Guía de Trabajos Prácticos de Pascal

# Parte 3

Cada ejercicio debe ser copiado/escrito en la versión de Pascal que posean y luego compilarlo, probarlo y “decorarlo”

1) Usando el tipo de dato Record, cargar 3 libros en memoria y después realizar un listado de todos los libros. De los libros se deben almacenar: El título, el nombre del autor, un breve resumen y el id del libro.

Teniendo el siguiente código, realice el codigo de una subacción donde el usuario pueda cargar a mano los 3 libros.

program testrecor01;

type

formato\_libro = record

titulo: string[50];

autor: string[50];

resumen: string[200];

libro\_id: longint;

end;

var

libro1, libro2, libro3 : formato\_libro;

procedure mostrarlibro(unlibro:formato\_libro);

begin

Writeln('Nombre: ', unlibro.titulo);

Writeln('Autor: ', unlibro.autor);

Writeln('resumen: ', unlibro.resumen);

Writeln('id del libro: ', unlibro.libro\_id);

end;

BEGIN

libro1.titulo:= 'El increible libro';

libro1.autor:= 'Don increible';

libro1.resumen:= 'Un libro increible.';

libro1.libro\_id:= 423232;

libro2.titulo:= 'Otra titulo';

libro2.autor:= 'El otro autor';

libro2.resumen:= 'un libro cuando no esta el primero.';

libro2.libro\_id:= 342345;

libro3.titulo:= 'Este es un titulo';

libro3.autor:= 'Este es el autor';

libro3.resumen:= 'Esto, es un pequeño resumen del libro.';

libro3.libro\_id:= 234534;

mostrarlibro(libro1);

writeln();

mostrarlibro(libro2);

writeln();

mostrarlibro(libro3);

END.

2) Contar todas las palabras de una secuencia

Para poder ejecutar este código es necesario crear un archivo (con nombre: archivo.txt) en el mismo lugar donde se encuentra el código del programa.

Colocar en el archivo el siguiente texto:

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Mauris eu eros odio. Nunc venenatis nibh et dolor commodo, eu dictum libero malesuada. Nullam ac urna nisl. Quisque condimentum augue sagittis.

Código:

program ej2guia3;

var

sec: file of char;

v : char;

error, cont: integer;

begin

Assign (sec, 'archivo.txt');

{$I-}

Reset (sec);

error:= IOResult;

{$I+}

if error <> 0 then

begin

writeln('Error: el archivo archivo.txt exite?');

halt(2);

end;

cont:=0;

read(sec, v);

while not Eof (sec) do

begin

while (v = ' ') do

begin

read(sec, v);

end;

cont := cont + 1;

while (v <> ' ') do

begin

read(sec, v);

end;

end;

writeln('La cantidad de palabras es: ', cont);

close(sec)

end.

3) Mostrar el promedio de palabras que hay por oración.

Se requiere ocupar el mismo archivo dado anteriormente.

program ej3guia3;

var

sec: file of char;

v : char;

error, pal, ora: integer;

prom:real;

begin

Assign (sec, 'archivo.txt');

{$I-}

Reset (sec);

error:= IOResult;

{$I+}

if error <> 0 then

begin

writeln('Error: el archivo archivo.txt exite?');

halt(2);

end;

pal:=0;

ora:=0;

read(sec, v);

while not Eof (sec) do

begin

while ((not eof(sec)) and (v <> '.')) do

begin

while ((not eof(sec)) and (v = ' ')) do

begin

read(sec, v);

end;

pal := pal + 1;

while ((not eof(sec)) and(v <> ' ')) do

begin

read(sec, v);

end;

end;

if ((not eof(sec)) and (v = '.')) then

ora := ora + 1;

read(sec, v);

end;

if ora = 0 then

writeln('No hay oraciones en el texto')

else

begin

prom:=pal / ora;

writeln('El promedio de palabras por oraciones es: ', prom:4:3);

end;

close(sec)

end.

