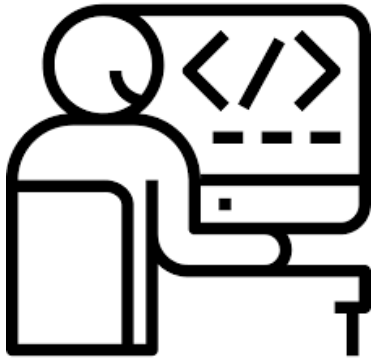


Conceptos de Algoritmos Datos y Programas

CADP – **TEMAS**

- Estructura de datos ARREGLO
- Agregar elementos
- Insertar elementos
- Eliminar elementos

VECTORES - OPERACIONES

Carga de valores

Lectura / Escritura

Recorridos

Dimensión física y lógica

Agregar elementos

Insertar elementos

Borrar elementos

Búsqueda de un elemento

VECTORES - AGREGAR

Significa agregar en el vector un elemento detrás del último elemento cargado en el vector. Puede pasar que esta operación no se pueda realizar si el vector está lleno

Dl = ~~4~~ 5

45

a

34	10	-1	5						
1	2	3	4	5	6	7	8	9	10

Qué pasos considero?

VECTORES - AGREGAR

Significa agregar en el vector un elemento detrás del último elemento cargado en el vector. Puede pasar que esta operación no se pueda realizar si el vector está lleno

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.

Cómo se implementa?

CADP – TIPOS DE DATOS

VECTORES - AGREGAR

Dado un vector de números enteros (10 elementos como máximo) realice un programa que lea un nuevo número e invoque a un módulo que agregue el elemento en el vector.

```
Program uno;
  const
    fisica = 10;
  type
    numeros= array [1..fisica]  of integer;

var
    VN: numeros;
    dimL, valor:integer;
    ok:boolean;

Begin
  cargar (VN,dimL);
  read(valor);
  agregar(VN,dimL,ok,valor);
End.
```

VN

?	?	?	?	?	?	?	?	?	?

dimL = 4

VN

4	-1	10	3	?	?	?	?	?	?

valor = 7 dimL = 5 ok = true

VN

4	-1	10	3	7	?	?	?	?	?

VECTORES - AGREGAR

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

 Begin
   pude:= false;
                    Verifico si hay espacio
   if ((dl + 1) <= física) then
    begin
      pude:= true;          Registro que se pudo realizar
      dL:= dL + 1;          Incremento la dimensión lógica
      a[dL]:= num;          Agrego elelemento
    end;
 end.
```

VECTORES -INSERTAR

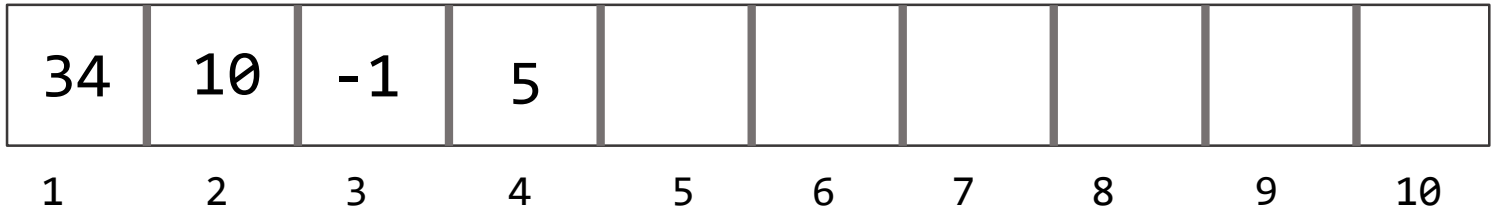
Significa agregar en el vector un elemento en una posición determinada. Puede pasar que esta operación no se pueda realizar si el vector está lleno o si la posición no es válida

Dl = 4

45

pos = 2

a



Qué pasos considero?

VECTORES -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

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.

Cómo se implementa?

CADP – TIPOS DE DATOS

VECTORES - INSERTAR

Dado un vector de números enteros (10 elementos como máximo) realice un programa que lea un nuevo número y una posición e invoque a un módulo que inserte el elemento en el vector en la posición leída.

```
Program uno;
  const
    fisica = 10;
  type
    numeros= array [1..fisica]  of integer;

var
 VN: numeros;
 dimL, valor,pos:integer;
 ok:boolean;
Begin
 cargar (VN,dimL);
 read(valor); read(pos);
 insertar(VN,dimL,valor,ok,pos);
End.
```

VN

?	?	?	?	?	?	?	?	?	?

dimL = 4

VN

4	-1	10	3	?	?	?	?	?	?

valor = 7 pos= 2 dimL = 5 ok = true

VN

4	7	-1	10	3	?	?	?	?	?

CADP – TIPOS DE DATOS

VECTORES - INSERTAR

```
Procedure insertar (var a :números; var dL:integer; var pude:boolean;
                                num:integer; pos: integer);
Var
 i:integer;

