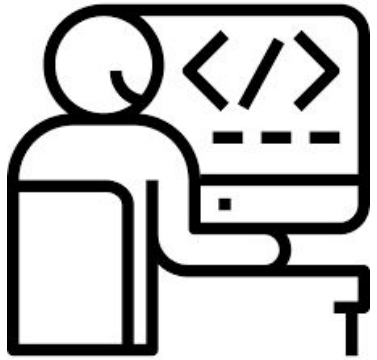


Algoritmos y Programación I

AyPI – **Temas de la clase pasada**

- Alcance de variables
- Comunicación entre módulos

AyPI – **Temas de la clase de hoy**

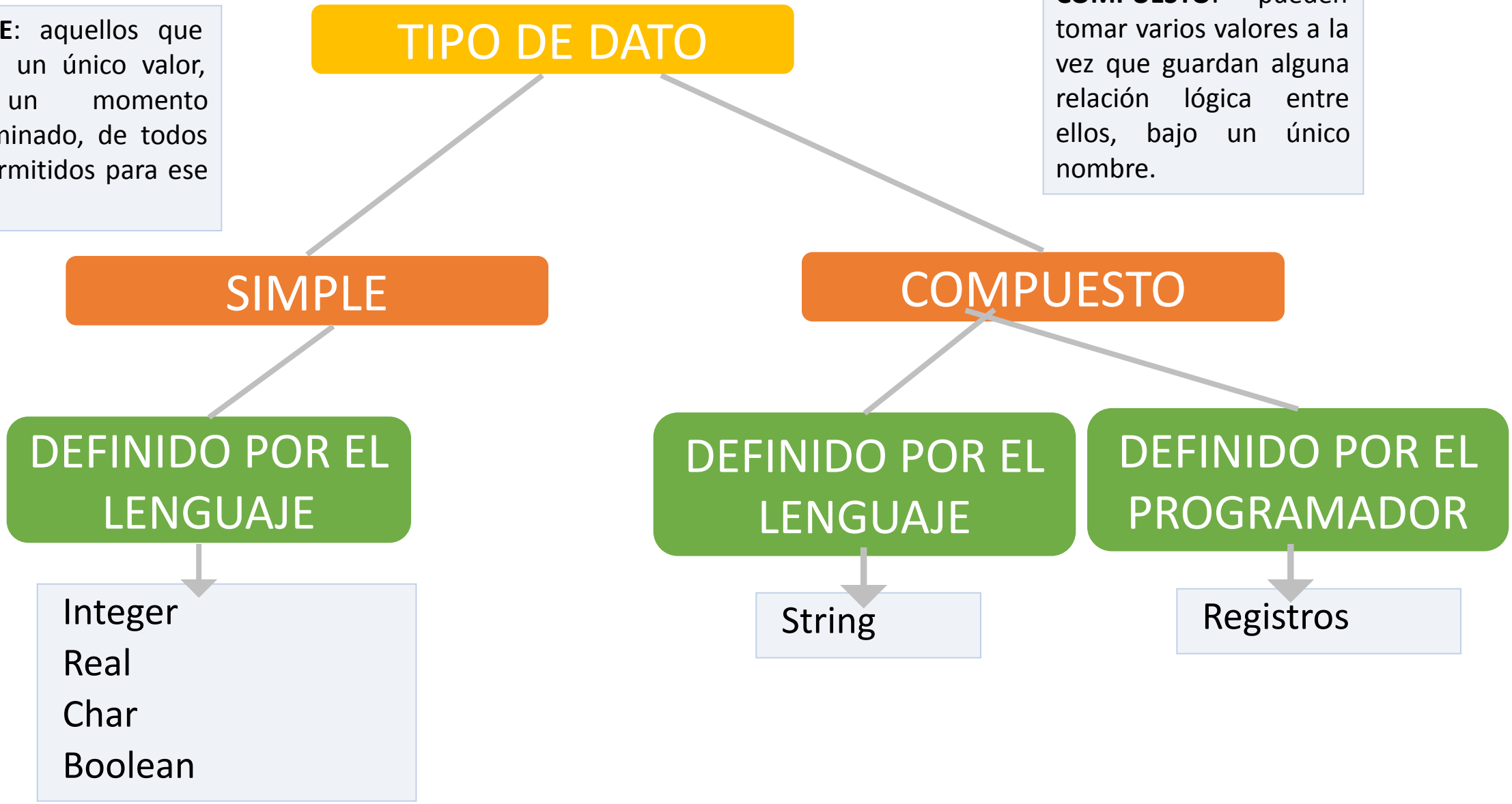
- Tipo de Datos Estructurado
- Clasificación de tipos de Datos Estructurados
- Tipo de datos REGISTRO
- Corte de control

AyPI – **TIPOS DE DATOS**

ESTRUCTURADOS

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

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



AyPI – **TIPOS DE DATO STRING**

- Un tipo de dato string es una sucesión de caracteres de un largo determinado, que se almacenan en un área contigua de la memoria.
- Existe en la mayoría de los lenguajes como un tipo predefinido.
- No es un dato simple.

