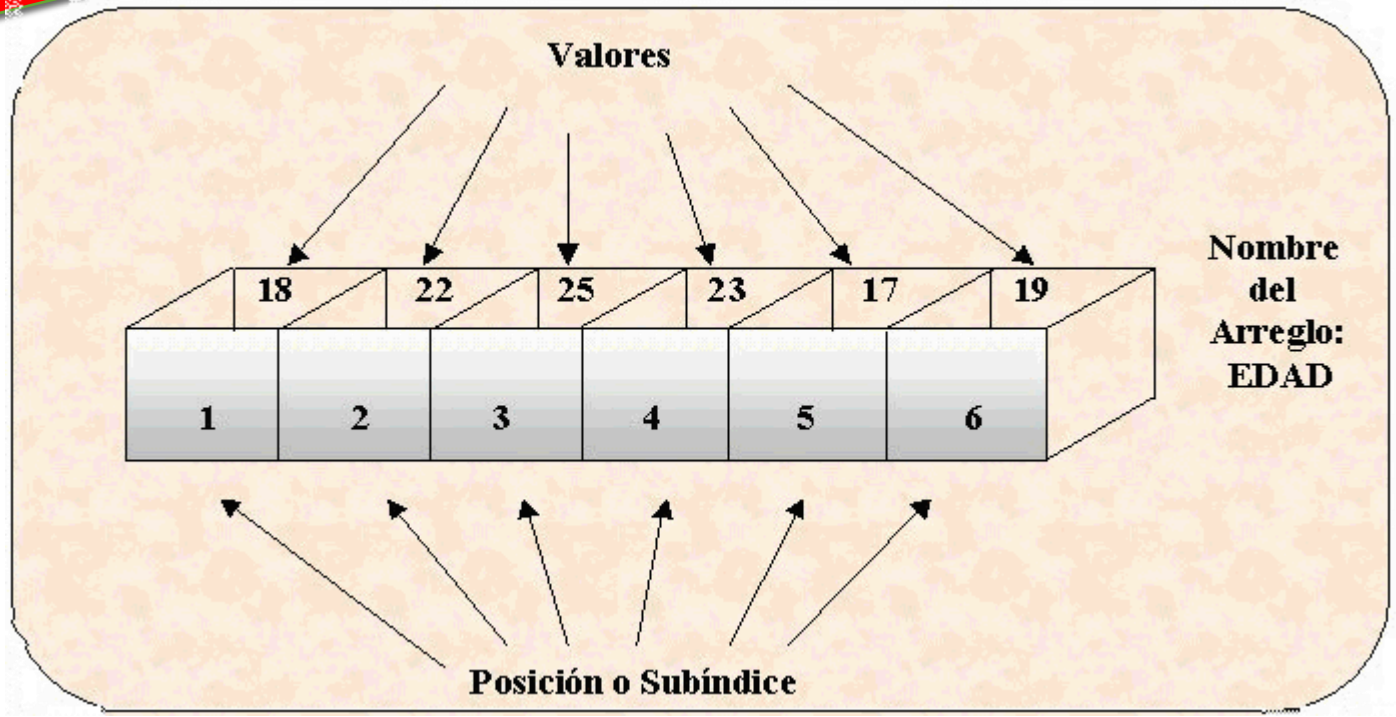
PROGRAMACIÓN I

TEORÍA – CECILIA SANZ

Temas

- ✓ Operaciones
- ✓ Ejemplos

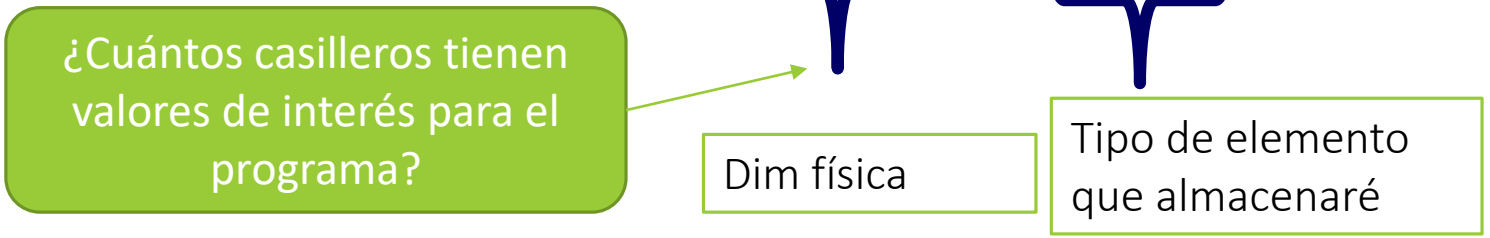
Motivación


Valores
18
22
25
23
17
19
1
2
3
4
5
6
Nombre del Arreglo: EDAD
Posición o Subíndice

Estructura de datos-VECTOR-Concepto

¿Cómo declaramos un vector en Pascal?

