



# Conceptos de Algoritmos Datos y Programas

# CADP – TEMAS



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

Carga de valores

Lectura / Escritura

Recorridos

Dimensión física y lógica

Agregar elementos

Insertar elementos

Borrar elementos

**Búsqueda de un elemento**





**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

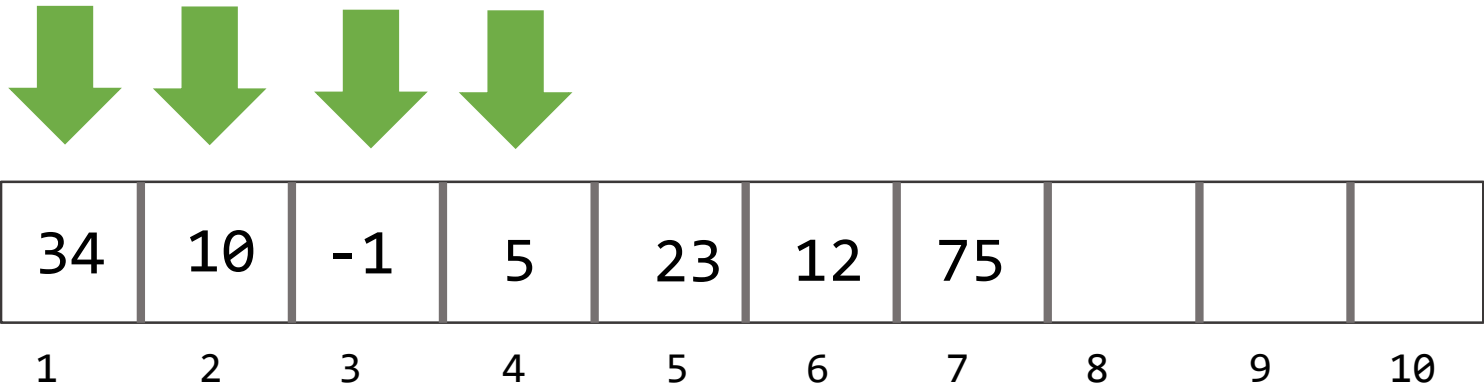
- Se debe recorrer el vector teniendo en cuenta el orden:
  - BUSQUEDA MEJORADA
  - BUSQUEDA BINARIA

Vector Desordenado

D1 = 7

5

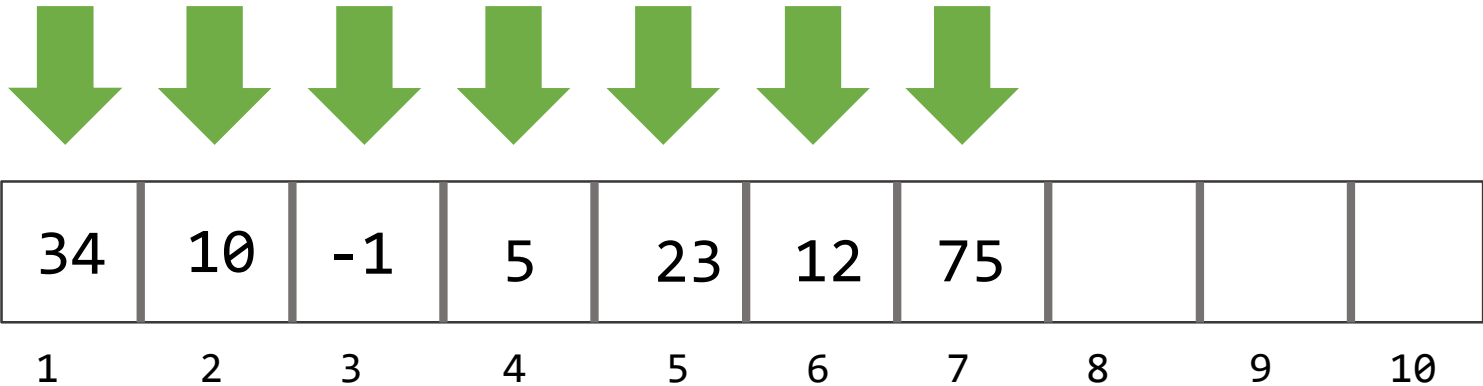
a



D1 = 7

25

a



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;
```

VN

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

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

dimL = 5

VN

4	-1	10	3	7	?	?	?	?	?
---	----	----	---	---	---	---	---	---	---

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

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 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.
```

