

UNIVERSIDAD NACIONAL DE INGENIERÍA

FACULTAD DE ELECTROTECNIA Y COMPUTACIÓN

Algoritmización y Estructuras de Datos

Profesor:

Adilson G. López

Grupo: 2M1 - CO

Autor:

Gabriel A. Ortiz

2020 - 0325U

Índice

1. Introducción	2
2. Código Fuente	3
2.1. Tabla de herramientas - propiedades - eventos	3
2.2. Interfaz gráfica	4
2.3. Variables globales	5
2.4. Eventos	6
2.4.1. Evento de cargar formulario	6
2.4.2. Evento KeyPress del TextBox	6
2.4.3. Evento del botón limpiar	7
2.4.4. Evento al cambiar método de ordenamiento	7
2.5. Métodos	8
2.5.1. Método establecer arreglo	8
2.5.2. Método invertir posición	8
2.5.3. Método para imprimir	9
2.5.4. Método Inserción directa	9
2.5.5. Método Shell	10
2.5.6. Método Shaker Sort	11
3. Repositorio	12
4. Conclusión	13

1. Introducción

Los arreglos ordenados poseen distintos métodos para efectuar su ordenamiento, los tres más comunes es el algoritmo de burbuja mayor, el algoritmo de burbuja menor y el algoritmo de burbuja con señal. Estos algoritmos tratan de distintas maneras el ordenamiento de números en un arreglo.

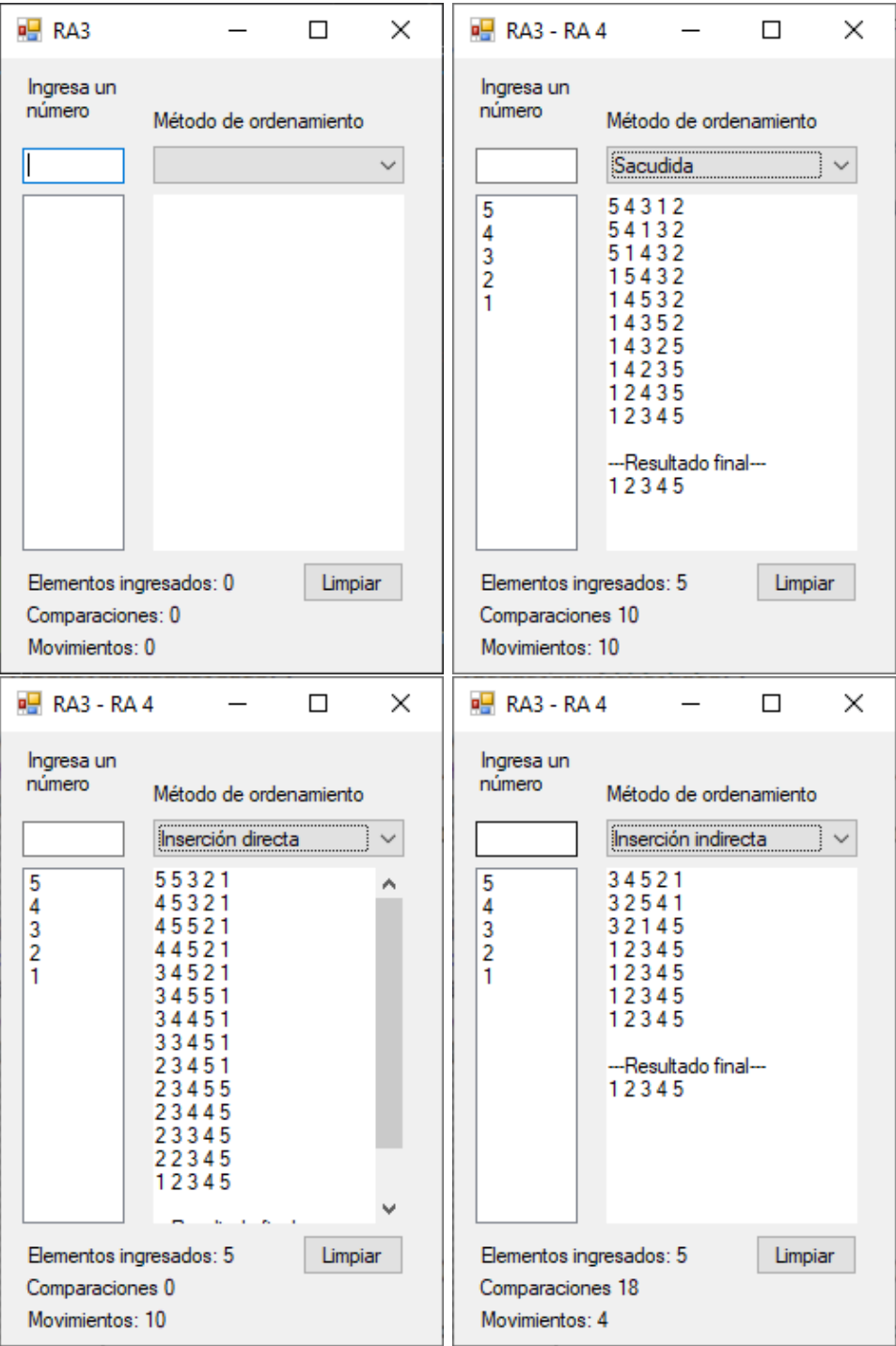
El lenguaje utilizado para este laboratorio es C#, lenguaje creado por Microsoft. Utilizamos en su misma medida .NET Framework con el IDE de Visual Studio 2019. Con el fin de aprender las herramientas que posee tanto el lenguaje con el entorno de desarrollo a la vez que se aprenden distintos algoritmos de manejo de datos.

