

# TEMAS DE INTRO I

- Pilas (datos/estructuras de control)
- Filas (datos/estructuras de control)
- Modularización y Parámetros
- Variables
- Método de Desarrollo
- Funciones
- Arreglos
- Ordenamientos
- Matrices

# Matrices

## Arreglos de más de una dimensión

### Const

Inicio1=1;

Fin1=2;

Inicio2=1;

Fin2=3;

### Type

Matriz: Array [inicio1..fin1, inicio2..fin2 ] of **tipo Primitivo**;

### Var

Mat: Matriz;

inicio1

fin1


inicio2

fin2

Se accede por celda indicando cada dimensión

**Mat[1,3]**

**Const**

.....

**Type**

**Matriz: Array [inicio1..fin1, inicio2..fin2, ] of Integer;**

**Var**

**Mat: Matriz;**

inicio1

fin1

		44

inicio2

fin2

**Mat[1,3]:=44;**

**Const**

.....

**Type**

**Matriz: Array [inicio1..fin1, inicio2..fin2, ] of Integer;**

**Var**

**Mat: Matriz;**

inicio1

fin1

		54

inicio2

fin2

**Mat[1,3]:=44;**

**Mat[1,3]:= Mat[1,3] + 10;**

Se puede “graficar” de las dos maneras, pero luego se debe respetar la convención, no mezclar los índices

## Const

AñoInicio=1;

AñoFin=4;

Cuat1=1;

Cuat2=2

## Type

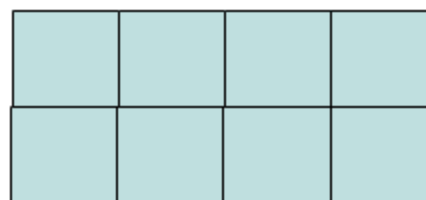
Materias: Array [AñoInicio..AñoFin,Cuat1..Cuat2] of TipoPrimitivo;

1				
2				
	1	2	3	4

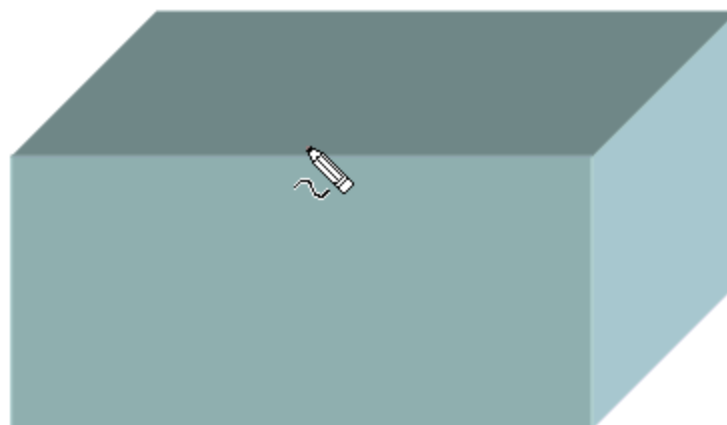
1		
2		
3		
4		
	1	2

# Se pueden graficar?

Var **Nombre2D** : array [**Inicio1**..**Fin1**, **Inicio2**..**Fin2**] of **TipoPri**;



Var **Nombre3D** : array [**Inicio1**..**Fin1**, **Inicio2**..**Fin2**, **Inicio3**..**Fin3**] of **TipoPri**;

