

Ordenamiento y Búsqueda de Información

Ing. Armandina J. Leal Flores

Algoritmos de Ordenamiento

► ¿Qué son?

- Son secuencias de instrucciones cuya función es acomodar los valores de un arreglo de menor a mayor.

► ¿Para qué se utilizan?

- Para poder buscar un valor más rápidamente.
- Ejemplo: Directorio Telefónicos

Algoritmos más comunes

- ▶ Burbuja
- ▶ Intercambio
- ▶ Selección Directa
- ▶ Inserción Directa

- ▶ Merge Sort
- ▶ Quicksort

- ▶ Heap Sort
- ▶ Shell Sort

Medida de Eficiencia

Una medida de eficiencia es:

- ▶ Contar el # de comparaciones
- ▶ Contar el # de movimientos de elementos

Estos están en función de el #(n) de elementos a ser ordenados.

La eficiencia de los algoritmos se mide por el número de comparaciones e intercambios que tienen que hacer.

Burbuja

- ▶ El método recibe como parámetro un arreglo unidimensional que contiene los elementos a ordenar.
- ▶ El método va comparando cada elemento del arreglo con el que sigue; si el elemento es mayor que el otro, se intercambian.
- ▶ Este proceso dejará el valor más grande al final del arreglo.
- ▶ Se debe repetir el proceso hasta que no haya ningún intercambio.

Ejemplo

Primera Pasada



Intercambio



Intercambio



Intercambio



Segunda Pasada



Intercambio



Intercambio



No hay Intercambio

Tercera Pasada



Como hubo
intercambios
se realiza otra
pasada

Burbuja

```
void burbuja (int iArr [], int iCantidad)
{
    bool bIntercambio = true;
    for (int iPasada = 1; iPasada < iCantidad && bIntercambio; iPasada++)
    {
        bIntercambio=false;
        for (int ij = 0; ij < iCantidad - iPasada; ij++)
        {
            if (iArr[ij] > iArr [ij+1] )
            {
                int iAux = iArr [ij];
                iArr [ij] = iArr [ij+1];
                iArr [ij+1] = iAux;
                bIntercambio = true;
            }
        }
    }
}
```

Intercambio

- ▶ Compara el elemento inferior de la lista con los restantes y efectúa el intercambio cuando el orden resultante de la comparación no es el correcto.

Ejemplo

Primera Pasada

8	7	1	6
---	---	---	---

Intercambio

7	8	1	6
---	---	---	---

7	8	1	6
---	---	---	---

Intercambio

1	8	7	6
---	---	---	---

1	8	7	6
---	---	---	---

No hay Intercambio

1	8	7	6
---	---	---	---

1	8	7	6
---	---	---	---

Intercambio

1	7	8	6
---	---	---	---

1	7	8	6
---	---	---	---

Intercambio

1	6	8	7
---	---	---	---

Segunda Pasada

1	6	8	7
---	---	---	---

Intercambio

1	6	7	8
---	---	---	---

Tercera Pasada

Intercambio

```
void intercambio(int iArr[], int iCantidad)
{
    int iAux;
    for (int il = 0; il < iCantidad-1; il++)
    {
        for(int ij = il+1; ij < iCantidad; ij++)
        {
            if ( a[il] > a[ij])
            {
                iAux = iArr[il];
                iArr[il] = iArr[ij];
                iArr[ij] = iAux;
            }
        }
    }
}
```

Selección Directa

- ▶ El método recibe como parámetro el arreglo que contiene los elementos a ordenar.
- ▶ *Busca en el arreglo el valor más pequeño.*
- ▶ *Intercambia este valor por el que se encuentra en la primera posición.*
- ▶ Repite el proceso con los valores restantes del arreglo (ya no se incluyen los valores que ya se acomodaron).
- ▶ *Nota: También puede hacerse en sentido contrario, en lugar de buscar el menor se busca el mayor y se coloca en la primera posición del arreglo.*

Ejemplo



Selecciona el menor (1)
Intercambia menor con casilla 0

Primera Pasada



Busca el menor: 6
Intercambia menor con casilla 1

Segunda Pasada



Busca el menor: 7
Intercambia menor con casilla 2

Tercera Pasada



Selección Directa

```
void seleccionDirecta ( int iArr[], int iCantidad )
{
    for ( int il = 0; il < iCantidad - 1; il++)
    {
        //Busca el valor más chico
        int iMenor = iArr[il];
        int iPos = il;
        for ( int ij = il+1; ij < iCantidad; ij++)
        {
            if ( iArr[ij] < iMenor )
            {
                iMenor = iArr[ij];
                iPos = ij;
            }
        }
        //intercambia el valor
        iArr[iPos] = iArr[il];
        iArr[il] = iMenor;
    }
}
```

Inserción Directa

- ▶ El método recibe como parámetro el arreglo que contiene los elementos a ordenar.
- ▶ Compara cada elemento del arreglo con los que se encuentran en las posiciones anteriores.
- ▶ Si resulta que el elemento con el que se está comparando es mayor, se recorre a la derecha
- ▶ Si resulta que el elemento con el que se está comparando es menor, se detiene el proceso debido a que ya se encontró la posición del elemento.