AyPI – **TIPOS DE DATO STRING**

```
Program uno;
var
    nombre: string; apellido: string;
    nombre_y_apellido: string;
begin
  nombre:= ‘Rosa’;   apellido:= ‘Gonzalez’;
  nombre_y_apellido:= apellido + ‘, ‘ + nombre;
  write(nombre_y_apellido);

  read(nombre);
  if(nombre = ‘Rosa’) then
     write(‘Hola Rosita, amiga mia‘);
  else
     write(‘Hola ‘ + nombre);
end.
```

Gonzalez, Rosa

AyPI – TIPOS DE DATOS ESTRUCTURADOS

CLASIFICACIÓN

ESTRUCTURA DE DATOS

Permite al programador definir un tipo al que se asocian diferentes datos que tienen valores lógicamente relacionados y asociados bajo un nombre único.

Elementos	Acceso	Tamaño	Linealidad
Homogénea	Secuencial	Dinámica	Lineal
Heterogénea	Directo	Estática	No Lineal

AyPI – **TIPOS DE DATOS ESTRUCTURADOS** CLASIFICACIÓN

ELEMENTOS

Depende si los elementos son del mismo tipo o no.

Homogénea → Los elementos que la componen son del mismo tipo

Heterogénea → Los elementos que la componen pueden ser de distinto tipo

AyPI – **TIPOS DE DATOS ESTRUCTURADOS**

CLASIFICACIÓN

TAMAÑO

Hace referencia a si la estructura puede variar su tamaño durante la ejecución del programa.

ESTATICA → El tamaño de la estructura no varía durante la ejecución del programa

DINAMICA → El tamaño de la estructura puede variar durante la ejecución del programa

AyPI – **TIPOS DE DATOS ESTRUCTURADOS**

CLASIFICACIÓN

ACCESO

Hace referencia a cómo se pueden acceder a los elementos que la componen.

SECUENCIAL → Para acceder a un elemento particular se debe respetar un orden predeterminado, por ejemplo, pasando por todos los elementos que le preceden, por ese orden.

DIRECTO → Se puede acceder a un elemento particular, directamente, sin necesidad de pasar por los anteriores a él, por ejemplo, referenciando una posición.

AyPI – **TIPOS DE DATOS ESTRUCTURADOS**

CLASIFICACIÓN

LINEALIDAD

Hace referencia a cómo se encuentran almacenados los elementos que la componen.

LINEAL

Está formada por ninguno, uno o varios elementos que guardan una relación de adyacencia ordenada donde a cada elemento le sigue uno y le precede uno, solamente.

NO LINEAL

Para un elemento dado pueden existir 0, 1 ó mas elementos que le suceden y 0, 1 ó mas elementos que le preceden.

AyPI – **Tipos de Datos definidos por el programador**

HASTA AHORA

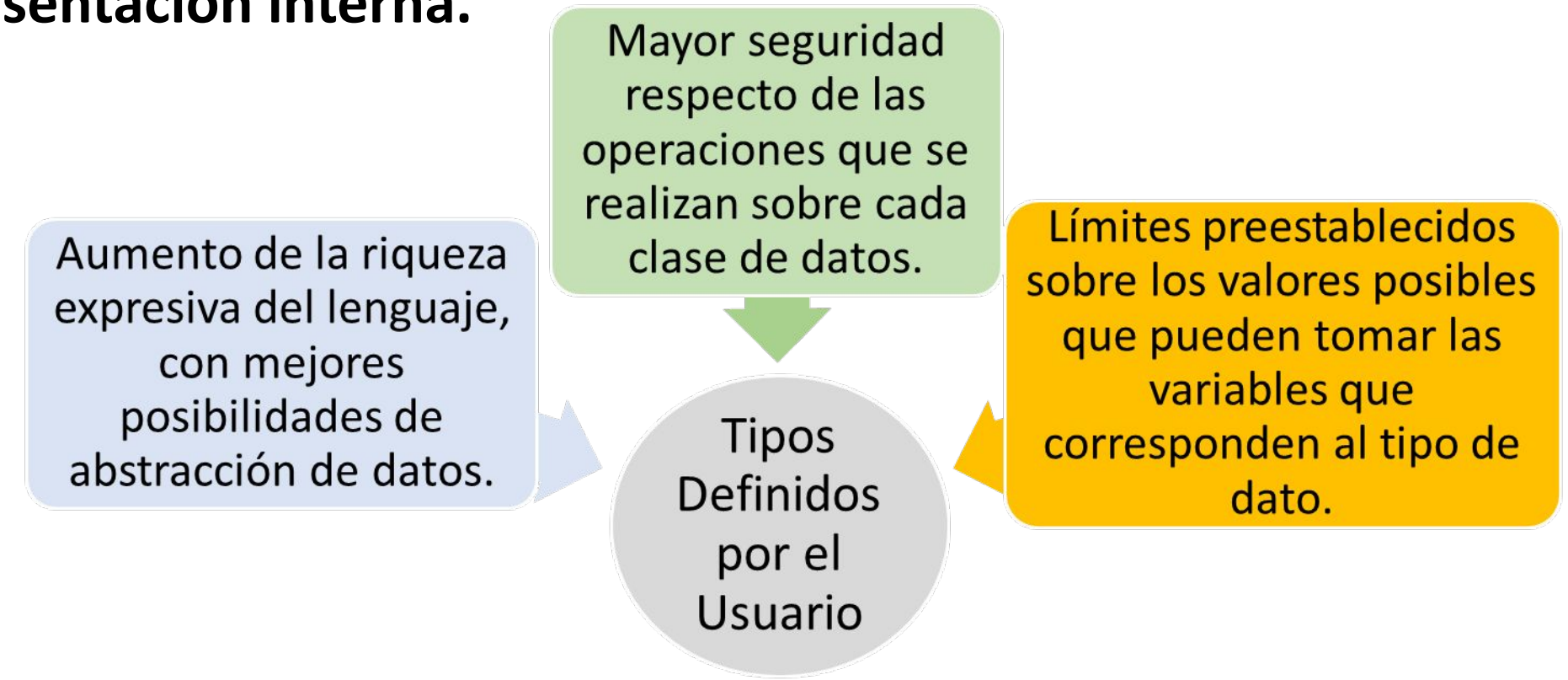
Hemos trabajado los tipos de datos simples que se pueden considerar estándar en la mayoría de los lenguajes de programación.

