);
    with registro_celular do
    begin
      writeln(archivo_txt,codigo,' ',precio:0:2,' ',marca);
      writeln(archivo_txt,stock_disponible,' ',stock_minimo,' ',descripcion);
      writeln(archivo_txt,nombre);
    end;
  end;
  close(archivo_celulares);
  close(archivo_txt);
  textcolor(green); write('Se ha exportado el archivo creado en el inciso (a) a un archivo de
texto denominado '); textcolor(yellow); write('"celulares2.txt"'); textcolor(green); writeln('
con todos los celulares del mismo');
end;
function control_unicidad(var archivo_celulares: t_archivo_celulares; codigo: int16): boolean;
var
  registro_celular: t_registro_celular;
  ok: boolean;
begin
  ok:=false;
  while ((not eof(archivo_celulares)) and (ok=false)) do
  begin
    read(archivo_celulares,registro_celular);
    if (registro_celular.codigo=codigo) then
      ok:=true;
  end;
  control_unicidad:=ok;
end;
procedure agregar_celular(var archivo_celulares: t_archivo_celulares);
var
  registro_celular: t_registro_celular;
  celulares: int16;
begin
```

```
  celulares:=0;
  reset(archivo_celulares);
  leer_celular(registro_celular);
  while (registro_celular.codigo<>codigo_salida) do
  begin
    if (control_unicidad(archivo_celulares,registro_celular.codigo)=false) then
    begin
      seek(archivo_celulares,filesize(archivo_celulares));
      write(archivo_celulares,registro_celular);
      celulares:=celulares+1;
    end;
    leer_celular(registro_celular);
  end;
  close(archivo_celulares);
  textcolor(green); write('Se han agregado '); textcolor(yellow); write(celulares);
textcolor(green); writeln(' celulares al final del archivo');
end;
procedure modificar_stock_celular(var archivo_celulares: t_archivo_celulares);
var
  registro_celular: t_registro_celular;
  codigo: int16;
  ok: boolean;
begin
  codigo:=1+random(1000);
  ok:=false;
  reset(archivo_celulares);
  while ((not eof(archivo_celulares)) and (ok=false)) do
  begin
    read(archivo_celulares,registro_celular);
    if (registro_celular.codigo=codigo) then
    begin
      registro_celular.stock_disponible:=random(101);
      seek(archivo_celulares,filepos(archivo_celulares)-1);
      write(archivo_celulares,registro_celular);
      ok:=true;
    end;
  end;
  close(archivo_celulares);
  if (ok=true) then
  begin
    textcolor(green); write('Se ha modificado el stock del celular con código ');
textcolor(yellow); write(codigo); textcolor(green); writeln(' en el archivo');
  end
  else
  begin
    textcolor(green); write('No se ha encontrado el celular con código '); textcolor(yellow);
write(codigo); textcolor(green); writeln(' en el archivo');
  end;
end;
procedure exportar_archivo_txt2(var archivo_celulares: t_archivo_celulares);
var
  registro_celular: t_registro_celular;
  archivo_txt: text;
begin
  reset(archivo_celulares);
  assign(archivo_txt,'SinStock.txt');
  rewrite(archivo_txt);
  while (not eof(archivo_celulares)) do
  begin
    read(archivo_celulares,registro_celular);
    if (registro_celular.stock_disponible=stock_disponible_corte) then
      with registro_celular do
      begin
        writeln(archivo_txt,codigo,' ',precio:0:2,' ',marca);
        writeln(archivo_txt,stock_disponible,' ',stock_minimo,' ',descripcion);
        writeln(archivo_txt,nombre);
```