4) A partir de la secuencia anterior, genere una secuencia de salida (archivo2.txt) que posea todas las palabras que comienzan con vocal.

program ej4guia3;

const

vocales = ['a', 'e', 'i', 'o', 'u'];

var

sec, sal: file of char;

v : char;

error: integer;

esVocal: boolean;

begin

Assign (sal, 'archivo2.txt');

rewrite(sal);

Assign (sec, 'archivo.txt');

{$I-}

Reset (sec);

error:= IOResult;

{$I+}

if error <> 0 then

begin

writeln('Error: el archivo archivo.txt exite?');

halt(2);

end;

read(sec, v);

while not Eof (sec) do

begin

while ((not eof(sec)) and (v = ' ')) do

begin

read(sec, v);

end;

if ((not eof(sec)) and (LowerCase(v) in vocales)) then

esVocal := true;

while ((not eof(sec)) and (v <> ' ')) do

begin

if esVocal then

write(sal, v);

read(sec, v);

end;

esVocal := false;

end;

close(sal);

close(sec);

end.

5) Se posee una secuencia con los mensajes internos enviados entre los empleados de una empresa, la secuencia está formada de la siguiente manera: 3 caracteres para el destino, 3 caracteres para el origen, 3 caracteres que indican la cantidad de caracteres que tiene el mensaje y el mensaje. No existe ningún carácter que separe un mensaje de otro.

se pide:

1. Escribir una secuencia de salida con todos los mensajes (incluyendo origen y destino) que van hacia el área de Mantenimiento (que destino comience con 1).
2. Contar la cantidad de mensajes que se dirigen hacia el área de sistemas (que destino comience con 23).

Use la siguiente secuencia como ejemplo:

100001031vengan a arreglar esto rapido!120230043Existe una falla en el aire acondicionado.233100063El arreglo electrico recien va a poder ser posible el 23042018230001039No me anda la impresora, estoy apurado

program ej5guia3;

uses math;

var

sec, sal: file of char;

v : char;

i, error, cont: integer;

esMantenimiento, esSistemas: boolean;

long: integer;

function aEntero(caracter: char): integer;

begin

case caracter of

'0': aEntero:= 0;

'1': aEntero:= 1;

'2': aEntero:= 2;

'3': aEntero:= 3;

'4': aEntero:= 4;

'5': aEntero:= 5;

'6': aEntero:= 6;

'7': aEntero:= 7;

'8': aEntero:= 8;

'9': aEntero:= 9;

end;

end;

begin

Assign (sal, 'mensajesalida.txt');

rewrite(sal);

Assign (sec, 'mensaje.txt');

{$I-}

Reset (sec);

error:= IOResult;

{$I+}

if error <> 0 then

begin

writeln('Error: el archivo archivo.txt exite?');

halt(2);

end;

read(sec, v);

cont := 0;

while not eof(sec) do

begin

esMantenimiento := false;

esSistemas := false;

if v = '1' then

esMantenimiento := true

else

begin

if v = '2' then

esSistemas := true;

end;

if esMantenimiento then

write(sal, v);

read(sec, v);

if (v = '3') and esSistemas then

cont := cont + 1;

for i:= 1 to 4 do

begin

if esMantenimiento then

write(sal, v);

read(sec, v);

end;

long:= 0;

for i := 2 downto 0 do

begin

read(sec, v);

if esMantenimiento then

write(sal, v);

long := long + aEntero(v) \* round(Power(10, i));

end;

for i:= 1 to long do

begin

if esMantenimiento then

write(sal, v);

read(sec, v);

end;

if esMantenimiento then

write(sal, '#');

end;

writeln('cantidad de mensajes para Sistemas: ', cont);

close(sal);

close(sec);

end.

6) Realice un programa que cargue un archivo que contenga la definición de los libros dada en el ejercicio 1.

Mejorar el algoritmo para que el usuario tenga que ingresar si o no, para cuando quiere salir o cargar otro libro.

program ej6guia3;

type

formato\_libro = record

titulo: string[50];

autor: string[50];

resumen: string[200];

libro\_id: longint;

end;

var

arch: file of formato\_libro;

reg: formato\_libro;

ops: (‘si’, ‘no’);

BEGIN

Assign (arch, 'libros.dat');

rewrite(arch);

write('Quiere ingresar un libro? (Ingrese si/no): ');

readln(ops);

while ops = 'si' do

begin

Write('Ingrese un Titulo: ');

readln(reg.titulo);

Write('Ingrese un autor: ');

readln(reg.autor);

Write('Ingrese un Resumen: ');

readln(reg.resumen);

Write('Ingrese el ID del libro: ');

readln(reg.libro\_id);

write(arch, reg);

write('Quiere ingresar un libro? (Ingrese si/no): ');

readln(ops);

end;

close(arch);

END.

¿Existe una forma de mejorar este algoritmo para que cada vez que se ejecute permita al usuario agregar un nuevo libro al final del archivo sin borrarlo? que hay que modificar?