Esto significa que el conjunto de valores de ese tipo, las operaciones que se pueden efectuar y su representación están definidas y acotadas por el lenguaje.

Esta definición nos indica que podemos requerir la representación de elementos "NO de tipo estándar" que se asocien con el fenómeno real a tratar.

AyPI – **Tipos de Datos definidos por el programador**

Un aspecto muy importante en los lenguajes de programación es la capacidad de especificar y manejar datos no estándar, indicando valores permitidos, operaciones válidas y su representación interna.



AyPI – **Tipos de Datos definidos por el programador**

TIPO DE DATOS DEFINIDOS POR EL USUARIO

Un tipo de dato definido por el usuario es aquel que no existe en la definición del lenguaje, y el programador es el encargado de su especificación.

Type

```
identificador = tipo;
```

Dónde se declara?

Un tipo estandar

Un tipo definido por el lenguaje

AyPI – **Tipos de Datos definidos por el programador**

TIPO DE DATOS DEFINIDOS POR EL USUARIO (TDDU)

```
program uno;
Const
   ...
Type
     identificador = tipo;
Módulos
Var
    x: identificador;
    …
Begin
   ...
End.
```

AyPI – **TIPOS DE DATOS**

ESTRUCTURADOS

Supongamos que se quiere representar la información de los distintos inmuebles con los que trabaja una inmobiliaria. Para simplificar el problema la inmobiliaria trabaja con casas o departamentos, y sólo los alquila.

¿Qué información debería conocer la inmobiliaria?

AyPI – **TIPOS DE DATOS**

ESTRUCTURADOS

- Tipo propiedad
- Cantidad de habitaciones
- Cantidad de baños
- Precio Alquiler
- Localidad

Hasta ahora no conocemos una forma de poder agrupar toda esta información en una sola variable.

AyPI – **TIPOS DE DATOS**

ESTRUCTURADOS

Tipo propiedad
Cantidad de habitaciones
Cantidad de baños
Precio Alquiler
Localidad

Tipo propiedad
Cantidad de habitaciones
Cantidad de baños
Precio Alquiler
Localidad

INMUEBLE

AyPI – **TIPOS DE DATOS**

ESTRUCTURADOS

ESTRUCTURA DE DATOS

Permite al programador definir un tipo al que se asocian diferentes datos que tienen valores lógicamente relacionados y asociados bajo un nombre único.

Tipo propiedad
Cantidad de habitaciones
Cantidad de baños
Precio Alquiler
Localidad

INMUEBLE

AyPI – TIPOS DE DATOS ESTRUCTURADOS REGISTRO

REGISTRO

Es uno de los tipos de datos estructurados, que permiten agrupar diferentes clases de datos en una estructura única bajo un sólo nombre

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

Tipo propiedad
Cantidad de habitaciones
Cantidad de baños
Precio Alquiler
Localidad

REGISTRO INMUEBLE

AyPI – **TIPOS DE DATOS ESTRUCTURADOS** REGISTRO

Tipo propiedad
Cantidad de habitaciones
Cantidad de baños
Precio Alquiler
Localidad

REGISTRO INMUEBLE

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)

¿Cómo lo defino?

Campos

Representan cada uno de los datos que forman el registro

AyPI – **TIPOS DE DATOS ESTRUCTURADOS** REGISTRO

```
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 inmueble?

AyPI – TIPOS DE DATOS ESTRUCTURADOS REGISTRO

```
Program uno;
Const
   ....
Type

inmueble = record
    tipo: string;
    cantHab: integer;
    cantBaños:integer;
    precio: real;
    localidad: string;
end;

Var
    inmu1, inmu2: inmueble;
```

¿Cómo trabajamos el registro inmueble?

AyPI – TIPOS DE DATOS ESTRUCTURADOS REGISTRO

CON LA VARIABLE REGISTRO

```
Program uno;
Const
   ....
Type

inmueble = record
    tipo: string;
    cantHab: integer;
    cantBaños:integer;
    precio: real;
    localidad: string;
end;

Var
    inmu1, inmu2: inmueble;
```

```
Begin

 ...
 inmu2:= inmu1;
 ...
End.
```

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

AyPI – TIPOS DE DATOS ESTRUCTURADOS REGISTRO

CON LOS CAMPOS DEL REGISTRO

¿Cómo le damos valor?