Begin
  pude:= false;

  if ((dL + 1) <= física) and (pos>= 1) and (pos <= dL) )then begin

   for i:= dL downto pos do
        a[i+1]:= a[i];

   pude:= true;
   a[pos]:= num;
   dL:= dL + 1;
  end;
end;
```

Verifico si hay espacio y si la posición es válida

Corro los elementos empezando desde atrás hasta la posición a insertar para hacer el hueco donde se va a insertar el elemento

Registro que se pudo realizar
Inserto el elemento
Incremento la dimensión lógica

VECTORES -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

pos = 2

DL = 4

a

34	10	-1	5						
1	2	3	4	5	6	7	8	9	10

Qué pasos considero?

VECTORES -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

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

Cómo se implementa?

CADP – TIPOS DE DATOS

VECTORES - ELIMINAR

Dado un vector de números enteros (10 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 = 10;
  type
    numeros= array [1..fisica]  of integer;

var
 VN: numeros;
 dimL,pos:integer;
 ok:boolean;
Begin
 cargar (VN,dimL);
  read(pos);
  eliminar(VN,dimL,ok,pos);
End.
```

VN

?	?	?	?	?	?	?	?	?	?

dimL = 4

VN

4	-1	10	3	?	?	?	?	?	?

pos= 2 dimL = 3 ok = true

VN

4	10	3	3	?	?	?	?	?	?

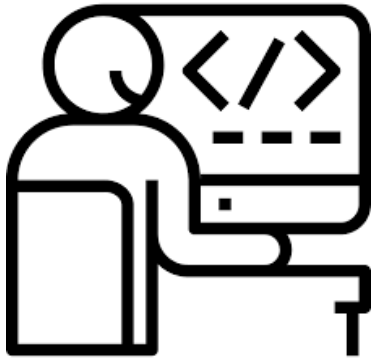
VECTORES - ELIMINAR

```
Procedure eliminar (var a :números; var dL:integer; var pude:boolean;pos: integer);
Var
 i:integer;

Begin
  pude:= false;

  if ((pos>= 1) and (pos <= dL) )then begin

   for i:= pos to (dL-1) do
      a[i]:= a[i+1];

   pude:= true;
   dL:= dL - 1;
  end;
end;
```

Verifico si la posición es válida

Corro los elementos empezando desde la posición hasta la dimensión lógica-1 para "tapar" el elemento a eliminar

Registro que se pudo realizar
Decremento la dimensión lógica



Conceptos de Algoritmos Datos y Programas

CADP – **TEMAS**

- Estructura de datos ARREGLO
- Búsqueda en un vector desordenado
- Búsqueda en un vector ordenado

VECTORES - OPERACIONES

Carga de valores

Lectura / Escritura

Recorridos

Dimensión física y lógica

Agregar elementos

Insertar elementos

Borrar elementos

Búsqueda de un elemento

VECTORES - BUSQUEDAS

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 en 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 en el que se terminó el vector.

Vector Ordenado

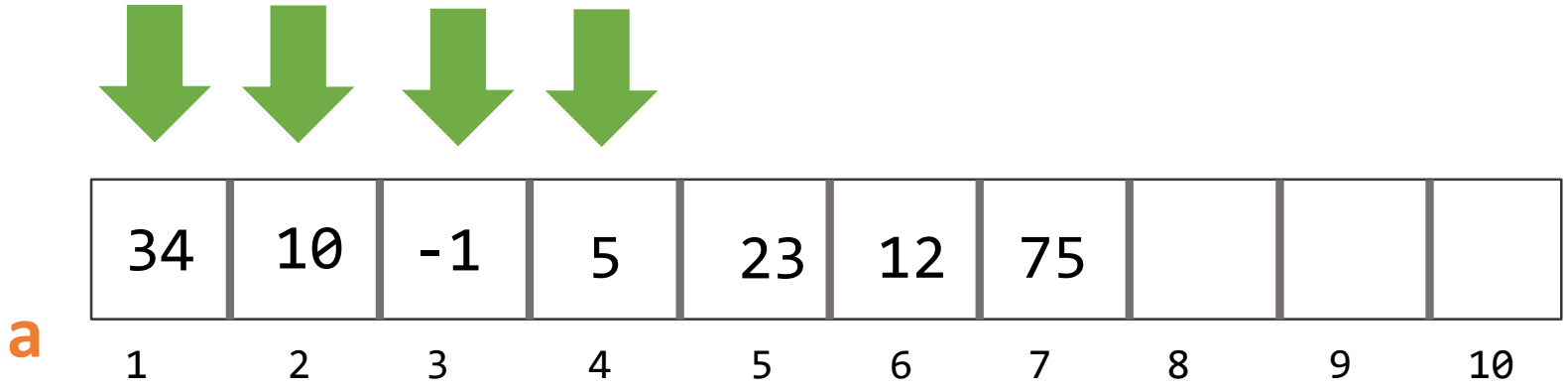
- Se debe recorrer el vector teniendo en cuenta el orden:
  - BUSQUEDA MEJORADA
  - BUSQUEDA BINARIA

