

PROGRAMACIÓN I

TEORÍA – CECILIA SANZ

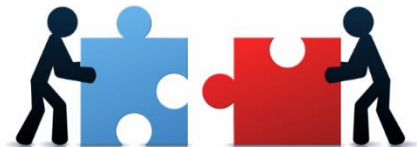
Temas

- ✓ Estructura de datos arreglo
- ✓ Definición
- ✓ Operaciones
- ✓ Ejemplos

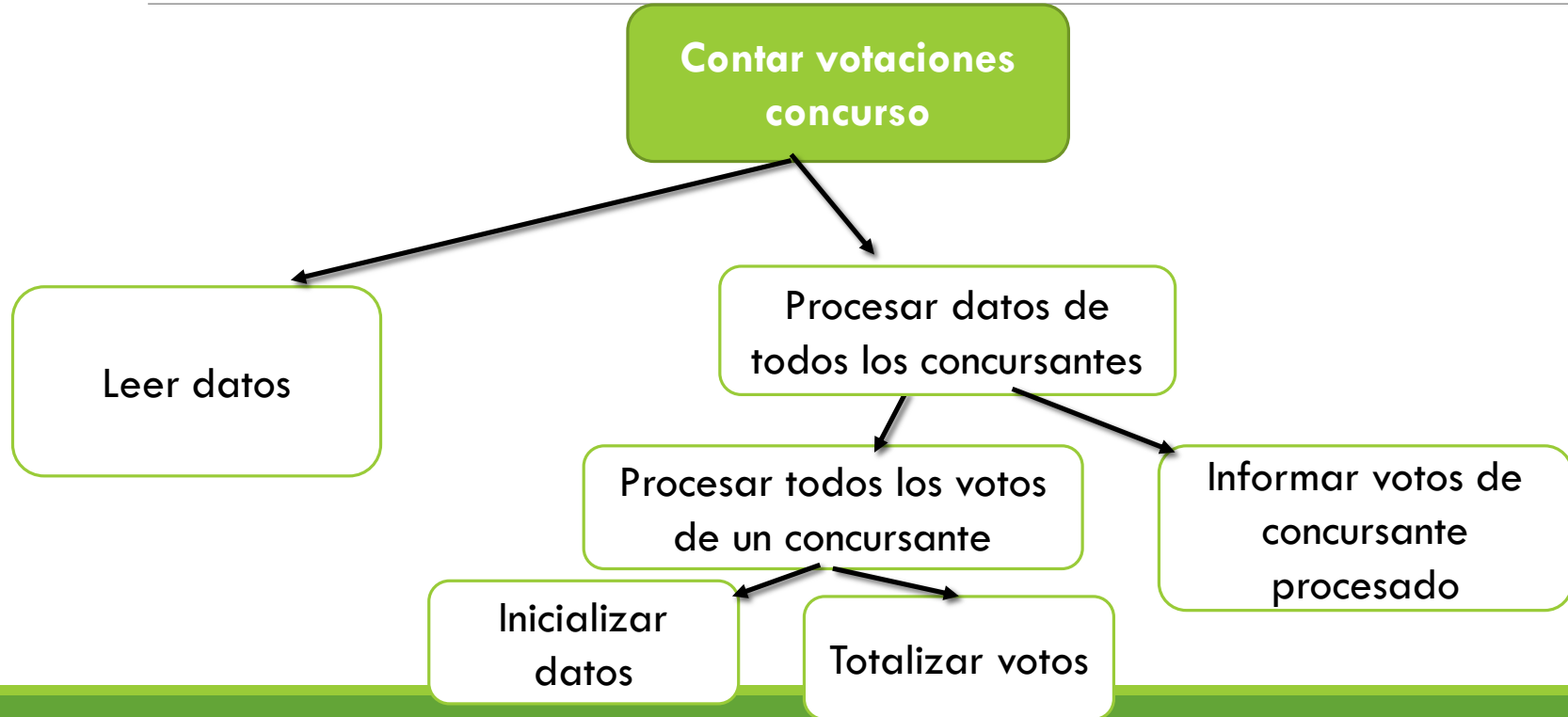
REPASO

Supongamos que estamos recibiendo votaciones para diferentes concursantes de un programa de TV. Cada voto indica a qué participante se está votando a través de su código que va de 1 a 20 y de qué provincia proviene el voto. **Los votos vienen ordenados por código de participante.** Debo informar el total de votos por cada participante.

¿Cómo lo resolvemos?



