



# *Unidad III*

---

## *Arreglos*



# Subarreglos

---

- Un subarreglo contiene componentes de otro arreglo llamado arreglo origen.
- Si las componentes del arreglo origen son de tipo simple, las del subarreglo deben ser del mismo tipo.
- Un subarreglo tiene la misma cantidad de componentes que el arreglo origen.
- Puede usarlas todas o una parte.
- Si usa una parte hay que saber cuantas componentes son y proteger la variable que tiene ese valor.



# Subarreglos

---

## Algoritmo SUB

void **carga** ( real ed[50])

Comienzo

entero i

    Para i Desde 0 Hasta 49

        Leer ed[i]

    FinPara

    retorna()

Fin

entero **carga\_subarreglo** ( real ed[50], real sel[50])

Comienzo

entero i,c

    c=0

    Para i Desde 0 Hasta 49

        Si (ed[ i ] >25)

            Entonces sel[ c ] = ed [ i ]

            c=c+1

        FinSi

    FinPara

    retorna c

Fin

void **mostrar\_subarreglo** ( real sel[50], entero can)

Comienzo

entero i,

    Para i Desde 0 Hasta can-1

        Escribir sel[ i ]

    FinPara

    retorna()

Fin

Comienzo/\* **Algoritmo Principal** /\*

entero edad[50] , selec[50], c

**carga** (edad)

    c=**carga\_subarreglo** (edad, selec)

**mostrar\_subarreglo** (selec, c)

Fin



# Búsqueda de un elemento en un arreglo

---

Como principales algoritmos de búsqueda en arreglos veremos:

- Búsqueda Secuencial
- Búsqueda Binaria



# Búsqueda Secuencial

---

Consiste en recorrer y examinar cada uno de los elementos del arreglo, desde el primero, hasta encontrar el elemento buscado o hasta que se hayan examinado todos los elementos del arreglo sin éxito.



# Búsqueda Secuencial

---

**Forma 1:** utilizando una bandera lógica

Algoritmo **Bandera**

Constante N=50

void **carga** ( entero ar[N])

Booleano **buscar** ( entero ar [N], entero el)

Comienzo

entero i

booleano esta

i=0

esta=falso

Mientras ((i<N) y ( esta == falso))

    Si (ar[ i] == el)

        Entonces esta = verdadero

        Sino i = i+1

    FinSi

FinMientras

retorna (esta)

Fin

Comienzo   /\***Algoritmo principal**/\*

entero arre [N], elem

booleano p

carga(arre)

Leer elem

p=buscar (arre, elem)

Si (p ==verdadero)

    Entonces

        Escribir "el elemento se encuentra en el  
        arreglo"

    Sino

        Escribir "el elemento no se encontró en el  
        arreglo"

FinSi

Fin



# Búsqueda Secuencial

---

**Forma 2:** sin utilizar una bandera lógica

Algoritmo **Sin\_Bandera**

Constante N=50

void **carga** ( entero ar[N])

entero **buscar** ( entero ar [N], entero elem)

Comienzo

entero i

i=0

Mientras ((i<N) y (ar[ i] != elem))

    i=i+1

FinMientras

retorna (i)

Fin

Comienzo **/\*Algoritmo principal/\***

entero arre [N], p, elem

carga(arre)

Leer elem

p=buscar (arre, elem)

Si (p <N)

    Entonces

        Escribir "el elemento se encuentra en la  
                posición", p

    Sino

        Escribir "el elemento no se encontró en el  
                arreglo"

FinSi

Fin



# Búsqueda Secuencial

---

**Forma 3:** utilizando un elemento centinela.

## Algoritmo **Centinela**

Constante  $N=50$

void **carga** ( entero ar[N+1])

entero **buscar** ( entero ar [N+1], entero elem)

Comienzo

entero  $i=0$

ar[N]=elem

Mientras (ar[i]!=elem)

$i=i+1$

FinMientras

retorna (i)

Fin

Comienzo **/\*Algoritmo principal/\***

entero arre [N+1], p, elem

carga(arre)

Leer elem

arre [N]= elem

p=buscar (arre, elem)

Si ( $p == N$ )

Entonces

    Escribir "el elemento no se encontró en el arreglo"

Sino

    Escribir "el elemento se encuentra en la posición", p

FinSi

Fin





# Búsqueda Binaria

---

Es un algoritmo de búsqueda que encuentra la posición de un valor en un arreglo **ordenado**.

**Compara el valor** con **el elemento en el medio** del arreglo, si no son iguales, la mitad en la cual el valor no puede estar es eliminada y la búsqueda continúa en la mitad restante hasta que el valor se encuentre o se quede sin espacio de búsqueda.



# Búsqueda Binaria

---

## Algoritmo **Búsqueda\_Binaria**

Comienzo

Constante N=9

void **carga** ( entero ar[N])

entero **buscar** ( entero ar [N], entero elem)

**Comienzo**

entero inf, sup, medio

inf=0

sup=N-1

medio=(inf+sup) div 2

Mientras ((inf<=sup) y (elem!=ar[medio]))

    Si ( elem<ar[medio] )

        Entonces sup=medio-1

        Sino inf=medio+1

    FinSi

    medio=(inf+sup) div 2

FinMientras

Si ( inf<=sup )

    Entonces retorna (medio)

    Sino retorna ( -1)

FinSi

Fin

Comienzo/\***Algoritmo principal** \*/

entero arre [ N], p, elem

carga(arre)

Leer elem

p=buscar (arre, elem)

Si (p != -1)

    Entonces

        Escribir "el elemento se encuentra en la posición", p

    Sino

        Escribir "el elemento no se encontró en el arreglo"

FinSi

Fin