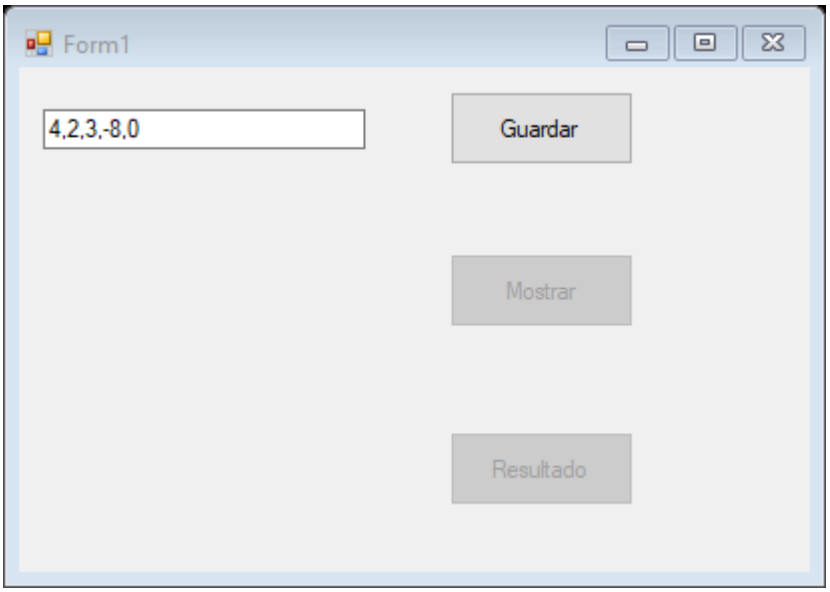
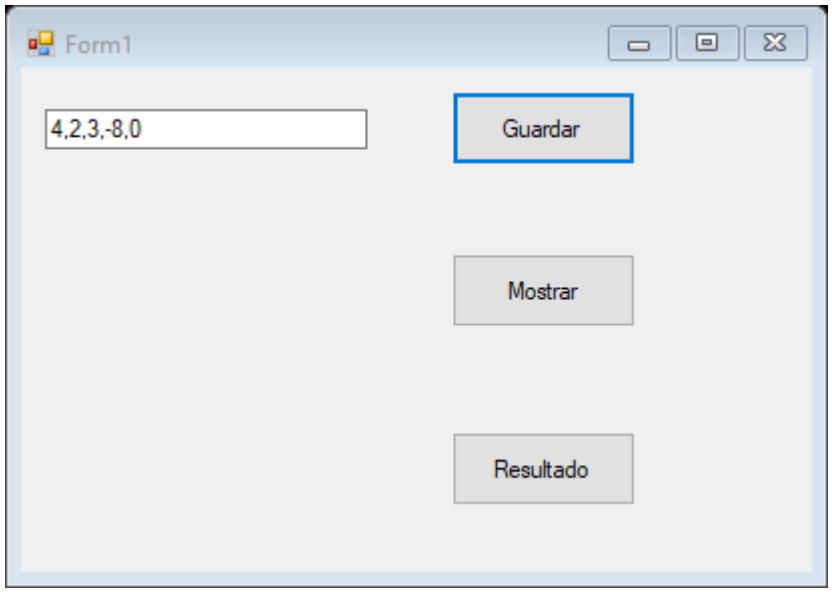
¿Qué es un FrameWork?

