PROGRAMACIÓN I

Teoría – Cecilia Sanz

REGISTROS Y PROCESAMIENTO CON CORTE DE CONTROL

REGISTROS

OPERACIONES

ASIGNACIÓN

Estructura de datos – REGISTRO –

Operaciones

Con la variable REGISTRO

Sólo se puede realizar la operación de asignación.

```
peli1:= peli2;
```

siendo peli1 y peli2 del mismo tipo registro

Con los campos del registro

Las operaciones con los campos del registro son las que el tipo de campo permita. Para acceder a un campo se utiliza la notación calificada:

```
nombrevarregistro.campo
peli1.director
```

En este caso permite todas las operaciones válidas para un string.

Estructura de datos – REGISTRO

¿Cómo le asignamos valor a un registro?

```
Program uno;
Type  periodo=1950..2014;
      cate= 'A'..'E';
 pelicula = record
              titulo: string;
              director:string;
              año:periodo;
              categoria: cate;
            end;
```

```
Var
  p1:pelicula;
Begin
  p1.titulo:="La era del hielo";
  p1.director:= "Chris Wedge";
  p1.año:= 2002;
  p1.categoria:='A';
 End.
```

¿Qué pasa si no asignamos todos los campos?

REGISTROS

OPERACIONES

ENTRADA Y SALIDA

Estructura de datos – REGISTRO

¿Cómo le asignamos valor a un registro?

```
Program uno;
Type
 cadena= string[50];
 codcolor=1..10;
 auto = record
            marca: cadena;
            modelo:cadena;
            precio:real;
            color: codcolor;
        end;
```

```
Var
  a1:auto;
Begin
  read(a1.marca);
  read(a1.modelo);
  read(a1.precio)
  read(a1.color);
End.
```

¿Qué pasa si no leemos todos los campos?

NO SE PUEDE
read(a1);

¿Cómo modularizamos?

Estructura de datos – REGISTRO

¿Cómo le asignamos valor a un registro?

```
Program uno;
Type
 cadena= string[50];
 codcolor=1..10;
 auto = record
            marca: cadena;
            modelo:cadena;
            precio:real;
            color: codcolor;
        end;
```

```
Var   a1:auto;
Procedure Leer (var a: auto);
Begin
   readln(a.marca);
   readln(a.modelo);
   readln(a.precio);
   readln(a.color);
End;
Begin
   Leer (a1);
End.
```

¿Cómo imprimimos los datos de un registro?

Estructura de datos – REGISTRO –

¿Cómo le imprimimos un registro?

```
Program uno;
Type
 cadena= string[50];
 codcolor=1..10;
 auto = record
            marca: cadena;
            modelo:cadena;
            precio:real;
            color: codcolor;
        end;
Var   a1:auto;
```

```
Procedure Leer (var a: auto);
Begin
   …
End;
Procedure Informar (a: auto);
Begin
   writeln(a.marca);
   writeln(a.modelo);
   writeln(a.precio);
   writeln(a.color);
End;
```

NO SE PUEDE
write(a1);

```
Begin
  Leer (a1);
  Informar(a1);
End.
```

REGISTROS

OPERACIONES

COMPARACIÓN

Estructura de datos – REGISTRO

¿Cómo comparamos dos registros?

Se debe realizar la comparación campo a campo.

NO se puede realizar en forma directa entre dos variables registro, es decir,

```
if (a1 = a3) then
```

Estructura de datos – REGISTRO

¿Cómo comparamos dos registros?

```
Program uno;
Type
  auto= record
    …
  end;
Procedure leer(VAR a:auto);
begin
  …
end;
```

```
Var a1,a2: auto;
Begin
  leer(a1);
  leer(a2);
  if (a1.marca = a2.marca)and
     (a1.modelo = a2.modelo) and
     (a1.precio = a2.precio) and
     (a1.color = a2.color)
  then ….
End.
```

REGISTROS

EJEMPLOS

Estructura de datos – REGISTRO Ejercicio

Realice un programa que lea información sobre las app con mayores descargas de google play en el último mes. La lectura se realiza hasta leer una app de nombre “FIN”. De cada app se conoce: nombre, categoría y puntaje.
Informe el máximo puntaje dado a una app, y la cantidad de app con categoría “juego”

Estructura de datos – REGISTRO Ejercicio

```
Program aplicaciones;
 Const fin= “FIN”;
 Type
  cadena=string[40];

  app=record
        nombre: cadena;
        categoría: cadena;
        puntaje: real;
      end;
```