```
Program uno;
Const
   ...
Type

inmueble = record
    tipo: string;
    cantHab: integer;
    cantBaños:integer;
    precio: real;
    localidad: string;
end;

Var
    inmu1, inmu2: inmueble;
```

```
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

inmu1.cantHab

AyPI – **TIPOS DE DATOS ESTRUCTURADOS** REGISTRO

```
Program uno;
Const
   ….
Type

inmueble = record
    tipo: string;
    cantHab: integer;
    cantBaños:integer;
    precio: real;
    localidad: string;
end;

Var
    inmu1, inmu2: inmueble;
```

```
Begin
 inmu1.tipo:='Casa';
 inmu1.cantHab:= 2;
 inmu1.cantBaños:= 1;
 inmu1.precio:= 15.500;
 inmu1.localidad:= `La Plata';
End.
```

```
Begin
 read (inmu1.tipo);
 read(inmu1.cantHab);
 read(inmu1.cantBaños);
 read(inmu1.precio);
 read(inmu1.localidad);
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 (inmu1)

AyPI – TIPOS DE DATOS ESTRUCTURADOS REGISTRO

```
Procedure leer (var i:inmueble);

Begin
    read(i.tipo);
    read(i.cantHab);
    read(i.cantBaños);
    read(i.precio);
    read(i.localidad);
end;
```

¿Cómo muestro el contenido de un registro?

¿Debo asignarlos en el orden en que se declararon?

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

¿Puede ser una función en vez de un procedimiento?

```
Program uno;
Const
  ...
Type
inmueble = record
   tipo: string;
   cantHab: integer;
   cantBaños:integer;
   precio: real;
   localidad: string;
end;
Procedure leer (var i:inmueble);
begin
 ...
end;

Var
   inmu1, inmu2: inmueble;
Begin
 leer (inmu1);
 inmu2:= inmu1;
End.
```

AyPI – **TIPOS DE DATOS ESTRUCTURADOS** REGISTRO

```
Procedure leer (var i:inmueble);

Begin
    read(i.tipo);
    read(i.cantHab);
    read(i.cantBaños);
    read(i.precio);
    read(i.localidad);
end;
```

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

(El campo sin asignar queda con dato basura)

¿Debo asignarlos en el orden en que se declararon?

(no necesariamente)

¿Puede ser una función en vez de un procedimiento?

(Si, la función devuelve un dato tipo inmueble)

AyPI – TIPOS DE DATOS ESTRUCTURADOS REGISTRO

```
Program uno;
Const
   ....
Type

inmueble = record
    tipo: string;
    cantHab: integer;
    cantBaños:integer;
    precio: real;
    localidad: string;
end;

Var
    inmu1, inmu2: inmueble;
```

```
Begin
 leer (inmu1);
 write (inmu1.tipo);
 write (inmu1.cantHab);
 write (inmu1.cantBaños);
 write (inmu1.precio);
 write (inmu1.localidad);
End.
```

No se puede hacer write (inmu1)

¿Qué ocurre si no le imprimo todos los campos?

¿MODULARIZAR?

AyPI – TIPOS DE DATOS ESTRUCTURADOS REGISTRO

```
Procedure imprimir (i:inmueble);

Begin
    write(i.tipo);
    write(i.cantHab);
    write(i.cantBaños);
    write(i.precio);
    write(i.localidad);
end;
```

¿Cómo comparo el contenido de dos registros?

¿Debo imprimir en el orden en que se declararon?

¿Qué ocurre si no imprimo todos los campos?

¿Puede ser una función en vez de un procedimiento?

```
Program uno;
Const
  ...
Type
inmueble = record
   tipo: string;
   cantHab: integer;
   cantBaños:integer;
   precio: real;
   localidad: string;
end;
Procedure imprimir (i:inmueble);
begin
 ...
end;

Var
   inmu1, inmu2: inmueble;
Begin
 leer (inmu1);
 imprimir(inmu1);
End.
```

AyPI – **TIPOS DE DATOS ESTRUCTURADOS** REGISTRO

```
Procedure imprimir (i:inmueble);

Begin
    write(i.tipo);
    write(i.cantHab);
    write(i.cantBaños);
    write(i.precio);
    write(i.localidad);
end;
```

¿Qué ocurre si no imprimo todos los campos?

(El campo no se muestra por pantalla)

¿Debo imprimir en el orden en que se declararon?

(no necesariamente)

¿Puede ser una función en vez de un procedimiento?

(Terminantemente NO!!!!!)

AyPI – TIPOS DE DATOS ESTRUCTURADOS REGISTRO

```
Program uno;
Const
   ...
Type

inmueble = record
    tipo: string;
    cantHab: integer;
    cantBaños:integer;
    precio: real;
    localidad: string;
end;

Var
    inmu1, inmu2: inmueble;
```

```
Begin
 leer (inmu1);
 leer (inmu2),

 if ((inmu1.tipo = inmu2.tipo) and
     (inmu1.cantHab = inmu2.cantHab) and
     (inmu1.cantBaños = inmu2.cantBaños) and
     (inmu1.precio = inmu2.precio) and
     (inmu1.localidad = inmu2.localidad)) then
  write (`Los registros tienen los mismos valores`)
End.
```

No se puede hacer
inmu1 = inmu2

¿MODULARIZAR?

AyPI – TIPOS DE DATOS ESTRUCTURADOS REGISTRO

```
function iguales (i1,i2:inmueble):boolean;
Var
 ok:boolean;
Begin
    if ((i1.tipo = i2.tipo)and
     (i1.cantHab = i2.cantHab) and
     (i1.cantBaños = i2.cantBaños) and
     (i1.precio = i2.precio) and
     (i1.localidad = i2.localidad))
    then ok:= true
    else ok:= false;
    iguales:= ok;
end;
```

