



# Conceptos de Algoritmos Datos y Programas

# CADP – TEMAS



## Estructura de Datos REGISTRO

# CADP – ESTRUCTURA DE DATOS

## REGISTRO



**COMPUESTO:** pueden tomar varios valores a la vez que guardan alguna relación lógica entre ellos, bajo un único nombre.

**SIMPLE:** aquellos que toman un único valor, en un momento determinado, de todos los permitidos para ese tipo.

### TIPO DE DATO

#### SIMPLE

#### COMPUESTO

DEFINIDO POR EL LENGUAJE

DEFINIDO POR EL PROGRAMADOR

DEFINIDO POR EL LENGUAJE

DEFINIDO POR EL PROGRAMADOR

Integer  
Real  
Char  
Boolean

Subrango

String

Registro



Supongamos que se quiere representar la información de las distintas razas de animales que existen en una veterinaria. Para simplificar el problema supongamos que la veterinaria atiende solamente perros. De cada animal se conoce la raza, el nombre, la edad.



Qué información es relevante para un perro?

*Con lo que sabemos hasta ahora como lo representamos?*



## REGISTRO

Es un tipo de datos estructurado, que permite agrupar diferentes clases de datos en una estructura única bajo un sólo nombre

Raza  
Nombre  
Edad

Una manera natural y lógica de agrupar los datos de cada perro en una sola estructura es declarar un tipo **REGISTRO** asociando el conjunto de datos de cada uno.

**REGISTRO**  
**PERRO**



**Heterogénea**



Los elementos pueden ser de distinto tipo (puede haber registros con todos elementos del mismo tipo)

**Estática**

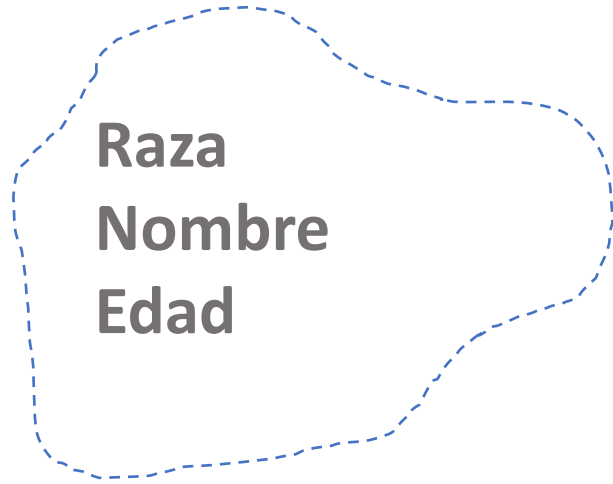


El tamaño no cambia durante la ejecución (se calcula en el momento de compilación)

**Campos**



Representan cada uno de los datos que forman el registro



**REGISTRO  
PERRO**

**Cómo se  
define?**



```
Program uno;
```

```
Const
```

```
... .
```

```
Type
```

```
nombre = record
```

```
    campo1: tipo;
```

```
    campo2: tipo;
```

```
    ...
```

```
end;
```

```
Var
```

```
    variable: nombre;
```



Se nombra cada campo.

Se asigna un tipo a cada campo.

Los tipos de los campos deben ser estáticos.

*Cómo declaro el  
registro PERRO?*



```
Program uno;
```

```
Const
```

```
...
```

```
Type
```

```
    perro = record
```

```
        raza: string;
```

```
        nombre: string;
```

```
        edad: integer;
```

```
end;
```

```
Var
```

```
    ani1, ani2: perro;
```

La característica principal es que un registro permite representar la información en una única estructura.