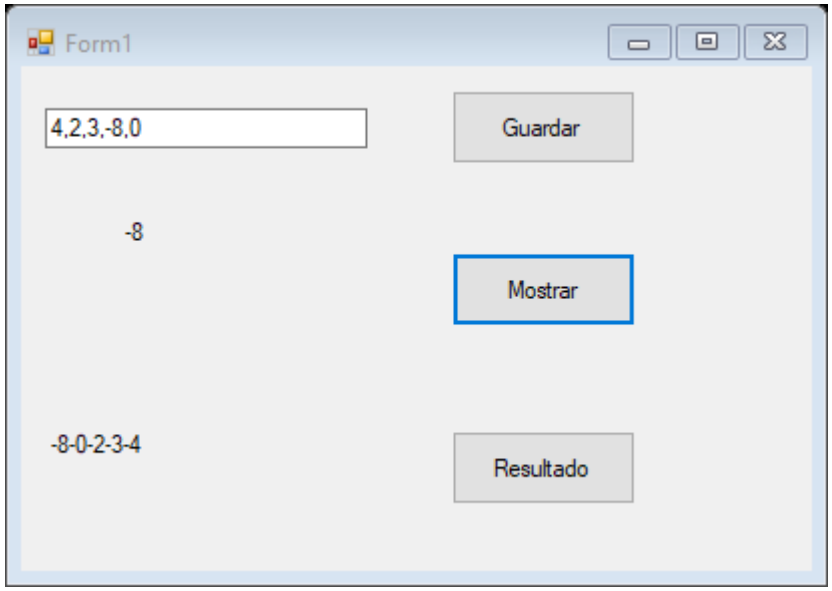
Para definir un FrameWork primero debemos de dar una traducción al termino y lo más correcto sería “Marco o Plantilla de Trabajo”. Se supone que un framework es todo el estructurado que existe detrás de una aplicación como lo pueden ser: Diagramas, ficheros, bases de datos, etc. Todo lo relacionado al desarrollo de un proyecto. Haciendo una analogía, un framework se podría comparar al esqueleto de un edificio en construcción.

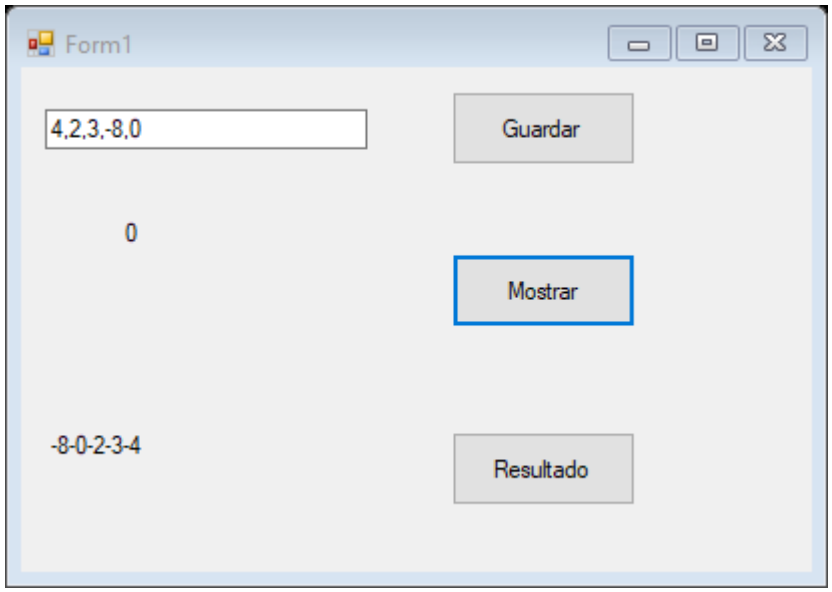
¿Qué es Visual Studio .NET?