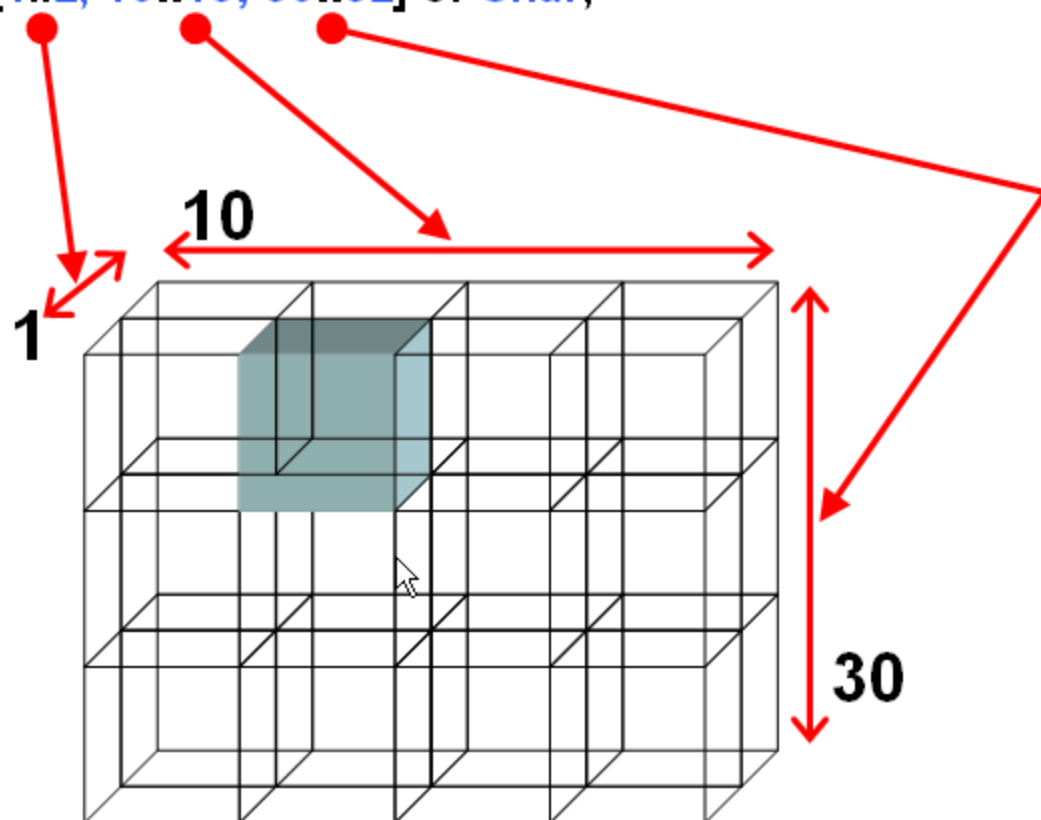


Var **Nombre4D** : array [**Inicio1**..**Fin1**, **Inicio2**..**Fin2**, **Inicio3**..**Fin3** , **Inicio4**..**Fin4**] of **TipoPri**;