Estructura de datos – REGISTRO Ejercicio

```
Var
  a:app;
  cantJuego: integer;
  max:real;
  maxNom:cadena;
Begin
  leer(a);
  max:=-1;
  cantJuego:=0;
  maxNom:=‘ ‘;
```

```
while (a.nombre<>fin) do
 begin
     if (a.categoria=“juego”)
     then cantJuego:= cantJuego + 1;
     if (a.puntaje > max) then
      begin
         max:= a.puntaje;
         maxNom:=a.nombre;
      end;
      leer(a);
   end;
 write(maxNom, cantJuego);
End.
```

REGISTROS

CLAUSULA WITH

Estructura de datos – REGISTRO –

WITH

Cuando se trabaja con registros, hay ocasiones en que el acceso a los campos a través de la calificación suele ser tediosa. Para solucionar este inconveniente, el lenguaje Pascal provee una sentencia **with** que permite que un registro sea nombrado una vez, y luego sea accedido directamente.

Estructura de datos – REGISTRO –

WITH

Su forma general es:

with nombre-variable-registro

do begin

{*se referencian solo los campos del registro*}

end;

¿Cómo lo aplicamos al procedimiento leer?

Estructura de datos – REGISTRO – WITH

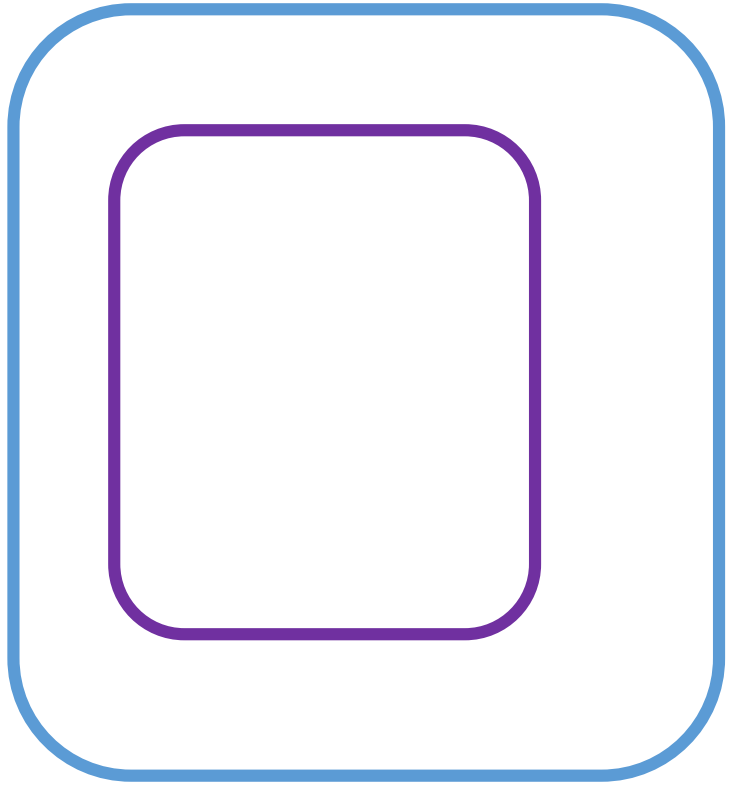
SIN UTILIZAR WITH

```
Procedure leer (var a:app)
Begin
   readln(a.nombre);
   readln(a.categoria);
   readln(a.puntaje);
end;
```

UTILIZANDO WITH

```
Procedure leer (var a:app)
Begin
  WITH a do
    begin
       readln(nombre);
       readln(categoria);
       readln(puntaje);
    end;
 End;
```

REGISTROS ANIDADOS



Estructura de datos – REGISTRO de REGISTROS

```
Program anidados;
 type cadena= string[35]; años:2000..2023;
        dias=1..31; meses=1..12;
         fecha=record
                      dia:dias;
                      mes:meses;
                      año:años;
                  end;
         alumno = record
                         nombre:cadena;
                         fNac: fecha;
                         añoIngreso:años;
                       end;
Var   a:alumno;
```