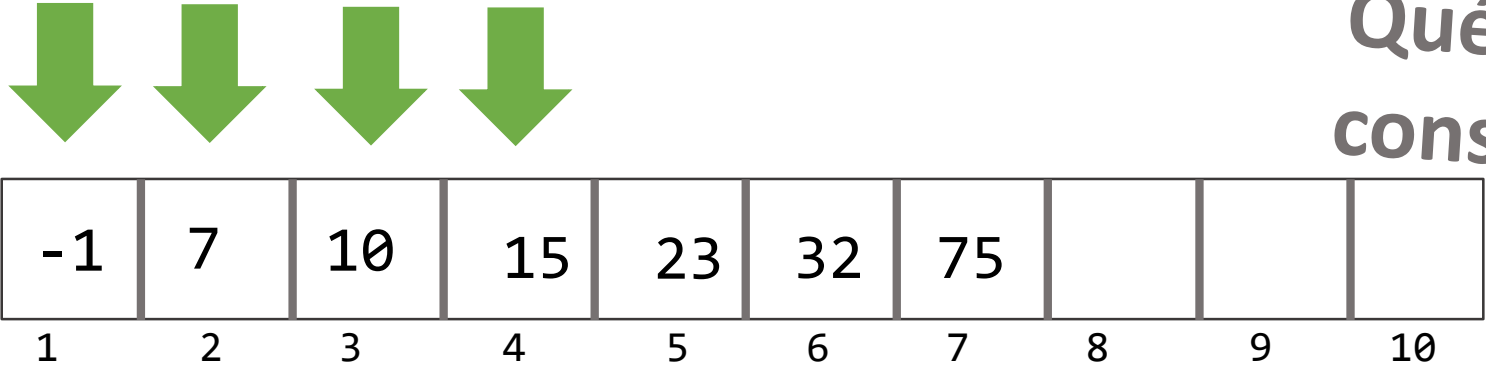
## Vector Ordenado Búsqueda Mejorada

Qué pasos considero?