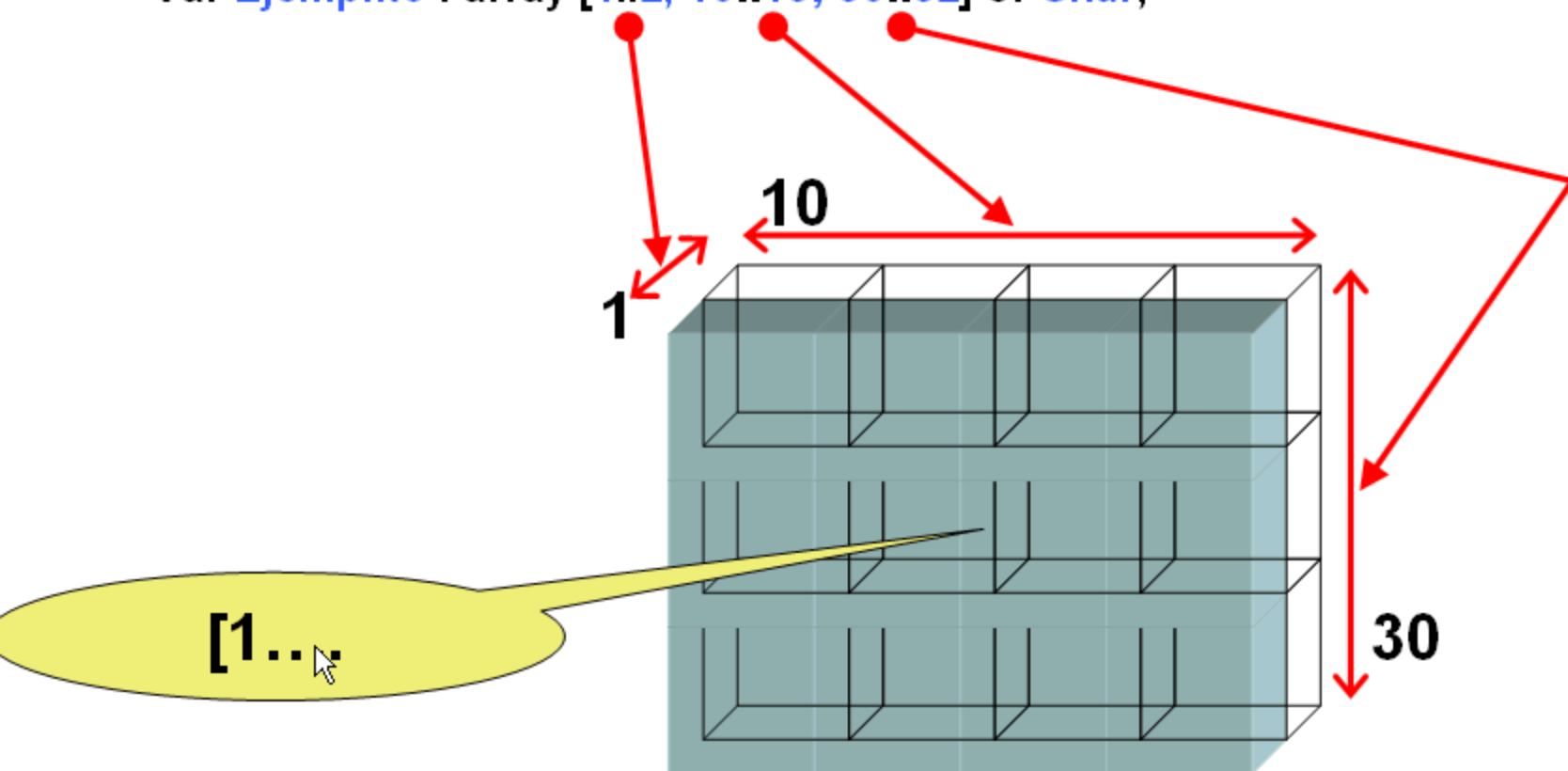
# Ejemplo de tres dimensiones

Var **Ejemplito** : array [1..2, 10..13, 30..32] of Char;



