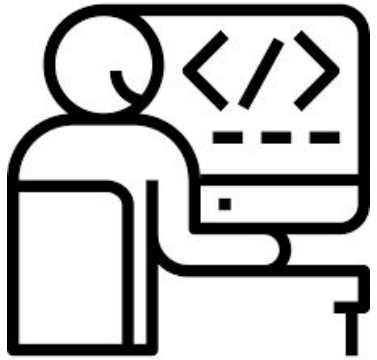


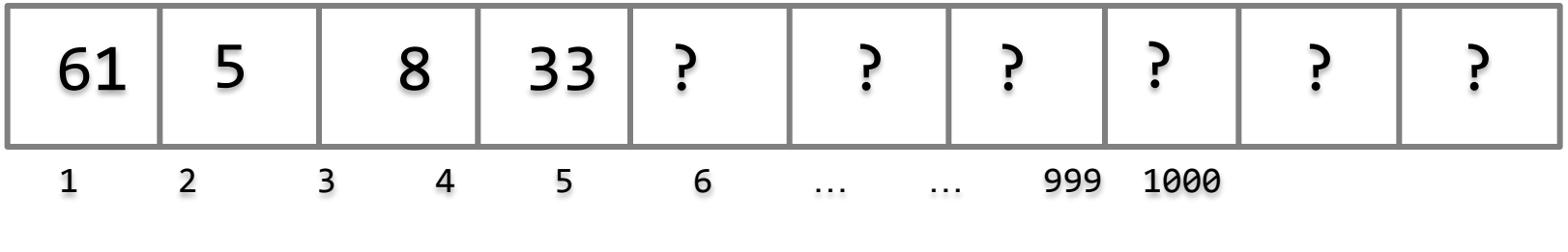
Conceptos de Algoritmos Datos y Programas

CADP – **Temas de la clase de hoy**

- Tipos de Datos Arreglo
- Operaciones con vectores

DIMENSIONES

Supongamos que se declara un vector de enteros cuya dimensión física es de 1000, y por algún motivo sólo se cargan las 4 primeras posiciones.



Si se quiere obtener la suma de los elementos, hasta donde debo considerar? Tengo que sumar los 1000 elementos? Qué valores tienen las posiciones que no fueron cargadas?

DIMENSION FISICA
DIMENSION LOGICA

DIMENSIONES

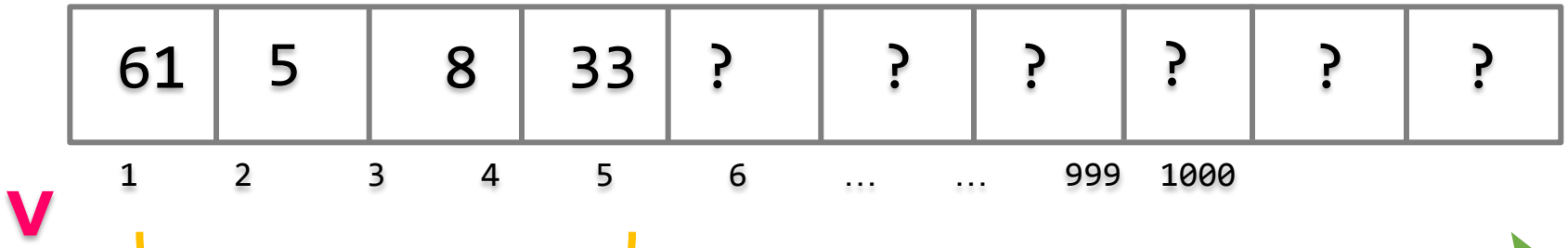
DIMENSION FISICA

Se especifica en el momento de la declaración y determina su ocupación máxima de memoria.
La cantidad de memoria total reservada no variará durante la ejecución del programa.

DIMENSION LOGICA

Se determina cuando se cargan contenidos a los elementos del arreglo.
Indica la cantidad de posiciones de memoria ocupadas con contenido real. Nunca puede superar la dimensión física.

DIMENSIONES



Cuándo se determina cada una?

Dónde se declaran?

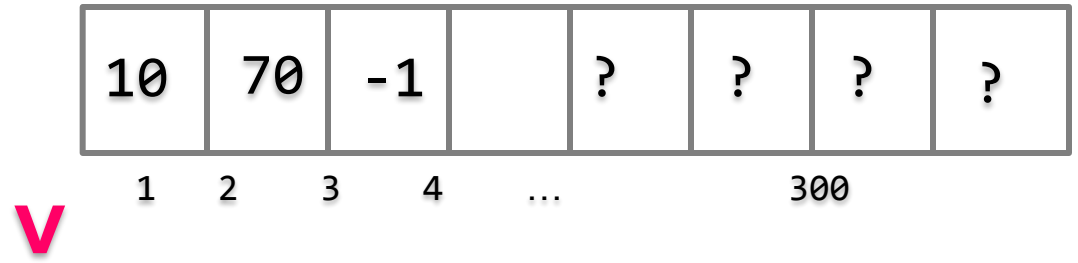
Veamos un ejemplo…

CADP – **VECTOR** DIMENSIONES

Realizar un programa que cargue un arreglo con números enteros hasta leer el número 50, a lo sumo se cargan 300 números.
Luego de terminar la carga informe cuál es el número mas grande de los leídos.

Hasta cuando se lee?

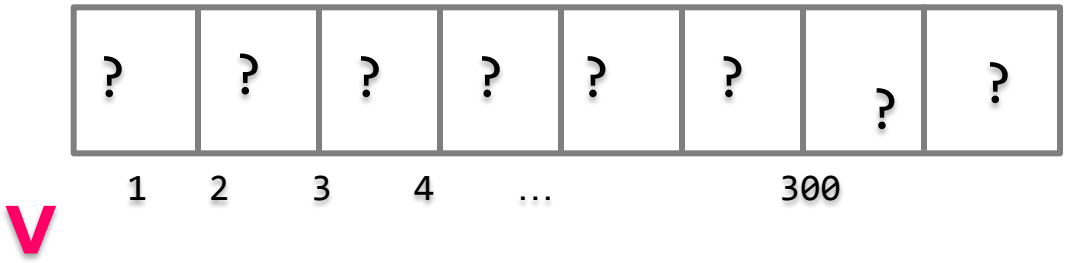
10
70
-1
50



DF = 300
DL= 3

Cuál es la dimensión física?