*Cómo se  
trabaja con un  
registro?*





## CON LA VARIABLE REGISTRO

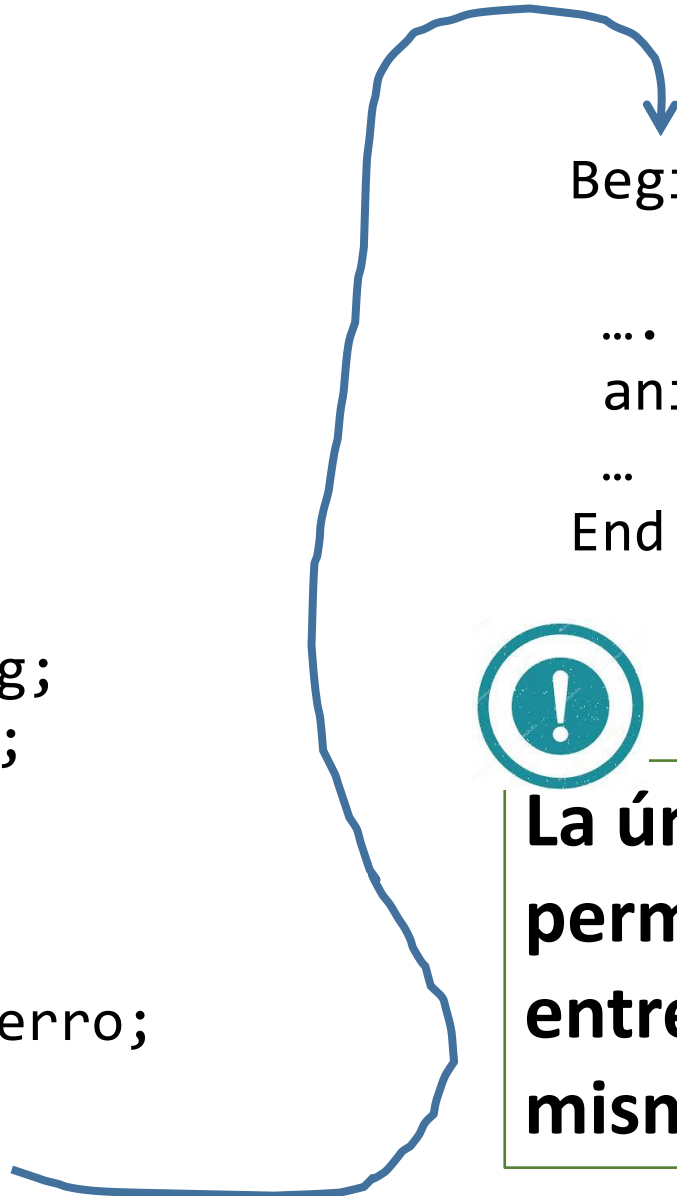
```
Program uno;  
Const  
    ...  
Type  
  
    perro = record  
        raza: string;  
        nombre: string;  
        edad: integer;  
    end;  
  
Var  
    ani1, ani2: perro;
```

Begin

```
    ...  
    ani2:= ani1;  
    ...  
End.
```



**La única operación permitida es la asignación entre dos variables del mismo tipo**



# CADP – ESTRUCTURA DE DATOS

## REGISTRO



### CON LOS CAMPOS DEL REGISTRO

```
Program uno;  
Const  
    ...  
Type  
  
    perro = record  
        raza: string;  
        nombre: string;  
        edad: integer;  
    end;  
  
Var  
    ani1, ani2: perro;
```

Cómo se le  
da valor?

Begin

...

Puedo realizar las  
operaciones permitidas  
según el tipo de campo  
del registro

