Ordenamiento

Ordenamiento

Ya sabemos como declarar arrays, ahora nosotros vamos a saber cómo ordenar dichos arrays.

Ordenamiento

¿Para qué queremos ordenar arrays? muchos se preguntaran.

La respuesta es simple, imaginemos que tenemos una cantidad grande de números y necesitamos que nos muestren un array de esos números ordenados de menor a mayor o de mayor a menor, o un lista de personas ordenadas por por ejemplo su apellido , para eso yo debo modificar el orden de los elementos de mi array para que dicho ordenamiento sea posible, eso se consigue con los algoritmos de ordenamiento.

Ordenamiento

Nosotros vamos a enseñar el burbujeo, pero este es uno de los cuantos algoritmos que existen.

El metodo burbujeo es un algoritmo muy usado a la hora de ordenar un array, el algorimo compara elemento por elemento del vector, cumpliendo con una determinada lógica, si dicha lógica se cumple se van cambiando de lugar los elementos dentro del array.

Ordenamiento

Para esto nosotros debemos usar una variable auxiliar (como la que usabamos en los máximos y mínimos), que nos va a servir para que no se nos pierda la información por cada intercambio (**swap**), este método módifica nuestro array en tiempo real hasta ordenarlo por completo. Cómo compara elemento por elemento, este algoritmo es bastante poco óptimo, pero si eficiente, ya que siempre lograra su cometido (el de ordenar el vector)

Burbujeo

```
for(i=0;i<5-1;i++)
{
    for(j=i+1;j<5;j++)
    {
        if(numeros[i]>numeros[j])
        {
            aux=numeros[i];
            numeros[i]=numeros[j];
            numeros[j]=aux;
        }
    }
}
```

Así se declararia un burbujeo para ordenar 5 números.

Si quisiera ordenar mil números, cambiaria los 5 por 1000 y los ordenaria normalmente.

Burbujeo

Se divide en dos for, un for principal (variable i), que va a iterar por la cantidad de elementos del array - 1, ¿por qué -1? porque siempre se van a comparar posiciones del array distintas (no me serviria comparar la posición 0 de mi array con la posición 0 de mi array)

Y un for secundario (variable j) que va a ser la posición distinta que se va a comparar con el indice i, nunca i y j van a valer lo mismo cuando se compare uno con el otro, ya que el algoritmo no tendria sentido, el valor de i se va a comparar con todos los valores de j, hasta que el for principal termine.

Burbujeo

Imaginemos que tenemos un array de 5 números en este orden

array[0] = 6

array[1] = 5

array[2] = 9

array[3] = 1

array[4] = 3

Burbujeo

El algoritmo haria lo siguiente. la posición 0 de mi array, se va comparar con la posición 1, 2, 3 y 4. En cada comparación va a preguntar si lo que está en ese momento en la posición 0, es mayor a lo que está en la posición 1, 2 , 3 y 4.

array[0] > array[1]

Sabiendo que array[0] vale 6 esto se cumple, por ende, luego del swapeo el array quedaria así.

array[0] = 5 array[1] = 6 array[2] = 9 array[3] = 1 array[4] = 3

Burbujeo

-Luego se compararia el indice 0 con el 2, en el indice 0 ya no hay un 6, hay un 5, y en el indice 2 tenemos un 9 -> 5 > 9 (NO) por ende no ocurre swap.

-Luego comparo el indice 0 con el 3 (en el 3 hay un 1) -> 5 > 1 (SI, modifico el array)

array[0] = 1 array[1] = 6 array[2] = 9 array[3] = 5 array[4] = 3

-En la última iteración del for secundario tenemos al indice 0 que se compara con el 4.

1 no es mayor a 3, por ende no se modificaria el array.

Burbujeo

Una vez que muere el for secundario, se produce la primer iteración del for principal por ende i=1 y j que se inicia con (i+1), va a valer 2 en su primer iteración por ende el indice 1, se va a comparar con el indice 2, 3 y 4.

El array estaba así

array[0] = 1 array[1] = 6 array[2] = 9 array[3] = 5 array[4] = 3

y termina así

array[0] = 1 array[1] = 3 array[2] = 9 array[3] = 6 array[4] = 5

Burbujeo

Muere el for secundario y el for principal realiza otra iteración. Ahora i = 2 y j empieza valiendo 3.

El indice 2, se va a comparar con el 3 y el 4.

El array estaba así

array[0] = 1 array[1] = 3 array[2] = 9 array[3] = 6 array[4] = 5

y termina así

array[0] = 1 array[1] = 3 array[2] = 5 array[3] = 9 array[4] = 6

Burbujeo

Muere el for secundario, y nuevamente ocurre otra iteración en el for principal. Ahora i vale 3 y j comienza valiendo 4. Acá el indice 3 se va a comparar solamente con el indice 4

En el indice 3 está el 9 y en el 4 está el 6.

¿9 es mayor a 6? -> Si, por ende se produce el swap y el array queda así.

array[0] = 1 array[1] = 3 array[2] = 5 array[3] = 6 array[4] = 9

Quedo totalmente ordenado.

Ordenamiento

El array quedo totalmente ordenado, el algoritmo relaciona todos con todos, y hay momentos en los que hace el swap y momentos que no, por eso, no es el más óptimo que existe, pero si es más eficiente, ya que siempre va a ordenar de forma correcta.

Hay muchos algoritmos de ordenamiento, nosotros en la cursada con que sepan manejar burbujeo es suficiente, pero también tienen el método inserción que de otra forma, consigue el mismo objetivo que el burbujeo pero de una forma más óptima.

Inserción

Lo que realiza el inserción es tomar el segundo item (posición 1), hasta la última posición (posición) y compararlo con todos los que hay a su izquierda.

Por ejemplo indice 1 con el 0

En otra iteración Indice 2 con el 1 y el 0

Indice 3 con el 2, el 1 y el 0

Y el indice 4 con el 3, el 2 y el 1

Inserción

No vamos a entrar tan en detalle como hicimos con el burbujeo porque con que sepan cómo funciona el burbujeo ya es suficiente, pueden ver en el apunte de la clase una explicación más detallada del algoritmo, pero básicamente hace lo que le dijimos recién, comparar cada elemento, empezando desde el indice 1, con todos los de su izquierda. El ítem tomado, será “movido o insertado” de modo que a su izquierda no tenga ítems mayores.

Inserción

```
for(i=1;i<5;i++)
{
    aux = numeros[i];
    j = i - 1;

    while(j>-1 && aux<numeros[j])
    {
        numeros[j+1] = numeros[j];
        j--;
    }

    numeros[j+1]=aux;
}
```

En vez de usar dos for, usamos un while, que me indique que si el número que comparo a mi izquierda es mayor y el indice que está a mi izquierda es 0 o mayor.

Si el número que comparo a mi izquierda es mayor, hago el swap.