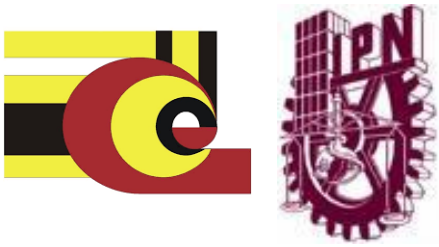
Ordenamiento por el método de la burbuja.



Diplomado de Java - Módulo I

Departamento de Diplomados y Extensión Profesional

Centro de Investigación en Computación

Instituto Politécnico Nacional

Modulo I – Ordenamiento burbuja.

Profesor:

Alan Badillo Salas

Integrantes:

Erick Alvarez Barcena

becker7000@outlook.es

Fecha de entrega:

30 de abril de 2022

Algoritmo de ordenamiento por el método de la burbuja.

Con este algoritmo podemos ordenar una lista pequeña de números en orden creciente o decreciente, una lista de n elementos requiere una cantidad hasta n-1 pasadas. Por cada pasada se comparan parejas de elementos adyacentes y en caso de que en la comparación el elemento izquierdo sea mayor al derecho entonces se intercambian, esto mismo da un efecto de burbujeo de los números más grandes hasta el final. Cada que vamos haciendo una pasada se ordena por lo menos un número, eso quiere decir que entre más pasadas a la lista menos comparaciones son requeridas.

Pseudocodigo del algoritmo de ordenamiento por el método de la burbuja.

```
//Algoritmo de ordenamiento de la burbuja: toma una lista de números y los ordena de forma ascendente.
//Aunque también podemos modificar el algoritmo para que los ordene de forma descendente.
Proceso OrdenamientoPorBurbuja

    //Datos de entrada:
    Definir N como entero; //La constante N guardará el tamaño de la lista de numeros.
    Imprimir "Dame el tamaño de la lista: "; //Se muestra un mensaje en pantalla con instrucciones.
    leer N; //Se registra el valor de N con el teclado.
    Definir lista como entero; //Se define el tipo de dato de lista como entero.
    Dimension lista[N]; //Se crea una lista con tamaño N de elementos.

    //Procedemos a leer cada uno de los valores:
    Definir i como entero; // La variable i será una variable de control que nos ayuda a iterar el ciclo ´para´.
    para i<-0 hasta (N-1) con paso 1 Hacer // Se crea el ciclo para con un recorrido desde 0 hasta N-1.
        Imprimir "Valor del elemento ",(i+1),": "; //Se muestra en pantalla el elemento a registrar valor.
        Leer lista[i];  //Se registra con el teclado un valor de la lista.
    FinPara

    //Inicia el ordenamiento con el algoritmo de la burbuja, definimos una variable que guardará un valor de la lista,
    Definir actual Como Entero;//este valor se va a comparar con el valor del elemento ´siguiente´ en la lista.
    Definir siguiente Como Entero; //Definimos otra variable que guardará el valor a la derecha de ´actual´ en la lista.

    //La variable de control i será usada para cada pasada desde 0 hasta N-1.
    Definir j Como Entero; //Variable de control será usada para comparar cada pareja consecutiva de la lista.

    //Iniciamos el recorrido de la lista, con cada pasada llevamos el valor más grande al final de la lista.
    //De ahí el nombre Burubuja haciendo referencia a las burbujas de un refresco que sólo las más grandes suben hasta el final.
    para i <- 0 hasta N-1 Con Paso 1  Hacer //Bucle para hacer cada una de las pasadas a la lista.
        //Bucle para hacer cada comparación posible, entre más pasadas menos comparaciones serán necesarias.
        para j<- 0 hasta N-2 Con Paso 1 Hacer
            actual <- lista[j]; //Aseguramos en una variable los valores actual y siguiente.
            siguiente <- lista [j+1];
            si actual>siguiente Entonces  //En caso de estar desordenados, se ordenan.
           //si usamos la condición actual>siguiente los ordenará de forma creciente.
           //si usamos la condición actual<siguiente los ordenará de forma descreciente.
                lista[j]=siguiente;  //Se asigna el valor menor a la posición a la izquierda de la posición j+1.
                lista[j+1]=actual; //Se asigna el valor mayor a la posición a la derecha de la posicón j.
            FinSi
        FinPara //Al finalizar este ciclo un elemento de la lista se ordenará.
    FinPara //Al finalizar este ciclo todos los elementos ya estarán ordenados.

    //Mostramos en pantalla como quedó la lista luego de ordenarla.
    para i <- 0 Hasta N-1 con paso 1 Hacer //Al momento de imprimir sumamos 1 a i para mostrar una enumeración
        Imprimir "Elemento ",(i+1),": ",lista[i]; //de los elementos más amigable. Ejemplo: Elemento 1: 45
    FinPara
FinProceso
```