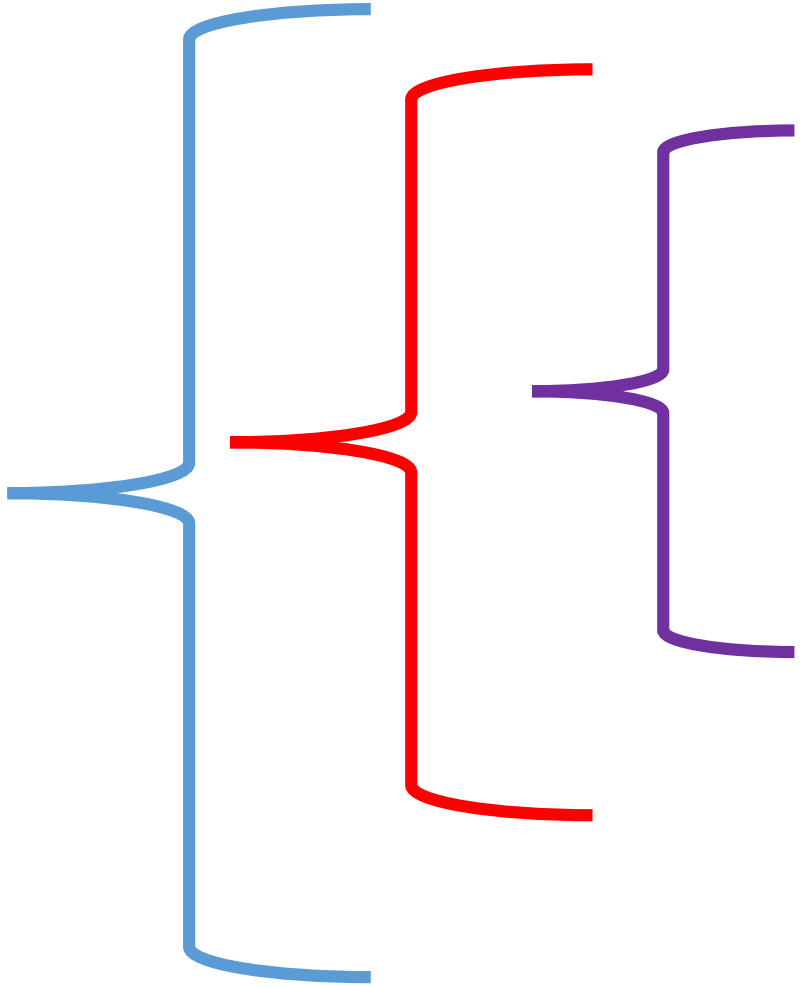
```
Begin
  a.nombre:=“Juan Martini”;
  a.fNac.dia:= 2;
  a.fNac.mes:=6;
  a.fNac.año:= 2002;
  a.añoIngreso:= 2020;
End.
```

Estructura de datos – REGISTRO de REGISTROS

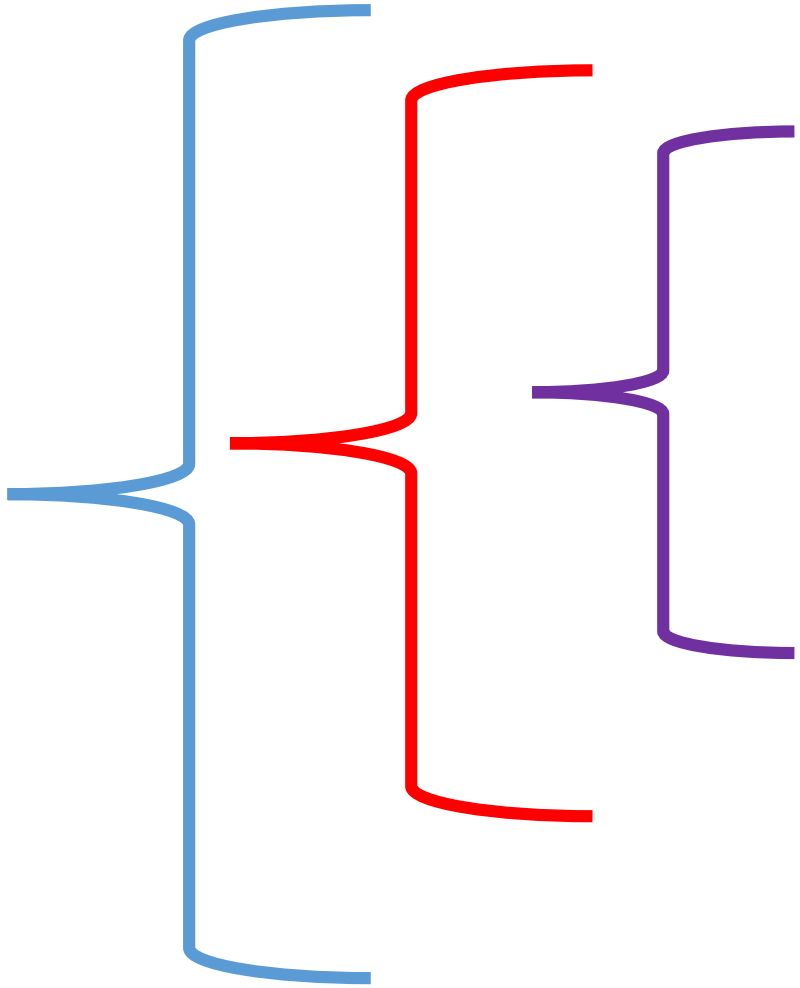
```
Program anidados;
 type cadena= string[35]; años:2000..2023;
        dias=1..31; meses=1..12;
         fecha=record
                        dia:dias;
                        mes:meses;
                        año:años;
                    end;
         alumno = record
                            nombre:cadena;
                            fNac: fecha;
                            añoIngreso:años;
                        end;
Var    a:alumno;
```

```
Procedure leer (var a:alumno);
Begin
   readln(a.nombre);
   readln(a.fNac.dia);
   readln(a.fNac.mes);
   readln(a.fNac.año);
   readln(a.añoIngreso);
end;
```