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



Dimensión física y lógica del vector

Arreglos: dimensiones…

Cuando se trabaja con arreglos se deben hacer consideraciones importantes acerca de algunos aspectos:

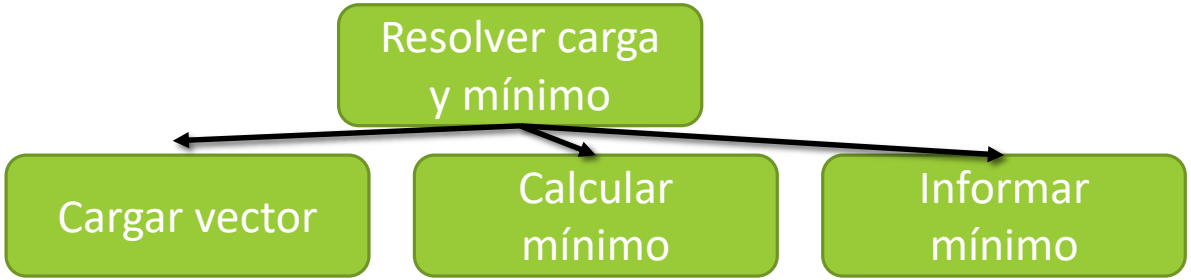
- Dimensión Física del arreglo
- Dimensión Lógica del arreglo

Ejemplo

Supongamos que se quiere cargar una estructura que permita almacenar números enteros **hasta leer el número 99**. Luego se pide hacer un módulo que recorra la estructura y calcule el valor mínimo. Se sabe que a lo sumo son 1000 números.

Ejemplo

Supongamos que se quiere cargar una estructura que permita almacenar números enteros hasta leer el número 99. Luego se pide hacer un módulo que recorra la estructura y calcule el valor mínimo. Se sabe que a lo sumo son 1000 números.



Estructura de datos-ARREGLO – Dim Física-Lógica

```
Program cuatro;
Const DIMF=1000; {dimensión física del vector}
        FIN = 99;
 Type  rango=0..DIMF;
         numeros = array [1..DIMF]  of integer;
  Var
    nume: numeros;
    dimL: rango; {dimensión lógica del vector}
  Begin
    cargarNumeros(nume,dimL);
    write('el menor numero leido es', minimo(nume,dimL));
  End.
```

Estructura de datos-ARREGLO – Dim Física-Lógica

```
Procedure cargarNumeros (var n:numeros; var DimL:rango);
 var   valor:integer;
 begin
   DimL:=0;
   readln(valor);
   while (DimL < DIMF) and (valor <> FIN) do
   begin
       dimL:= dimL + 1;
       n[dimL]:= valor;
       if (dimL<>DimF) then readln(valor);
     end;
 end;
```

Siempre se debe controlar no ingresar mas números que la dim. Fisica declarada

Estructura de datos-ARREGLO – Dim Física-Lógica

```
function minimo (n:numeros; DimL:rango):integer;
  var
    i: rango;  min:integer;
  Begin
    min:=9999;
    for i:= 1 to DimL do
        if (n[i]< min) then  min:= n[i];
    minimo:= min;
end;
```

Prestar atención a que se recorre hasta la Dimensión Lógica

Agregar un elemento nuevo Al final del vector

Arreglos: Agregar al final (append)

Aspectos a considerar:

- dimF : dimensión del vector (tamaño especificado en la declaración del vector)
- dimL : cantidad de elementos en el vector. Luego de la operación la cantidad de elementos es dimL+1
- espacio suficiente: debe verificarse que dimL < dimF.
  - El elemento a agregar ocupará la posición dimL+1.
  - Luego de la operación la cantidad de elementos es dimL+1.