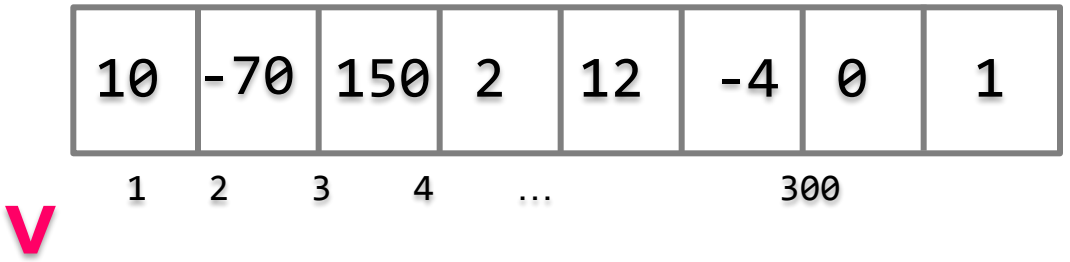
5
0



DF = 300
DL= 0

Cuál es la dimensión lógica?

10
70
15
0



DF = 300
DL= 300

CADP – **VECTOR** DIMENSIONES

Realizar un programa que cargue un arreglo con números enteros hasta leer el número 50, a lo sumo se cargan 300 números.
Luego de terminar la carga informe cuál es el número máximo entre los números leídos.

```
begin
   cargarNumeros ();
   calcularMaximo ();
end.
```

Qué tipos de datos debo definir?

cargarNumeros que tipo de módulo es? Parámetros?

calcularMaximo que tipo de módulo es? Parámetros?

CADP – **VECTOR** DIMENSIONES

Realizar un programa que cargue un arreglo con números enteros hasta leer el número 50, a lo sumo se cargan 300 números.
Luego de terminar la carga informe cuál es el número máximo entre los números leídos.

```
Program uno;
  const
    dimFisica = 300;

  type
    numeros= array [1..dimFisica]  of integer;

var
    VN: numeros;
    dimL: integer;

begin
  cargarNumeros (VN, dimL);
  write (“El número mayor es:”, calcularMaximo(VN,dimL));
end.
```

En el proceso, se leen los valores, se cargan y me devuelve el arreglo cargado y cuantos elementos cargo

La función recibe el vector y cuantos elementos tiene cargados y devuelve el máximo

CADP – **VECTOR** DIMENSIONES

Realizar un programa que cargue un arreglo con números enteros hasta leer el número 50, a lo sumo se cargan 300 números.
Luego de terminar la carga informe cuál es el número máximo entre los números leídos.

```
Procedure cargarNumeros (var a: números; var  dL:integer);

Var
 num:integer;

Begin
 dL:=0;
 read (num);
 while (num <> 50) do
  begin
   a[dL]:= num;
   read(num);
  end;
End;
```

Es correcto?

Cómo dL, está inicializado en 0, la primera vez se accede a la posición a[0] y no es válida

CADP – **VECTOR** DIMENSIONES

Realizar un programa que cargue un arreglo con números enteros hasta leer el número 50, a lo sumo se cargan 300 números.
Luego de terminar la carga informe cuál es el número máximo entre los números leídos.

```
Procedure cargarNumeros (var a: números; var  dL:integer);

Var
 num:integer;

Begin
 dL:=1;
 read (num);
 while (num <> 50) do
  begin
   a[dL]:= num;
   read(num);
  end;
End;
```

Es correcto?

Cómo dL, nunca se incrementa, entonces carga siempre en la misma posición a[1]

CADP – **VECTOR** DIMENSIONES

Realizar un programa que cargue un arreglo con números enteros hasta leer el número 50, a lo sumo se cargan 300 números.
Luego de terminar la carga informe cuál es el número máximo entre los números leídos.

```
Procedure cargarNumeros (var a: números; var  dL:integer);

Var
 num:integer;

Begin
 dL:=1;
 read (num);
 while (num <> 50) do
  begin
   a[dL+1]:= num;
   read(num);
  end;
End;
```

Es correcto?

Cómo dL, nunca se incrementa, entonces carga siempre en la misma posición a[1]

CADP – **VECTOR** DIMENSIONES

Realizar un programa que cargue un arreglo con números enteros hasta leer el número 50, a lo sumo se cargan 300 números.
Luego de terminar la carga informe cuál es el número máximo entre los números leídos.

```
Procedure cargarNumeros (var a: números; var  dL:integer);

Var
 num:integer;

Begin
 dL:=1;
 read (num);
 while (num <> 50) do
  begin
   a[dL]:= num;
   dL:= dL+1;
   read(num);
  end;
End;
```

Es correcto?

Si el primer número leído es 50, entonces dL devuelve 1

CADP – **VECTOR** DIMENSIONES

Realizar un programa que cargue un arreglo con números enteros hasta leer el número 50, a lo sumo se cargan 300 números.
Luego de terminar la carga informe cuál es el número máximo entre los números leídos.

```
Procedure cargarNumeros (var a: números; var  dL:integer);

Var
 num:integer;

Begin
 dL:=0;
 read (num);
 while (num <> 50) do
  begin
   dL:= dL+1;
   a[dL]:= num;
   read(num);
  end;
End;
```

Es correcto?

Qué pasa si leo mas de 300 números (el valor 50 no apareció y ya leí 300 valores)

CADP – **VECTOR** DIMENSIONES

Realizar un programa que cargue un arreglo con números enteros hasta leer el número 50, a lo sumo se cargan 300 números.
Luego de terminar la carga informe cuál es el número máximo entre los números leídos.

```
Procedure cargarNumeros (var a: números; var  dL:integer);

Var
 num:integer;

Begin
 dL:=0;
 read (num);
 while ((dL < dimFisica) and (num <> 50)) do
  begin
   dL:= dL+1;
   a[dL]:= num;
   read(num);
  end;
End;
```

Es correcto?

Si! En dL quedó guardada la cantidad de elementos que realmente se cargaron

CADP – **VECTOR** DIMENSIONES

Realizar un programa que cargue un arreglo con números enteros hasta leer el número 50, a lo sumo se cargan 300 números.
Luego de terminar la carga informe cuál es el número máximo entre los números leídos.