EJERCICIO de REPASO



```
Program votación;  
Const maxPart= 20; maxProv=23; fin=0;  
Type codigospartic= 0.. maxPart; codigosprov= 1.. maxProv;  
    votos = record  
        cod_participante: codigospartic;  
        prov_votante: codigosprov;  
    end;  
Var v: votos; actual: codigospartic; totalvotos: integer;
```

{Aquí va el proceso de Leer y el de ProcesarUnconcurante}

```
Begin  
    Leer(v);  
    While (v.cod_participante<> fin) do  
        begin  
            ProcesarUnConcurante (v, actual, totalvotos);  
            Writeln("Participante", actual, "tiene: ", totalvotos);  
        End;  
End.
```

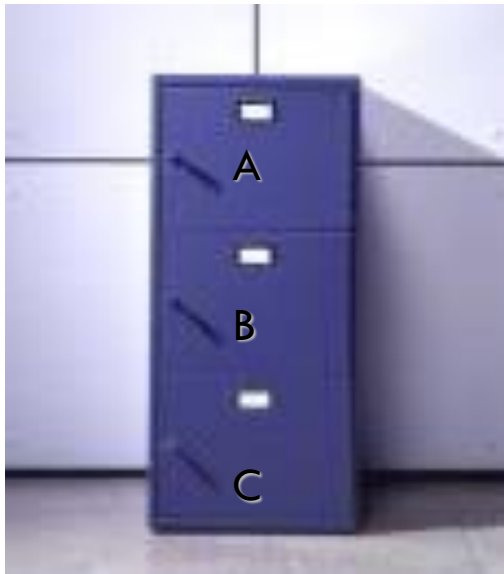
```
Procedure Leer (VAR v:votos);  
Begin  
    readln(v.cod_participante);  
    if (v.cod_participante <> fin)  
        then readln(v.prov_votante);  
End;
```

```
Procedure ProcesarUnConcursante (VAR v: votos; VAR actual: codigospartic; VAR  
totalvotos:integer);  
Begin  
    actual:= v.cod_participante;  
    totalvotos:=0;  
    While (actual = v.cod_participante) do  
        Begin  
            totalvotos:=totalvotos + 1;  
            Leer(v);  
        end;  
End;
```



Motivación

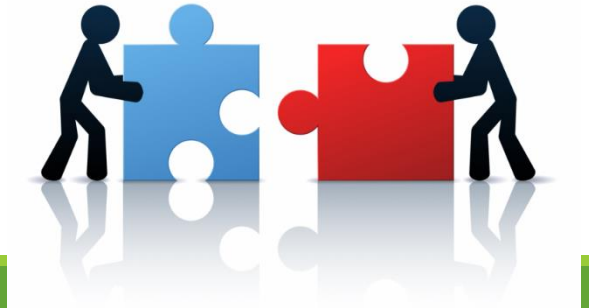
Motivación



Motivación

Supongamos que estamos recibiendo votaciones para diferentes concursantes de un programa de TV. Cada voto indica a qué participante se está votando a través de su código que va de 1 a 20. Debo informar el total de votos por cada participante.

¿Cómo lo resolvemos?



Motivación

- 
- Para resolver estos problemas podemos usar una estructura de datos tipo **arreglo**.

Un tipo de dato *Arreglo* es una colección de elementos a los que se accede mediante un índice.