Implementando el seudocódigo en código Java.
(Con cada línea comentada).

Este código fue escrito, compilado y ejecutado usando la tecnología IntelliJ IDEA versión Community.

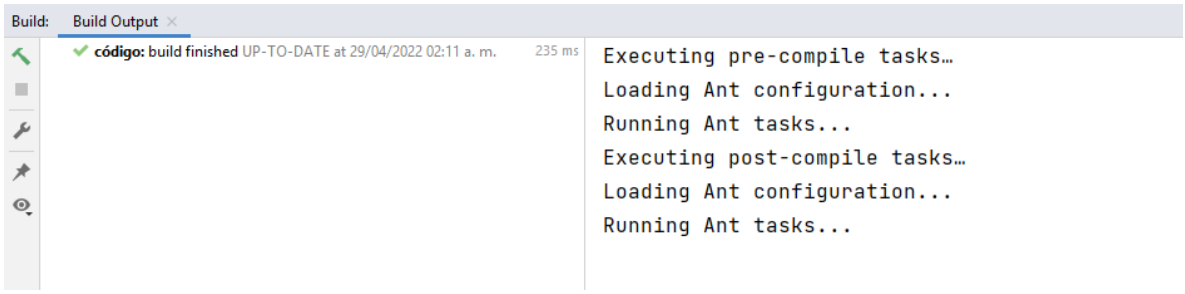
```java
//Se importal la librería para poder usar la clase Scanner
import java.util.Scanner;

//Nombre de la clase siguiendo el estilo UpperCamellCase
public class TestOrdenamintoBurbuja {
    //Método estatico vacío que recibe un arreglo llamado args para
    //poder ejecutar la clase TestOrdenamientoBurbuja.
    public static void main(String[] args) {
        //Instanciando un objeto de la clase Scanner
        Scanner entrada = new Scanner(System.in);

        //Creando una constante que guardara el tamaño de la lista:
        final int N;
        //Dando una instruccion al ususario:
        System.out.print("\n\tDame el tamaño de la lista: ");
        //Registrando el valor de la constante N con el teclado:
        N = Integer.parseInt(entrada.nextLine());
        //Creando un arreglo de tamaño N de tipo entero:
        int lista[] = new int[N];

        //Ciclo for que llega hasta un elemento antes de la longitud de lista:
        for (int i = 0; i < lista.length; i++) {
            //Se muestra al usuario el elemento que se va a guardar:
            System.out.print("\n\t Valor del elemento "+(i+1)+": ");
            //Se registra con el teclado un elemento del arreglo lista:
            lista[i]=Integer.parseInt(entrada.nextLine());
        }

        //Se crean dos varibles enteras, ´actual´ guardará
        //un elemento de la lista, ´siguiente´ guardará el elemento
        //adyacente a ´actual´ en el arreglo.
        int actual;
        int siguiente;