7) Realice un programa que lea el archivo anteriormente creado

Incorpora la funcionalidad de poder pasar al próximo libro cuando uno lo necesite.

program ej7guia3;

Uses sysutils;

type

formato\_libro = record

titulo: string[50];

autor: string[50];

resumen: string[200];

libro\_id: longint;

end;

var

arch: file of formato\_libro;

reg: formato\_libro;

error: integer;

procedure mostrarlibro(unlibro:formato\_libro);

begin

Writeln('Nombre: ', unlibro.titulo);

Writeln('Autor: ', unlibro.autor);

Writeln('resumen: ', unlibro.resumen);

Writeln('id del libro: ', unlibro.libro\_id);

end;

BEGIN

Assign (arch, 'libros.dat');

{$I-}

Reset (arch);

error:= IOResult;

{$I+}

if error <> 0 then

begin

writeln('Error: el archivo libros.dat existe?');

halt(2);

end;

read(arch, reg);

while not EOF(arch) do

begin

mostrarlibro(reg);

read(arch, reg);

end;

close(arch);

END.

8) Se tiene un archivo con los empleados de una empresa que posee los siguientes campos: Planta, Categoría y Nombre del empleado.

La categoría es un número entero que va del 1 al 4.

Se pide mostrar el nombre de cada trabajador, y además la cantidad de trabajadores que hay por categoría, planta y total general.

Para poder trabajar con este ejercicio se tiene que correr el siguiente programa que genera un archivo trabajadores.dat, con los datos necesarios para ejecutar el programa.

program ej8guia3Generador;

type

formato\_empleado = record

plan: string[20];

cate: 1..4;

nomb: string[50];

end;

var

arch: file of formato\_empleado;

reg: formato\_empleado;

procedure cargarEmpleado(var emp: formato\_empleado; p: string; c:smallint; n:string);

begin

emp.plan:=p;

emp.cate:=c;

emp.nomb:=n;

end;

BEGIN

Assign (arch, 'empleados.dat');

rewrite(arch);

cargarEmpleado(reg, 'a', 1, 'Aranda');

write(arch, reg);

cargarEmpleado(reg, 'a', 1, 'Vera');

write(arch, reg);

cargarEmpleado(reg, 'a', 2, 'Arce');

write(arch, reg);

cargarEmpleado(reg, 'a', 2, 'Basualdo');

write(arch, reg);

cargarEmpleado(reg, 'a', 4, 'Corrales');

write(arch, reg);

cargarEmpleado(reg, 'a', 4, 'Derqui');

write(arch, reg);

cargarEmpleado(reg, 'a', 4, 'Gutierrez');

write(arch, reg);

cargarEmpleado(reg, 'a', 4, 'Morales');

write(arch, reg);

cargarEmpleado(reg, 'b', 1, 'Biondi');

write(arch, reg);

cargarEmpleado(reg, 'b', 1, 'Casas');

write(arch, reg);

cargarEmpleado(reg, 'b', 1, 'Davis');

write(arch, reg);

cargarEmpleado(reg, 'b', 1, 'Nogueira');

write(arch, reg);

cargarEmpleado(reg, 'b', 2, 'Milán');

write(arch, reg);

cargarEmpleado(reg, 'b', 2, 'Molina');

write(arch, reg);

cargarEmpleado(reg, 'b', 2, 'Portales');

write(arch, reg);

cargarEmpleado(reg, 'b', 3, 'Cortezs');

write(arch, reg);

cargarEmpleado(reg, 'b', 3, 'Mendez');

write(arch, reg);

cargarEmpleado(reg, 'b', 3, 'Nogueira');

write(arch, reg);

cargarEmpleado(reg, 'b', 3, 'Salerno');

write(arch, reg);

cargarEmpleado(reg, 'b', 4, 'Casas');

write(arch, reg);

close(arch);

END.

