

Universidad Nacional de Ingeniería

Facultad de Electrotecnia y Computación -.Ingeniería en Computación

Algoritmización y estructuras de datos Laboratorio No. 4

Alumno:

Marcel Enmanuel Díaz Largaespada (2020-1384U)

Docente:

Adilson González López

Fecha de entrega:

Miércoles 25 de agosto del 2021

Índice

1. Introducción	3
2. Diseño del <i>form</i>	4
3. Código Fuente	
3.1. Declaraciones y métodos iniciales	
3.2. Métodos de ordenamiento	6
3.3. Métodos de interacción con la interfaz	9
4. Ejecución del programa	10
5. Conclusiones	

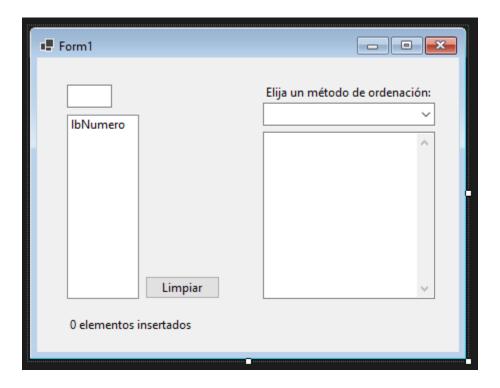
1. Introducción

En este laboratorio de Algoritmización y Estructuras de Datos se trabajó nuevamente con métodos de ordenación para reagrupar los elementos de un arreglo en un orden de menor a mayor. A modo de finalización, los métodos con los que se trabajó fueron los siguientes: *Ordenación por sacudida, inserción directa e inserción decreciente*. A mi consideración, son formas algo rebuscadas y tediosas de reagrupar elementos, sin embargo, eso no les sustrae la eficacia y la funcionalidad.

Como ya es de costumbre, en el siguiente documento serán descritas todas las partes del código y adjuntas a ellas habrá capturas del mismo, en el IDE de Visual Studio.

2. Diseño del form

El diseño de esta interfaz no buscaba innovar por sobre las anteriores. Al contrario, se trabajó en una interfaz genérica.



3. Código Fuente

3.1. Declaraciones y métodos iniciales.

Estas son las declaraciones de variables que estaremos empleando a lo largo del código. Al mismo tiempo, debajo de ellas hay la función de que limpiará lo mostrado en el panel que nombramos **lbNumeros**:

En la función de inicialización de nuestro *form* se le añaden las opciones de selección a la caja correspondiente a la que le asignamos por nombre **cbMetodo**:

```
1referencia
private void Form1_Load(object sender, EventArgs e)
{
    cbMetodo.Items.Add("Shaker Sort/Sacudida");
    cbMetodo.Items.Add("Insercion directa");
    cbMetodo.Items.Add("Shell/InsercionIncrementoDecrecientes");
    cbMetodo.DropDownStyle = ComboBoxStyle.DropDownList;
}
```

Esta función genérica nos permite imprimir en el textbox denominado **txtOrden** los elementos del arreglo según se vayan ordenando en el arreglo.

3.2. Métodos de ordenamiento.

Metódo Shake Sort u ordenación por sacudida

```
1 referencia
void InsercionDirecta()
   N = 0;
    foreach (int Elemento in lbNumero.Items)
        Array.Resize(ref Numeros, N + 1);
       Numeros[N] = Elemento;
        N++;
    for (i = 1; i < N; i++)
        aux = Numeros[i];
        k = i - 1;
        while ((k \ge 0) \&\& (aux < Numeros[k]))
            Numeros[k + 1] = Numeros[k];
            k = k - 1;
            Imprimir();
            Numeros[k + 1] = aux;
        Imprimir();
```

