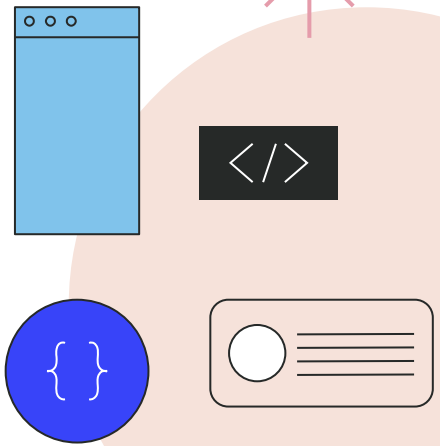
Estructura de datos y algoritmos

Ordenamiento y búsqueda

Viviana Gasull



Contenidos

Metodos de ordenamiento

- **Introducción**
- **Bubble Sort**
- **Insertion Sort**
- **Shell sort**

Algoritmos de ordenamiento

Ordenamiento

El **ordenamiento** o **clasificación** de datos (*sort*, en inglés) es una operación consistente en disponer un conjunto —estructura— de datos en algún determinado orden con respecto a uno de los ***campos/atributos*** de los elementos del conjunto.
Uno de los problemas fundamentales de la ciencia de la computación es ordenar una lista de objetos.
Para lograr este objetivo existen distintos algoritmos de ordenamiento.
Los algoritmos varían en complejidad y en eficiencia.

Algoritmos de Ordenamiento

- Un **algoritmo de ordenamiento** tiene como finalidad ubicar los elementos de una lista en determinado orden.
- Por lo tanto ordenar es el proceso de reacomodar los elementos de acuerdo a un orden preestablecido. El ordenamiento generalmente se realiza in-situ es decir sobre la misma estructura intercambiando elementos.
- El objetivo del ordenamiento es facilitar las búsquedas posteriores de los datos.
- En general se realizan ordenamientos sobre estructuras de datos de tipo numéricos, char y string.
- Para los datos tipo string y char el orden es el alfabético.
- Se han realizado distintos algoritmos buscando optimizar el tiempo de ordenamiento.

Algoritmos de Ordenamiento

Clasificación de algoritmos de Sorting

En las ciencias de la computación los algoritmos de ordenamiento se clasificados según:

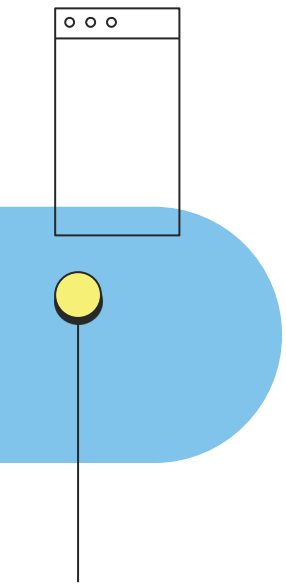
- **Complejidad computacional.** Son aquellos que se clasifican en términos de la eficiencia que tenga determinado algoritmo para organizar una lista de tamaño n. Normalmente, un buen comportamiento es aquel en que el número de comparaciones es n log n Clasificación de algoritmos de Sorting

Algoritmos de Ordenamiento

- **Uso de memoria:**

    - **Internal Sort:** se le llama a cualquier algoritmo que utilice exclusivamente la memoria principal, durante el ordenamiento. Esto asume una “high-speed random acces to all memory”.
    - **External Sort:** se le llama a cualquier algoritmo que utilice memoria externa, como lo podría ser un disco o u otro dispositivo.

¿Como harían par ordenar una lista de elementos?



Bubble sort

Es el método de ordenamiento más fácil de realizar y más fácil de comprender. Este se considera el más simple y es utilizado a nivel mundial.

Recibe este nombre porque los valores más pequeños “flotan” hacia la parte superior del arreglo, como las burbujas en el agua, mientras las más pesadas (valores mayores) se hunden hacia el fondo del arreglo.