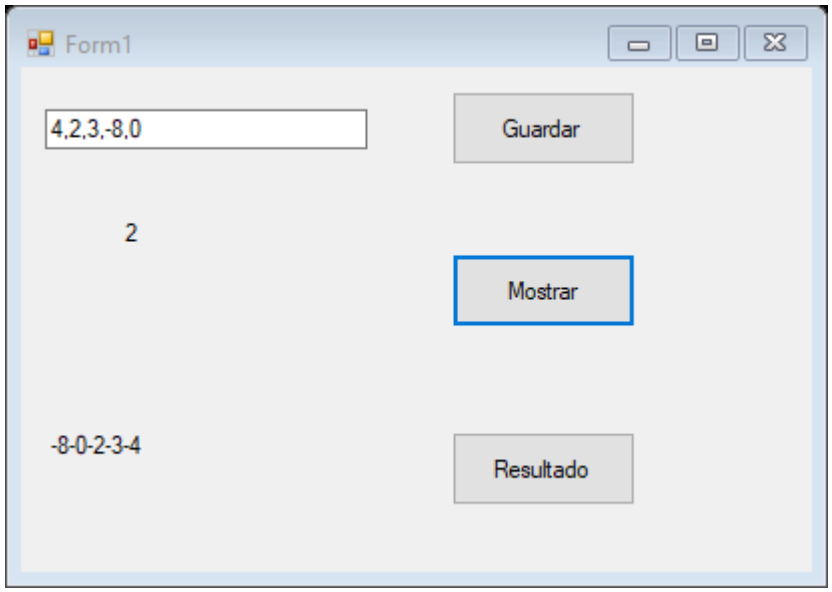
Visual Studio .NET desarrollado por Microsoft es una herramienta que permite crear aplicaciones o Servicios web basados en la implementación de XML (Extensible Markup Language). Utiliza el framework .NET el cual incluye distintos lenguajes para programación entre los cuales se encuentran: Visual Basic, Visual C++, Visual C# entre otras aplicaciones. Es considerado un IDE el cual puede ser operado de manera Visual o desde la Consola.

