



Algorítmica I

SESION 06

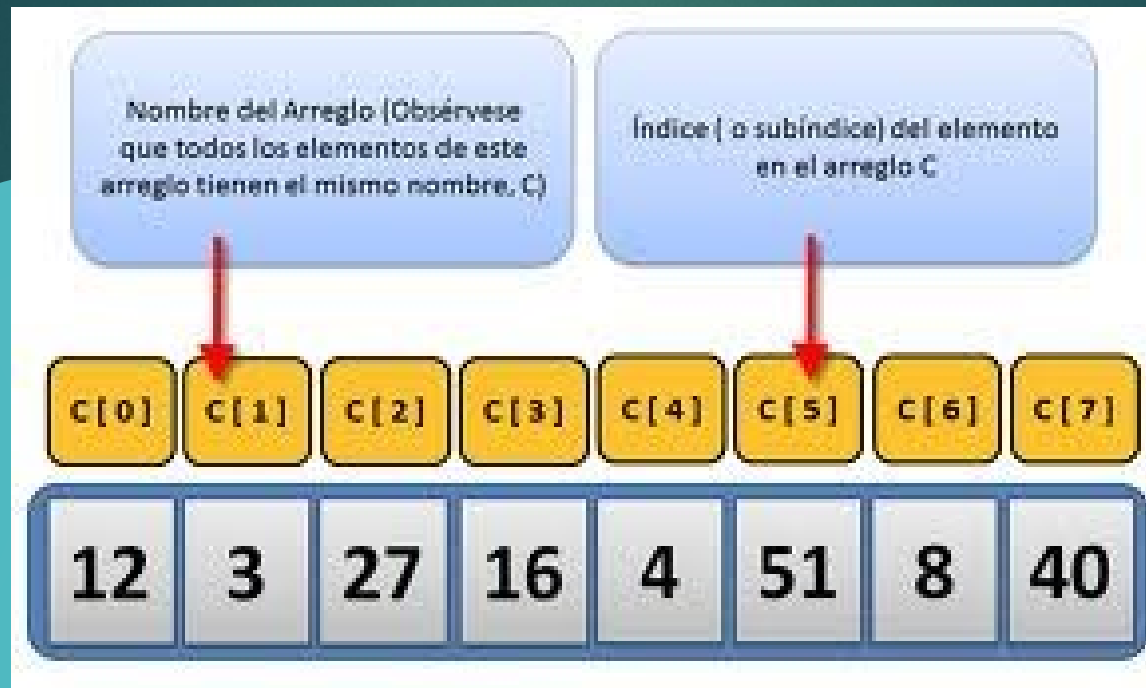
- Arreglos de una dimensión (Vectores)
- Métodos de Ordenamiento Simple

Contenido

- ▶ Arreglos de una dimensión (Vectores)
 - ▶ Definición de Vectores
 - ▶ Inicialización de Vectores
 - ▶ Procesos con vectores
 - ▶ Métodos de Ordenamiento

Arreglos Unidimensionales

Un arreglo es un conjunto finito de componentes del mismo tipo, los cuales se diferencian o relacionan a través de un sub índice.



ARREGLO UNIDIMENSIONAL (VECTORES)

- ▶ **Definición:** Los arreglos son un grupo de posiciones en memoria relacionadas entre sí por el hecho de que todas tienen el mismo nombre y los datos que contiene son todos del mismo tipo.
- ▶ Son entidades estáticas ya que conservan el mismo tamaño durante toda la ejecución del programa.
- ▶ Para poder referirnos a una posición en particular o al dato dentro de esa posición del arreglo, se especifica el nombre del arreglo y el número de posición del elemento. Las posiciones generalmente se cuentan a partir del cero como primera posición.