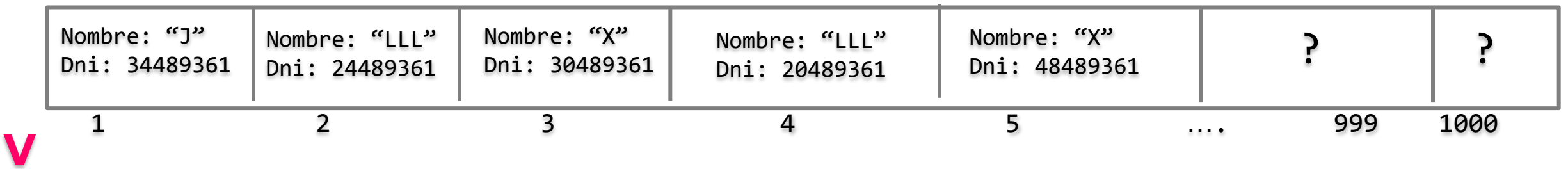
```
function calcularMaximo (a: números; dL:integer):integer;

Var
 max,i:integer;

Begin
 max:=-9999;
 for i:= 1 to dL do
  begin
   if (a[i]>= max) then max:= a[i];
  end;

 calcularMaximo:= max;
End;
```

Recibe la cantidad de elementos reales que fueron cargados en el vector

Recorre el vector hasta la cantidad de elementos cargados

CADP – **VECTOR** VECTOR DE REGISTROS

Supongamos que se dispone de un vector de a lo sumo 1000 personas. De cada persona se conoce nombre y dni. Como se implementaría el módulo que recibe el vector y cuenta la cantidad de personas con nombre “ LLL”.



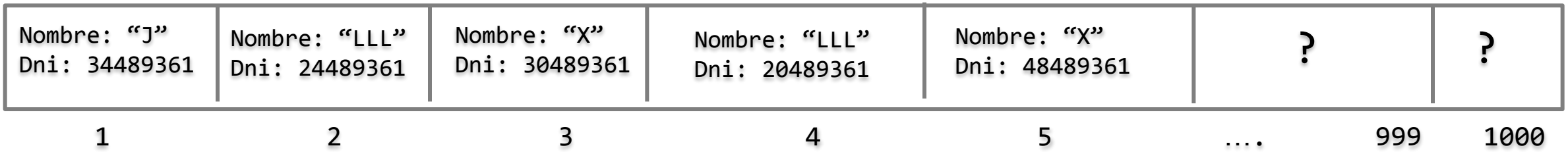
Debo recorrer el vector hasta su dimensión lógica e ir contando cuantos tienen el nombre LLL

CADP – **VECTOR** VECTOR DE REGISTROS

Supongamos que se dispone de un vector de a lo sumo 1000 personas. De cada persona se conoce nombre y dni. Como se implementaría el módulo que recibe el vector y cuenta la cantidad de personas con nombre “ LLL”.

```
Program uno;
  const
    dimFisica = 1000;

  type
    persona = record
      nombre:string;
      dni:integer;
    end;
    personas= array [1..dimFisica]  of persona;
var
    VP: personas;
    dimL: integer;
begin
  cargarPersonas (VP, dimL);  //no se implementa ya que se dispone
  write (“La cantidad es:”, cantidad(VP,dimL));
end.
```

CADP – **VECTOR** VECTOR DE REGISTROS



OPCION 1

```
Function cantidad (v: personas;dL:integer):integer;
Var
 i,cant:integer;
begin
  cant:= 0;
  for i:= 1 to dL do
    if (v[i].nombre = "LLL" )
     then cant:= cant + 1;

   cantidad:= cant;
end.
```

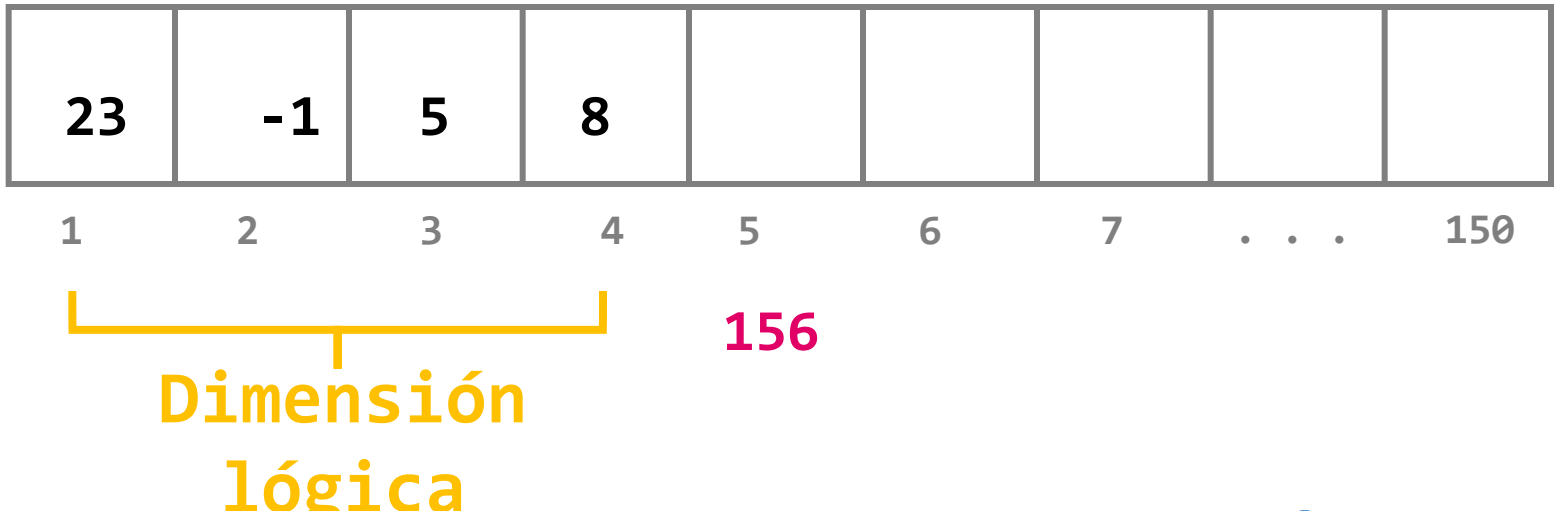
OPCION 2

```
Function cantidad (v: personas;dL:integer):integer;
Var
 i,cant:integer;
begin
  cant:= 0;
  for i:= 1 to dL do
    if (v[i.nombre] = "LLL" )
       then  cant:= cant + 1;

   cantidad:= cant;
end.
```

AGREGAR

Significa poner al final de los elementos que tiene el vector un nuevo elemento. Puede pasar que esta operación no se pueda realizar si el vector está lleno.