```
function iguales (i1,i2:inmueble):boolean;
Begin
    iguales:= ((i1.tipo = i2.tipo) and
     (i1.cantHab = i2.cantHab) and
     (i1.cantBaños = i2.cantBaños) and
     (i1.precio = i2.precio) and
     (i1.localidad = i2.localidad));
end;
```

AyPI – TIPOS DE DATOS ESTRUCTURADOS REGISTRO

```
Program uno;
Const
  ...
Type
inmueble = record
   tipo: string;
   cantHab: integer;
   cantBaños:integer;
   precio: real;
   localidad: string;
end;
function iguales (i1,i2:inmueble): boolean;
begin
 ...
end;

Var
   inmu1, inmu2: inmueble;
```

```
Begin
 leer (inmu1);
 leer (inmu2);
 if (iguales (inmu1,inmu2) = true) then
    write (`Los registros son iguales`)
 else write (`Los registros no son iguales`);
End.
```

```
Begin
 leer (inmu1);
 leer (inmu2);
 if (iguales (inmu1,inmu2)) then
    write (`Los registros son iguales`)
 else write (`Los registros no son iguales`);
End.
```

AyPI – **REGISTRO**

EJERCITACIÓN

Escriba un programa que lea inmuebles hasta leer un inmueble cuya localidad es `XXX` Al finalizar informe de los inmuebles en la localidad de `La Plata` cuantos tienen al menos 2 habitaciones

```
Tipo `Depto`
cantHab:2
cantBaños 1
precio 15.200
Localidad La Plata
```

```
Tipo `Casa`
cantHab:3
cantBaños 2
precio 23.000
Localidad Gonnet
```

```
Tipo `Casa`
cantHab:5
cantBaños 3
precio 55.400
Localidad La Plata
```

```
Tipo `Casa`
cantHab:1
cantBaños 2
precio 18.000
Localidad La Plata
```

```
Tipo `Casa`
cantHab:4
cantBaños 1
precio 10.000
Localidad XXX
```

2

Al leer la Localidad XXX, ¿necesito leer todos los otros campos?

AyP – **REGISTRO**

EJERCITACIÓN

Escriba un programa que lea inmuebles hasta leer un inmueble cuya localidad es `XXX` Al finalizar informe de los inmuebles en la localidad de `La Plata` cuantos tienen al menos 2 habitaciones

```
Inicializar contadores (cant)
Leer registro (inmu)
While (no sea el ultimo registro) do
 begin
  if (inmu es de La Plata con al menos dos habitaciones) then
    incremento (cant)
  leer registro (inmu)
 end;
Write (`La cantidad es`, cant);
```

¿Cómo verifico las condiciones?

¿Qué modularizo?

AyPI – **REGISTROS**

EJERCITACIÓN

```
Program uno;
Const
  ...
Type
inmueble = record
   tipo: string;
   cantHab: integer;
   cantBaños:integer;
   precio: real;
   localidad: string;
end;

// módulos


Var
   inmu1, inmu2: inmueble;
   cant: integer;
```

```
Begin
 cant:=0;

 leer (inmu);

 while (inmu.localidad <> `XXX`) do
  begin
   if (cumple (inmu1) = true) then
     cant:= cant + 1;
   leer (inmu);
  end;
  write (`La cantidad es`, cant);
End.
```

AyPI – **REGISTROS**

EJERCITACIÓN

```
procedure leer (var i:inmueble);

Begin
    read(i.tipo);
    read(i.cantHab);
    read(i.cantBaños);
    read(i.precio);
    read(i.localidad);
end;
```

¿Qué alternativa conviene?

```
procedure leer (var i:inmueble);

Begin
    read(i.localidad);
    if (i.localidad <> `XXX`) then
     begin
      read(i.cantHab);
      read(i.cantBaños);
      read(i.precio);
      read(i.tipo);
     end;
end;
```

AyPI – **REGISTROS**

EJERCITACIÓN

```
function cumple(i:inmueble): boolean;
var
 ok:boolean;

begin
 if (i.localidad = `La Plata`) and
    (i.cantHab >= 2) then
  ok:= true
 else
  ok:= false;
 cumple:= ok;
end;
```

Otra opción

```
function cumple(i:inmueble): boolean;
begin
 cumple:= (i.localidad = `La Plata`)
                  and (i.cantHab >= 2);
end;
```

AyPI – TIPOS DE DATOS ESTRUCTURADOS REGISTRO

```
Program uno;
Const
   ....
Type

inmueble = record
    tipo: string;
    cantHab: integer;
    cantBaños:integer;
    precio: real;
    localidad: string;
end;

Var
    inmu1: inmueble;
```

¿Qué cambiamos si ahora se quiere agregar la fecha desde que el inmueble está disponible?

AyPI – TIPOS DE DATOS ESTRUCTURADOS REGISTRO

Opción 1

```
Program uno;
Type
inmueble = record
    tipo: string;
    cantHab: integer;
    cantBaños:integer;
    precio: real;
    localidad: string;
    dia:integer;
    mes:integer;
    año:integer;
end;
Var
    inmu1: inmueble;
```

¿Otra opción?

AyPI – TIPOS DE DATOS ESTRUCTURADOS REGISTRO

Opción 2

```
Program uno;
Type
fecha = record
 dia: integer;
 mes:integer;
 año:integer;
end;
inmueble = record
    tipo: string;
    cantHab: integer;
    cantBaños:integer;
    precio: real;
    localidad: string;
    fechaPub:fecha;
end;
```

¿Cómo hacemos ahora el proceso de lectura?

AyPI – TIPOS DE DATOS ESTRUCTURADOS REGISTRO