Arreglos Unidimensionales

- UN ARREGLO UNIDIMENSIONAL ES UN TIPO DE DATOS ESTRUCTURADO QUE ESTÁ FORMADO DE UNA COLECCIÓN FINITA Y ORDENADA DE DATOS DEL MISMO TIPO.
- ES LA ESTRUCTURA NATURAL PARA MODELAR LISTAS DE ELEMENTOS IGUALES.
- EL TIPO DE ACCESO A LOS ARREGLOS UNIDIMENSIONALES ES EL ACCESO DIRECTO, ES DECIR, PODEMOS ACCEDER A CUALQUIER ELEMENTO DEL ARREGLO SIN TENER QUE CONSULTAR A ELEMENTOS ANTERIORES O POSTERIORES, ESTO MEDIANTE EL USO DE UN ÍNDICE PARA CADA ELEMENTO DEL ARREGLO QUE NOS DA SU POSICIÓN RELATIVA.
- PARA IMPLEMENTAR ARREGLOS UNIDIMENSIONALES SE DEBE RESERVAR ESPACIO EN MEMORIA, Y SE DEBE PROPORCIONAR LA DIRECCIÓN BASE DEL ARREGLO, LA CANTIDAD DE ELEMENTOS MAXIMO.

Arreglos Unidimensionales

- ▶ Un arreglo de una dimensión es una lista de variables, todas de un mismo tipo a las que se hace referencia por medio de un nombre común.
- ▶ Una variable individual del arreglo se llama elemento del arreglo.
- ▶ Para declarar un arreglo de una sola dimensión se usa el formato general:

tipo_dato identificador[tamaño];

```
int arreglo[3];
```

arreglo[0]

arreglo[1]

arreglo[2]

Arreglos Unidimensionales

Ejemplo:

`int lista[10];` //declara un arreglo **lista** con **10** elementos de tipo **int**.



Los elementos del arreglo lista del ejemplo serian:

`lista[0] = 34` primer elemento del arreglo

`lista[1] = 15` segundo elemento del arreglo

.....

`lista[9] = 18` último elemento del arreglo

Declaración de un Vector

Se le asigna un nombre y se define su tamaño y el tipo de datos que va a contener (caracteres, enteros, reales, etc.)

En C++

```
int arreglo[10]; /* un arreglo de 10 enteros */
```

```
float arreglo[10]; /* un arreglo de 10 reales */
```

```
char arreglo[10]; /* un arreglo de 10 caracteres */
```


Inicializacion de un Vector

```
for (i = 0; i < 4; i++){  
    A[i] = 0;  
}  
int v[]={7,9,2,7,15,-3};
```

Por ejemplo, para crear un arreglo de diez elementos enteros, se escribe:

```
int num[10];
```

Esta declaración hace que el compilador reserve espacio suficiente para contener diez valores enteros.

Las definiciones de arreglos pueden incluir, si se desea, la asignación de valores iniciales. Los valores iniciales deben aparecer en el orden en que serán asignados a los elementos individuales del arreglo, encerrados entre llaves y separados por comas. La forma general es:

```
tipo nombre_arreglo[número_de_elementos] = {valor1,valor2,...,valorN};
```

La presencia del número de elementos del arreglo, es opcional cuando los valores iniciales están presentes. El método para inicializar arreglos mediante valores constantes después de su definición, es adecuado cuando el número de elementos es pequeño. Por ejemplo, para inicializar un arreglo de diez enteros con los valores 5, 1, 3, 8, 9, 3, 0, 1, 5, 7, se escribe:

```
int num[] = {5, 1, 3, 8, 9, 3, 0, 1, 5, 7};
```

Es importante señalar que cualquier elemento del arreglo puede ser manipulado en la misma manera que una variable. Por ejemplo, se puede incrementar en una unidad el contenido del tercer elemento de tres formas diferentes:

```
i = 2;           /* accederemos al tercer elemento*/  
a[i] = a[i] + 1; /* a[2] contiene el número 4*/  
a[i] += 1;       /* a[2] contiene el número 5*/  
a[i]++;          /* a[2] contiene el número 6*/
```

O llevar a cabo operaciones más complejas, como elevar al cubo el contenido del tercer elemento así:

```
a[i] *= a[i] * a[i]; /* a[2] contiene el número 216*/
```

Errores comunes que se pueden cometer:

1. Declarar un arreglo dándole dimensión mediante una variable.
2. Olvidar indicar el tamaño n del arreglo entre corchetes al momento de ser declarado cuando no se le asignen valores iniciales.
3. Olvidar que el primer elemento del arreglo tiene el subíndice cero.
4. Olvidar que el último elemento del arreglo tiene el subíndice $n-1$.