# Ejemplo de tres dimensiones

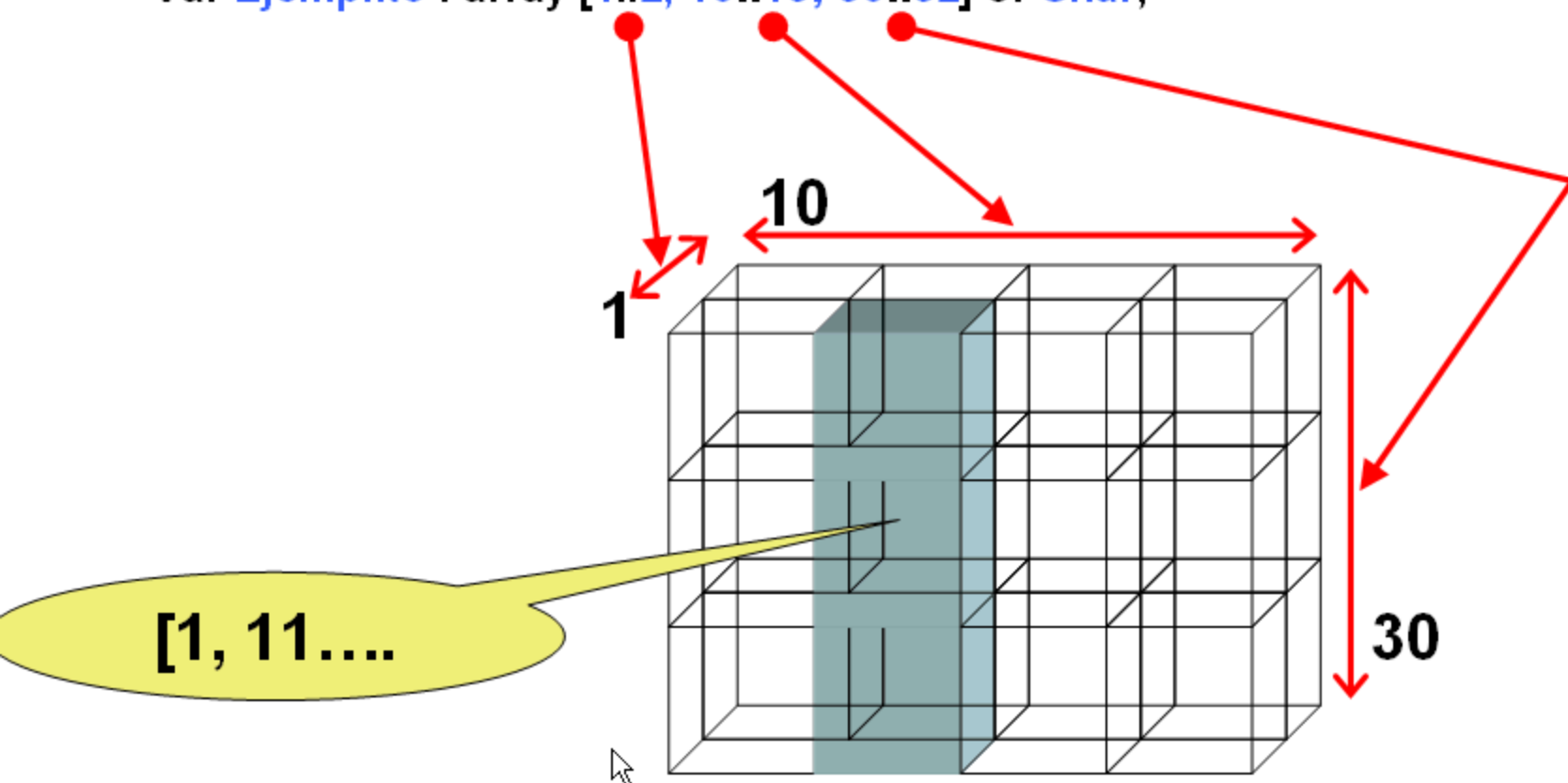
Var **Ejemplito** : array [1..2, 10..13, 30..32] of Char;