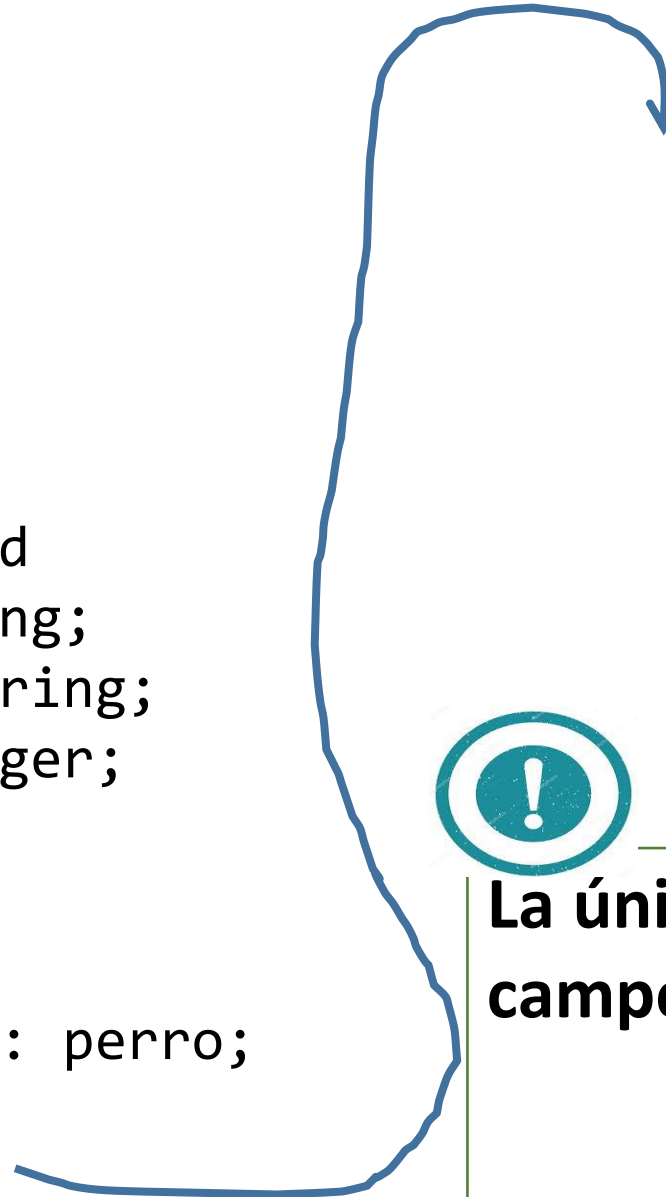
...

End.




La única forma de acceder a los  
campos es **variable.nombrecampo**

**ani1.nombre**





```
Program uno;  
Const  
    ...  
Type  
  
perro = record  
    raza: string;  
    nombre: string;  
    edad: integer;  
end;  
  
Var  
    ani1, ani2: perro;
```



```
Begin  
    ani1.raza := 'Callejero';  
    ani1.nombre := 'Bob';  
    ani1.edad := 1;  
End.
```

```
Begin  
    read (ani1.raza);  
    read(ani1.nombre);  
    read(ani1.edad);  
End.
```

Qué ocurre si no le  
doy valor a todos los  
campos?

Debo asignarlos en  
el orden en que se  
declararon?

**MODULARIZAR?**



No se puede hacer  
read (ani1)

# CADP – ESTRUCTURA DE DATOS

```
Procedure leer (var p:perro);
```

```
Begin  
  read (p.raza);  
  read(p.nombre);  
  read(p.edad);  
End.
```

Debo asignarlos en  
el orden en que se  
declararon?

Puede ser una  
función en vez de un  
procedimiento?

*Cómo muestro  
el contenido de  
un registro?*

Qué ocurre si no le  
doy valor a todos los  
campos?