D1 = 7

15

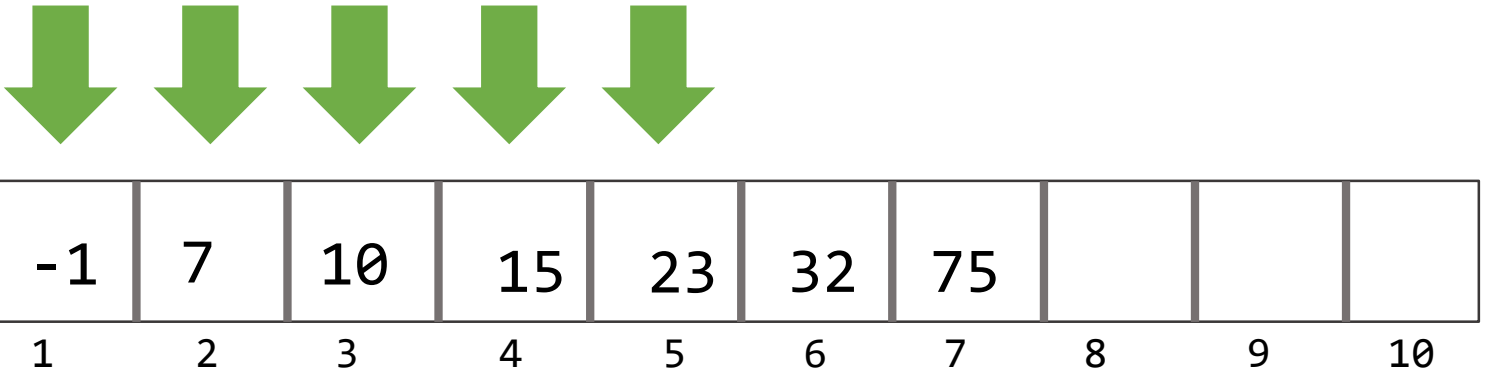
a



D1 = 7

16

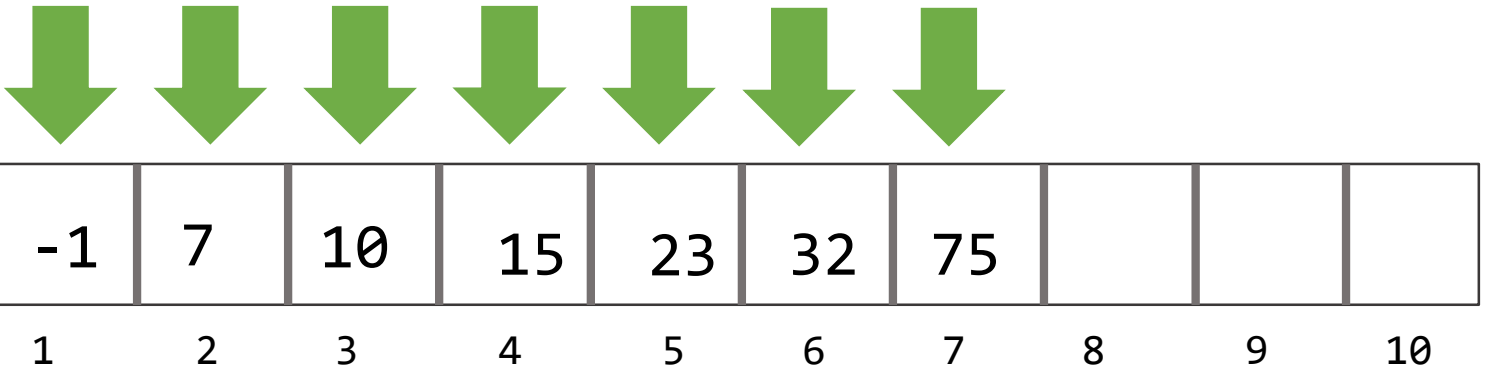
a



D1 = 7

80

a





## **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;
```

VN

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

dimL = 4

```
var  
  VN: numeros;  
  dimL,pos:integer;  
  ok:boolean;
```

VN

4	-1	10	3	?	?	?	?	?	?
---	----	----	---	---	---	---	---	---	---

```
Begin  
  cargar (VN,dimL);  
  read(valor);  
  ok:= existe(VN,dimL,valor);
```

valor= -1   dimL = 4   ok = true

VN

4	-1	10	3	?	?	?	?	?	?
---	----	----	---	---	---	---	---	---	---

```
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?*

# CADP – TIPOS DE DATOS

# VECTORES - BUSQUEDAS

## Vector Ordenado

## Búsqueda DICOTOMICA

Qué pasos considero?