VECTORES - BUSQUEDAS

Vector Desordenado

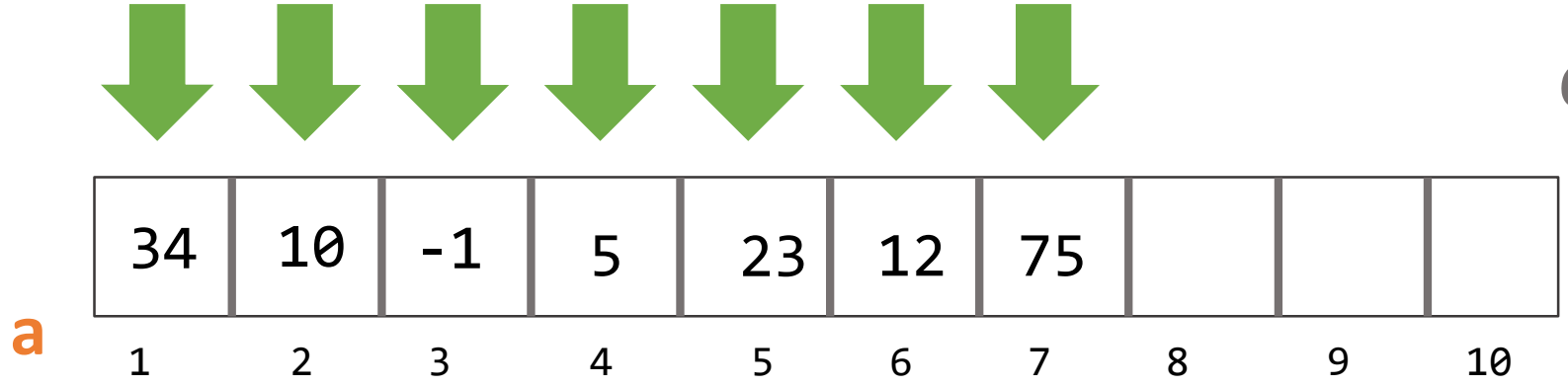
```
Dl = 7
```

```
5
```




```
Dl = 7
```

```
25
```



Qué pasos considero?

CADP – TIPOS DE DATOS VECTORES – BUSCAR 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 en el que se terminó el vector.

1- Inicializar la búsqueda desde la posición 1 (pos).

2- Mientras ((el elemento buscado no se igual al valor en el arreglo[pos]) y (no se termine el arreglo))

  2.1 Avanzo una posición

3- Determino porque condición se ha terminado el while y devuelvo el resultado.

Cómo se implementa?

CADP – TIPOS DE DATOS VECTORES – BUSCAR DESORDENADO

Dado un vector de números enteros (10 elementos como máximo) realice un programa que lea un nuevo número y determine si el valor se encuentra en el vector.

```
Program uno;
  const
    fisica = 10;
  type
    numeros= array [1..fisica]  of integer;

  var
     VN: numeros;
     dimL, valor:integer;
     ok:boolean;

  Begin
    cargar (VN,dimL);
    read(valor);
    res:= buscar(VN,dimL,valor);
  End.
```

VN

?	?	?	?	?	?	?	?	?	?

dimL = 5

VN

4	-1	10	3	7	?	?	?	?	?

valor = 10 dimL = 5 ok = true

VN

4	-1	10	3	7	?	?	?	?	?

CADP – TIPOS DE DATOS VECTORES – BUSCAR DESORDENADO

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

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

Es correcto?

Si el elemento no está, pos en este caso quedaría en 11, y en la última línea de la función estaría asignando el resultado de comparar a[11] = valor

CADP – TIPOS DE DATOS VECTORES – BUSCAR DESORDENADO

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

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

Es correcto?

Si el elemento no está, pos en este caso quedaría en 11, y en en el while se pregunta a[11] y no es válido

CADP – TIPOS DE DATOS VECTORES – BUSCAR DESORDENADO

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

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

Es correcto?