```
      end;
  end;
  close(archivo_celulares);
  close(archivo_txt);
  textcolor(green); write('Se ha exportado el contenido del archivo binario a un archivo de
texto denominado '); textcolor(yellow); write('"SinStock.txt"'); textcolor(green); writeln('
con aquellos celulares que tienen stock 0');
end;
procedure leer_opcion(var opcion: int8);
begin
  textcolor(red); writeln('MENÚ DE OPCIONES');
  textcolor(yellow); write('OPCIÓN 1: '); textcolor(green); writeln('Crear un archivo de
registros no ordenados de celulares y cargarlo con datos ingresados desde un archivo de texto
denominado "celulares1.txt"');
  textcolor(yellow); write('OPCIÓN 2: '); textcolor(green); writeln('Listar en pantalla los
datos de aquellos celulares que tengan un stock menor al stock mínimo');
  textcolor(yellow); write('OPCIÓN 3: '); textcolor(green); writeln('Listar en pantalla los
celulares del archivo cuya descripción contenga una cadena de caracteres proporcionada por el
usuario');
  textcolor(yellow); write('OPCIÓN 4: '); textcolor(green); writeln('Exportar el archivo
creado en el inciso (a) a un archivo de texto denominado "celulares2.txt" con todos los
celulares del mismo');
  textcolor(yellow); write('OPCIÓN 5: '); textcolor(green); writeln('Añadir uno o más
celulares al final del archivo con sus datos ingresados por teclado');
  textcolor(yellow); write('OPCIÓN 6: '); textcolor(green); writeln('Modificar el stock de un
celular dado');
  textcolor(yellow); write('OPCIÓN 7: '); textcolor(green); writeln('Exportar el contenido del
archivo binario a un archivo de texto denominado "SinStock.txt" con aquellos celulares que
tengan stock 0');
  textcolor(yellow); write('OPCIÓN 0: '); textcolor(green); writeln('Salir del menú de
opciones');
  textcolor(green); write('Introducir opción elegida: '); textcolor(yellow); readln(opcion);
  writeln();
end;
procedure menu_opciones(var archivo_celulares: t_archivo_celulares; var archivo_carga: text);
var
  opcion: int8;
begin
  leer_opcion(opcion);
  while(opcion<>opcion_salida) do
  begin
    case opcion of
      1: cargar_archivo_celulares(archivo_celulares,archivo_carga);
      2: imprimir1_archivo_celulares(archivo_celulares);
      3: imprimir2_archivo_celulares(archivo_celulares);
      4: exportar_archivo_txt1(archivo_celulares);
      5: agregar_celular(archivo_celulares);
      6: modificar_stock_celular(archivo_celulares);
      7: exportar_archivo_txt2(archivo_celulares);
    else
        textcolor(green); writeln('La opción ingresada no corresponde a ninguna de las
mostradas en el menú de opciones');
    end;
    writeln();
    leer_opcion(opcion);
  end;
end;
var
  archivo_celulares: t_archivo_celulares;
  archivo_carga: text;
begin
  randomize;
  assign(archivo_carga,'celulares1.txt');
  cargar_archivo_carga(archivo_carga);
  assign(archivo_celulares,'celulares2');
  menu_opciones(archivo_celulares,archivo_carga);
```

```
end.
```

Ejercicio 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.

```
program TP1_E7;
{$codepage UTF8}
uses crt, sysutils;
const
  codigo_salida=0;
  opcion_salida=0;
type
  t_string20=string[20];
  t_registro_novela=record
    codigo: int16;
    nombre: t_string20;
    genero: t_string20;
    precio: real;
  end;
  t_archivo_novelas=file of t_registro_novela;
function random_string(length: int8): t_string20;
var
  i: int8;
  string_aux: string;
begin
  string_aux:='';
  for i:= 1 to length do
    string_aux:=string_aux+chr(ord('A')+random(26));
  random_string:=string_aux;
end;
procedure leer_novela(var registro_novela: t_registro_novela; ok: boolean);
var
  i: int8;
begin
  if (ok=true) then
  begin
    i:=random(100);
    if (i=0) then
      registro_novela.codigo:=codigo_salida
    else
      registro_novela.codigo:=1+random(1000);
  end;
  if (registro_novela.codigo<>codigo_salida) then
  begin
    registro_novela.nombre:=random_string(5+random(15));
    registro_novela.genero:=random_string(5+random(15));
    registro_novela.precio:=100+random(9001)/10;
  end;
end;
```

```
procedure cargar_archivo_carga(var archivo_carga: text);
var
  registro_novela: t_registro_novela;
begin
  rewrite(archivo_carga);
  leer_novela(registro_novela,true);
  while (registro_novela.codigo<>codigo_salida) do
  begin
    with registro_novela do
    begin
      writeln(archivo_carga,codigo,' ',precio:0:2,' ',genero);
      writeln(archivo_carga,nombre);
    end;
    leer_novela(registro_novela,true);
  end;
  close(archivo_carga);
end;
procedure cargar_archivo_novelas(var archivo_novelas: t_archivo_novelas; var archivo_carga:
text);
var
  registro_novela: t_registro_novela;
begin
  rewrite(archivo_novelas);
  reset(archivo_carga);
  while (not eof(archivo_carga)) do
  begin
    with registro_novela do
    begin
      readln(archivo_carga,codigo,precio,genero); genero:=trim(genero);
      readln(archivo_carga,nombre);
    end;
    write(archivo_novelas,registro_novela);
  end;
  close(archivo_novelas);
  close(archivo_carga);
  textcolor(green); writeln('El archivo binario de novelas fue creado y cargado con éxito');
end;
function control_unicidad(var archivo_novelas: t_archivo_novelas; codigo: int16): boolean;
var
  registro_novela: t_registro_novela;
  ok: boolean;
begin
  ok:=false;
  while ((not eof(archivo_novelas)) and (ok=false)) do
  begin
    read(archivo_novelas,registro_novela);
    if (registro_novela.codigo=codigo) then
      ok:=true;
  end;
  control_unicidad:=ok;
end;
procedure agregar_novela(var archivo_novelas: t_archivo_novelas);
var
  registro_novela: t_registro_novela;
  novelas: int16;
begin
  novelas:=0;
  reset(archivo_novelas);
  leer_novela(registro_novela,true);
  while (registro_novela.codigo<>codigo_salida) do
  begin
    if (control_unicidad(archivo_novelas,registro_novela.codigo)=false) then
    begin
      seek(archivo_novelas,filesize(archivo_novelas));
      write(archivo_novelas,registro_novela);
      novelas:=novelas+1;
```