Arreglos: Agregar al final

```
{Este programa agrega varios elementos invocando al procedure Agregar}

Program AgregarElementos;
const  dimF = 10; FIN=0;
Type  rango= 0..dimF;
        vector = array [1..dimF] of integer;
{aquí va el procedure Agregar}
Var  v : vector;
      elem: integer;
      exito: boolean; {controla si se pudo agregar o no}
      dimL, i: rango;
```

Arreglos: Agregar al final

```
Procedure AGREGAR (var v: vector; var dimL:rango; elemento: integer;
                   var exito : boolean);
Begin
 {verificar espacio suficiente}
  If (dimL < DimF) then
            begin
              dimL := dimL + 1; {Se debe incrementar la dim. lógica}
              v[dimL]:= elemento; {Se guarda el valor en el vector}
              exito:=true;
            end
            else exito := false; {Se usa para indicar que no se pudo hacer}
 End;
```

Arreglos: Agregar al final

```
Begin  {prog. ppal}
    readln (elem);
    exito:=true; dimL:=0;
    while (elem <> FIN) and (exito) do
     begin
         AGREGAR ( v, dimL, elem, exito);
         if (exito) then readln (elem);
      end;
     write ('El vector resultante es: ');
     Informar (v, dimL);
End.
```

INSERTAR
un elemento en el vector

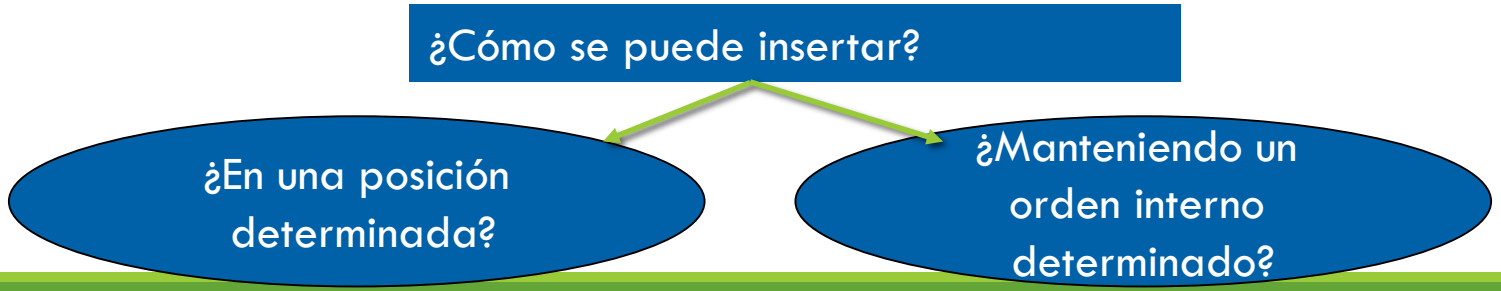
VECTOR: Insertar un elemento

Aspectos a considerar:

➔ dimF : dimensión física del vector (tamaño especificado en la declaración del vector).

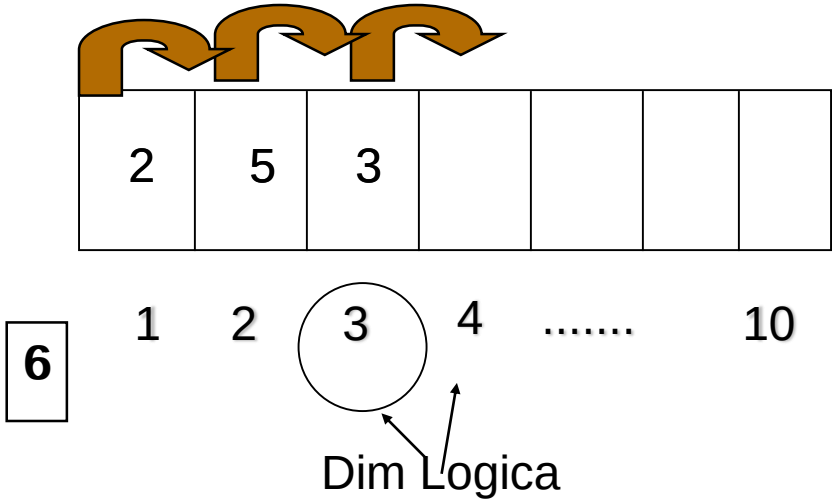
➔ dimL: cantidad de elementos en el vector. De ser posible la operación, la cantidad de elementos es dimL+1.