2. Código Fuente


2.1. Tabla de herramientas - propiedades - eventos

Herramienta	Propiedades		Eventos
	Nombre	Texto	
label	label1	Ingrese un numero	
	label2	Metodo de ordenamiento	
	lblElements	Elementos ingresados: 0	
	lblComparisons	Comparaciones: 0	
	lblMovements	Movimientos: 0	
textbox	txtInput		txtInput_KeyPress
listbox	lbNumbers		
combobox	cbOrderMethods		cbOrderMethods_ItemSelectedChanged
richtextbox	txtOrder		
button	btnClean		btnOrder_Click

2.2. Interfaz gráfica



2.3. Variables globales

A code editor window with a dark background and three colored window control buttons (red, yellow, green) in the top left corner. It contains two lines of Java code.

```
1 int[] numbers;  
2 int n = 0, comparisons = 0, movements = 0;
```

2.4. Eventos

2.4.1. Evento de cargar formulario

```
1 private void RA3_Load(object sender, EventArgs e)
2 {
3     cbOrderMethods.Items.Add("Burbuja menor");
4     cbOrderMethods.Items.Add("Burbuja mayor");
5     cbOrderMethods.Items.Add("Burbuja por señal");
6     cbOrderMethods.Items.Add("Sacudida");
7     cbOrderMethods.Items.Add("Inserción directa");
8     cbOrderMethods.Items.Add("Inserción indirecta");
9     cbOrderMethods.DropDownStyle = ComboBoxStyle.DropDownList;
10 }
```

2.4.2. Evento KeyPress del TextBox

```
1 private void TxtInput_KeyPress(object sender, KeyPressEventArgs e)
2 {
3     if (e.KeyChar != 13)
4     {
5         txtInput.Clear();
6         return;
7     }
8
9     int newNumber;
10
11     if (!int.TryParse(txtInput.Text, out newNumber))
12         MessageBox.Show("No se ha ingresado un número");
13
14     else
15     {
16         lbNumbers.Items.Add(newNumber);
17         lblElements.Text = "Elementos ingresados: " + lbNumbers.Items.Count
18     ;
19         txtInput.Clear();
20         txtInput.Focus();
21     }
22 }
```

2.4.3. Evento del botón limpiar

```
1 private void BtnClean_Click(object sender, EventArgs e)
2 {
3     txtInput.Clear();
4     txtOrderResult.Clear();
5     lbNumbers.Items.Clear();
6     lblElements.Text = "Elementos ingresados: 0";
7     lblMovements.Text = "Movimientos: 0";
8     lblComparisons.Text = "Comparaciones: 0";
9 }
```