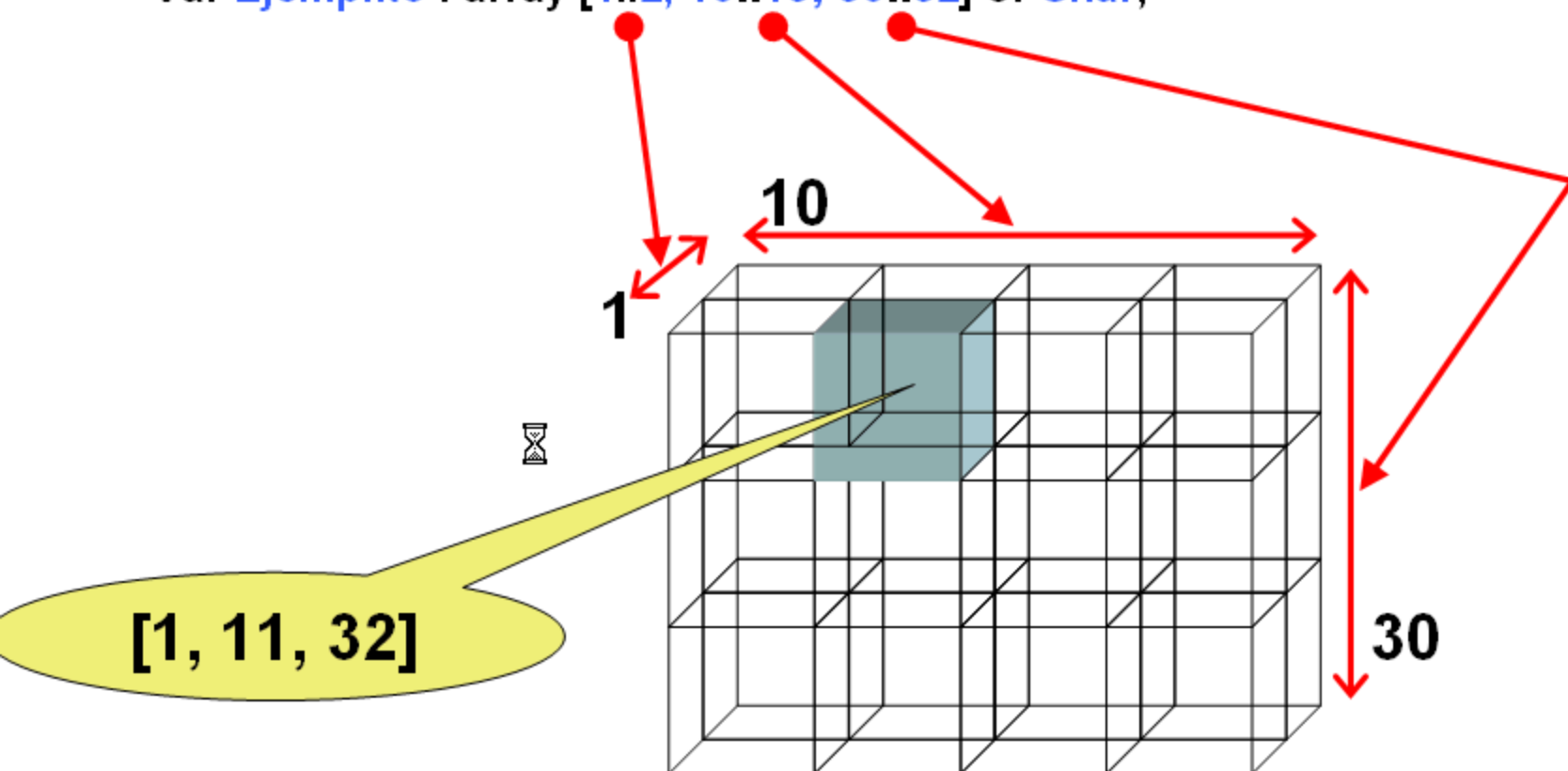
# Ejemplo de tres dimensiones

Var **Ejemplito** : array [1..2, 10..13, 30..32] of Char;



# Ejemplo de tres dimensiones

Var **Ejemplito** : array [1..2, 10..13, 30..32] of Char;



- Definir un matriz de dos dimensiones de enteros y cargarla por teclado
- Implementar un módulo que pase como parámetro una matriz de dos dimensiones de enteros, y un valor a buscar y devuelva la primer posición en la que está. Si el elemento no está, devuelve (-1,-1). Una vez que encuentra la primer ocurrencia del elemento, abandona la búsqueda

```
program EjemploMatriz;  
    {busca si un elemento esta en la matriz}  
    const  
        maxfila=2;  
        maxcol=2;  
    type  
        matriz= array [1..maxfila, 1..maxcol] of integer;  
  
    procedure cargarmatriz(var matint:matriz);  
    var  
        fi, co : integer;  
    begin  
        for fi:=1 to maxfila do  
            for co:=1 to maxcol do  
                readln(matint[fi,co]);  
            end;  
        end;
```

```
program EjemploMatriz;  
  {busca si un elemento esta en la matriz}  
  const  
    maxfila=2;  
    maxcol=2;  
  type  
    matriz= array [1..maxfila, 1..maxcol] of integer;  
  .....
```

```
Function estaenFila(matint: matriz; fil,dato:integer): boolean;  
  var  
    col: integer;  
  begin  
    col:=1;  
    while (col<= maxcol) and (matint[fil, col]<> dato) do  
      col:= col+1;  
    if col > maxcol  
      then estaenFila:= false  
      else estaenFila:= true;  
  end;
```

program EjemploMatriz;

.....

var

dato, fil: integer;

matint:matriz;

encontre: boolean;

begin

cargarMatriz(matint);

writeln('ingrese dato');

Readln (dato);

encontre:=false;

fil:=1;

while (fil<=maxfila) and not encontre do

begin

encontre:= estaenFila(matint,fil,dato);

fil:= fil +1

end;

if encontre

then writeln (' el elemento esta')

else writeln (' el elemento no esta');

end.