## REGISTRO



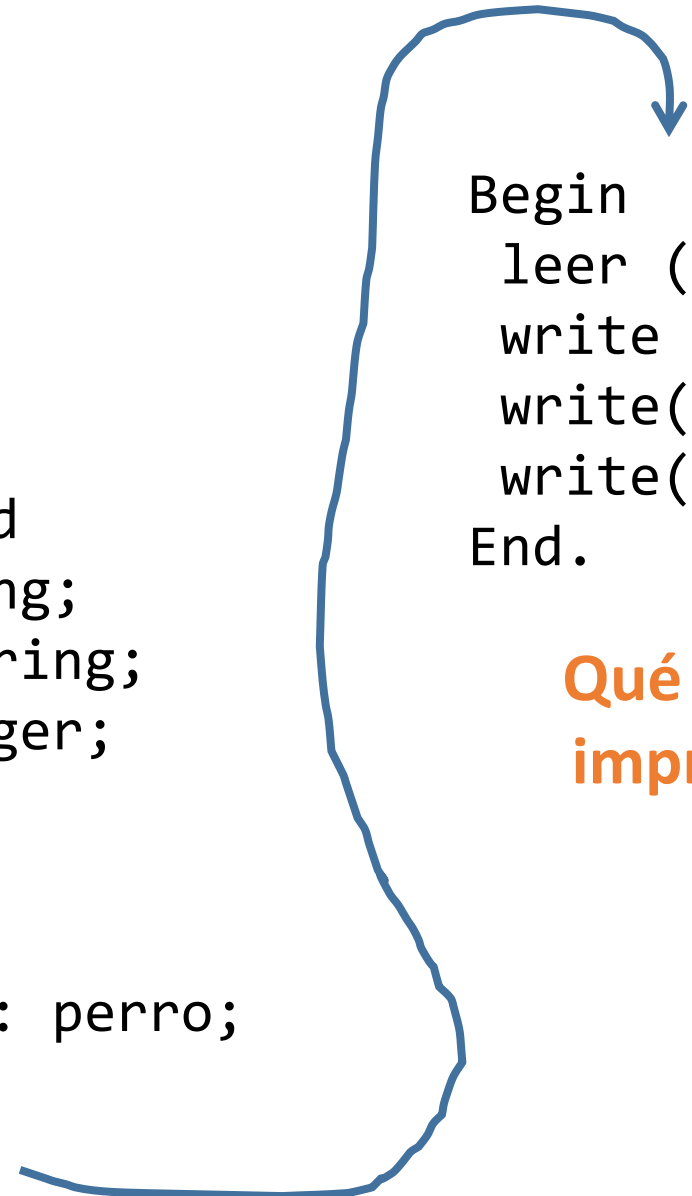
```
Program uno;  
Const  
  ...  
Type  
  perro = record  
    raza: string;  
    nombre: string;  
    edad: integer;  
  end;  
  
Procedure leer (var p:perro);  
begin  
  ...  
end;  
  
Var  
  ani1, ani2: perro;  
  
Begin  
  leer (ani1);  
  ani2:= ani1;  
End.
```

# CADP – ESTRUCTURA DE DATOS

## REGISTRO



```
Program uno;  
Const  
    ...  
Type  
  
perro = record  
    raza: string;  
    nombre: string;  
    edad: integer;  
end;  
  
Var  
    ani1, ani2: perro;
```



```
Begin  
    leer (ani1);  
    write (ani1.raza);  
    write(ani1.nombre);  
    write(ani1.edad);  
End.
```

Qué ocurre si no le  
imprimo todos los  
campos?



No se puede hacer  
write (ani1)

**MODULARIZAR?**

# CADP – ESTRUCTURA DE DATOS

```
Procedure imprimir (p:perro);
```

```
Begin
  write (p.raza);
  write(p.nombre);
  write(p.edad);
End.
```

Debo asignarlos en el orden en que se declararon?

Puede ser una función en vez de un procedimiento?

Cómo se comparan dos registros?

Qué ocurre si no le imprimo todos los campos?

## REGISTRO



```
Program uno;
Const
  ...
Type
  perro = record
    raza: string;
    nombre: string;
    edad: integer;
  end;
Procedure leer (var p:perro);
begin
  ...
end;
Procedure imprimir (p:perro);
begin
  ...
end;
Var
  ani1, ani2: perro;
Begin
  leer (ani1);
  imprimir(ani1);
End.
```



```
Program uno;  
Const  
    ...  
Type  
  
perro = record  
    raza: string;  
    nombre: string;  
    edad: integer;  
end;  
  
Var  
    ani1, ani2: perro;
```

```
Begin  
    leer (ani1);  
    leer (ani2),  
  
    if ((ani1.raza = ani2.raza)and  
        (ani1.nombre = ani2.nombre) and  
        (ani1.edad = ani2.edad))  
    then  
        write (`Los registro son iguales`);  
    End.
```



