# SOLUTION OF THE PARTY OF THE PA

## UNIVERSIDAD CATÓLICA DE EL SALVADOR

FACULTAD DE INGENIERÍA Y ARQUITECTURA INGENIERÍA EN SISTEMAS INFORMÁTICOS

CICLO: I - 2020

MATERIA: PROGRAMACIÓN II

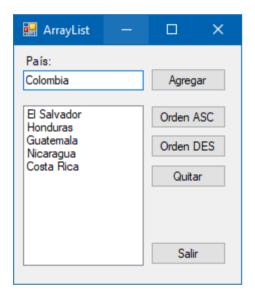
DOCENTE: ING. GIOVANNI ACOSTA HENRIQUEZ (giovanni.acosta@catolica.edu.sv)

# PRÁCTICA 6: COLECCIONES, ESTRUCTURAS Y ENUMERACIONES

## **Objetivos:**

- ✓ Almacenar datos en colecciones: ArrayList, List y Dictionary
- ✓ Crear estructuras de datos y enumeraciones
- ✓ Desarrollar aplicaciones utilizando colecciones, estructuras de datos y enumeraciones
- 1. Ejecutar Visual Studio .NET
- 2. Crear un nuevo proyecto de tipo Aplicación de Windows Forms (Windows Form App)
- 3. Crear un formulario para cada uno de los siguientes ejemplos.

# Ejemplo 1: uso de colección ArrayList

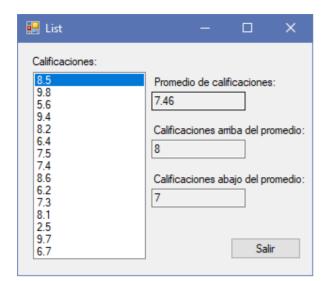


Control	Name	Text
Form	frmArrayList	
textBox1	txtPais	
listBox1	lstPaises	
button1	btnAgregar	Agregar
button2	btnOrdenAsc	Orden ASC
button3	btnOrdenDes	Orden DES
button4	btnQuitar	Quitar
button5	btnSalir	Salir

Crear el ArrayList en las declaraciones generales, fuera de los métodos:

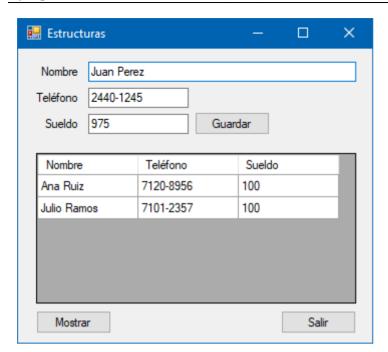
```
private static ArrayList paises = new ArrayList();
5 references
public static void LlenarLista(ListBox lista)
    lista.Items.Clear();
    foreach (string pais in paises)
        lista.Items.Add(pais);
    }
}
1 reference
private void frmArrayList Load(object sender, EventArgs e)
{
    paises.Add("El Salvador");
    paises.Add("Honduras");
    paises.Add("Guatemala");
    paises.Add("Nicaragua");
    paises.Add("Costa Rica");
    LlenarLista(lstPaises);
}
```

```
private void btnAgregar_Click(object sender, EventArgs e)
    if (txtPais.Text != String.Empty)
    {
        paises.Add(txtPais.Text);
        LlenarLista(lstPaises);
        txtPais.Clear();
        txtPais.Focus();
    }
}
private void btnOrdenAsc_Click(object sender, EventArgs e)
{
    if (lstPaises.Items.Count>1)
        paises.Sort();
        LlenarLista(lstPaises);
    }
}
private void btnOrdenDes_Click(object sender, EventArgs e)
    if (lstPaises.Items.Count > 1)
    {
        paises.Sort();
        paises.Reverse();
        LlenarLista(lstPaises);
    }
}
private void btnQuitar_Click(object sender, EventArgs e)
    if (lstPaises.SelectedIndex >= 0)
    {
        lstPaises.Items.Remove(lstPaises.SelectedIndex);
        paises.RemoveAt(lstPaises.SelectedIndex);
        LlenarLista(lstPaises);
    }
}
```



Control	Name	ReadOnly
Form	frmList	
listBox1	lstCalificaciones	
textBox1	txtPromedio	true
textBox2	txtArriba	true
textBox3	txtAbajo	true
button1	btnSalir	

```
private double ArribaPromedio(List<double> notas, double promedio)
    int cant = 0;
    foreach (double nota in notas)
        if (nota > promedio) cant++;
    return cant;
}
private double AbajoPromedio(List<double> notas, double promedio)
    int cant = 0;
    foreach (double nota in notas)
        if (nota < promedio) cant++;</pre>
    return cant;
}
private void frmList_Load(object sender, EventArgs e)
    List<double> calificaciones = new List<double> { 8.5, 9.8, 5.6, 9.4, 8.2, 6.4,
      7.5, 7.4, 8.6, 6.2, 7.3, 8.1, 2.5, 9.7, 6.7 };
    lstCalificaciones.DataSource = calificaciones;
    double prom = calificaciones.Average();
    txtPromedio.Text = prom.ToString("N2");
    txtArriba.Text = ArribaPromedio(calificaciones, prom).ToString();
    txtAbajo.Text = AbajoPromedio(calificaciones, prom).ToString();
}
```