El paso de arreglos a funciones es una de las operaciones más comunes en la programación. Para que una función pueda recibir un arreglo unidimensional simplemente se especifica su tipo y se coloca después de su identificador, un par de corchetes vacíos. Por ejemplo en el siguiente prototipo de función:

```
float Prom( float x[ ], int n);
```

El compilador reconoce que el primer argumento que la función va a recibir es un arreglo por el par de corchetes vacío. El segundo argumento, que es una cantidad entera, representa el número de elementos del arreglo que será procesado. También se puede especificar, por ejemplo, si se va a pasar un arreglo de 10 elementos, la dimensión del arreglo así:

```
float Prom(float x[10], int n);
```

pero como no es necesario, rara vez se utiliza.

Errores más comunes que se pueden cometer:

1. Omitir el tipo de valor de regreso definido para una función en la declaración del prototipo, olvidando escribir un valor de retorno.
2. Regresar un valor de una función que fue declarada como void.
3. Volver a definir dentro de la función una variable como local, siendo que ésta fue declarada como variable paramétrica dentro de los paréntesis de la función.

Ejercicio 1

Elaborar un diagrama de flujo que represente la solución al siguiente requerimiento.

- ▶ Ingresar 10 valores numéricos enteros por teclado
- ▶ Visualizar solo los valores pares
- ▶ Obtener el promedio de esos valores pares

Ejercicio Practico

- ▶ Crear una aplicación en C++ que permita registrar por teclado 10 notas de alumnos (las notas deben ser enteras entre 0 y 20).
- ▶ Obtener lo siguiente:
 - ▶ El promedio ponderado
 - ▶ Lista de las notas desaprobadas
 - ▶ Lista de las notas aprobadas

Ejercicio 1

- ▶ Cargar dos vectores `vec1` y `vec2` de 20 posiciones cada uno y comparar los valores correspondientes a las mismas posiciones e imprimir el valor y la posición de aquellas que tengan el mismo valor.

Ejercicio 2

- ▶ Cargar un vector con 20 elementos enteros al azar y generar un vector que contenga solo los números pares y otro vector que contenga los números impares.
- ▶ Mostrar el vector original y los vectores par e impar

Métodos de ordenamiento

- ▶ BURBUJA
- ▶ POR SELECCIÓN
- ▶ POR INSERCIÓN
- ▶ POR INTERCAMBIO

Método de Ordenamiento Burbuja

Método Burbuja

- ▶ La **Ordenación de burbuja** (**Bubble Sort** en inglés) es un sencillo algoritmo de ordenamiento.
- ▶ Funciona revisando cada elemento de la lista que va a ser ordenada con el siguiente, intercambiándolos de posición si están en el orden equivocado.
- ▶ Es necesario revisar varias veces toda la lista hasta que no se necesiten más intercambios, lo cual significa que la lista está ordenada.
- ▶ Este algoritmo obtiene su nombre de la forma con la que suben por la lista los elementos durante los intercambios, como si fueran pequeñas "burbujas".
- ▶ También es conocido como el **método del intercambio directo**. Dado que solo usa comparaciones para operar elementos
- ▶ Es considerado un algoritmo de comparación, siendo el más sencillo de implementar.

Ejemplo:

- ▶ Elementos (A = 50, 20, 40, 80, 30), donde se introduce una variable interruptor para detectar si se ha producido intercambio en la pasada.