No se puede hacer  
ani1 = ani2

**MODULARIZAR?**



```
procedure iguales (p,p1:perro; var ok:boolean);  
Begin  
    if( (p.raza = p1.raza)and  
        (p.nombre = p1.nombre) and  
        (p.edad = p1.edad))  
  
    then ok:= true  
    else ok:= false;  
end;
```

*Puede ser una función  
en vez de un  
procedimiento?*





```
function iguales (p,p1:perro):boolean;  
Var  
  ok:Boolean;  
Begin  
  if( (p.raza = p1.raza)and (p.nombre = p1.nombre) and (p.edad = p1.edad))  
  then ok:= true  
  else ok:= false;  
  iguales:= ok;  
end;
```

---

```
function iguales (p,p1:perro):boolean;  
Var  
  ok:Boolean;  
Begin  
  ok:= ((p.raza = p1.raza)and  
        (p.nombre = p1.nombre)  
        and (p.edad = p1.edad))  
  iguales:= ok;  
end;
```

```
function iguales (p,p1:perro):boolean;  
Begin  
  iguales := ((p.raza = p1.raza)  
              and (p.nombre= p1.nombre)  
              and (p.edad = p1.edad));  
end;
```



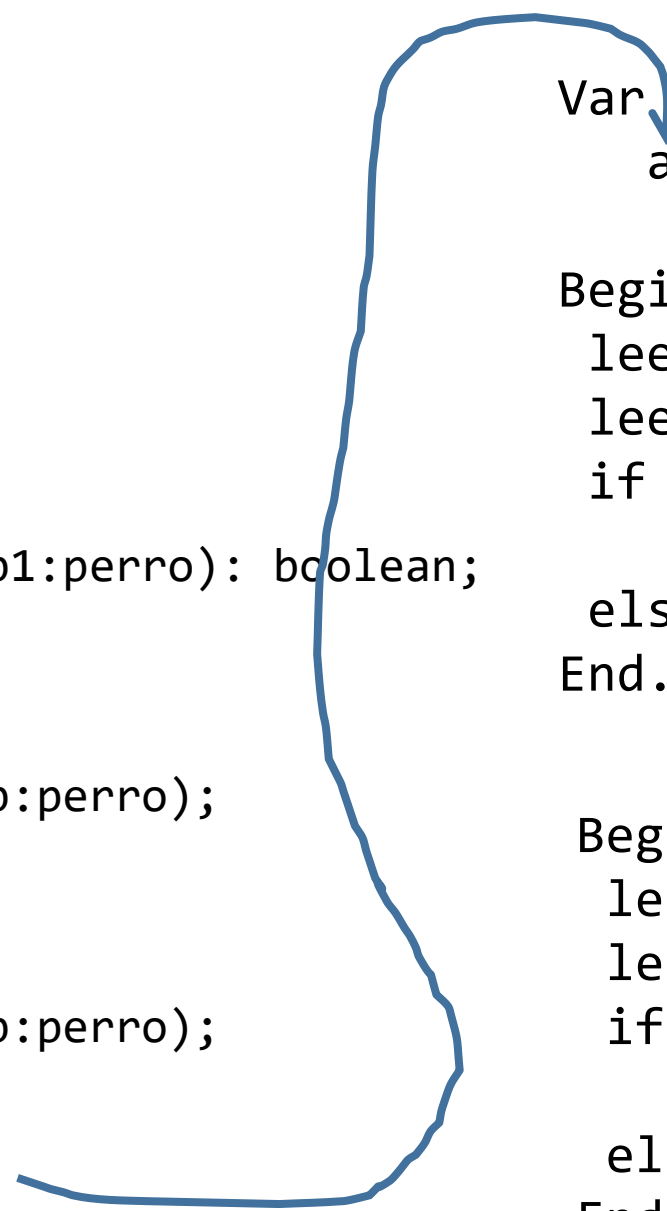
```
Program uno;  
Const  
    ...  
Type  
perro = record  
    raza: string;  
    nombre: string;  
    edad: integer;  
end;  
function iguales (p,p1:perro): boolean;  
begin  
    ...  
end;  
Procedure leer (var p:perro);  
begin  
    ...  
end;  
Procedure imprimir (p:perro);  
begin  
    ...  
end;
```