SI TIENE UN SOLO INDICE SE DENOMINA VECTOR

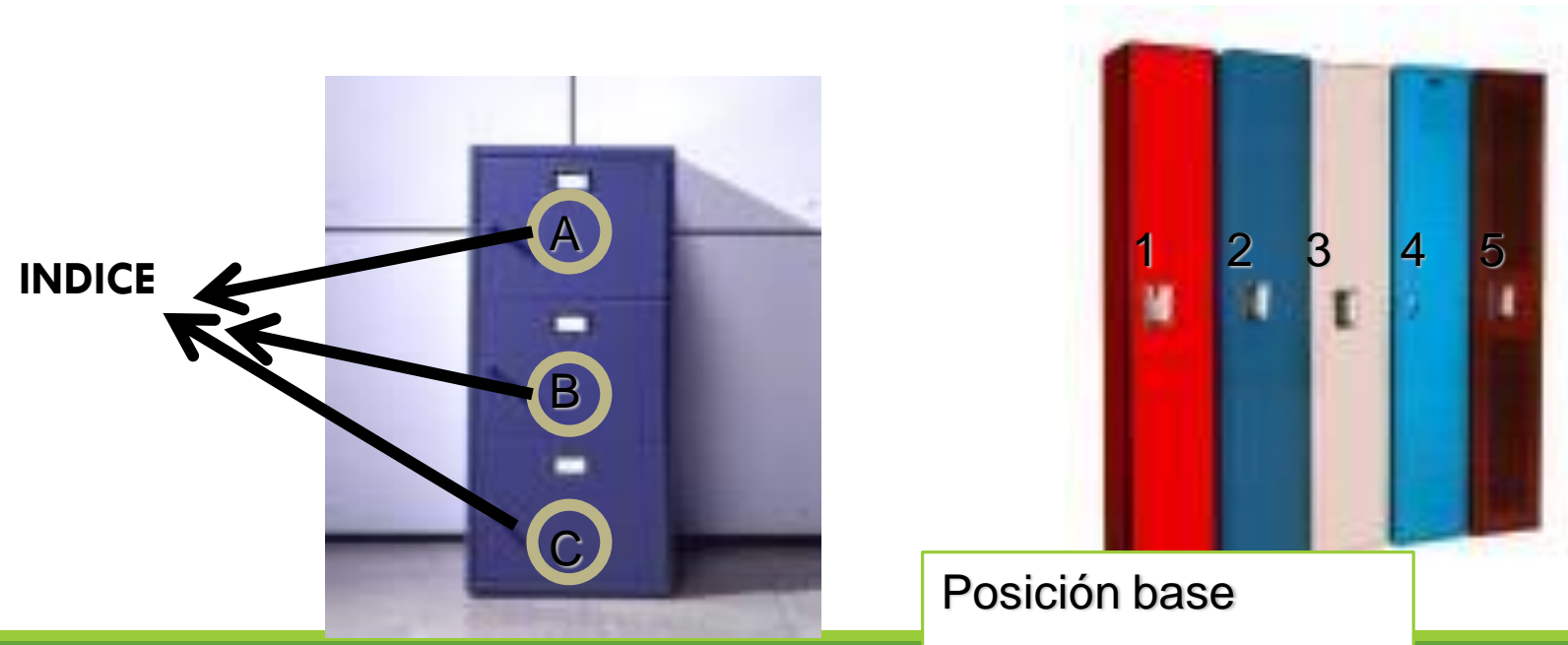
Un arreglo (**ARRAY**) es una estructura de datos que permite acceder a cada componente a través de una **variable índice**, que da la **posición** de la componente dentro de la estructura de datos.