- ▶ **Pasada 1**

50	20	40	80	30
----	----	----	----	----

Intercambio 50 y 20

20	50	40	80	30
----	----	----	----	----

Intercambio 50 y 40

20	40	50	80	30
----	----	----	----	----

50 y 80 ordenados

20	40	50	80	30
----	----	----	----	----

Intercambio 80 y 30

20	40	50	30	80
----	----	----	----	----

Elemento mayor es 80
interruptor = TRUE

► Pasada 2

20	40	50	30	80
----	----	----	----	----



20 y 40 ordenados

20	40	50	30	80
----	----	----	----	----



40 y 50 ordenados

20	40	50	30	80
----	----	----	----	----



Se intercambian 50 y 30

20	40	30	50	80
----	----	----	----	----



- 50 y 80 elementos mayores ordenados y
- interruptor = TRUE

► **Pasada 3.- Solo se hacen dos comparaciones.**

20	40	30	50	80
----	----	----	----	----

20 y 40 ordenados

20	30	40	50	80
----	----	----	----	----

Se intercambian 40 y 30
interruptor = TRUE

- **Pasada 4.-** Se hace una única comparación de 20 y 30, y no se produce intercambio:

20	30	40	50	80
----	----	----	----	----

20 y 30 ordenados

20	30	40	50	80
----	----	----	----	----

Lista ordenada
interruptor = FALSE

The background is a dark teal gradient. It features several decorative elements: a large teal circle in the top right, a red vertical rectangle to its right, a large teal circle on the right side, a large teal circle in the bottom left, and a smaller teal circle in the bottom right.

► Ejercicio 01

Desarrollar el algoritmo para ordenar n elementos por el método Burbuja

Recomendaciones

- Uso de vectores
- Uso de For anidados
- Uso de un Flag o bandera

The background features a dark teal color with a faint, repeating pattern of a molecular or network structure, consisting of small dark spheres connected by thin lines. Overlaid on this are several decorative elements: a large light blue circle on the left, a medium light blue circle on the right, a smaller light blue circle at the bottom center, and a vertical red bar in the top right corner.

Método de Ordenamiento por Selección

Ordenamiento por Selección

Consiste en lo siguiente:

- ▶ Buscar el elemento más pequeño de la lista.
- ▶ Se intercambia con el elemento ubicado en la primera posición de la lista.
- ▶ Buscar el segundo elemento más pequeño de la lista.
- ▶ Se intercambia con el elemento que ocupa la segunda posición en la lista.
- ▶ Se repite este proceso hasta que esté ordenada toda la lista.

Método Selección

- Los métodos de ordenación por selección se basan en dos principios básicos:

Seleccionar el elemento más pequeño (o más grande) del arreglo. Colocarlo en la posición más baja (o más alta) del arreglo. A diferencia del método de la burbuja, en este método el elemento más pequeño (o más grande) es el que se coloca en la posición final que le corresponde.

- Consideremos un array A con 5 valores enteros 51, 21, 39, 80, 36:

$A[0]$	$A[1]$	$A[2]$	$A[3]$	$A[4]$
51	21	39	80	36

↓
Pasada 1

21	51	39	80	36
----	----	----	----	----

↓
Pasada 2

Pasada 0.
Seleccionar 21
Intercambiar 21 y $A[0]$

Pasada 1.
Seleccionar 36
Intercambiar 36 y $A[1]$



21	36	39	80	
----	----	----	----	--

Pasada 3

21	36		80	51
----	----	--	----	----

Pasada 4

21	36		51	
----	----	--	----	--

Pasada 2.
Seleccionar 39
Intercambiar 39 y
A[2]

Pasada 3.
Seleccionar 51
Intercambiar 51 y
A[3]

Lista ordenada


The background is a dark teal color. It features several large, semi-transparent teal circles of varying sizes. In the top right corner, there is a solid red vertical rectangle.

► Ejercicio 02

Desarrollar el algoritmo para ordenar n elementos aplicando el método por Selección