Qué consideraciones debo tener?



Veamos los pasos …

AGREGAR

1- Verificar si hay espacio (cantidad de elementos actuales es menor a la cantidad de elementos posibles)

2- Agregar al final de los elementos ya existentes el elemento nuevo.

3- Incrementar la cantidad de elementos actuales.

CADP – **TIPOS DE DATOS VECTOR** AGREGAR

Dado un vector de números enteros (150 elementos como máximo) realice un programa que lea un número e invoque a un módulo que agregue el elemento en el vector.

```
Program uno;
  const
    fisica = 150;
  type
    numeros= array [1..fisica]  of integer;

var
    VN: numeros;
    dimL, valor:integer;
    ok:boolean;
begin
  llenarNumeros (VN, dimL); No se implementa
  read (valor);
  agregar (VN, dimL, ok, valor);
end.
```

CADP – **TIPOS DE DATOS VECTOR** AGREGAR

Dado un vector de números enteros (150 elementos como máximo) realice un programa que lea un número e invoque a un módulo que agregue el elemento en el vector.

```
Procedure agregar (var a :números; var dL:integer; var pude:boolean;
                   valor:integer);

Begin
  pude:= false;
  if ((dl + 1) <= física) then begin
   pude:= true;
   dL:= dL + 1;
   a[dL]:= valor;
  end;
end.
```

Verifico si hay espacio

Registro que se puede hacer la operación

Aumento la cantidad de elementos

Agrego el elemento

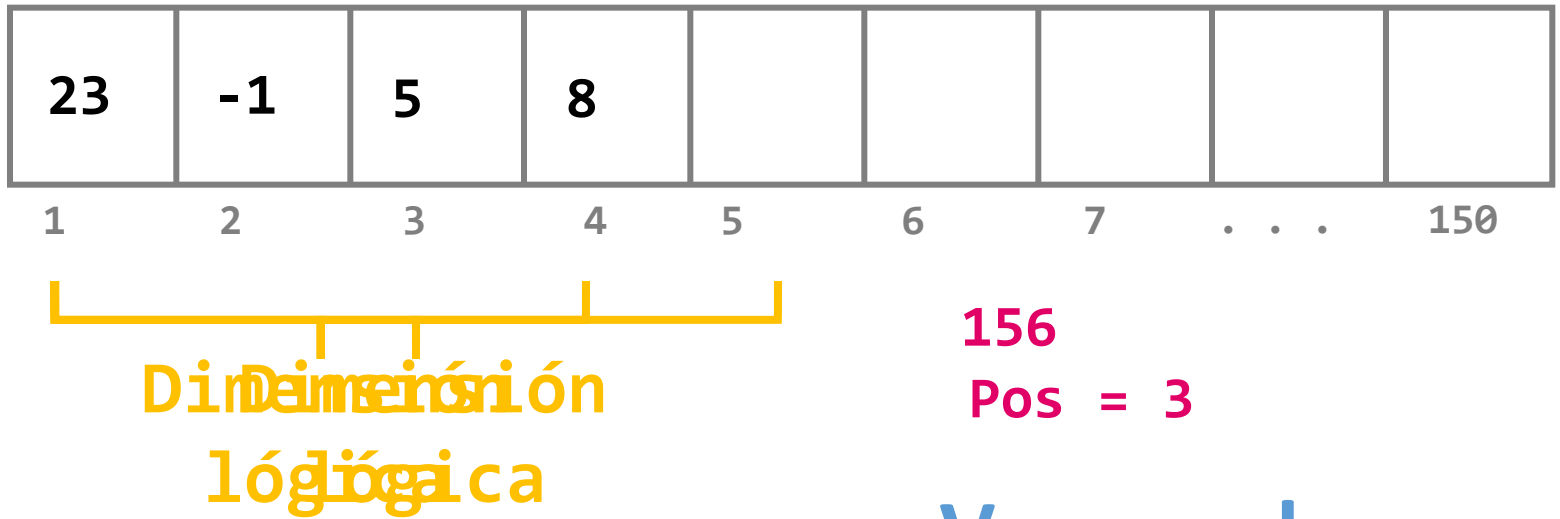
CADP – **TIPOS DE DATOS VECTOR**

INSERTAR

Insertar

Significa agregar en el vector un elemento en una posición determinada. Puede pasar que esta operación no se pueda realizar si el vector está lleno o si la posición no es válida

Qué consideraciones debo tener?



Veamos los pasos …

INSERTAR

1- Verificar si hay espacio (cantidad de elementos actuales es menor a la cantidad de elementos posibles)

2- Verificar que la posición sea válida (esté entre los valores de dimensión definida del vector y la dimensión lógica).

3- Hacer lugar para poder insertar el elemento.

4- Incrementar la cantidad de elementos actuales.

CADP – **TIPOS DE DATOS VECTOR** INSERTAR

Dado un vector de números enteros (150 elementos como máximo) realice un programa que lea un número y una posición e invoque a un módulo que inserte el elemento en el vector en la posición leída.

```
Program uno;
  const
    fisica = 150;
  type
    numeros= array [1..fisica]  of integer;

var
    VN: numeros;
    dimL, valor, pos:integer;
    ok:boolean;
begin
  llenarNumeros (VN, dimL); No se implementa
  read (valor); read (pos);
  insertar (VN, dimL, ok, valor, pos);
end.
```

CADP – **TIPOS DE DATOS VECTOR** INSERTAR

Dado un vector de números enteros (150 elementos como máximo) realice un programa que lea un número y una posición e invoque a un módulo que inserte el elemento en el vector en la posición leída.

```
Procedure agregar (var a :números; var dL:integer; var pude:boolean;
                   valor:integer; pos:integer);
Var
 i:integer;
Begin
  pude:= false;
  if ((dL + 1) <= física) and (pos>= 1) and (pos <= dL) )then begin
   for i:= dL down to pos do
        a[i+1]:= a[i];
   pude:= true;
   a[pos]:= valor;
   dL:= dL + 1;
  end;
end;
```