Conceptos iniciales



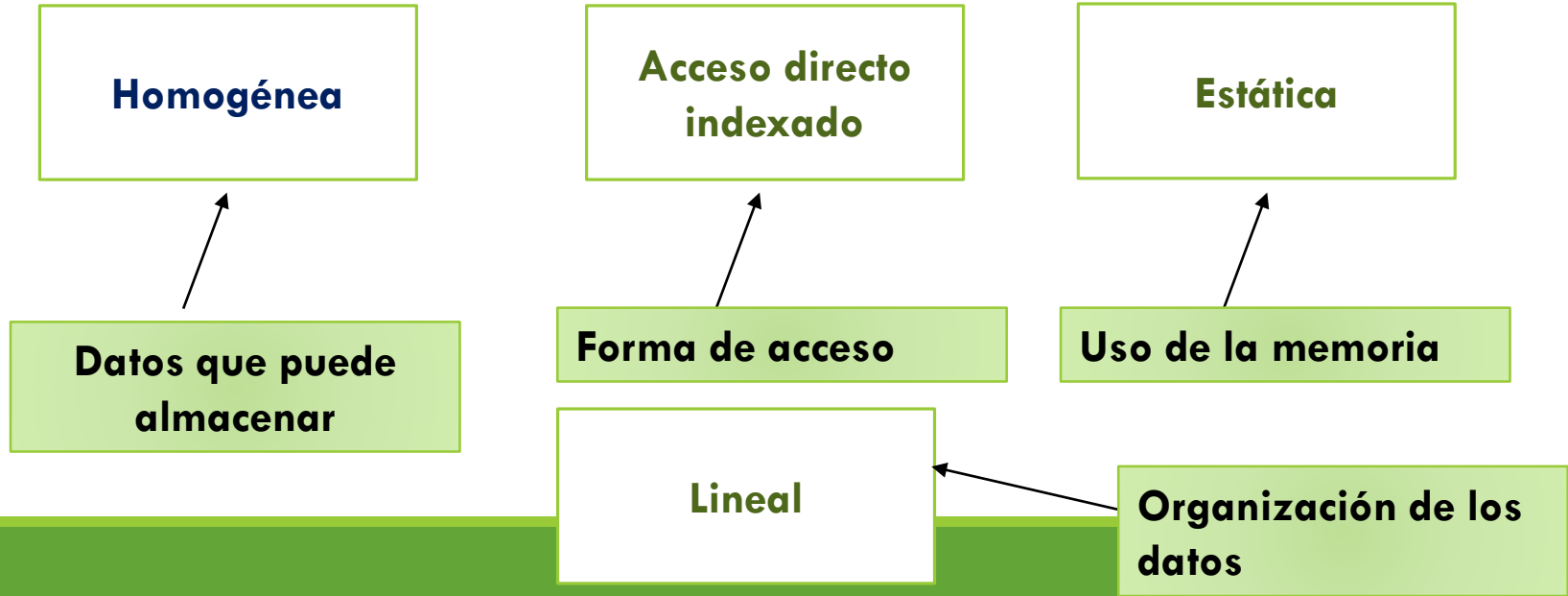
Estructura de datos-ARREGLO-Concepto

Un tipo de dato *Arreglo* es una colección *indexada* de elementos.



Estructura de datos-ARREGLO-Concepto

- Un tipo de dato *Arreglo* es una colección *indexada* de elementos.



Estructura de datos-ARREGLO-Declaración

Program uno;

type concursantes= **array**[indice] **of** tipo;

Subrangos de tipos ordinales

integer

char

boolean

enumerativo

Los **elementos** de un arreglo pueden pertenecer a **cualquier tipo de datos de asignación estática:**

- Enteros
- Reales
- Caracteres
- Registros, Otro arreglo

Estructura de datos-ARREGLO-Concepto



¿Qué es el índice de un arreglo?

Es una variable de tipo ordinal.

Permite acceder a cualquier elemento de manera directa.

Representa un desplazamiento desde la posición inicial del arreglo

Estructura de datos-VECTOR-Concepto

¿Cómo declaramos un vector en Pascal?

```
TYPE vector = array [1..10] of integer;
```



Dim física

Tipo de elemento
que almacenaré

Estructura de datos-VECTOR-Concepto


En el ejemplo

```
TYPE vector= array [1..20] of integer;
```



Dim física

A blue bracket is drawn under the range [1..20] in the code above, pointing down to the text 'Dim física'.



Tipo de elemento
que almacenaré

A blue bracket is drawn under 'of integer;' in the code above, pointing down to the text 'Tipo de elemento que almacenaré'.

Declaraciones en Pascal - Ejemplos

Type

Sucursales = Array [1..100] of real;

Temperaturas = Array [1..30] of real;

AñosAutos = Array [2000..2012] of integer;

Categorias = Array ['A' .. 'D'] of real;

Frecuencia = Array [char] of integer;

Var

Autos : añosAutos; {este vector tiene 11 componentes de tipo entero}

Frec : frecuencia; {Este vector tendrá 255 componentes de tipo entero}

Sucursal : Sucursales; {Este vector tendrá 100 componentes de tipo real}

TotalCategorias: categorias; { este vector tiene 4 componentes de tipo real}

TempMensual: Temperaturas; { este vector tiene 30 componentes de tipo real}

VECTORES



Estructura de datos-ARREGLO-Operaciones

- Asignar elementos.
- Lectura/escritura.
- Recorridos.
- Agregar elementos.
- Insertar elementos.
- Borrar elementos.
- Búsqueda de un elemento.
- Ordenación de un arreglo.

OPERACIÓN DE ASIGNACIÓN

Estructura de datos-VECTOR-Operaciones

Program uno;

type

concurantes=array [1..20] of integer;

var

c: concurantes;

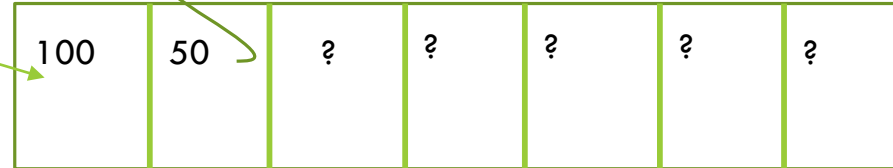
begin

c[1]:= 100;

c[2]:= 50;

.....

end.



OPERACIÓN DE CARGAR UN VECTOR (LECTURA DESDE TECLADO)

Estructura de datos-VECTOR-Operaciones

Program uno;

Const max=20;

type

concursantes= array [1..max] of integer;

var

c: concursantes;

x,i:integer;

begin

for i:= 1 to max do

begin

readln (x);

c[i]:= x;

End;

.....

for i:= 1 to max do

readln (c[i]);

?	?	?	?	?	?	?
---	---	---	---	---	---	---

100	500	200	1000	90	900	1003
-----	-----	-----	------	----	-----	------

Ejemplo

Suponga que se hay un concurso de poesía y se registran la cantidad de inscriptos por día durante el mes de septiembre. Informar los días que superan el promedio de inscriptos del mes.