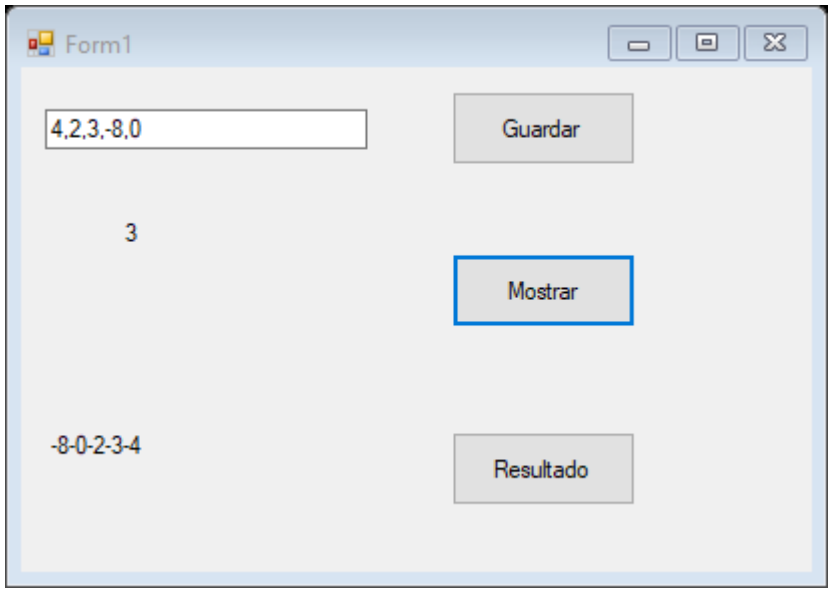


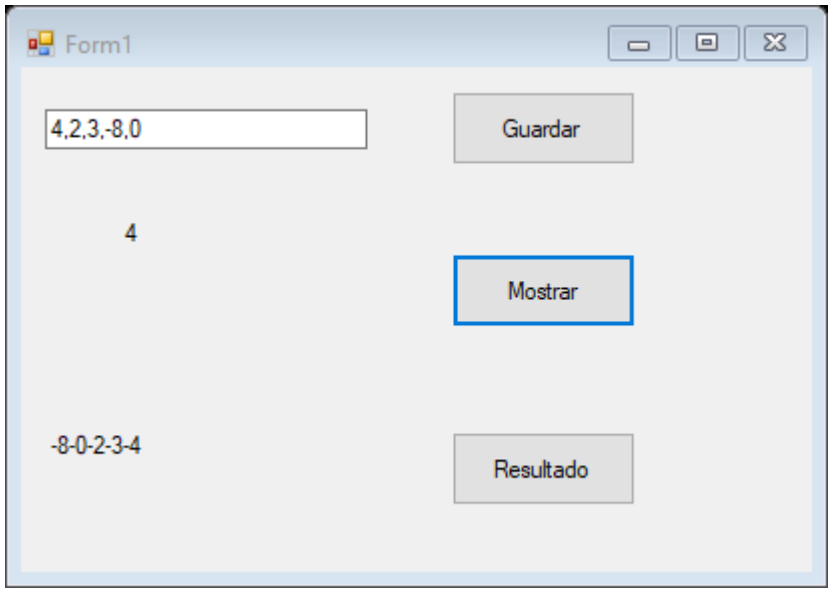

Form1
4,2,3,-8,0
Guardar
Mostrar
Resultado

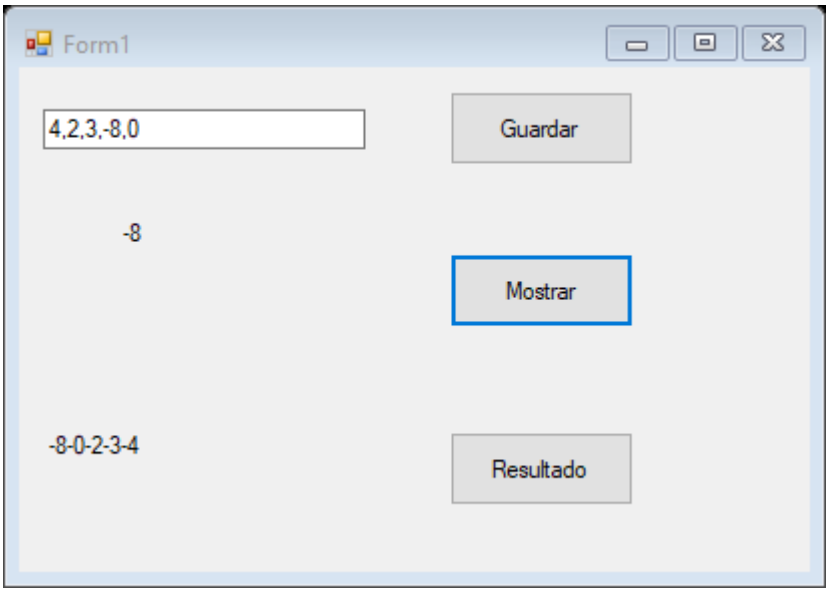

Form1
4,2,3,-8,0
Guardar
-8
Mostrar
-8-0-2-3-4
Resultado


Form1
4,2,3,-8,0
Guardar
0
Mostrar
-8-0-2-3-4
Resultado


Form1
4,2,3,-8,0
Guardar
2
Mostrar
-8-0-2-3-4
Resultado


Form1
4,2,3,-8,0
Guardar
3
Mostrar
-8-0-2-3-4
Resultado


Form1
4,2,3,-8,0
Guardar
4
Mostrar
-8-0-2-3-4
Resultado




```
14  public partial class Form1 : Form
15  {
16      int[] values; //Aca se almacenan los numeros
17      int iterator = 0;//Iterador para recorrer la matriz values
18      public Form1()
19      {
20          InitializeComponent();
21      }
22      //Boton Guardar
23      private void button1_Click(object sender, EventArgs e)
24      {
25          label2.Text = "";//Limpia el texto
26          label1.Text = "";//Limpia el texto
27          iterator = 0;//Regresa el iterador a 0
28          int written = 0;//Datos escritos en la matriz de numeros
29          String text = maskedTextBox1.Text;//Obtiene el texto
30          if (!text.Equals(""))
31          {
32              String[] characters = text.Split(',');//Guarda las cadenas exceptuando las ','
33              int internalIterator = 0;//Crea e inicializa un contador interno
34              for (int i = 0; i < characters.Length; i++)//Recorre la matriz
35              {
36                  var pattern = @"^(-?[1-9]+\d*([.]\d+)?)$|^(-?0[.]\d*[1-9]+)$|^0$|^0.0$";//Crea patron numerico
37                  if (Regex.Match(characters[i], pattern, RegexOptions.IgnoreCase).Success)//Verifica si la cadena cumple el patron
38                  {
39                      internalIterator++;//Aumenta el iterador interno
40                  }
41              }
42              int quantity = internalIterator;//Obtiene el numero de cadenas almacenadas
43              values = new int[quantity];//Inicializa la matriz y le da un tamaño
44              for (int i = 0; i < characters.Length; i++)//For utilizado para recorrer la matriz de cadenas
45              {
46                  var pattern = @"^(-?[1-9]+\d*([.]\d+)?)$|^(-?0[.]\d*[1-9]+)$|^0$|^0.0$";//Crea patron numerico
47                  if (Regex.Match(characters[i], pattern, RegexOptions.IgnoreCase).Success)//Verifica si la cadena cumple el patron
```

```
            for (int i = 0; i < characters.Length; i++)//For utilizado para recorrer la matriz de cadenas
            {
                var pattern = @"^(-?[1-9]+\d*([.]\d+)?)$|^(-?0[.]\d*[1-9]+)$|^0$|^0.0$";//Crea patron numerico
                if (Regex.Match(characters[i], pattern, RegexOptions.IgnoreCase).Success)//Verifica si la cadena cumple el patron