Verifico si hay espacio y la posición es válida

Hago el corrimiento
Para hacer lugar

Inserto el elemento en la
posición pos

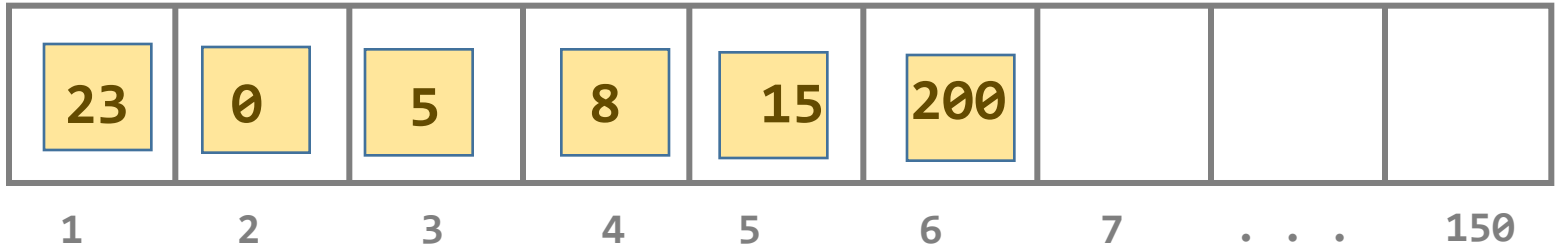
CADP – **TIPOS DE DATOS VECTOR**

Ejercicio

Dado un arreglo de números enteros (150 elementos como máximo) realice un programa que genere otro arreglo sólo con los números pares.

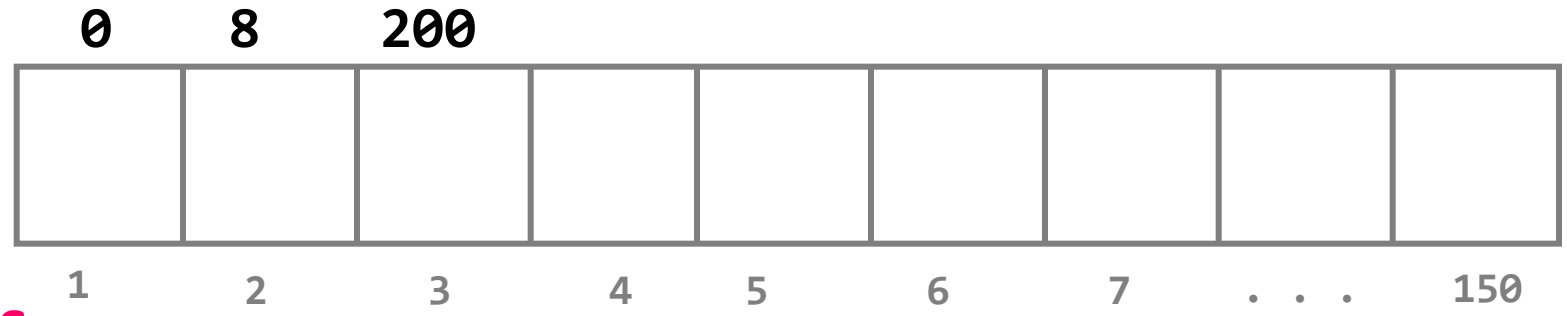
Qué consideraciones debo tener?

V



dimL = 6

Vres



dimL = 1

dimL = 2

dimL = 3

Veamos los pasos ...

CADP – **TIPOS DE DATOS VECTOR** Ejercicio

Dado un arreglo de números enteros (150 elementos como máximo) realice un programa que genere otro arreglo sólo con los números pares.

Cargar vector1 *(no hay que implementarlo)*.

Para cada elemento en la posición i del vector 1

Si (el elemento es par) entonces

Agregar el elemento de la posición i del vector1 en el vector2 en la posición correspondiente.

Aumento la dimensión lógica del vector2

CADP – **TIPOS DE DATOS VECTOR** Ejercicio

Dado un arreglo de números enteros (150 elementos como máximo) realice un programa que genere otro arreglo sólo con los números pares.

```
Program uno;
  const
    tam = 150;
  type
    numeros= array [1..tam]  of integer;


  var
    a1,a2: numeros;
    dim1L, dim2L: integer;

  begin
    llenarNumeros (a1,dim1L);  //no se implementa
    procesar(a1,dim1L, a2,dim2L);
  end.
```

CADP – **TIPOS DE DATOS VECTOR** Ejercicio

Dado un arreglo de números enteros (150 elementos como máximo) realice un programa que genere otro arreglo sólo con los números pares.

```
Procedure procesar (a1:números; dim1:integer;
                    var a2:números; var dim2:integer);
var
 i: integer;
begin
 dim2:=0;
 for i:= 1 to dim1 do
   begin
     if (esPar(a1[i])) then
      begin
        dim2:= dim2 + 1;
        a2[dim2]:= a1[i];
      end;
 end;
```

Para cada elemento del vector 1

Si el elemento es par

Agrego el elemento en la posición

CADP – **TIPOS DE DATOS VECTOR** Ejercicio

Dado un arreglo de números enteros (150 elementos como máximo) realice un programa que genere otro arreglo sólo con los números pares.

```
Function esPar (valor:integer):boolean;

begin
 esPar:= (valor MOD 2 = 0);
end;
```

Esta función recibe el valor que esta almacenado en el vector en la posición i.

Sería lo mismo que en vez de recibir el valor la función reciba el vector y la posición y chequee si el valor que hay en el vector en la posición recibida es par?

CADP – **TIPOS DE DATOS VECTOR** Ejercicio

Dado un arreglo de números enteros (150 elementos como máximo) realice un programa que genere otro arreglo sólo con los números pares.

```
Procedure procesar (a1:números; dim1:integer;
                    var a2:números; var dim2:integer);
var
 i: integer;
begin
 dim2:=0;
 for i:= 1 to dim1 do
   begin
     if (esPar(a1[i])) then
      begin
        dim2:= dim2 + 1;
        a2[dim2]:= a1[i];
      end;
 end;
```

Cómo se puede reescribir el proceso con módulos que ya hemos realizado?