Si pos no es <= dL no significa que haya estado el elemento

CADP – TIPOS DE DATOS VECTORES – BUSCAR DESORDENADO

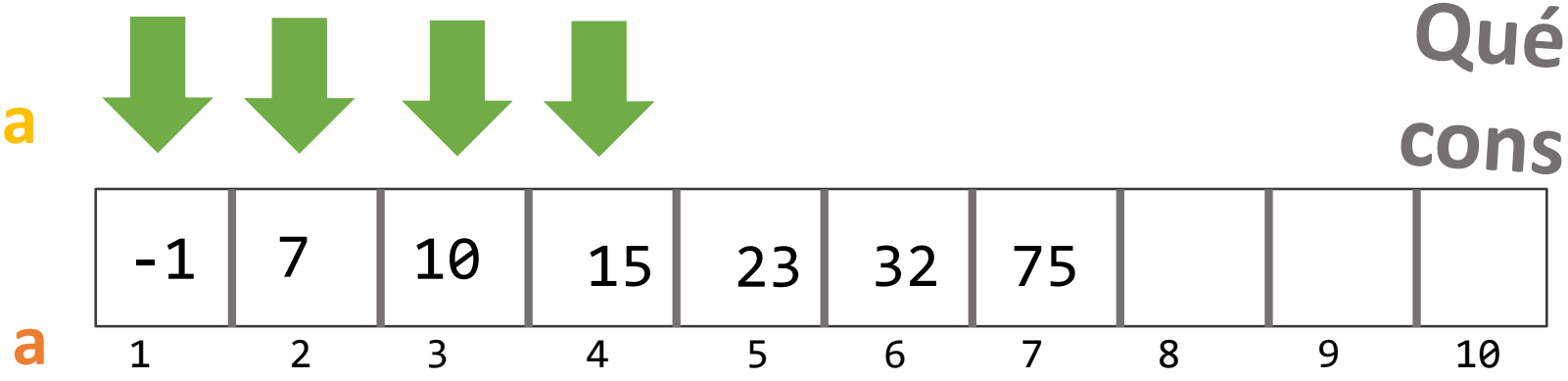
```
function buscar (a :números; dL:integer; valor:integer): boolean;
 Var
   pos:integer;
   esta:boolean;

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

VECTORES - BUSQUEDAS

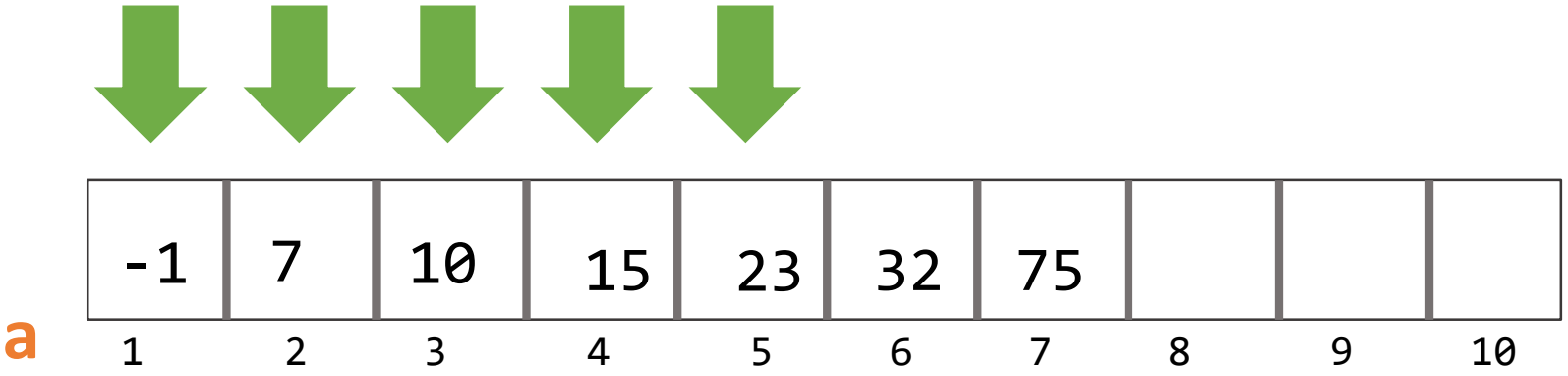
Vector Ordenado
Búsqueda Mejorada

Qué pasos considero?

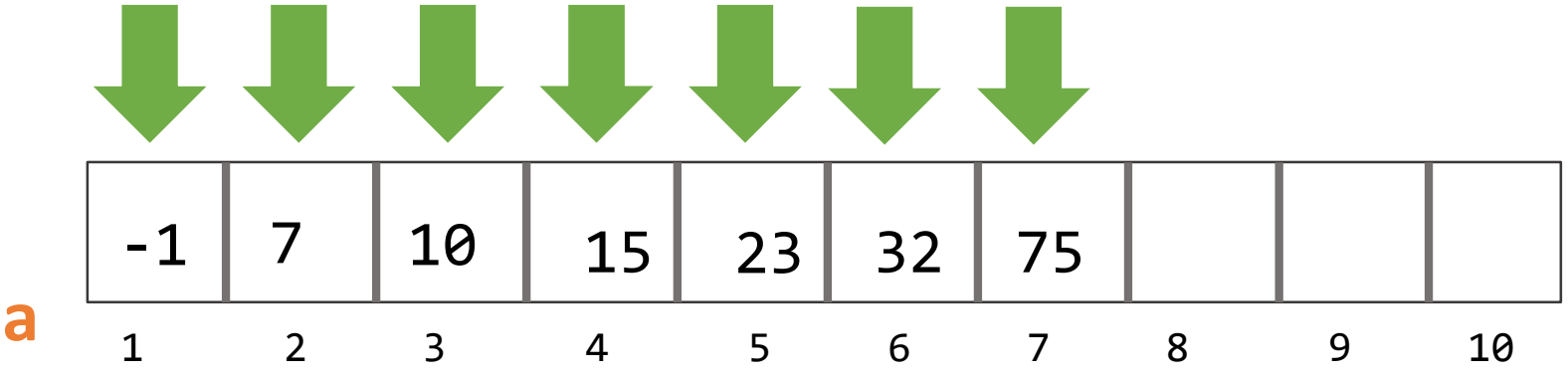
```
Dl = 7
15
```