                {
                    values[written] = Int32.Parse(characters[i]);//Agrega el numero a la matriz de numeros
                    written++;//Se aumenta el numero de datos escritos
                }
            }
            button3.Enabled = true;//Activa el boton de Resultado
            button2.Enabled = true;//Activa el boton de Mostrar
        }
    }

    //Metodo para ordenar la matriz de numeros
    void Ordenar(int[] numbers)
    {
        for (int i = 0; i < numbers.Length - 1; i++)//Se recorre la matriz
        {
            for (int j = 0; j < numbers.Length - 1; j++)//Se recorre la matriz
            {
                if (numbers[j] > numbers[j + 1])//Comprueba si el actual es mayor que el siguiente
                {
                    int aux = numbers[j + 1];//Se crea una variable auxiliar donde se guarda el siguiente
                    numbers[j + 1] = numbers[j];//A la posicion siguiente se le da el valor de la posicion actual
                    numbers[j] = aux;//La posicion actual recibe el valor del auxiliar
                }
            }
        }
        values = numbers;//Se reescribe la matriz de numeros con los valores ya ordenados
    }
```

```
            values = numbers;//Se reescribe la matriz de numeros con los valores ya ordenados
        }
        //Boton Resultado
        private void button3_Click(object sender, EventArgs e)
        {
            Ordenar(values);//Se ordenan los valores de la matriz numeros
            label2.Text = "";//Limpia el texto
            for (int i = 0; i < values.Length; i++)//Se recorre la matriz
            {
                if (label2.Text.Equals(""))//Verifica si es el primer numero en ingresarse al label
                {
                    label2.Text = values[i].ToString();//Se ingresa el numero al label
                }
                else
                {
                    label2.Text = label2.Text + "-" + values[i].ToString();//Concatena el valor del label con el nuevo numero
                }
            }
        }
        //Boton Mostrar
        private void button2_Click(object sender, EventArgs e)
        {
            if (iterator < values.Length)//Verifica que el iterador no desborde la matriz
            {
                label1.Text = values[iterator].ToString();//Escribe en el label el valor de la posicion
                iterator++;//Incrementa el iterador
            }
            else
            {
                iterator = 0;//Regresa el iterador a su posicion inicial
                label1.Text = values[iterator].ToString();//Escribe en el label el valor de la posicion
                iterator++;//Incrementa el iterador
            }
        }
    }
```