Método por inserción directa

```
1 referencia
void InsercionDecreciente()
   N = 0;
   foreach (int Elemento in lbNumero.Items)
       Array.Resize(ref Numeros, N + 1);
       Numeros[N] = Elemento;
       N++;
   bool band;
   inta = N;
   while (inta > 0)
        inta = inta / 2;
        band = true;
        while (band)
           band = false;
            i = 0;
            while ((i + inta) \le N - 1)
                if (Numeros[i] > Numeros[i + inta])
                    aux = Numeros[i];
                    Numeros[i] = Numeros[i + inta];
                    Numeros[i + inta] = aux;
                    band = true;
                i = i + 1;
        Imprimir();
```

Método Shell o por inserción decreciente

3.3. Métodos de interacción con la interfaz.

Cuando se selecciona un método de ordenamiento la caja de selección, se llama a la función correspondiente a dicho método:

```
//Evento: cbMetodo Click
1referencia
private void cbMetodo_SelectedIndexChanged(object sender, EventArgs e)
{
    if (cbMetodo.SelectedIndex == 0)
    {
        txtOrden.Clear();
        Sacudida();
    }
    if (cbMetodo.SelectedIndex == 1)
    {
        txtOrden.Clear();
        InsercionDirecta();
    }
    if (cbMetodo.SelectedIndex == 2)
    {
        txtOrden.Clear();
        InsercionDecreciente();
    }
}
```

Esta función nos permitirá ingresar números y mostrarlos en pantalla por medio de la caja de listas que nosotros denominamos como **lbNumero:**

```
referencia
private void txtNumero_KeyPress(object sender, KeyPressEventArgs e)
{
    if (e.KeyChar == 13)
    {
        int Num = int.Parse(txtNumero.Text);
        lbNumero.Items.Add(Num);
        lEtiqueta.Text = Convert.ToString(lbNumero.Items.Count + " elementos insertados.");
        txtNumero.Text = "";
        txtNumero.Focus();
    }
}
```

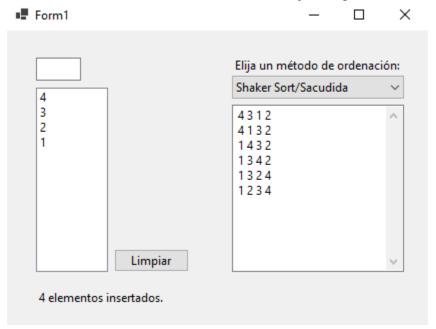
Esta pequeña línea de código declara un arreglo de un tamaño indefinido¹:

int[] Numeros;

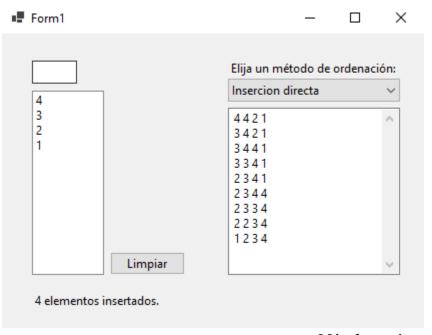
¹ Seguramente debió de haber sido declarada en el segmento de **declaraciones globales**, pero por alguna razón se encontraba ahí y ahí mismo lo deje. No por nada bien se dice que, si algo funciona apropiadamente de la manera en la que está, por conveniencia, que así se deje.

4. Ejecución del programa

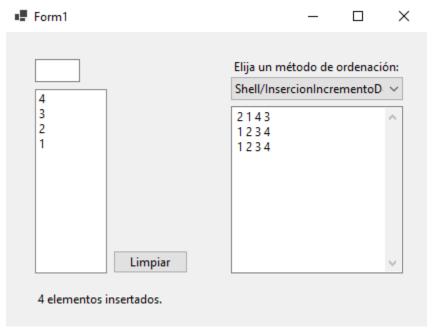
Finalmente, el resultado de nuestro laboratorio arrojó la siguiente interfaz:



Ingresando el «(in)famous» 4, 3, 2, 1, y escogiendo Shake Sort como método de ordenamiento



Método por inserción directa



Método Shell o Inserción Decreciente

5. Conclusiones

Con estos tres métodos se culmina con los algoritmos de ordenación de elementos en un arreglo. Los resultados fueron óptimos y fructíferos, puesto que próximamente tendremos que emplearlos en nuestros códigos para ordenar los arreglos de la misma forma.