Var  
 ani1, ani2: perro;

```
Begin  
    leer (ani1);  
    leer (ani2);  
    if (iguales (ani1,ani2) = true) then  
        write (`Los registros son iguales`)  
    else write (`Los registros no son iguales`);  
End.
```

```
Begin  
    leer (ani1);  
    leer (ani2);  
    if (iguales (ani1,ani2)) then  
        write (`Los registros son iguales`)  
    else write (`Los registros no son iguales`);  
End.
```





Escriba un programa que lea perros hasta leer un perro cuya raza es `XXX` Al finalizar informe de los perros en con nombre `Bob` y que tienen al menos 2 años

Raza `Ovejero`  
Nombre `Bob`  
edad:2

Raza `Callejero`  
Nombre `Bob`  
edad:1

Raza `Golden`  
Nombre `Aragon`  
edad:5

Raza `Ovejero`  
Nombre `Lucy`  
edad:3

Raza `Salchicha`  
Nombre `Scoby`  
edad:1



1



Escriba un programa que lea perros hasta leer un perro cuya raza es `XXX` Al finalizar informe de los perros en con nombre `Bob` y que tienen al menos 2 años

```
Inicializar contadores (cant)
Leer registro (ani)
While (no sea el ultimo registro) do
  begin
    if (ani tiene nombre `Bob`) then
      if (ani tiene al menos dos años) then
        incremento (cant)
      leer registro (ani)
    end;
  Write (`La cantidad es`, cant);
```

**Cuál es la estructura de datos?**

**Como verifico las condiciones?**

**Qué modularizo?**



```
Program uno;  
Const  
    ...  
Type  
perro = record  
    raza: string;  
    nombre: string;  
    edad: integer;  
end;  
  
// módulos  
  
Var  
    ani: perro;  
    cant: integer;
```

**Begin**

cant:=0;

leer (ani);

while (ani.raza <> `XXX`) do

begin

if (cumpleNombre (ani) = true) then

if (edad (ani) = true) then

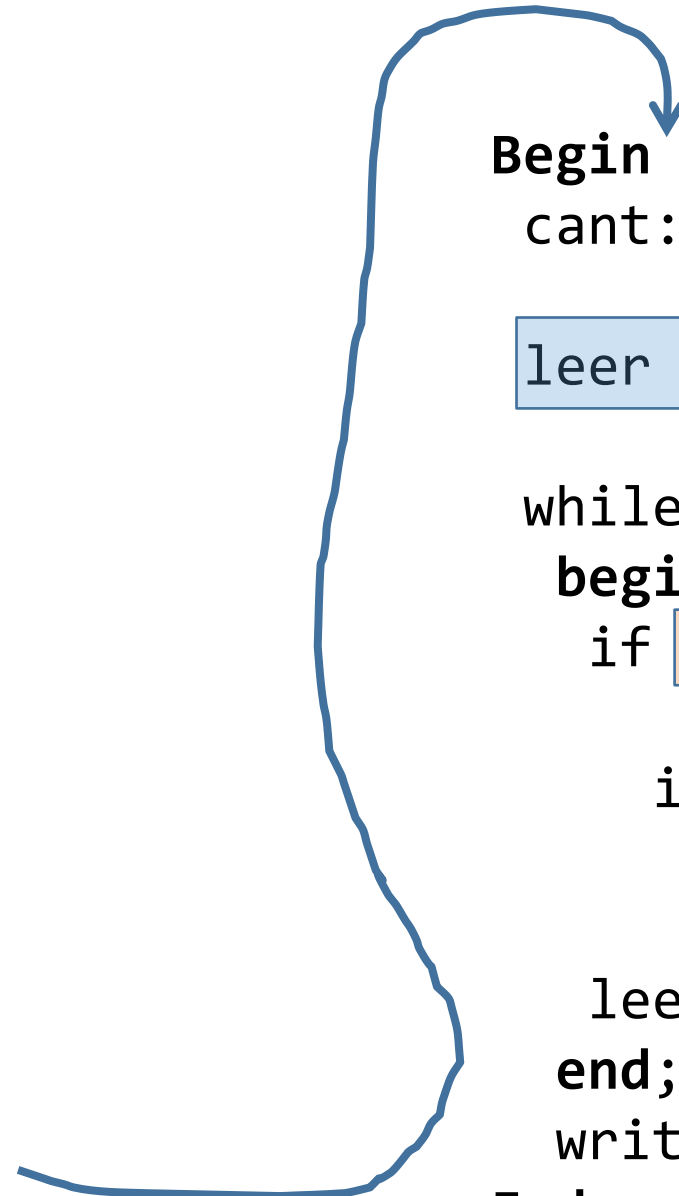
cant:= cant + 1;