Recomendaciones

- Uso de vectores
- Uso de For anidados



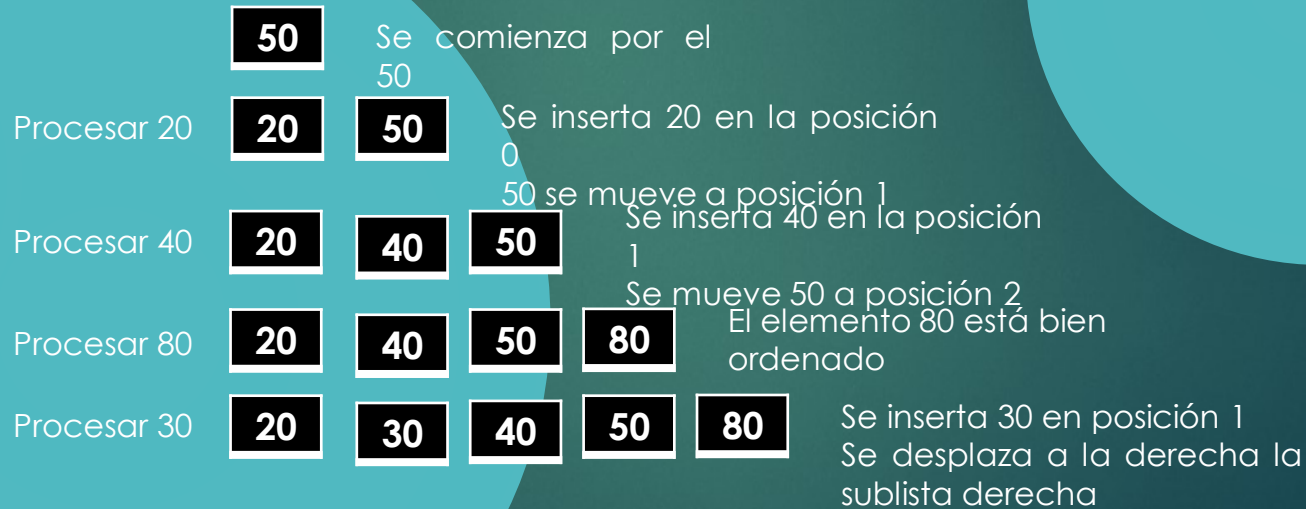
Método de Ordenamiento por Inserción

Ordenamiento por Inserción

- ▶ El algoritmo consiste en realizar varias pasadas sobre la lista de elementos.
- ▶ En cada pasada se analiza un elemento, y se intenta encontrar su orden relativo entre los analizados en pasadas anteriores.
- ▶ Cada elemento a analizar se desplaza por esa lista hasta encontrar su lugar.
- ▶ Cuando todos los elementos de la lista han sido analizados, la lista está completamente ordenada

Método Inserción

- ▶ El método de ordenación por inserción es similar al proceso típico de ordenar tarjetas de nombres (cartas de una baraja) por orden alfabético, que consiste en insertar un nombre en su posición correcta dentro de una lista o archivo que ya está ordenado.
- ▶ Así el proceso en el caso de la lista de enteros $A = 50, 20, 40, 80, 30$.



The background is a dark teal color. It features several light teal circles of different sizes. There are also several light teal arrows pointing to the right, some of which are partially obscured by the circles. In the top right corner, there is a vertical red bar.

Método de Ordenamiento por Intercambio

Método Intercambio

- Se encarga de ordenar los elementos de una lista en orden ascendente. Este algoritmo se basa en la lectura sucesiva de la lista a ordenar, comparando el elemento inferior de la lista con los restantes y efectuando intercambio de posiciones cuando el orden resultante de la comparación no sea el correcto.

Pasada 0

A[0] A[1] A[2] A[3]

8	4		2
---	---	--	---



4	8		2
---	---	--	---



4	8		2
---	---	--	---



Lista inicial

Se realiza intercambio

4	8		2
---	---	--	---

No se realiza intercambio

4	8		2
---	---	--	---

Se realiza intercambio

2	8		4
---	---	--	---

Lista resultante

► **Pasada 1**



Intercambi
o



Intercambi
o



Lista inicial

Lista resultante

Pasada 2

La sublista a considerar ahora es 8, 6 ya que 2, 4 está ordenada. Una comparación única se produce entre los dos elementos de la sublista



Intercambi
o



Lista inicial

Lista resultante



50	39	30	28	22	18	10	5
----	----	----	----	----	----	----	---

Búsqueda Binaria

N=8

Buscar 39

$$f=8/2=4$$

2

1

50	39	30	28
----	----	----	----

22	18	10	5
----	----	----	---

Buscado 39 es mayor a 22 buscar en segmento 1
Buscado 39 es menor a 28 busca en segmento 2

$$f=4/2=2$$

50	39
----	----

30	28
----	----