➔ VALIDAR SI HAY espacio suficiente: Debe verificarse que dimL < dimF



Arreglos: Insertar

Supongamos que deseamos insertar el VALOR 6 en la POSICIÓN 1 del vector



La posición donde debo insertar debe ser VÁLIDA

Arreglos: Insertar

Pasos a seguir:

1. Validar la posición a insertar
2. Verificar espacio en el arreglo
3. Hacer corrimiento de los valores del arreglo (a partir de la dimensión lógica)
4. Asignar el valor
5. Aumentar la dimensión lógica

Arreglos: Insertar

Declaraciones

```
Program insertarelementos;
 Const  dimF = 1000;
 Type  rango= 0..dimF;
vector = array [1..dimF] of integer;
{aquí van los procesos}
var      v: vector;
         elem : integer;
         pos , DimL:  rango;
         sepuede : boolean;
```

Arreglos: Insertar

```
Procedure INSERTAR (var num:vector; var dimL:rango; elem:integer; pos:rango; var
exito:boolean );
var i : rango;
Begin
   if  (dimL < DimF ) and (pos >=1) and (pos <=DimL) then {podría ponerse una constante}
                              begin
                                 exito := true;
                                 For i:= DimL downto pos do
                                    num[i+1]:= num[i];
                                 num [pos] := elem;
                                 DimL := DimL + 1; {Incremento dim logica}
                              end
    else exito := false;
 end;
```

Efectuamos el corrimiento

Guardamos el nuevo valor

Incrementamos DIML

Arreglos: Insertar

```
Program insertarelementos;
 Const  dimF = 1000; FIN=0;
 Type  rango= 0..dimF;
vector = array [1..dimF] of integer;
{procesos}
var     v: vector;
      elem : integer;
      pos , DimL:  rango;
      sepuede : boolean;
```

```
begin
    cargar-vector (v, DimL);
    sepuede := true;
    readln (elem);
    readln(pos);
    while (elem <> FIN) and (sepuede) do
    begin
        INSERTAR ( v, DimL, elem, pos, sepuede);
        if (sepuede)
        then begin
                readln (elem);
                readln(pos);
             end;
    end;
End.
```

BORRAR
Un Elemento Existente En El Vector

Arreglos: Borrar

Aspectos a considerar:

DimL : cantidad de elementos en el vector. Luego de la operación la cantidad de elementos disminuye en 1.

➔**Pos** : posición del elemento del vector que se quiere borrar. Debe verificarse que pos<= DimL

{Consideremos un programa que borra varios elementos en posiciones que se ingresan como dato, invocando al procedure Borrar}

Arreglos: Borrar

Declaraciones

```
program borrarelementos;
const  dimF = 1000;
type  rango= 0..dimF;
      vector = array [1..dimF] of integer;
var    v: vector;
       elem : integer;
        pos , DimL: rango;
      sepuede : boolean;
```

Arreglos: Borrar

```
Procedure BORRAR (var num: vector; var DimL: rango; var elemento: integer; pos: rango; var exito: boolean );
Var j: rango;
Begin
      if (pos <= DimL) and (pos >=1) then
                  begin  {verifica valor de pos, valida ese valor}
                      exito:=true;
                       elemento :=  num [pos] ; {Guarda el valor del elemento que se borra }
                       for j:= pos  to   DimL-1  do {Se hacen los corrimientos a izquierda }
                           num [ j ]  :=  num [ j + 1 ] ;
                       DimL := DimL - 1 ;
                   end
                  else exito := false;
  End;
```

Arreglos: Borrar

```
program borrarelementos;
const  dimF = 1000; FIN=0;
type  rango= 0..dimF;
      vector = array [1..dimF] of integer;
var    v: vector;
       elem : integer;
        pos , DimL: rango;
       exito: boolean;
```

```
begin
    cargarvector (v, DimL);
    read (pos);
    BORRAR ( v, DimL, elem, pos, exito);
    if exito
        then writeln ("El elemento borrado es: ", elem)
        else writeln ("No se pudo borrar el elemento");
End.
```