Material descargable

JAVA Programación Orientada a Objetos

egg

Algoritmo de Ordenamiento Burbuja

El algoritmo de ordenamiento burbuja es uno de los algoritmos más simples para ordenar una colección de elementos. Puede aplicarse tanto a arreglos estáticos como a listas dinámicas (ArrayList) en Java. Además, es útil para ordenar tanto datos numéricos como alfabéticos.

Funcionamiento

1. **Comparación de Elementos:**
   - Comienza desde el primer elemento de la colección y compara con el siguiente elemento.
   - Si el primer elemento es mayor (o menor, dependiendo del orden deseado) que el segundo, se intercambian.
   - Este proceso continúa hasta que se haya comparado cada par de elementos adyacentes en la colección.
2. **Recorrido de la Colección:**
   - Después de un recorrido completo, el elemento más grande (o más pequeño) se moverá a la última posición de la colección.
   - Como resultado, después del primer recorrido, el elemento más grande estará en la última posición.
3. **Iteración:**
   - El proceso anterior se repite para los elementos restantes de la colección, es decir, para los primeros n-1 elementos en la segunda iteración, luego para los primeros n-2 elementos en la tercera iteración, y así sucesivamente.
4. **Complejidad Temporal:**
   - La complejidad temporal del algoritmo de ordenamiento burbuja es O(n^2), donde 'n' es el número de elementos en la colección.
   - Debido a su complejidad cuadrática, el algoritmo de ordenamiento burbuja puede ser ineficiente en colecciones grandes.

Ventajas y desventajas

Ventajas

- Es fácil de entender e implementar.
- Puede aplicarse tanto a arreglos estáticos como a listas dinámicas (ArrayList).
- Adecuado para listas pequeñas o casi ordenadas.
- Puede ordenar tanto datos numéricos como alfabéticos.

Desventajas

- Tiene una complejidad temporal cuadrática, lo que lo hace ineficiente para colecciones grandes.
- No es adecuado para datos grandes o colecciones ya ordenadas.

Veamos un ejemplo de implementación:

```java
import java.util.ArrayList;

public class App {
    public static void main(String[] args) {
        // CASO 1: MANEJO DE ARREGLO ESTÁTICO
        // Agrego info al arreglo de manera predeterminada
        int[] arr = { 64, 34, 25, 12, 22, 11, 90 };
        System.out.println("Array antes de ordenar:");
        printArray(arr);// Imprimo el arreglo inicial
        bubbleSort(arr);// Invoco al método de ordenamiento
        System.out.println("\nArray ordenado:");
        printArray(arr); // Imprimo el arreglo ya ordenado

        // CASO 2: MANEJO DE ARREGLO DINÁMICO
        // Agrego info al Array de manera predeterminada
        ArrayList<Integer> list = new ArrayList<>();
        list.add(64);
        list.add(34);
        list.add(25);
        list.add(12);
        list.add(22);
        list.add(11);
        list.add(90);
        System.out.println("\nArrayList antes de ordenar:");
        printArrayList(list); // Imprimo el arreglo inicial
        bubbleSort(list);// Invoco al metodo de ordenamiento
        System.out.println("\nArrayList ordenado:");
        printArrayList(list);// Imprimo el arreglo ya ordenado
```

```
    }

    // Ordenamiento Burbuja para arreglos estáticos (int[])
    static void bubbleSort(int[] arr) {
        int n = arr.length;
        for (int i = 0; i < n - 1; i++) {
            for (int j = 0; j < n - i - 1; j++) {
                if (arr[j] > arr[j + 1]) {
                    int temp = arr[j];
                    arr[j] = arr[j + 1];
                    arr[j + 1] = temp;
                }
            }
        }
    }

    // Ordenamiento Burbuja para listas dinámicas (ArrayList<Integer>)
    static void bubbleSort(ArrayList<Integer> list) {
        int n = list.size();
        for (int i = 0; i < n - 1; i++) {
            for (int j = 0; j < n - i - 1; j++) {
                if (list.get(j) > list.get(j + 1)) {
                    int temp = list.get(j);
                    list.set(j, list.get(j + 1));
                    list.set(j + 1, temp);
                }
            }
        }
    }

    // Método para imprimir arreglos
    static void printArray(int[] arr) {
        for (int i = 0; i < arr.length; i++) {
            System.out.print(arr[i] + " ");
        }
        System.out.println();
    }

    // Método para imprimir ArrayList
    static void printArrayList(ArrayList<Integer> list) {
        for (int i = 0; i < list.size(); i++) {
            System.out.print(list.get(i) + " ");
        }
        System.out.println();
    }

}
```

El algoritmo de ordenamiento burbuja es versátil y fácil de implementar en Java, ya sea para ordenar arreglos estáticos o listas dinámicas. Además, puede manipular tanto datos numéricos como alfabéticos. Aunque es útil para fines educativos y para listas pequeñas o casi ordenadas, su complejidad cuadrática lo hace ineficiente para colecciones grandes. En tales casos, se prefieren otros algoritmos de ordenamiento más eficientes como el mergesort, quicksort, o heapsort.