```
Dl = 7
16
```




```
Dl = 7
80
```



CADP – TIPOS DE DATOS VECTORES – BUSCAR ORDENADO

BUSQUEDA MEJORADA

1- Inicializar la búsqueda desde la posición 1 (pos).

2- Mientras ((el elemento buscado sea menor al valor en el arreglo[pos]) y (no se termine el arreglo))
   2.1 Avanzo una posición

3- Determino porque condición se ha terminado el while y devuelvo el resultado.

Cómo se implementa?

CADP – TIPOS DE DATOS

Dado un vector de números enteros (10 elementos como máximo) ordenado realice un programa que lea un número e invoque a un módulo que retorne si el número se encuentra en el vector.

```
Program uno;
  const
    fisica = 10;
  type
    numeros= array [1..fisica]  of integer;

var
 VN: numeros;
 dimL,pos:integer;
 ok:boolean;
Begin
  cargar (VN,dimL);
 read(valor);
 ok:= existe(VN,dimL,valor);
End.
```

VN

?	?	?	?	?	?	?	?	?	?

dimL = 4

VN

4	-1	10	3	?	?	?	?	?	?

valor= -1 dimL = 4 ok = true

VN

4	-1	10	3	?	?	?	?	?	?

VECTORES - INSERTAR

```
Function existe (a:números; dL:integer; valor:integer):boolean;

Var
  pos:integer;

Begin
  pos:=1;

  while ( (pos <= dL) and (a[pos]< valor)) do
     begin
       pos:= pos + 1;
     end;