Estructura de datos-VECTOR-Ejercicio

```
Program Ejemplo1;  
  CONST max=30;  
  type  
    rango: 1..max;  
    inscriptos= array [1..max] of integer;  
  var insc: inscriptos; i: rango; prom: real;  
begin  
  CargarInscriptos (insc);  
  prom: = promedio(insc);  
  For i:= 1 to max do  
    If (insc [i] > prom) then writeln ('El día ', i, 'supera el promedio del mes');  
end.
```

Estructura de datos-ARREGLO-Ejercicio

```
Procedure CargarInscriptos (var insc: inscriptos);
```

```
Var i: rango;
```

```
Begin
```

```
  For i:= 1 to max do
```

```
    readln(insc[i]);
```

```
  end.
```

```
Function Promedio (insc:inscriptos): real;
```

```
Var i: rango; total: integer;
```

```
Begin
```

```
  total:=0;
```

```
  for i:=1 to max do
```

```
    total:=total+ insc[i];
```

```
  promedio:= total/max;
```

```
End;
```

1	2	3	4	30	31
?	?	?	?	?	?	?

OPERACIÓN DE RECORRIDO

Estructura de datos-ARREGLO- Recorridos



Recorrido completo.

Recorrido condicional.

Vector : Operación de Recorrido

La operación de Recorrido en un vector consiste en recorrer el vector de manera total o parcial, para realizar algún proceso sobre sus elementos.

La operación de Recorrido Total, implica analizar **todos** los elementos del vector, lo que lleva a recorrer completamente la estructura.

La operación de Recorrido Parcial, implica analizar los elementos del vector, **hasta** encontrar aquel que cumple con lo pedido. Puede ocurrir que se recorra todo el vector.

¿Qué estructuras de control conviene utilizar en cada caso?

¿Ejemplos?

Ejemplo

Arreglos: Recorridos

Ejemplo de Recorrido Total

Por ejemplo si nos piden saber la cantidad de veces que aparece el valor 23 en un vector de 100 componentes, habrá que recorrerlo completo, contando las apariciones del 23 (**ocurrencias del valor 23**):

Arreglos: Recorrido Total

```
Program REC-TO23;  
  Const DIMF= 100; valor=23;  
  Type   rango= 1..DIMF;  
          vector = array [ 1..DIMF ] of integer ;  
  
  Var a: vector; J : rango; Cant : integer ;  
  
  Begin  
    Cant := 0;  
    For J := 1 to DIMF do {cargando datos en todas las posiciones del vector}  
      Read ( a [ J ] );  
      For J := 1 to DIMF do {recorrido total del vector}  
        If ( a[ J ] = valor ) then Cant := Cant + 1;  
      Writeln ( 'Veces que aparece', valor, 'es: ', Cant );  
  end.
```


Ejemplo

Arreglos: Recorrido Condicional

Ejemplo de Recorrido Parcial

Encontrar la posición del primer cero en un vector de 200 componentes, sabiendo que por lo menos existe un cero.

Es claro que en este caso no conviene hacer un recorrido total del vector, pues nos piden la posición del primer cero sabiendo que existe al menos uno.

Arreglos: Recorrido Condicional

```
Program POSCERO;  
Const DIMF=200; valor=0;  
Type rango= 1..DIMF;  
       vector = Array [ 1..DIMF] of integer;  
Var V: vector; J : rango;  
Begin  
    CargarVector(V);  
    J := 1;  
    { Recorre el vector hasta encontrar el cero }  
    While ( V [ J ] <> valor )do  
        J := J + 1;  
    WriteLn ( 'La posición es : ', J );  
End.
```

OJO - Esto debiera modificarse en caso que no me aseguren que el cero está seguro

Arreglos: Recorrido Condicional

```
Program POSCE;  
Const DIMF=1000; valor=0;  
Type rango= 1..DIMF+1;  
      vector = Array [ 1..DIMF] of integer;  
Var A: vector; J : rango; encuentre:boolean;  
Begin  
  CargarVector(A); {Se considera que el vector se carga en todas las posiciones}  
  encuentre:=false; J := 1;  
  {Recorre el vector hasta encontrar el cero o hasta que se acabe el vector}  
  While ( not encuentre) and (J<=DIMF )do  
    If (A [J] = valor) then encuentre:= true  
      else J := J + 1;  
  If encuentre then Writeln ( 'La posición es : ', J ) ;  
End.
```