La técnica consiste en llevar a cabo varias pasadas, en cada pasada se comparan pares de elementos y si están en orden inverso se intercambian de lugar.

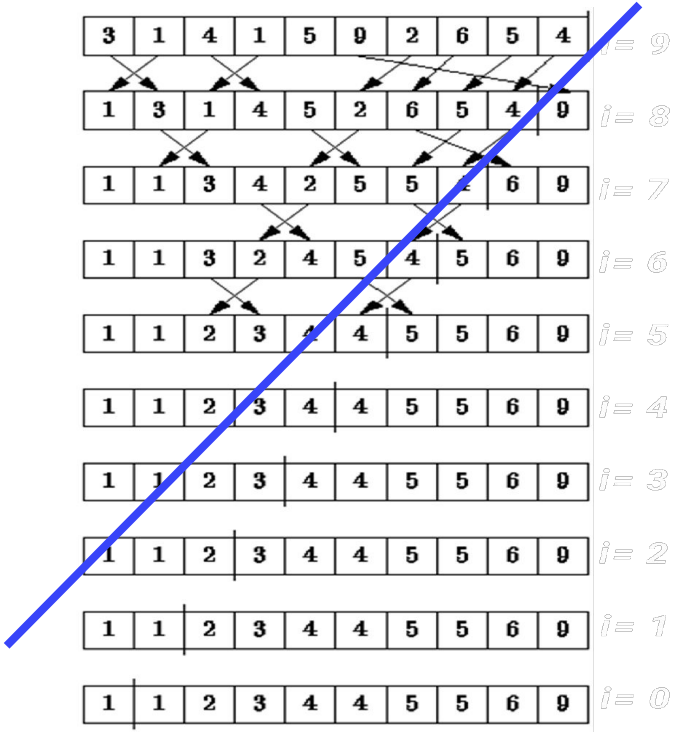
Bubble sort

El algoritmo inicia al principio de el conjunto de información a ordenar.

Compara los primeros dos elementos, y si el segundo es más chico que el primero, los intercambia y luego repite este procedimiento hasta que no hayan ocurrido cambios en la última evaluación.

El algoritmo realiza esto para cada par de elementos adyacentes, hasta que no tiene más elementos que comparar. Sin embargo este algoritmo es muy ineficiente, y es raramente utilizado, excepto para fines educacionales. Una variante de este método de ordenamiento es llamado Shuttle Sort

Bubble sort



A la derecha de la línea los elementos se encuentran ordenados.

Bubble sort

```
public class BubbleSort {
    public static void bubbleSort(int arreglo[], int tamaño) {
        int i, j, temp; // Variable temp para el intercambio
        for (i = (tamaño - 1); i >= 0; i--) {
            for (j = 1; j <= i; j++) {
                if (arreglo[j - 1] > arreglo[j]) {
                    temp = arreglo[j - 1];
                    arreglo[j - 1] = arreglo[j];
                    arreglo[j] = temp; } } } }
```

Qué modificarías en el algoritmo para que finalice cuando no haya más elementos fuera de orden.

Insertion Sort – Ordenamiento por inserción

Aunque éste algoritmo es más eficiente que el Bubble Sort, es ineficiente comparado con otros algoritmos de ordenamiento. Pero indudablemente Insertion Sort es una buena elección de método de ordenamiento para listas pequeñas de 30 elementos o menos.