leer (ani);

end;

write (`La cantidad es`, cant);

**End.**





```
procedure leer (var p:perro);
```

```
Begin
    read(p.raza);
    read(p.nombre);
    read(p.edad);
end;
```

Qué  
alternativa  
conviene?

```
procedure leer (var p:perro);
```

```
Begin
    read(p.raza);
    if (p.raza <> 'XXX') then
        begin
            read(p.nombre);
            read(p.edad);
        end;
end;
```



```
function cumpleNombre (p:perro): boolean;  
var  
    ok:boolean;  
  
begin  
    if (p.nombre = `Bob`) then  
        ok:= true  
    else  
        ok:= false;  
    cumpleNombre:= ok;  
end;
```

### Otra opción

```
function cumpleNombre (p:perro): boolean;  
  
begin  
    cumpleNombre:= (p.nombre = `Bob`);  
end;
```



```
function edad (p:perro): boolean;  
var  
    ok:boolean;  
  
begin  
    if (p.edad >= 2) then  
        ok:= true  
    else  
        ok:= false;  
    edad:= ok;  
end;
```

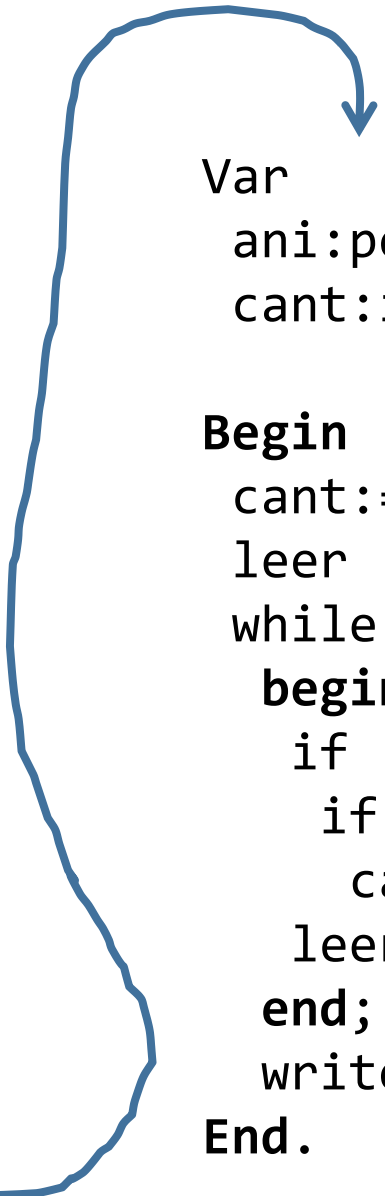
### Otra opción

```
function edad (p:perro): boolean;  
  
begin  
    edad:= (p.edad >= 2);  
end;
```





```
Program uno;  
Type  
  perro = record  
    raza: string;  
    edad: integer;  
    nombre: string;  
  end;  
Procedure leer (p:perro);  
begin  
end;  
function cumpleNombre (p:perro): boolean;  
begin  
  ...  
end;  
  
function edad (p:perro): boolean;  
begin  
  ...  
end;
```



```
Var  
  ani:perro;  
  cant:integer;
```