Ahora el Algoritmo:

program ej8guia3Consumidor;

type

formato\_empleado = record

plan: string[20];

cate: 1..4;

nomb: string[50];

end;

var

arch: file of formato\_empleado;

reg: formato\_empleado;

error: integer;

res\_plan: string[20];

res\_cate: 1..4;

contcate, contplan, conttot: integer;

procedure corte\_cate();

begin

writeln('El total de empleados en la planta: ', res\_plan);

writeln('en la categoria: ', res\_cate, ' es: ', contcate);

contplan:= contplan + contcate;

contcate:= 0;

res\_cate:=reg.cate;

end;

procedure corte\_plan();

begin

writeln('El total de empleados en la planta: ', res\_plan);

writeln('Es: ', contcate);

conttot:= conttot + contplan;

contplan:= 0;

res\_plan:=reg.plan;

end;

BEGIN

Assign (arch, 'empleados.dat');

{$I-}

Reset (arch);

error:= IOResult;

{$I+}

if error <> 0 then

begin

writeln('Error: el archivo empleados.dat exite?');

halt(2);

end;

contcate:=0;

contplan:=0;

conttot:=0;

read(arch, reg);

res\_plan:=reg.plan;

res\_cate:=reg.cate;

while not EOF(arch) do

begin

writeln('Nombre: ',reg.nomb);

contcate := contcate + 1;

if res\_plan <> reg.plan then

corte\_plan

else

begin

if res\_cate <> reg.cate then

corte\_cate;

end;

read(arch, reg);

end;

corte\_plan;

writeln('El total de empleados es de: ', conttot);

close(arch);

END.

Teniendo el Algoritmo, realizar las siguientes acciones:

* Mejorar el formato del informe.
* Guardar todos los empleados de categoría 4 en un archivo de salida que posea el mismo formato que el archivo de empleados.
* Generar un archivo de salida con el siguiente formato: planta y cantidad de empleados.

9.a) Generar un archivo histórico de vendedores, que tenga: dni, nombre, cantidad de total productos vendidos y monto total recaudado. El archivo debe estar ordenado por DNI.

program ej9guia3Generador;

type

formato\_vendedor = record

dni: longint;

nom: string[50];

cant: integer;

reca: real;

end;

var

arch: file of formato\_vendedor;

function cargarVendedor(d: longint; n:string; c:integer; r:real): formato\_vendedor;

begin

cargarVendedor.dni:=d;

cargarVendedor.nom:=n;

cargarVendedor.cant:=c;

cargarVendedor.reca:=r;

end;

BEGIN

Assign (arch, 'vendedores.dat');

rewrite(arch);

write(arch, cargarVendedor(10485801, 'Perez', 227, 186.57));

write(arch, cargarVendedor(15259293, 'Gutierrez', 288, 222.00));

write(arch, cargarVendedor(17435881, 'Aranda', 206, 33.74));

write(arch, cargarVendedor(17607118, 'Molina', 149, 259.23));

write(arch, cargarVendedor(18421670, 'Salerno', 66, 61.04));

write(arch, cargarVendedor(19242781, 'Mendez', 29, 162.19));

write(arch, cargarVendedor(19942300, 'Corrales', 123, 37.03));

write(arch, cargarVendedor(22793396, 'Derqui', 239, 132.28));

write(arch, cargarVendedor(26907917, 'Biondi', 224, 134.40));

write(arch, cargarVendedor(27096144, 'Nogueira', 7, 85.03));

write(arch, cargarVendedor(27638155, 'Casas', 344, 23.00));

write(arch, cargarVendedor(27685501, 'Portales', 158, 187.35));

write(arch, cargarVendedor(29919799, 'Morales', 330, 211.50));

write(arch, cargarVendedor(31776747, 'Arce', 388, 155.00));

write(arch, cargarVendedor(33056132, 'Cortezs', 329, 89.93));

write(arch, cargarVendedor(34964968, 'Milan', 209, 72.52));

write(arch, cargarVendedor(37139946, 'Davis', 210, 1981.50));

write(arch, cargarVendedor(37279602, 'Dias', 104, 104.84));

write(arch, cargarVendedor(38399826, 'Vera', 319, 81.33));

write(arch, cargarVendedor(39849870, 'Basualdo', 272, 167.63));

close(arch);

END.

9.b) Genere un archivo que posea las ventas que realizaron los vendedores en el dia, de los cuales se necesita los siguientes campos: DNI del vendedor, cantidad de productos vendidos y total recaudado.