```

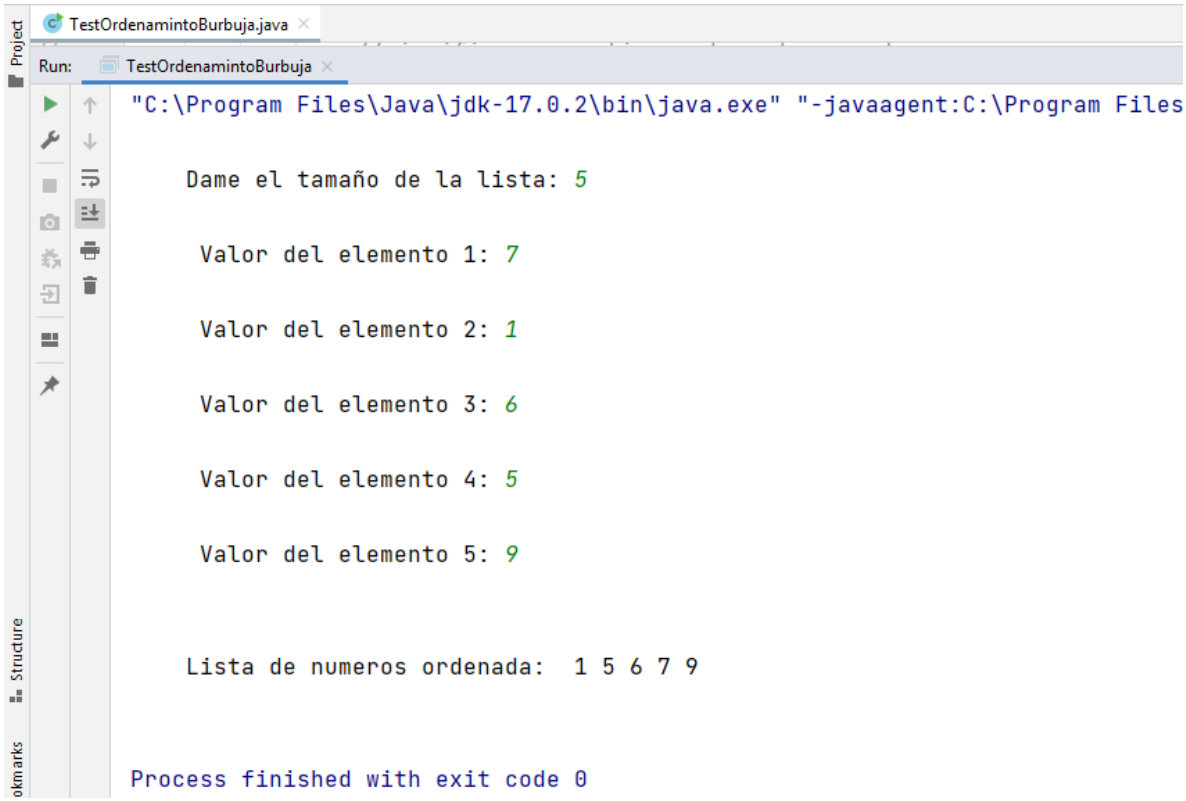
Continuación…

```
35          //Este ciclo hará el total de pasadas posibles a la lista:
36          for (int i = 0; i < lista.length; i++) {
37              //Este ciclo hará todas las compracaciones posibles
38              //por cada pasada:
39              for (int j = 0; j < lista.length-1; j++) {
40                  //Se asigna temporalmente un valor a ´actual´:
41                  actual=lista[j];
42                  //Se asigna temporalmente un valor a ´siguiente´:
43                  siguiente=lista[j+1];
44                  //En caso de que el elemento ´actual´que está a la
45                  //izquierda de ´siguiente´sea menor que ´siguiente´
46                  //significa que están en desorden y entonces se
47                  //intercambian sus valores:
48                  if (actual>siguiente){
49                      lista[j]=siguiente;
50                      lista[j+1]=actual;
51                  }
52              }
53          }
54
```

```
55          //Finalmente con un ciclo foreach mostramos en pantalla
56          //como quedó la lista ya ordenanda luego del proceso:
57          System.out.print("\n\n\tLista de numeros ordenada: ");
58          for (int x : lista ) {
59              System.out.print(" "+x);
60          }
61          //Salto de linea para mejor formato de presentación:
62          System.out.println("\n");
63      }
64  }
65
```

Compilando y ejecutando el código Java.





Conclusiones.

Éste algoritmo propone una forma fácil de ordenar los número de una lista, sin embargo, aunque es mucho más sencillo que cualquier otro algoritmo de ordenamiento nos ayuda bien a hacer ordenamientos de cantidades pequeñas de números. Nos ayuda también a entender el intercambio entre elementos de la lista que es un pilar importante de muchos algoritmos de ordenamiento.