Ejemplo

Compara con los valores anteriores hasta que encuentra uno menor o igual.

Primera Pasada

8	7	1	6

Se compara el 7 con los valores anteriores
Se mueve el 8 a la casilla 1
Se inserta el 7 en la casilla 0

Segunda Pasada

7	8	1	6

Se compara el 1 con los valores anteriores
Se mueve el 8 a la casilla 2
Se mueve el 7 a la casilla 1
Se inserta el 1 en la casilla 0

Tercera Pasada

1	7	8	6

Se compara el 6 con los valores anteriores
Se mueve el 8 a la casilla 3
Se mueve el 7 a la casilla 2
Se inserta el 6 en la casilla 1

1	6	7	8

Inserción Directa

```
void insercionDirecta (int iArr[], int iCantidad)
{
    for ( int il = 1; il < iCantidad; il++ )
    {
        int iAux = iArr[il];
        int ij = il - 1;
        while ( ij >= 0 && iAux < iArr[iJ] )
        {
            iArr[iJ + 1] = iArr[iJ];
            ij--;
        }
        iArr[iJ + 1] = iAux;
    }
}
```

Comparaciones entre Algoritmos de Ordenamiento

<http://www.herongyang.com/sort/index.html>

<http://www.cs.ubc.ca/spider/harrison/Java/sorting-demo.html>

Algoritmos de Búsquedas

- ▶ Son secuencias de instrucciones que permiten localizar un elemento dentro de un arreglo.
- ▶ Principales métodos:
 - Búsqueda Secuencial
 - Búsqueda Binaria

Búsqueda Secuencial

- ▶ Se busca en secuencia posición por posición.
- ▶ ¿Cuándo se detiene?
 - Si la información está **ordenada**
 - Se detiene cuando se encuentra un valor mayor.
 - Si la información **no está ordenada**
 - Se detiene cuando se termina de comparar con todos los elementos.

Ejemplos

Información **Ordenada**

Localizar el 29

Comparaciones: 5

3	8	15	17	29	30	42
---	---	----	----	----	----	----

Localizar el 28

Comparaciones: 5

3	8	15	17	29	30	42
---	---	----	----	----	----	----

Información **en Desorden**

Localizar el 28

Comparaciones: 7

29	3	42	17	30	15	8
----	---	----	----	----	----	---

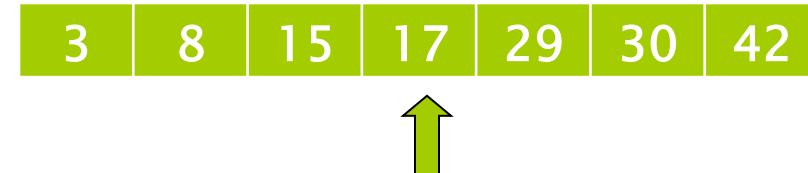
Búsqueda Binaria

- ▶ Se obtiene la posición central del arreglo.
- ▶ Se compara el valor que se quiere encontrar con el que está en la posición central.
- ▶ Si son iguales termina la búsqueda.
- ▶ Si el valor que se quiere encontrar es mayor que el que se encuentra en la posición, se descartan las posiciones de la 0 a la posición central; y se repite el proceso con las posiciones restantes.
- ▶ Si el valor que se quiere encontrar es menor que el que se encuentra en la posición, se descartan las posiciones de la posición central a la última; y se repite el proceso con las posiciones restantes.

Ejemplo:

Valor a encontrar: 29

Compara el valor que está en la casilla central con el valor a buscar (29 - 17)



Como el 17 es menor que 29 descarta los valores menores o iguales a 17

Compara el valor que está en la casilla central con el valor a buscar (29 - 30)



Como el 30 es mayor que 29 descarta los valores mayores o iguales a 17

Compara el valor que está en la casilla central con el valor a buscar (29 - 29)



FIN