```
procedure leer (var i:inmueble);
Begin
   read(i.tipo);
   read(i.cantHab);
   read(i.cantBaños);
   read(i.precio);
   read(i.localidad);
   read (i.fechaPub);
end;
```

NO SE PUEDE ya que **i.fechaPub** es un registro y no se puede hacer read directamente

```
procedure leer (var i:inmueble);

Begin
   read (i.tipo);
   read (i.cantHab);
   read (i.cantBaños);
   read (i.precio);
   read (i.localidad);
   read (i.fechaPub.dia);
   read (i.fechaPub.mes);
   read (i.fechaPub.año);
end;
```

¿Otra alternativa?

AyPI – TIPOS DE DATOS ESTRUCTURADOS REGISTRO

```
procedure leerFecha (var f:fecha);
Begin
    read(f.dia);
    read(f.mes);
    read(f.año);
end;
```

```
procedure leer (var i:inmueble);
var
 fec:fecha;
Begin
    read(i.tipo);
    read(i.cantHab);
    read(i.cantBaños);
    read(i.precio);
    read(i.localidad);
    leerFecha (fec);
    i.fechaPub:= fec;
end;
```

Otra opción

```
procedure leer (var i:inmueble);
 begin
    read(i.tipo);
    read(i.cantHab);
    read(i.cantBaños);
    read(i.precio);
    read(i.localidad);
    leerFecha (i.fechaPub);
 end;
```

AyPI – **REGISTRO**

EJERCITACIÓN

Escriba un programa que lea inmuebles hasta leer un inmueble cuya localidad es `XXX` Al finalizar informe de los inmuebles en la localidad de `La Plata` con al menos dos habitaciones; y cuántos inmuebles se publicaron en el verano del 2020-21.

```
Localidad La Plata
Tipo `Depto`
cantHab:2
cantBaños 1
precio 15.200
fechaPub: 13/12/2020

Localidad La Plata
Tipo `Casa`
cantHab:3
cantBaños 2
precio 23.000
fechaPub: 23/01/2021

Localidad Gonnet
Tipo `Casa`
cantHab:5
cantBaños 3
precio 55.400
fechaPub: 14/02/2021

Localidad La Plata
Tipo `Casa`
cantHab:1
cantBaños 2
precio 18.000
fechaPub: 1/04/2021
```

```
Localidad XXX
```

```
2
2
```

AyPI – **REGISTRO**

EJERCITACIÓN

Escriba un programa que lea inmuebles hasta leer un inmueble cuya localidad es `XXX` Al finalizar informe de los inmuebles en la localidad de `La Plata`con al menos dos habitaciones; y cuántos inmuebles se publicaron en el verano del 2020.

```
Inicializar contadores (cantLP,cantVe)
Leer registro (inmu)
While (no sea el ultimo registro) do
 begin
   if (inmu es de La Plata con al menos dos
habitaciones) then
   incremento (cant)
  if (inmu se publicó en verano 2020-21) then
   incremento (cantVe)
  leer registro (inmu)
 end;
Write (`Las cantidades son`, cant,cantVe);
```

¿Qué modularizo?

¿Cómo verifico las condiciones?

AyPI – **REGISTRO**

EJERCITACIÓN

¿Cómo es la función verano?

```
Program uno;
Type
fecha = record
 dia:integer;  mes:integer;  año:integer;
end;
inmueble = record
   tipo: string; cantHab: integer;
   cantBaños:integer;
   precio: real; localidad: string;
   fechaPub:fecha;        end;
Procedure leer (var i:inmueble);
begin
...
end;
function cumple(i:inmueble): boolean;
begin
...
end;
function verano (i:fecha): boolean;
begin
...
end;
```

```
Var
 inmu:inmueble;
 cant,cantVe:integer;
Begin
 cantLP:= 0; cantVe:=0;
 leer (inmu);
 while (inmu.localidad <> `XXX`) do
  begin
   if (cumple (inmu) then
     cantLP:= cantLP + 1;
   if (verano (inmu.fechaPub) then
     cantVe:= cantVe + 1;
   leer (inmu);
  end;
  write (`La cantidades son`, cantLP,cantV
End.
```

AyPI – **REGISTROS**

EJERCITACIÓN

¿Otra forma?

```
function verano (f:fecha): boolean;

Var
   esVerano:boolean;
Begin
  if ( (f.dia>= 21) and (f.mes = 12)
        and (f.año = 2020) ) then esVerano:= true

  else if ( (f.dia < 21) and (f.mes = 3)
          and (f.año = 2021) ) then esVerano:= true

  else if ( ( (f.mes = 1) or (f.mes = 2) )
       and (f.año = 2021) ) then esVerano:= true

  else esVerano:= false;

verano:= esVerano;
end;
```

AyPI – **REGISTRO**

EJERCITACIÓN

Escriba un programa que lea inmuebles hasta leer un inmueble cuya localidad es `XXX` Al finalizar informe el precio de alquiler del inmueble que tiene la máxima cantidad de habitaciones y baños (suma de habitaciones + baños).