  if ( (pos <= dL) and (a[pos]= valor)) then buscar:=true
  else buscar:= false;
end.
```

Importa el orden en la condición del while?

Alcanza con preguntar por sólo una de las dos condiciones?

VECTORES - BUSQUEDAS

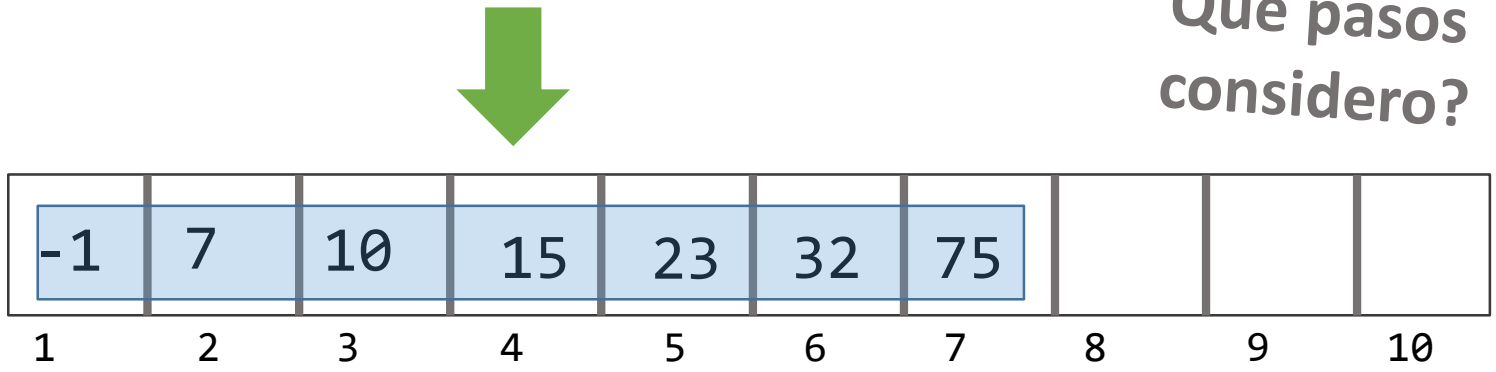
Vector Ordenado
Búsqueda DICOTOMICA

Qué pasos considero?

Dl = 7

10

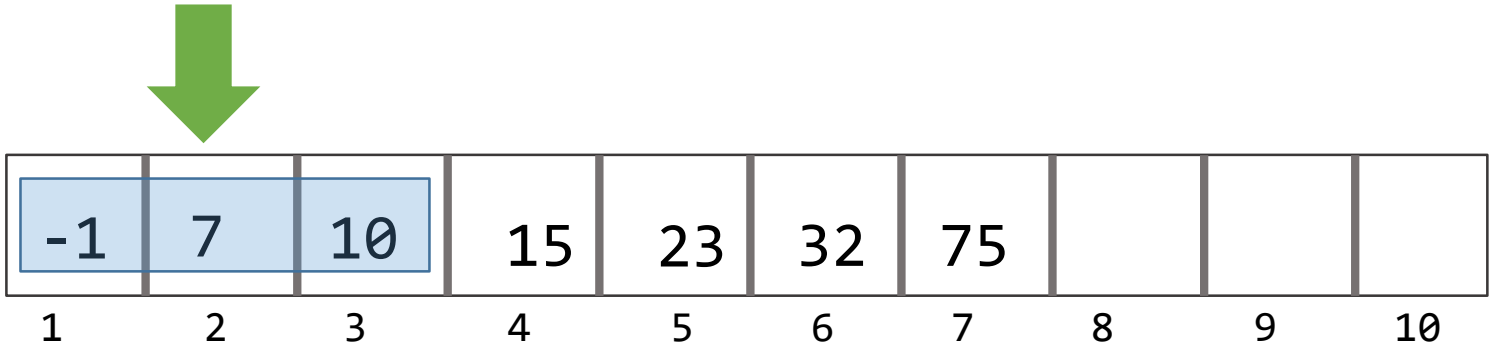
Inf 1
Sup 7
Medio 4



Dl = 7

10

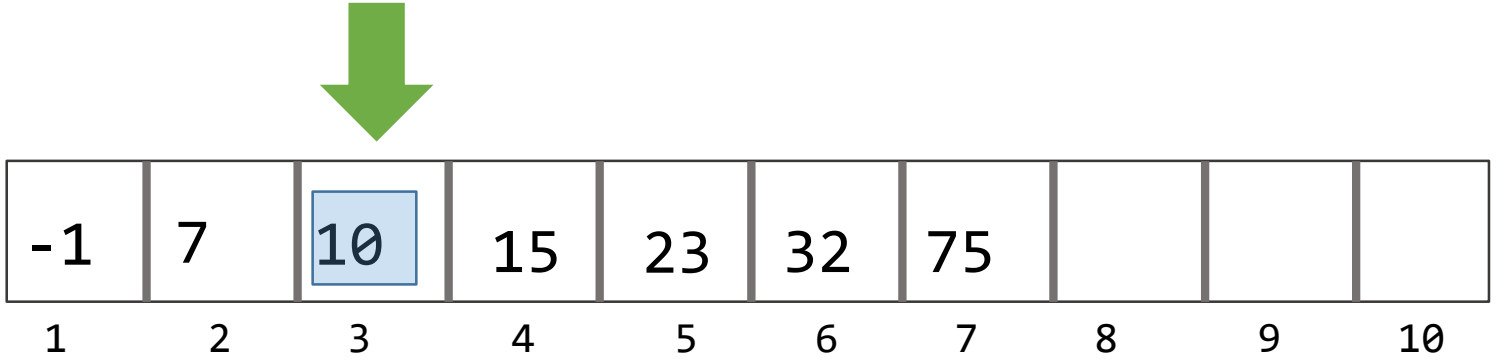
Inf 1
Sup 3 (medio-1)
Medio 2



Dl = 7

80

Inf 3(medio+1)
Sup 3
Medio 3



CADP – TIPOS DE DATOS VECTORES – BUSCAR ORDENADO

BUSQUEDA DICOTOMICA

Cómo se implementa?

1- Se calcula la posición media del vector (teniendo en cuenta la cantidad de elementos)

2- Mientras ((el elemento buscado sea <> arreglo[medio]) y (inf <= sup))

```
    Si ((el elemento buscado sea < arreglo[medio]) entonces
      Actualizo sup
    Sino
      Actualizo inf
    Calculo nuevamente el medio
```

3- Determino porque condición se ha terminado el while y devuelvo el resultado.

CADP – TIPOS DE DATOS

Dado un vector de números enteros (10 elementos como máximo) ordenado realice un programa que lea un número e invoque a un módulo que retorne si el número se encuentra en el vector.

```
Program uno;
  const
    fisica = 10;
  type
    numeros= array [1..fisica]  of integer;

var
 VN: numeros;
 dimL,pos:integer;
 ok:boolean;
Begin
  cargar (VN,dimL);
 read(valor);
 ok:= dicotomica(VN,dimL,valor);
End.
```

VN

?	?	?	?	?	?	?	?	?	?

dimL = 4

VN

4	-1	10	3	?	?	?	?	?	?

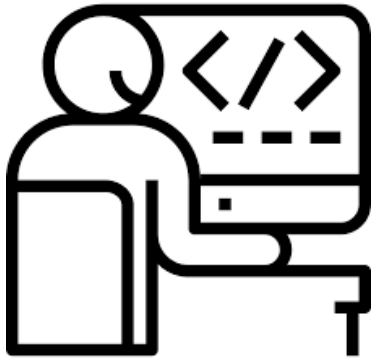
valor= -1 dimL = 4 ok = true

VN

4	-1	10	3	?	?	?	?	?	?

VECTORES - INSERTAR