CORTE DE CONTROL



EJEMPLOS con Corte de Control



Estructura de datos – REGISTRO – EJERCICIO

Se pide realizar un programa que lea información de grupos de música. De cada grupo se cuenta con (nombre, canción, votos recibidos). Se pide calcular e informar el grupo que ha obtenido en promedio más votos por sus canciones. Tenga en cuenta los siguientes puntos:

- La lectura termina cuando llega el grupo de nombre ZZZ.
- Un grupo puede tener más de una canción
- **Todas las canciones de un grupo se leen consecutivas**

Estructura de datos – REGISTRO – Ejercicio

Nombre= 'Abel Pintos'
Cancion= 'La llave'
Votos= 4000

Nombre= 'Abel Pintos'
Cancion= 'Oncemil'
Votos= 6000

Nombre= 'Abel Pintos'
Promedio = 5000

Nombre= 'Miranda'
Cancion='Es Mentira'
Votos= 1000

Nombre= 'Miranda'
Promedio = 1000

Nombre= 'Los Piojos'
Cancion= 'Llora'
Votos= 20000

Nombre= 'Los Piojos'
Cancion= 'Cancion de cuna'
Votos= 5000

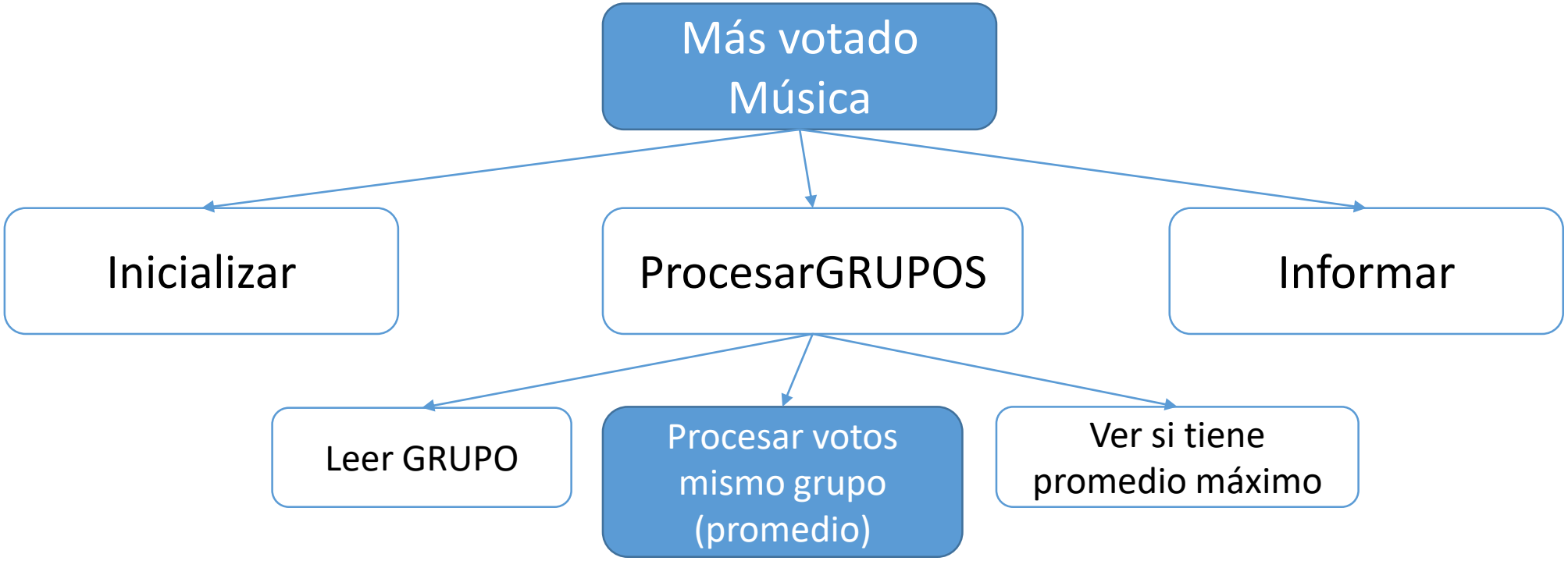
Nombre= 'Los piojos'
Cancion= 'Tan solo'
Votos= 12000