```
Localidad La Plata
Tipo `Depto`
cantHab:2
cantBaños 1
precio 15.200
```

```
Localidad Gonnet
Tipo `Casa`
cantHab:3
cantBaños 2
precio 23.000
```

```
Localidad La Plata
Tipo `Casa`
cantHab:5
cantBaños 3
precio 55.400
```

```
Localidad La Plata
Tipo `Casa`
cantHab:1
cantBaños 2
precio 18.000
```

```
Localidad XXX
```

55.400

AyPI – **REGISTRO**

EJERCITACIÓN

Escriba un programa que lea inmuebles hasta leer un inmueble cuya localidad es `XXX` Al finalizar informe de el precio de alquiler del inmueble que tiene la máxima cantidad de habitaciones y baños (suma de habitaciones + baños).

```
Inicializar máximo (max)
Leer registro (inmu)
While (no sea el ultimo registro) do
 begin
  sumadeHab:= inmu.cantHab + inmu.cantBaños
   if (sumadeHab supera el máximo) then
    actualizo el máximo (max)
    guardo el precio máximo (precioMax)
  leer registro (inmu)
 end;
Write (`El precio del inmueble con máxima
cantidad de habitaciones + baños es`, precioMax);
```

¿Cómo verifico las condiciones?

¿Qué modularizo?

AyPI – **REGISTROS**

EJERCITACIÓN

¿Qué cambiaría del programa si ahora se quiere informar tipo, habitaciones, baños precio y localidad del inmueble con mayor cantidad de habitaciones + baños?

```
Program uno;
Type
inmueble = record
    tipo: string;
    cantHab: integer;
    cantBaños:integer;
    precio: real;
    localidad: string;
end;

Procedure leer (var i:inmueble);
begin
...
end;

Var
 inmu:inmueble;
 sumadeHab:integer;
 max: integer;
 precioMax:real;
```

```
Begin
 max:= -1;
 leer (inmu);
 while (inmu.localidad <> `XXX`) do
  begin
   sumadeHab:=inmu.cantHabitaciones + inmu.cantBaños;
   if (sumadeHab >= max) then
     begin
       max:= sumadeHab;
       precioMax:= inmu.precio;
     end;
   leer (inmu);
  end;
  write (`El precio del inmueble con máxima cantidad
de habitaciones + baños es`, precioMax);
End.
```

AyPI – **REGISTROS**

EJERCITACIÓN

```
Program uno;
Type
inmueble = record
    tipo: string;
    cantHab: integer;
    cantBaños:integer;
    precio: real;
    localidad: string;
end;

Procedure leer (var i:inmueble);
begin
...
end;

Var
 inmu:inmueble;
 sumadeHab:integer;
 max: integer;
 tipoMax, locMax:string;
 cantHMax,cantBMax:integer;
 precioMax:real;
```

```
Begin
 max:= -1;  leer (inmu);
 while (inmu.localidad <> `XXX`) do
  begin
   sumadeHab:=inmu.cantHabitaciones + inmu.cantBaños;
   if (sumadeHab >= max) then
     begin
       max:= sumadeHab;
       tipoMax:= inmu.tipo;
       cantHMax:= inmu.cantHab;
       cantBMax:= inmu.cantBaños;
       precioMax:= inmu.precio;
       locMax:= inmu.localidad;
     end;
   leer (inmu);
  end;
  write (`El precio del inmueble con máxima cantidad
de habitaciones + baños es`, tipoMax, cantHMax,
cantBMax, precioMax,locMax);
End.
```

REGISTRO INMUEBLE

AyPI – **REGISTROS**

EJERCITACION

```
Program uno;
Type
inmueble = record
   tipo: string;
   cantHab: integer;
   cantBaños:integer;
   precio: real;
   localidad: string;
end;

Procedure leer (var i:inmueble);
begin
...
end;

Var
 inmu:inmueble;
 sumadeHab:integer;
 max: integer;
 inmuMax:inmueble;
```

```
Begin
 max:= -1;
 leer (inmu);
 while (inmu.localidad <> `XXX`) do
  begin
   sumadeHab:=inmu.cantHabitaciones + inmu.cantBaños;
   if (sumadeHab >= max) then
     begin
       max:= sumadeHab;
       inmuMax:= inmu;
     end;
   leer (inmu);
  end;
  write (`Los datos del inmueble con máxima cantidad
de habitaciones + baños son`, inmuMax.tipo,
inmuMax.cantHab, inmuMax.cantBaños, inmuMax.precio,
inmuMax.localidad);
End.
```

AyPI – **TIPOS DE DATOS ESTRUCTURADOS** REGISTRO

¿Qué cambio y consideraciones debo tener si quiero informar la cantidad de propiedades por localidad?

AyPI – **REGISTRO**

EJERCITACIÓN

Escriba un programa que lea inmuebles hasta leer un inmueble cuya localidad es `XXX`. Se pide informar la cantidad de inmuebles por Localidad.

```
Localidad La Plata
Tipo `Depto`
cantHab:2
cantBaños 1
precio 15.200
```

```
Localidad Gonnet
Tipo `Casa`
cantHab:3
cantBaños 2
precio 23.000
```

```
Localidad La Plata
Tipo `Casa`
cantHab:5
cantBaños 3
precio 55.400
```

```
Localidad La Plata
Tipo `Casa`
cantHab:1
cantBaños 2
precio 18.000
```

```
Localidad XXX
```

La Plata 3, Gonnet 1

¿Cómo hago?

AyPI – **REGISTRO**

EJERCITACIÓN

Escriba un programa que lea inmuebles hasta leer un inmueble cuya localidad es `XXX`. Se pide informar la cantidad de inmuebles por Localidad.