2.4.4. Evento al cambiar método de ordenamiento

```
1 private void CbOrderMethods_SelectedIndexChanged(object sender, EventArgs e)
2 {
3     SetNumbers();
4
5     if (cbOrderMethods.SelectedIndex == 0)
6         MinorBubble();
7     else if (cbOrderMethods.SelectedIndex == 1)
8         MajorBubble();
9     else if (cbOrderMethods.SelectedIndex == 2)
10        SignBubble();
11    else if (cbOrderMethods.SelectedIndex == 3)
12        ShakerSort();
13    else if (cbOrderMethods.SelectedIndex == 4)
14        DirectInsert();
15    else if (cbOrderMethods.SelectedIndex == 5)
16        Shell();
17
18    txtOrderResult.Text += "\n---Resultado final---\n";
19    Print();
20 }
```


2.5. Métodos

2.5.1. Método establecer arreglo

```
1 private void SetNumbers()
2 {
3     txtOrderResult.Clear();
4     n = comparisons = movements = 0;
5
6     foreach (int item in lbNumbers.Items)
7     {
8         Array.Resize(ref numbers, n + 1);
9         numbers[n] = item;
10        n++;
11    }
12 }
```

2.5.2. Método invertir posición

```
1 private void InvertPosition(int pos1, int pos2)
2 {
3     int temp;
4     movements++;
5     temp = numbers[pos1];
6     numbers[pos1] = numbers[pos2];
7     numbers[pos2] = temp;
8 }
```

2.5.3. Método para imprimir

```
1 private void Print()
2 {
3     foreach (int number in numbers)
4         txtOrderResult.Text += number + " ";
5     txtOrderResult.Text += "\r\n";
6
7     lblMovements.Text = "Movimientos: " + movements;
8     lblComparisons.Text = "Comparaciones " + comparisons;
9 }
```

2.5.4. Método Inserción directa

```
1 private void DirectInsert()
2 {
3     for(int i = 1; i < n; i++)
4     {
5         int temp = numbers[i];
6         int k = i - 1;
7
8         while((k ≥ 0) && (temp < numbers[k]))
9         {
10             numbers[k + 1] = numbers[k];
11             k = k - 1;
12             Print();
13             movements++;
14         }
15
16         numbers[k + 1] = temp;
17         Print();
18     }
19 }
```


2.5.5. Método Shell

```
1 private void Shell()
2 {
3     bool hasChange;
4     int inta = n;
5
6     while(inta > 0)
7     {
8         inta = (int)(inta / 2);
9         hasChange = true;
10        while(hasChange)
11        {
12            hasChange = false;
13            for(int i = 0; (i + inta) ≤ (n - 1); i++)
14            {
15                if (numbers[i] > numbers[i + inta])
16                {
17                    InvertPosition(i, i + inta);
18                    Print();
19                    hasChange = true;
20                }
21                comparisons++;
22            }
23        }
24        Print();
25    }
26
27 }
```

2.5.6. Método Shaker Sort

```
1 private void ShakerSort()
2 {
3     int left = 1, right = n - 1, k = right;
4
5     while(right ≥ left)
6     {
7         for(int i = right; i ≥ left; i--)
8         {
9             if(numbers[i - 1] > numbers[i])
10            {
11                InvertPosition(i - 1, i);
12                k = i;
13
14                Print();
15            }
16            comparisons++;
17        }
18        left = k + 1;
19        for (int i = left; i ≤ right; i++)
20        {
21            if (numbers[i - 1] > numbers[i])
22            {
23                InvertPosition(i - 1, i);
24                k = i;
25                Print();
26            }
27            comparisons++;
28        }
29        right = k - 1;
30    }
31 }
```

3. Repositorio

 Git: <https://github.com/include-minimaltools/AED>

4. Conclusión

Con esta práctica nos hemos introducido a nuevas herramientas del ambiente de .Net Framework como lo son los listbox, combobox, richtextbox y modificaciones en tiempo de ejecución en los labels para brindar una información directa al usuario e interactuar con él.

Hemos aprendido conceptos básicos de los algoritmos de ordenamiento de una serie de números aleatorias, así como la lógica funcional del programa, interactividad con eventos de formularios como «load», «click», «itemselectedchanged», «keypress» entre otros.