```
    end;
    leer_novela(registro_novela,true);
  end;
  close(archivo_novelas);
  textcolor(green); write('Se han agregado '); textcolor(yellow); write(novelas);
textcolor(green); writeln(' novelas al final del archivo');
end;
procedure modificar_novela(var archivo_novelas: t_archivo_novelas);
var
  registro_novela: t_registro_novela;
  codigo: int16;
  ok: boolean;
begin
  codigo:=1+random(1000);
  ok:=false;
  reset(archivo_novelas);
  while ((not eof(archivo_novelas)) and (ok=false)) do
  begin
    read(archivo_novelas,registro_novela);
    if (registro_novela.codigo=codigo) then
    begin
      leer_novela(registro_novela,false);
      seek(archivo_novelas,filepos(archivo_novelas)-1);
      write(archivo_novelas,registro_novela);
      ok:=true;
    end;
  end;
  close(archivo_novelas);
  if (ok=true) then
  begin
    textcolor(green); write('Se ha modificado la novela con código '); textcolor(yellow);
write(codigo); textcolor(green); writeln(' en el archivo');
  end
  else
  begin
    textcolor(green); write('No se ha encontrado la novela con código '); textcolor(yellow);
write(codigo); textcolor(green); writeln(' en el archivo');
  end;
end;
procedure exportar_archivo_txt(var archivo_novelas: t_archivo_novelas);
var
  registro_novela: t_registro_novela;
  archivo_txt: text;
begin
  reset(archivo_novelas);
  assign(archivo_txt,'novelas2.txt');
  rewrite(archivo_txt);
  while (not eof(archivo_novelas)) do
  begin
    read(archivo_novelas,registro_novela);
    with registro_novela do
    begin
      writeln(archivo_txt,codigo,' ',precio:0:2,' ',genero);
      writeln(archivo_txt,nombre);
    end;
  end;
  close(archivo_novelas);
  close(archivo_txt);
  textcolor(green); write('Se ha exportado el archivo creado en el inciso (a) a un archivo de
texto denominado '); textcolor(yellow); write('"novelas2.txt"'); textcolor(green); writeln('
con todas las novelas del mismo');
end;
procedure leer_opcion(var opcion: int8);
begin
  textcolor(red); writeln('MENÚ DE OPCIONES');
```

```
  textcolor(yellow); write('OPCIÓN 1: '); textcolor(green); writeln('Crear un archivo binario
a partir de la información almacenada en un archivo de texto denominado "novelas.txt"');
  textcolor(yellow); write('OPCIÓN 2: '); textcolor(green); writeln('Añadir una o más novelas
al final del archivo con sus datos ingresados por teclado');
  textcolor(yellow); write('OPCIÓN 3: '); textcolor(green); writeln('Modificar una novela
existente');
  textcolor(yellow); write('OPCIÓN 4: '); textcolor(green); writeln('Exportar el archivo
creado en el inciso (a) a un archivo de texto denominado "novelas2.txt" con todas las novelas
del mismo');
  textcolor(yellow); write('OPCIÓN 0: '); textcolor(green); writeln('Salir del menú de
opciones');
  textcolor(green); write('Introducir opción elegida: '); textcolor(yellow); readln(opcion);
  writeln();
end;
procedure menu_opciones(var archivo_novelas: t_archivo_novelas; var archivo_carga: text);
var
  opcion: int8;
begin
  leer_opcion(opcion);
  while(opcion<>opcion_salida) do
  begin
    case opcion of
      1: cargar_archivo_novelas(archivo_novelas,archivo_carga);
      2: agregar_novela(archivo_novelas);
      3: modificar_novela(archivo_novelas);
      4: exportar_archivo_txt(archivo_novelas);
    else
        textcolor(green); writeln('La opción ingresada no corresponde a ninguna de las
mostradas en el menú de opciones');
    end;
    writeln();
    leer_opcion(opcion);
  end;
end;
var
  archivo_novelas: t_archivo_novelas;
  archivo_carga: text;
begin
  randomize;
  assign(archivo_carga,'novelas1.txt');
  cargar_archivo_carga(archivo_carga);
  assign(archivo_novelas,'novelas2');
  menu_opciones(archivo_novelas,archivo_carga);
end.
```

Juan Menduiña

Trabajo Práctico N° 2:
Archivos Secuenciales Ordenados - Algorítmica Clásica.

Ejercicio 1.

Ejercicio 2.

Ejercicio 3.

Ejercicio 4.

Ejercicio 5.

Ejercicio 6.

Ejercicio 7.

Ejercicio 8.

Ejercicio 9.

Ejercicio 10.

Ejercicio 11.

Ejercicio 12.

Ejercicio 13.

Ejercicio 14.

Ejercicio 15.

Ejercicio 16.

Ejercicio 17.

Ejercicio 18.

Ejercicio 19.

Juan Menduiña

Trabajo Práctico N° 3:

.

Ejercicio 1.

Juan Menduiña

Trabajo Práctico N° 4:

.

Ejercicio 1.

Juan Menduiña

Trabajo Práctico N° 5:

.

Ejercicio 1.