El archivo tiene que estar ordenado por la clave la cual es el DNI, solo puede existir cero o un movimiento por cada empleado.

program ej9guia3Generador;

type

formato\_venta = record

dni: longint;

cant: integer;

reca: real;

end;

var

arch: file of formato\_venta;

function cargarVenta(d: longint; c:integer; r:real): formato\_venta;

begin

cargarVenta.dni:=d;

cargarVenta.cant:=c;

cargarVenta.reca:=r;

end;

BEGIN

Assign (arch, 'movimientos.dat');

rewrite(arch);

write(arch, cargarVenta(10485801, 81, 59.33));

write(arch, cargarVenta(17435881, 39, 86.50));

write(arch, cargarVenta(18421670, 84, 7.67));

write(arch, cargarVenta(19242781, 54, 11.88));

write(arch, cargarVenta(19942300, 76, 46.00));

write(arch, cargarVenta(26907917, 79, 183.00));

write(arch, cargarVenta(27096144, 37, 28.00));

write(arch, cargarVenta(27638155, 48, 11.60));

write(arch, cargarVenta(27685501, 62, 24.57));

write(arch, cargarVenta(29819799, 27, 2.60));

write(arch, cargarVenta(31776747, 88, 18.14));

write(arch, cargarVenta(33056132, 81, 21.00));

write(arch, cargarVenta(35964968, 15, 27.25));

write(arch, cargarVenta(37139946, 46, 8.13));

write(arch, cargarVenta(37279602, 25, 9.33));

write(arch, cargarVenta(38399826, 66, 18.60));

write(arch, cargarVenta(39849870, 46, 31.67));

close(arch);

END.

9.c) Tomando los archivos anteriores (maestro 9.a y movimientos 9.b) genere un nuevo archivo de vendedores actualizado. El cual debe actualizar los campos de cantidad de productos y total recaudado cuando el DNI es igual. Puede darse el caso de que el trabajador no fue a trabajar ese día por lo tanto no hay que actualizar los campos. En caso de encontrar un DNI que no exista en el maestro, informar del error al usuario y continuar con el próximo.

Código:

program ej9guia3Generador;

const

HV=2147483647;

type

formato\_venta = record

dni: longint;

cant: integer;

reca: real;

end;

formato\_vendedor = record

dni: longint;

nom: string[50];

cant: integer;

reca: real;

end;

var

mae, maea: file of formato\_vendedor;

reg\_mae, reg\_maea: formato\_vendedor;

mov: file of formato\_venta;

reg\_mov: formato\_venta;

error: integer;

procedure leer\_mae();

begin

read(mae, reg\_mae);

if filepos(mae)<filesize(mae) then

reg\_mae.dni := HV;

end;

procedure leer\_mov();

begin

filepos(mov)<filesize(mov)

read(mov, reg\_mov);

if filepos(mov)<filesize(mov) then

reg\_mov.dni := HV;

end;

BEGIN

Assign (mov, 'movimientos.dat');

{$I-}

Reset (mov);

error:= IOResult;

{$I+}

if error <> 0 then

begin

writeln('Error: el archivo movimientos.dat exite?');

halt(2);

end;

Assign (mae, 'vendedores.dat');

{$I-}

Reset (mae);

error:= IOResult;

{$I+}

if error <> 0 then

begin

writeln('Error: el archivo vendedores.dat exite?');

halt(2);

end;

Assign (maea, 'vendedores\_actualizado.dat');

rewrite(maea);

leer\_mae();

leer\_mov();

while (reg\_mae.dni <> HV) or (reg\_mov.dni <> HV) do

begin

if reg\_mae.dni < reg\_mov.dni then

begin

reg\_maea := reg\_mae;

write(maea, reg\_maea);

leer\_mae();

end

else

begin

if reg\_mae.dni > reg\_mov.dni then

begin

writeln('Error, DNI no existe en archivo maestro');

leer\_mov();

end

else

begin

reg\_maea := reg\_mae;

reg\_maea.cant := reg\_maea.cant + reg\_mov.cant;

reg\_maea.reca := reg\_maea.reca + reg\_mov.reca;

write(maea, reg\_maea);

leer\_mae();

leer\_mov();

end;

end;

end;

close(maea);

close(mae);

close(mov);

END.

Modificar el algoritmo para que pueda:

* En caso de que aparezca un DNI que no exista en el maestro, generar la alta de ese nuevo vendedor, solicitando al usuario que ingrese su nombre por consola.
* Generar un archivo de salida que tenga el DNI, nombre del empleado y la comisión del dia, la comisión del dia sera del 10% del total recaudado, en caso de que el vendedor haya vendido más de 20 productos el porcentaje para a ser el 20%.
* Al finalizar mostrar por pantalla cual es el vendedor que tiene la mayor cantidad de productos vendidos y cuál fue el vendedor que recaudó más en el dia.