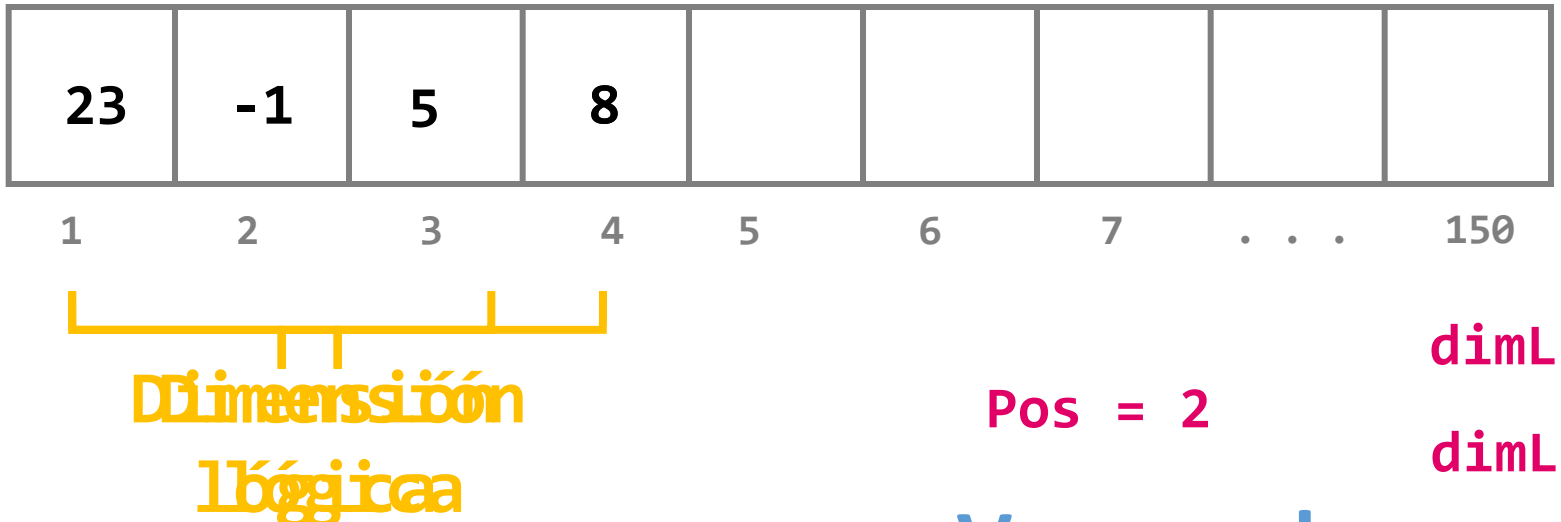
CADP – **TIPOS DE DATOS VECTOR**

ELIMINAR

Eliminar

Significa borrar (lógicamente) en el vector un elemento en una posición determinada, o un valor determinado. Puede pasar que esta operación no se pueda realizar si la posición no es válida, o en el caso de eliminar un elemento si el mismo no está

Qué consideraciones debo tener?



Veamos los pasos ...

ELIMINAR DE UNA POSICION

1- Verificar que la posición sea válida (esté entre los valores de dimensión definida del vector y la dimensión lógica).

2- Hacer el corrimiento a partir de la posición y hasta el final.

3- Decrementar la cantidad de elementos actuales.

CADP – **TIPOS DE DATOS VECTOR** ELIMINAR

Dado un vector de números enteros (150 elementos como máximo) realice un programa que lea una posición e invoque a un módulo que elimine el elemento en el vector en la posición leída.

```
Program uno;
  const
    fisica = 150;
  type
    numeros= array [1..fisica]  of integer;

var
    VN: numeros;
    dimL, valor, pos:integer;
    ok:boolean;
begin
  llenarNumeros (VN, dimL); No se implementa
  read (pos);
  eliminar (VN, dimL, ok, pos);
end.
```

CADP – **TIPOS DE DATOS VECTOR** ELIMINAR

Dado un vector de números enteros (150 elementos como máximo) realice un programa que lea una posición e invoque a un módulo que elimine el elemento en el vector en la posición leída.

```
Procedure borrar (var a :números; var dim:integer;
                  var pude:boolean; pos:integer;);

Begin
  pude:= false;
  if ((pos>=1) and (pos<=dim))then begin

    for i:= pos to (dim-1) do
      a[i]:= a[i+1];
    pude:= true;
    dim:= dim - 1;
  end;
end;
```

Verifico que la posición sea válida

Realizo el corrimiento

CADP – **TIPOS DE DATOS VECTOR** BUSQUEDA

Buscar

Significa recorrer el vector buscando un valor que puede o no estar en el vector. Se debe tener en cuenta que no es lo mismo buscar en un vector ordenado que uno que no lo este

Vector Desordenado

Se debe recorrer todo el vector (en el peor de los casos), y detener la búsqueda en el momento que se encuentra el dato buscado o que se terminó el vector.

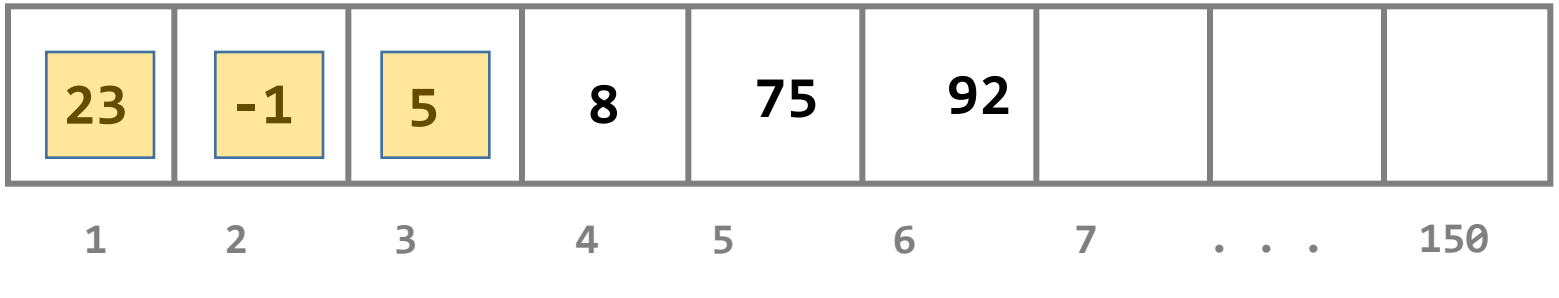
Vector Ordenado

Se debe aprovechar el orden, existen al menos dos formas: búsqueda mejorada y búsqueda dicotómica.