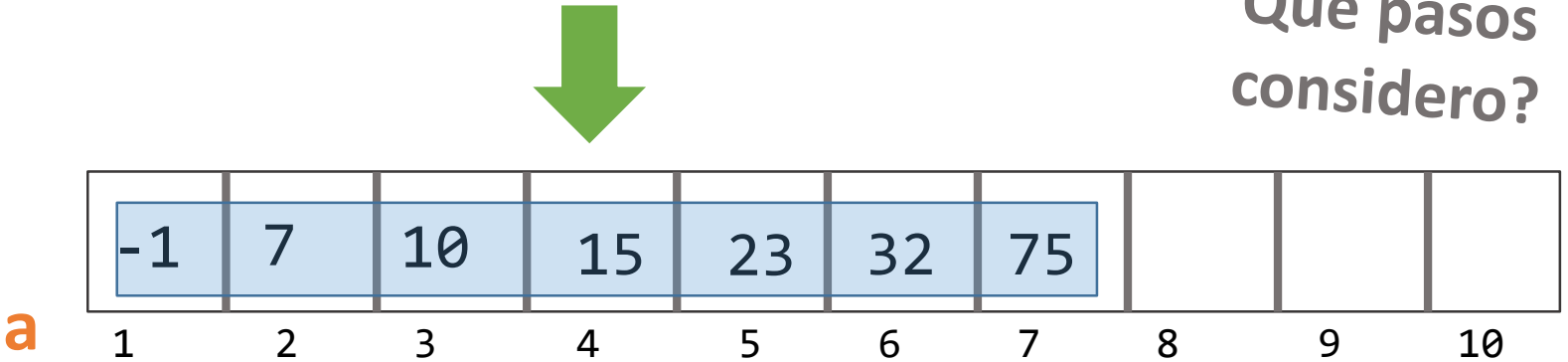
**D1 = 7**

**10**

Inf 1

Sup 7

Medio 4



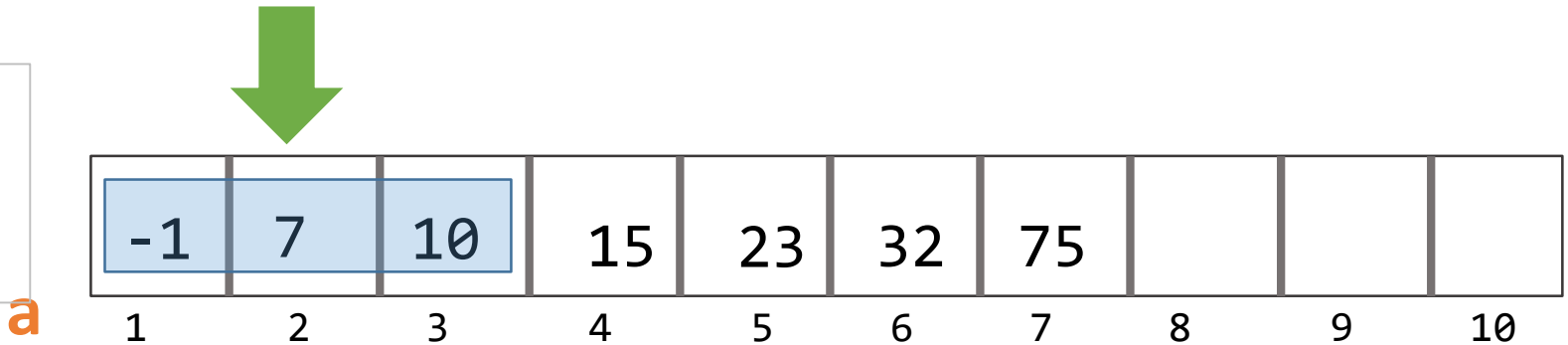
**D1 = 7**

**10**

Inf 1

Sup 3 (medio-1)

Medio 2



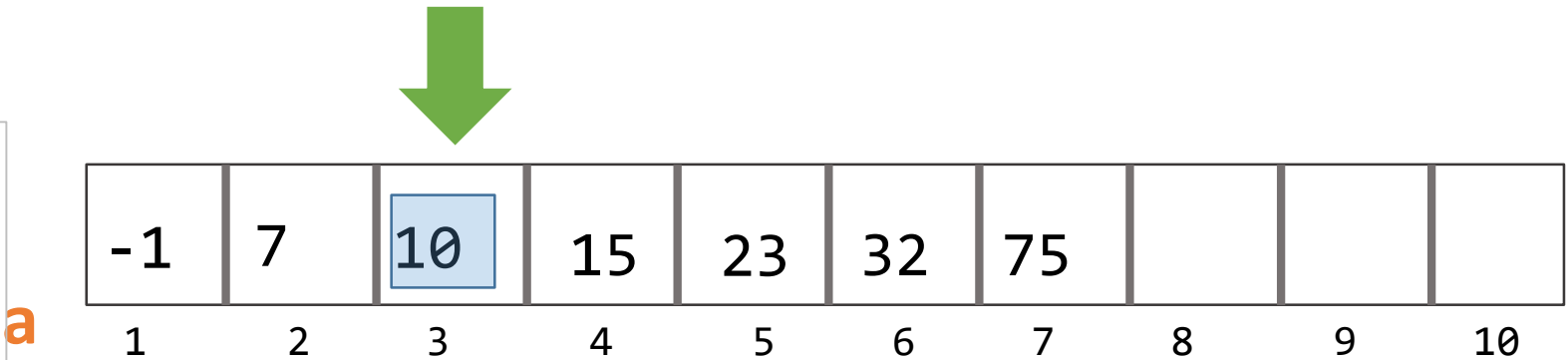
**D1 = 7**

**80**

Inf 3(medio+1)

Sup 3

Medio 3







## 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  $\neq$  arreglo[medio]) y (inf  $\leq$  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;
```

VN

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

dimL = 4

```
var  
  VN: numeros;  
  dimL,pos:integer;  
  ok:boolean;
```

VN

4	-1	10	3	?	?	?	?	?	?
---	----	----	---	---	---	---	---	---	---

```
Begin  
  cargar (VN,dimL);  
  read(valor);  
  ok:= dicotomica(VN,dimL,valor);
```

valor= -1   dimL = 4   ok = true

VN

4	-1	10	3	?	?	?	?	?	?
---	----	----	---	---	---	---	---	---	---

```
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.
```