```
Localidad La Plata
Tipo `Depto`
cantHab:2
cantBaños 1
precio 15.200
```

```
Localidad La Plata
Tipo `Casa`
cantHab:5
cantBaños 3
precio 55.400
```

```
Localidad La Plata
Tipo `Casa`
cantHab:1
cantBaños 2
precio 18.000
```

```
Localidad Gonnet
Tipo `Casa`
cantHab:3
cantBaños 2
precio 23.000
```

```
Localidad XXX
```

La Plata 3, Gonnet 1

NECESITO que los datos vengan “organizados” por el criterio que se quiere informar

AyPI – **REGISTRO**

EJERCITACIÓN

Escriba un programa que lea inmuebles hasta leer un inmueble cuya localidad es `XXX`. Se pide informar la cantidad de inmuebles por Localidad. **Los datos se leen organizados por localidad.**

```
Localidad La Plata
Tipo `Depto`
cantHab:2
cantBaños 1
precio 15.200

Localidad La Plata
Tipo `Casa`
cantHab:5
cantBaños 3
precio 55.400

Localidad La Plata
Tipo `Casa`
cantHab:1
cantBaños 2
precio 18.000
```

→ La Plata 3

```
Localidad Gonnet
Tipo `Casa`
cantHab:3
cantBaños 2
precio 23.000
```

→ Gonnet 1

¿Cómo hago?

```
Localidad XXX
```

AyPI – **REGISTRO**

EJERCITACIÓN

Escriba un programa que lea inmuebles hasta leer un inmueble cuya localidad es `XXX`. Se pide informar la cantidad de inmuebles por Localidad. **Los datos se leen organizados por localidad.**

```
Leo un inmueble (inmu)

while (no sea el último) do
 begin
  guardo la localidad actual
  inicializo contador de la localidad (cantI)
  while (no sea el ultimo) y (sea la misma localidad) do
   begin
    cuento un inmueble mas (cantI)
    leo un inmueble (inmu)
   end;
  informo cantidad de lugares por localidad (cantI)
 end;
End.
```

AyPI – **REGISTROS**

EJERCITACIÓN

```
Program uno;
Type
inmueble = record
    tipo: string;
    cantHab: integer;
    cantBaños:integer;
    precio: real;
    localidad: string;
end;

Procedure leer (var i:inmueble);
begin
...
end;

Var
 inmu:inmueble;
 locActual:string;
 cantI: integer;
```

¿Por qué se inicializa cantI en ese lugar?

¿Es correcto leer una sola vez dentro del segundo while?

```
Begin
 leer (inmu);
 while (inmu.localidad <> `XXX`) do
  begin
   locActual:=inmu.localidad;
   cantI:=0;
   while (inmu.localidad = locActual) do
      begin
         cantI:= cantI + 1;
         leer (inmu);
      end;
   write (`La cantidad de inmuebles es`, cantI);
  end;
End.
```

¿Qué cambio si quiero imprimir el nombre de la localidad?

AyPI – **REGISTROS**

EJERCITACIÓN

```
Begin
 leer (inmu);
 while (inmu.localidad <> `XXX`) do
  begin
   locActual:=inmu.localidad;
   cantI:=0;
   while (inmu.localidad = locActual) do
      begin
         cantI:= cantI + 1;
         leer (inmu);
      end;
   write (`La cantidad de inmuebles es`, cantI,
          `y la localidad es` inmu.localidad);
  end;
End.
```

```
Begin
 leer (inmu);
 while (inmu.localidad <> `XXX`) do
  begin
   locActual:=inmu.localidad;
   cantI:=0;
   while (inmu.localidad = locActual) do
      begin
         cantI:= cantI + 1;
         leer (inmu);
      end;
   write (`La cantidad de inmuebles es`, cantI,
          `y la localidad es` locActual);
  end;
End.
```

¿La dos soluciones cumplen lo pedido?

¿Qué cambios debo realizar para informar la localidad con más inmuebles?

AyPI – **REGISTRO**

EJERCITACIÓN

Escriba un programa que lea inmuebles hasta leer un inmueble cuyo precio es -1. Se pide informar la cantidad de inmuebles por Localidad.

```
Localidad La Plata
Tipo `Depto`
cantHab:2
cantBaños 1
precio 15.200
```

```
Localidad La Plata
Tipo `Casa`
cantHab:5
cantBaños 3
precio 55.400
```

```
Localidad La Plata
Tipo `Casa`
cantHab:1
cantBaños 2
precio 18.000
```

```
Localidad Gonnet
Tipo `Casa`
cantHab:3
cantBaños 2
precio 23.000
```

```
precio  -1
```

La Plata 3, Gonnet 1

AyPI – **REGISTROS**

EJERCITACIÓN

```
Begin
 leer (inmu);
 while (inmu.precio <> -1) do
  begin
   locActual:=inmu.localidad;
   cantI:=0;
   while (inmu.precio <> -1) and
         (inmu.localidad = locActual) do
      begin
         cantI:= cantI + 1;
         leer (inmu);
      end;
   write (`La cantidad de inmuebles es`, cantI,
          `y la localidad es` locActual);
  end;
End.
```

Válida

```
Begin
 leer (inmu);
 while (inmu.precio <> -1) do
  begin
   locActual:=inmu.localidad;
   cantI:=0;
   while (inmu.localidad = locActual) do
      begin
         cantI:= cantI + 1;
         leer (inmu);
      end;
   write (`La cantidad de inmuebles es`, ca
          `y la localidad es` locActual);
  end;
End.
```

Inválida ¿Por qué?