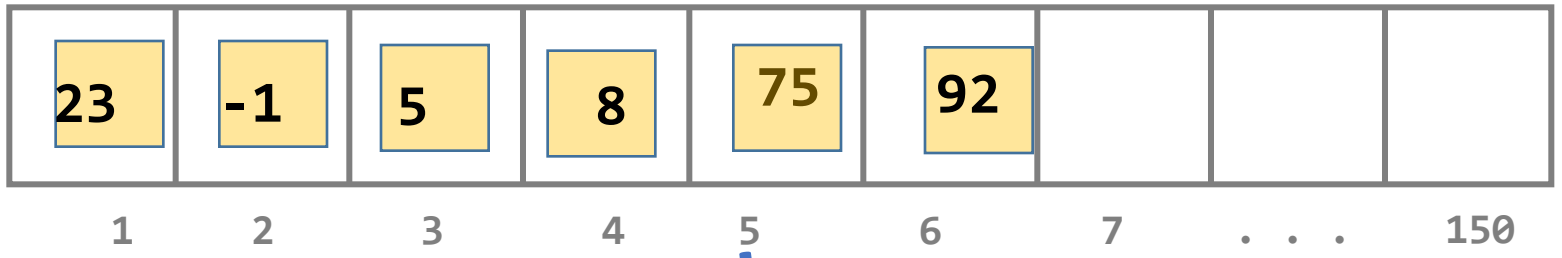
CADP – **TIPOS DE DATOS VECTOR** BUSQUEDA

Vector Desordenado

`valor = 5` `dimL = 6`



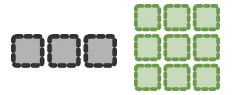
`valor = 15` `dimL = 6`



Qué estructura de control utilizo?

Qué tipo de módulo?

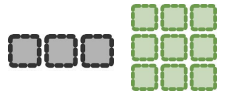
CADP – TIPOS DE DATOS VECTOR BUSQUEDA



Vector Desordenado

```
Program uno;
  const
    tam = 150;
  type
    numeros= array [1..tam]  of integer;
  var
    VN: numeros;  dimL, num:integer;

  begin
    read (num);
    cargarNumeros (VN,dimL);  //no se implementa
    if (buscar(VN,dimL,num)) then
      write (num, “Esta en el vector”)
    else ( num,”No se encuentra en el vector”);
end.
```

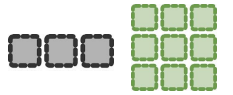
CADP – **TIPOS DE DATOS VECTOR** BUSQUEDA



```
function buscar (a:números; dim:integer, valor:integer): boolean;
Var
  pos:integer;
  esta:boolean;

Begin
  esta:= false;
  pos:=1;
  while ( (pos <= dim) and (not esta) ) do
     begin
       if (a[pos]= valor) then esta:=true
       else
          pos:= pos + 1;
     end;
   buscar:= esta;
end.
```

Por qué pos se incrementa en el else?

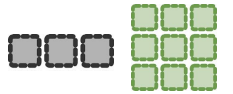
CADP – **TIPOS DE DATOS VECTOR** BUSQUEDA



```
function buscar (a:números; dim:integer, valor:integer): boolean;
Var
  pos:integer;

Begin
  pos:=1;
  while ( (pos <= dim) and (a[pos] <> valor) ) do
     begin
       pos:= pos + 1;
     end;
   buscar:= (a[pos]=valor);
end.
```

Es correcto?

CADP – **TIPOS DE DATOS VECTOR** BUSQUEDA



```
function buscar (a:números; dim:integer, valor:integer): boolean;
Var
  pos:integer;

Begin
  pos:=1;
  while ((a[pos] <> valor) and (pos <= dim) ) do
     begin
       pos:= pos + 1;
     end;
   buscar:= (a[pos]=valor);
end.
```

Es correcto?

CADP – **TIPOS DE DATOS VECTOR** BUSQUEDA

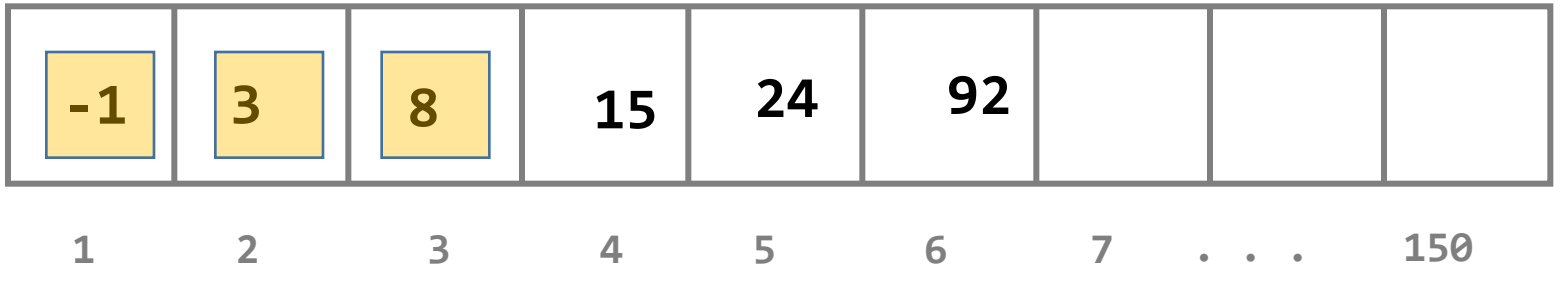
```
function buscar (a:números; dim:integer, valor:integer): boolean;
Var
  pos:integer;

Begin
  pos:=1;
  while ((pos <= dim) and (a[pos] <> valor)  ) do
     begin
       pos:= pos + 1;
     end;
   buscar:= (pos <= dim);
end.
```

Es correcto?