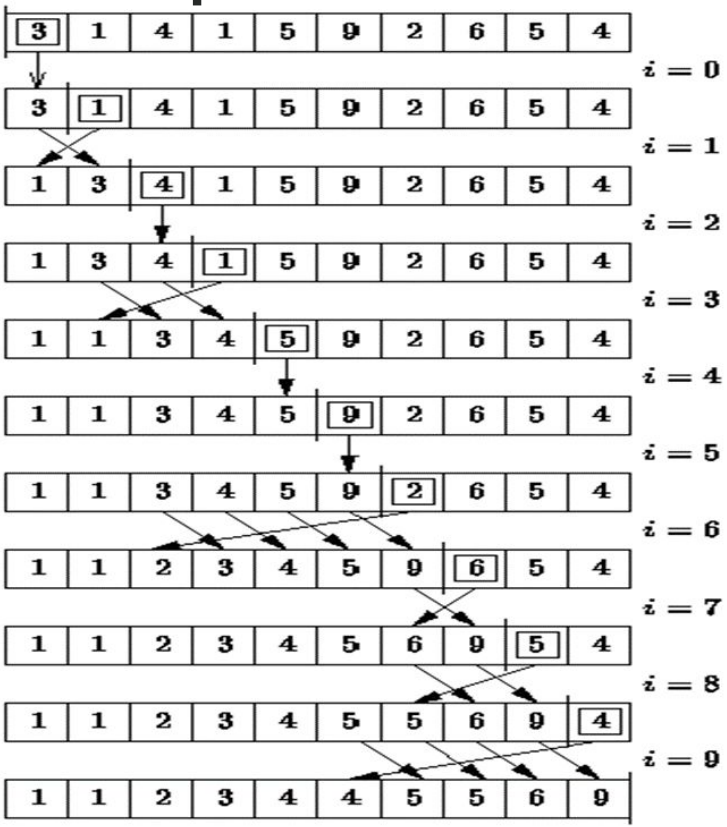
Una variante creada de insertion Sort que trabaja eficientemente con listas más grandes es shell Sort

Insertion Sort – por inserción

Veamos un pseudo código

```
Funcion insertionSort(arreglo, tamaño)
Para (i desde 1 hasta tamaño-1 )
Aux=arreglo[i];
j = i;
    Mientras j>0 ^ arreglo [j-1] > aux
        arreglo[j] = arreglo[j−1];
   j = j − 1;
    arreglo[j] = aux;
```

Insertion Sort – por inserción



Shell sort

Este método de ordenamiento fue propuesto en 1959 por Donald Shell. Mejora al ordenamiento Bubble y el Insertion sort.

El algoritmo Shell sort mejora el ordenamiento por inserción comparando elementos separados por un espacio de varias posiciones Gap. Esto permite que un elemento haga "pasos más grandes" hacia su posición esperada. Los pasos múltiples sobre los datos se hacen con tamaños de espacio cada vez más pequeños. El último paso del Shell sort es un simple ordenamiento por inserción, pero para entonces, ya está garantizado que los datos del vector están casi ordenados.

Aunque este método es ineficiente para grandes cantidades de información. Es uno de los algoritmos más rápidos para ordenar pequeñas cantidades de elementos.

Otra ventaja de este algoritmo es que requiere pequeñas cantidades de memoria.

Shell sort

- Imaginemos una lista de datos ordenables... por ejemplo, estos enteros:
  74, 14, 21, 44, 38, 97, 11, 78, 65, 88, 30
  n=11 (hay 11 elementos). Así que k<- n/2, e.d. K<-11/2 (K=5)

74, 14, 21, 44, 38, 97, 11, 78, 65, 88, 30
30, 14, 21, 44, 38, 74, 11, 78, 65, 88, 97

30, 14, 21, 44, 38, 74, 11, 78, 65, 88, 97

30, 11, 21, 44, 38, 74, 14, 78, 65, 88, 97

30, 11, 21, 44, 38, 74, 14, 78, 65, 88, 97

Queda como está y el 38 y 88 también

Shell sort

Pero de momento, nuestra k valía 5, así que ahora k<-k/2
e.d. K<-5/2 (K=2)

30, 11, 21, 44, 38, 74, 14, 78, 65, 88, 97

14, 11, 21, 44, 30, 74 , 38, 78, 65, 88, 97

Aplicamos Insertion y queda
11, 14, 21, 30, 38, 44, 65, 74, 78, 88, 97