```
Function dicotomica (a:números; dL:integer; valor:integer):boolean;
  Var
     pri, ult, medio : integer;
     ok:boolean

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

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



Conceptos de Algoritmos Datos y Programas

CADP – **TEMAS**

- Estructura de datos ARREGLO

- Dimensión física y dimensión lógica

CADP – TIPOS DE DATOS

VECTORES - OPERACIONES

Carga de valores

Lectura / Escritura

Recorridos

Dimensión física y lógica

Agregar elementos al final

Insertar elementos

Borrar elementos

Búsqueda de un elemento

Ordenación de los elementos

VECTORES

Supongamos que se existe un vector cargado de 10 elementos como máximo, pero por alguna circunstancia se cargaron sólo los primeros 4 valores.

20	77	68	2	?	?	?	?	?	?

a

Supongamos que sin saber que esto ocurrió se imprime el contenido del vector:

```
for i:= 1 to 10 do
   write (a[i]);
```

Que se obtendrá con la impresión?

VECTORES

DIMENSION FISICA

Se especifica en el momento de la declaración y determina su ocupación máxima de memoria.

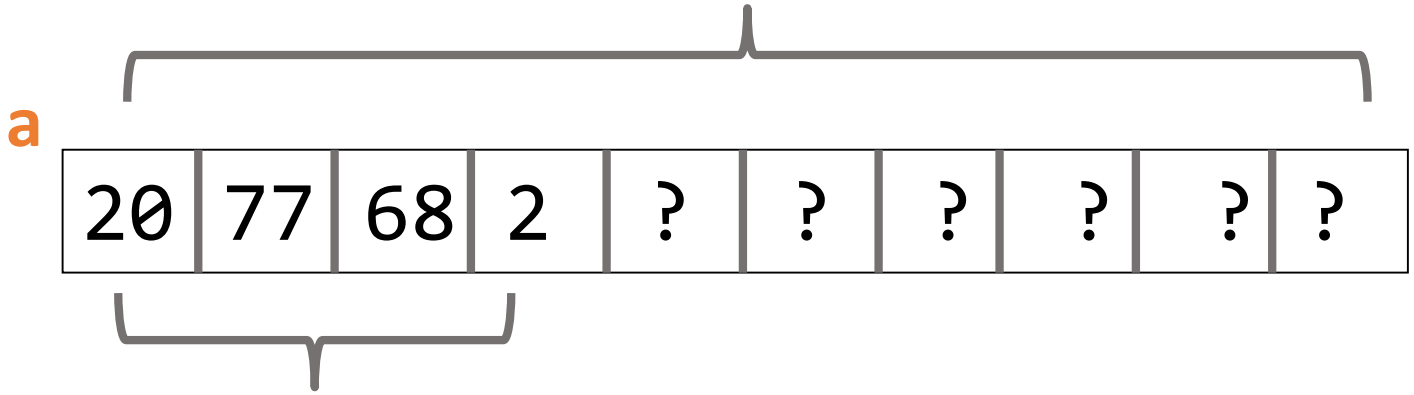
La cantidad de memoria total reservada no variará durante la ejecución del programa.

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.

VECTORES



Es la cantidad máxima de elementos que se pueden guardar en el arreglo.
No puede modificarse durante la ejecución del programa

Es la cantidad de elementos reales que se guardan en el arreglo.
Puede modificarse durante la ejecución del programa
Nunca puede ser mayor a la dimensión física (se debe controlar)

Cuándo se determina cada una?

Donde se declaran?

VECTORES

Realizar un programa que cargue un arreglo con números enteros hasta leer el número 50, a lo sumo se cargan 10 números.
Luego de terminar la carga informe cuál es el número mas grande de los leídos.

10
70
-1
50

a

10	70	-1	?	?	?	?	?	?	?

DF = 10
DL= 3

50

a

?	?	?	?	?	?	?	?	?	?

DF = 10
DL= 0

10
70
-1
4
80
-3
11
2
-1
6
...