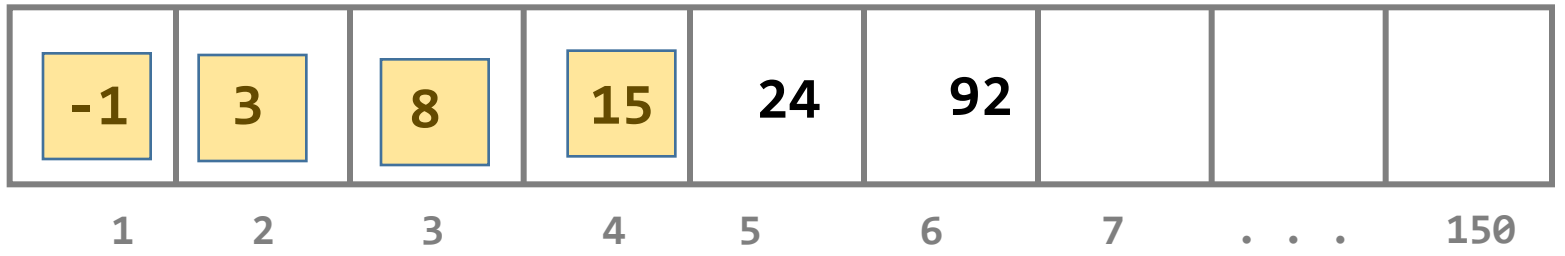
CADP – **TIPOS DE DATOS VECTOR** BUSQUEDA

Vector Ordenado

`valor = 8` `dimL = 6`



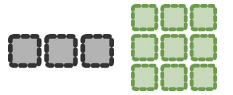
`valor = 9` `dimL = 6`



Qué estructura de control utilizo?

Qué tipo de módulo?

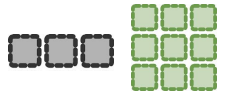
CADP – TIPOS DE DATOS VECTOR BUSQUEDA



Vector Ordenado

```
Program uno;
  const
    tam = 150;
  type
    numeros= array [1..tam]  of integer;
  var
    VN: numeros;  dimL, num:integer;

  begin
    read (num);
    cargarNumeros (VN,dimL);  //no se implementa
    if (buscar(VN,dimL,num)) then
      write (num, “Esta en el vector”)
    else ( num,”No se encuentra en el vector”);
end.
```

CADP – TIPOS DE DATOS VECTOR BUSQUEDA



Vector Ordenado

```
function buscar (a:números; dim:integer, num:integer): boolean;

Var
  pos:integer;

Begin
  pos:=1;
  while ( (pos <= dim) and (a[pos]<num)) do
     begin
       pos:= pos + 1;
     end;
  if ( (pos <= dim) and (a[pos]= num)) then buscar:=true
  else buscar:= false;
end.
```

Se puede invertir el orden en la pregunta?

Debo evaluar las dos condiciones antes de devolver el valor?

Se puede mejorar?

CADP – **TIPOS DE DATOS VECTOR** BUSQUEDA

Vector Ordenado – Búsqueda Binaria - Dicotómica

dim lógica=8 **dim física = 150** **BUSCADO= 155**

10	70	85	87	90	150	155	170		

Pri = 1 Ult = 8
Medio = 4 (87)

1- Se calcula el elemento que esta en la posición del medio

2- Si es el elemento que busco, entonces la búsqueda termino

Si NO es el elemento que busco, entonces
Comparo contra el valor del medio
Elijo del vector la mitad que me convenga

Como 87 < 155, entonces debo tomar la segunda mitad

dim lógica=8 **dim física = 150** **BUSCADO= 155**

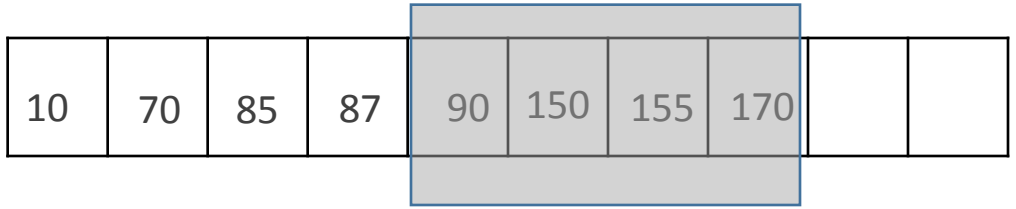
10	70	85	87	90	150	155	170		

Pri = 5
Ult = 8
Medio = 6 (150)

CADP – **TIPOS DE DATOS VECTOR** BUSQUEDA

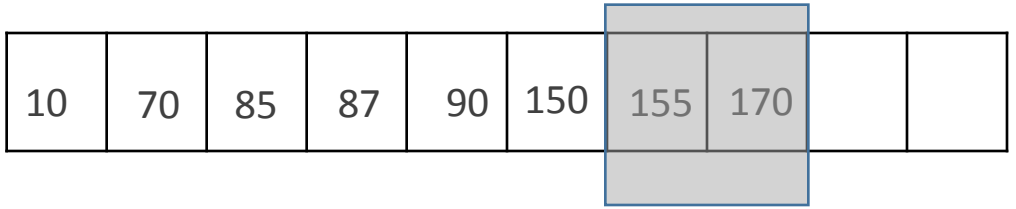
Vector Ordenado – Búsqueda Binaria - Dicotómica

dim lógica=8 dim física = 150 BUSCADO= 155



Pri = 5 Ult = 8
Medio = 6 (150)

Como 150 < 155, entonces debo tomar la segunda mitad

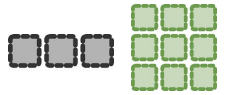


Pri = 7 Ult = 8
Medio = 7 (155)

Cuándo se termina si el elemento no existe?

Cómo se escribe?

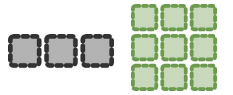
CADP – **TIPOS DE DATOS VECTOR** BUSQUEDA



Vector Ordenado – Búsqueda Binaria - Dicotómica

```
Procedure  BusquedaBinaria ( Var vec: números; dimL: integer;
                             bus: integer; var ok : boolean);
 Var
    pri, ult, medio : integer;

 Begin
     ok:= false;
     pri:= 1 ;  ult:= dimL;  medio := (pri + ult ) div 2 ;

     While  ( pri < = ult ) and ( bus <> vec[medio]) do
       begin
         if ( bus < vec[medio] ) then
           ult:= medio -1 ;
         else pri:= medio+1 ;
         medio := ( pri + ult ) div 2 ;
       end;
     if (pri <=ult) and (bus = vec[medio]) then ok:=true;
 end;
```

CADP – **TIPOS DE DATOS VECTOR** BUSQUEDA



COMPARACION

Búsqueda MEJORADA

- Se aplica cuando los elementos en la estructura tienen orden
- El número de comparaciones es en promedio (dimL+1)/2)

Búsqueda DICOTOMICA

- Se aplica cuando los elementos en la estructura tienen un orden.
- El número de comparaciones es en promedio $(1+\log_2(dimL+1)/2)$