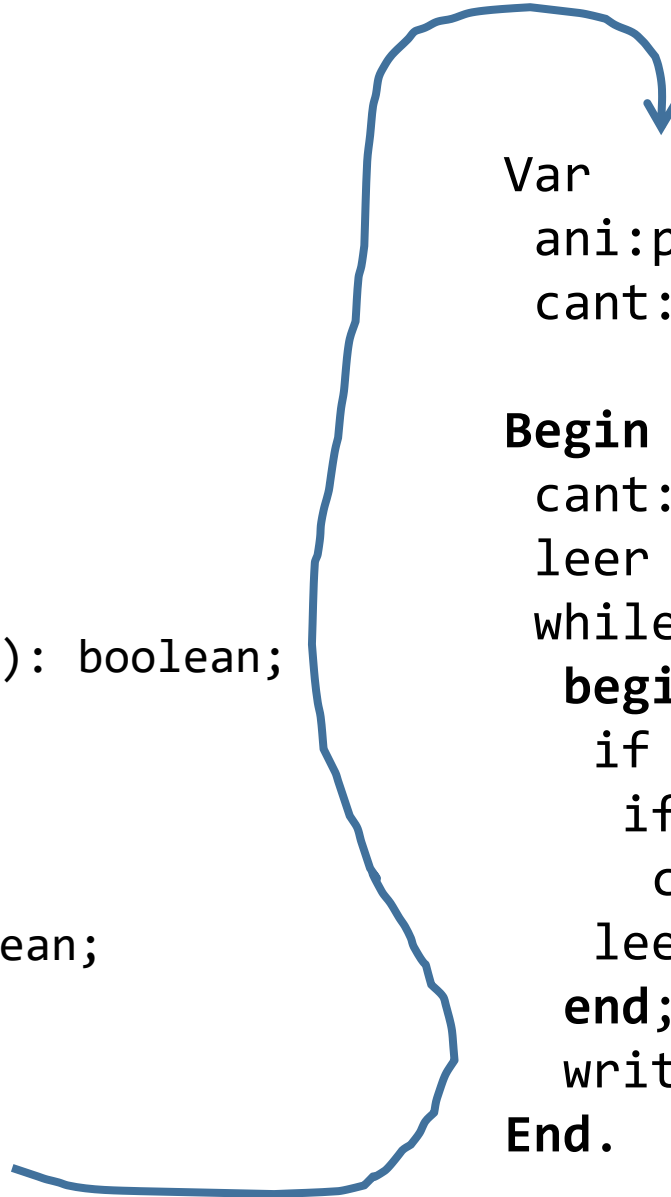
```
Begin  
  cant:= 0;  
  leer (ani);  
  while (ani.raza <> `XXX`) do  
    begin  
      if (cumpleNombre (ani)) then  
        if (edad (ani)) then  
          cant:= cant + 1;  
        leer (ani);  
      end;  
      write (`La cantidad es`, cant);  
    end;  
  End.
```

*Es necesario pasar  
todo el registro a  
las funciones?*



```
Program uno;  
Type  
perro = record  
    raza: string;  
    edad: integer;  
    nombre: string;  
end;  
Procedure leer (p:perro);  
begin  
end;  
function cumpleNombre (n:string): boolean;  
begin  
    ...  
end;  
  
function edad (e:integer): boolean;  
begin  
    ...  
end;
```

```
Var  
    ani:perro;  
    cant:integer;  
  
Begin  
    cant:= 0;  
    leer (ani);  
    while (ani.raza <> `XXX`) do  
        begin  
            if (cumpleNombre (ani.nombre)) then  
                if (edad (ani.edad)) then  
                    cant:= cant + 1;  
                    leer (ani);  
                end;  
            write (`La cantidad es`, cant);  
        end;  
    End.
```





```
function cumpleNombre (nom:string): boolean;  
var  
    ok:boolean;  
begin  
    if (nom = 'Bob') then ok:= true  
    else ok:= false;  
    cumpleNombre:= ok;  
end;
```

```
function edad (e:integer): boolean;  
var  
    ok:boolean;  
begin  
    if (e>= 2) then ok:= true  
    else ok:= false;  
    edad:= ok;  
end;
```