a

10	70	-1	4	80	-3	11	2	-1	6

DF = 10
DL= 10

CADP – TIPOS DE DATOS

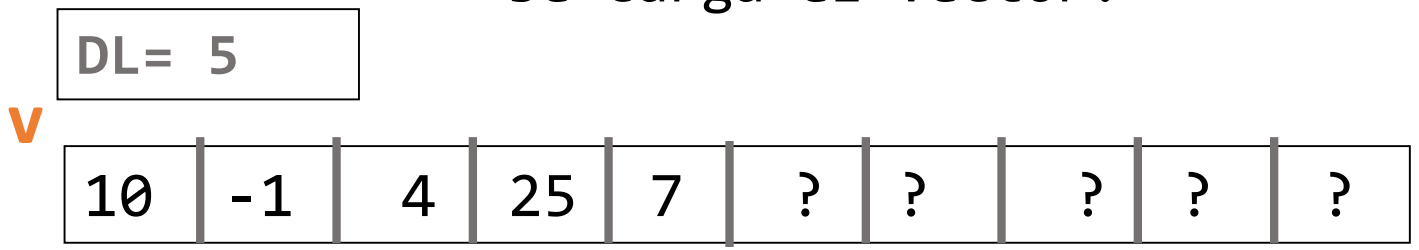
VECTORES

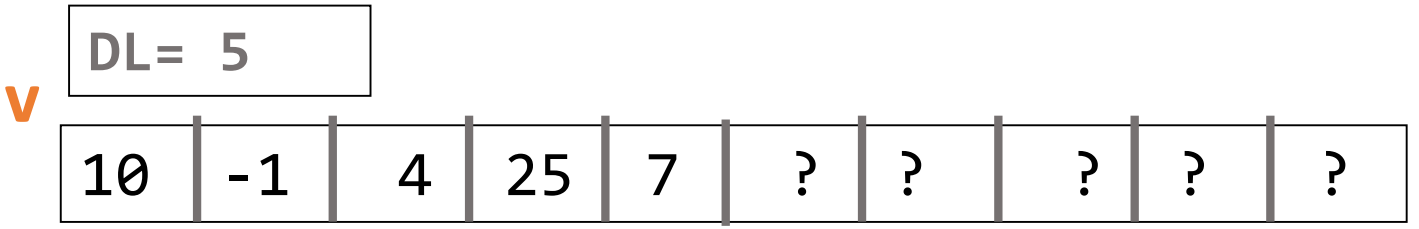
Realizar un programa que cargue un arreglo con números enteros hasta leer el número 50, a lo sumo se cargan 10 números.
Luego de terminar la carga informe cuál es el número mas grande de los leídos.

```
Program uno;
Const
 DF = 10
Type
   valores = array [1..DF] of integer;

Var
   v: valores;
   max:integer;
   dL:integer;
Begin
  cargarValores ( v , dL );
  max:= maximo (  v , dL );
End.
```

La dimensión física es una constante.

La dimensión lógica es una variable y toma valor cuando se carga el vector.





VECTORES

```
Procedure cargarValores (var a: números; var  dimL:integer);

 Var
  num:integer;

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

Es correcto?

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

VECTORES

```
Procedure cargarValores (var a: números; var  dimL:integer);

  Var
   num:integer;

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

Es correcto?

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

VECTORES

```
Procedure cargarValores (var a: números; var  dimL:integer);

  Var
   num:integer;

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

Es correcto?

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

VECTORES

```
Procedure cargarValores (var a: números; var  dimL:integer);

  Var
   num:integer;

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

Es correcto?

Si el primer número leído es 50, no entra al while, y como dimL está inicializado en 1, entonces devuelve que se cargó un elemento

```
Procedure cargarValores (var a: números; var  dimL:integer);

  Var
   num:integer;

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

Es correcto?

Qué pasa si leo mas de 10 números (el valor 50 no apareció y ya leí 10 valores)

VECTORES

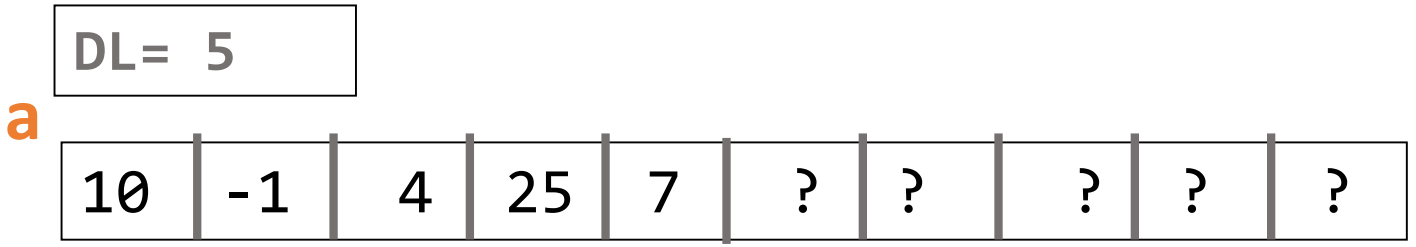
```
Procedure cargarValores (var a: números; var  dimL:integer);

  Var
   num:integer;

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

Es correcto?

SI!!!!

VECTORES

```
function maximo (a: números; dimL:integer):integer;

  Var
   max,i:integer;

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

   maximo:= max;
  End;
```



Es correcto?

NO! Sólo hay que recorrer hasta la cantidad de elementos cargados realmente

```
function maximo (a: números; dimL:integer):integer;

Var
 max,i:integer;

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

 maximo:= max;
End;
```