Nombre= 'Los Piojos'
Promedio = 12333

Nombre= 'ZZZ'
Cancion=
Votos=

LOS PIOJOS

Diseño de la solución



Estructura de datos – REGISTRO – Ejercicio

```
Program Ejemplo;
 CONST fin="zzz";
 TYPE  cadena=string[100];
       GruposM= Record
                   nombre:cadena;
                   cancion:cadena;
                   votos: integer;
                  End;
```

```
Procedure Leer (Var g: GruposM);
Begin
     readln(g.nombre);
     if (g.nombre <> fin) then begin
                                readln(g.cancion);
                                readln(g.votos);
                               end;
End;
```

Estructura de datos – REGISTRO – Ejercicio

SOLUCIÓN SIN MODULARIZAR (solo para mostrar el concepto de corte de control)

```
Var  max:real;
     nomMax:cadena;
Begin
  leer(g);
  max:= 0;
  while (g.nombre <> fin) do
   begin
     actual:= g.nombre; sum:= 0; cant:=0;
     while (actual = g.nombre) do
       begin
         sum:= sum + g.votos;
         cant:= cant + 1;
         leer(g);
       end;
     prom:= sum/cant; {calculo promedio}
     if (prom >= max) then {veo si prom es máximo}
       begin
         max:= prom;
         nomMax:= actual;
       end;
   end; {fin del while}
  writeln ('Grupo con máximo promedio', nomMax);
End.
```

Se procesan todos los registros del mismo grupo

Estructura de datos – REGISTRO – Ejercicio

SOLUCIÓN MODULARIZADA

```
Var  max:real;
     nomMax:cadena;

Begin {CUERPO DEL PROGRAMA – sigo diseño top - down}
  max:= 0;
  ProcesarGrupos(max, nomMax);
  writeln ('Grupo con maximo promedio', max,  nomMax);
End.
```

Estructura de datos – REGISTRO – Ejercicio

SOLUCIÓN MODULARIZADA

```
Procedure ProcesarGrupos (var max:real; var nommax: cadena);
var  g:grupoM;  prom: real;
       actual:cadena;
  Procedure ProcesarVotosGActual(var g:grupoM; actual:cadena;var prome:real);
      var sum, cant: integer;
  Begin
       sum:=0; cant:=0;
       while (actual = g.nombre) do
       begin
         sum:= sum + g.votos;
         cant:= cant + 1;
         leer(g);
       end;
       prome:= sum/cant;
  End;
Procedure PromedioMax (actual:cadena; prom:real; var max:real; var
nommax:cadena);
Begin
    if (prom >= max) then begin
                              max:= prom;
                              nomMax:= actual;
                             end;
End;
```

```
Begin
   leer (g);
   while (g.nombre <> fin) do
   begin
     actual:= g.nombre;
     ProcesarVotosGAtual(g, actual, prom);
     PromedioMax(actual, prom, max, nommax);
   end; {fin del while}
End.
```

Ver código en PASCAL

Estructura de datos – REGISTRO – Ejercicio

Puntos importantes del ejercicio:

- Debe mantenerse una variable (actual) que indique qué grupo se está procesando.
- Deben inicializarse los contadores para cada grupo.
- No debe leerse otro grupo (cuando se sale del segundo while) ya que se sale con uno leído.
- El nombre a asignar a max es actual ya que g tiene el siguiente registro.
- El máximo (max) debe inicializarse una única vez.
- Para aplicar corte de control, los datos deben estar ordenados según el criterio de procesamiento.