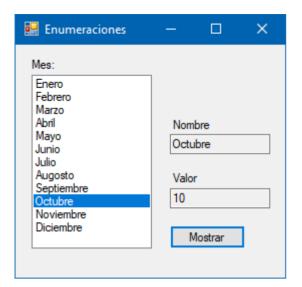
Control	Name	Text
Form	frmStruct	
TextBox1	txtNombre	
TextBox2	txtTelefono	
TextBox3	txtSueldo	
Button1	btnGuardar	Guardar
Button2	btnMostrar	Mostrar
Button3	btnSalir	Salir
DataGridView	dgvEmpleados	

Crear la estructura **Empleado** y la colección (List) **listaEmpleados** en las declaraciones generales, fuera de los métodos:

```
public struct Empleado
{
    public string nombre;
    public string telefono;
    private decimal sueldo;
    private const decimal bono = 100;
    1reference
    public void setSueldo(decimal sueldo)
    {
        if (sueldo >= 0)
        {
            this.sueldo = sueldo;
        }
    }
    1reference
    public decimal getSueldo()
    {
        return sueldo + bono;
    }
}
```

```
public static List<Empleado> listaEmpleados = new List<Empleado>();
```

```
private void btnGuardar_Click(object sender, EventArgs e)
  try
   {
        Empleado empleado = new Empleado();
        empleado.nombre = txtNombre.Text;
        empleado.telefono = txtTelefono.Text;
        empleado.setSueldo(Convert.ToDecimal(txtSueldo.Text));
        listaEmpleados.Add(empleado);
        txtNombre.Clear();
        txtTelefono.Clear();
        txtSueldo.Clear();
        txtNombre.Focus();
   catch (Exception ex)
        MessageBox.Show(ex.Message);
private void btnMostrar_Click(object sender, EventArgs e)
   dgvEmpleados.Rows.Clear();
   foreach (Empleado empleado in listaEmpleados)
        dgvEmpleados.Rows.Add(empleado.nombre, empleado.telefono, empleado.getSueldo());
   dgvEmpleados.ClearSelection();
```



Control	Name	Text
ListBox	lstMeses	
TextBox1	txtNombre	
TextBox2	txtValor	
Button1	btnMostrar	Mostrar

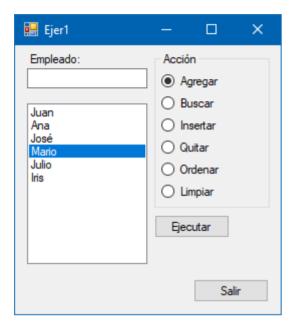
Crear la enumeración Meses en las declaraciones generales, fuera de los métodos:

```
public enum Meses
                              private void frmEnum Load(object sender, EventArgs e)
    Enero = 1,
                                   foreach (var mes in Enum.GetValues(typeof(Meses)))
    Febrero = 2,
                                   {
    Marzo = 3,
                                       lstMeses.Items.Add(mes);
    Abril = 4,
    Mayo = 5,
    Junio = 6,
    Julio = 7,
    Augosto = 8,
    Septiembre = 9,
    Octubre = 10,
    Noviembre = 11,
    Diciembre = 12
private void btnMostrar_Click(object sender, EventArgs e)
   if (lstMeses.SelectedIndex>=0)
       Meses mes = (Meses)Enum.ToObject(typeof(Meses), lstMeses.SelectedIndex + 1);
        int valor = (int)mes;
        txtNombre.Text = mes.ToString();
        txtValor.Text = valor.ToString();
```

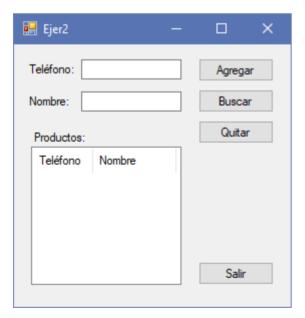


Indicaciones: agregar a la solución un proyecto de C# de tipo Aplicación Windows Forms y programar los siguientes formularios.

1. Desarrollar una aplicación que almacene en una lista (**List**) los nombres de empleados, que permita realizar las siguientes acciones sobre la lista: agregar un nombre al final de la lista, buscar el nombre especificado, insertar un nombre en la posición seleccionada en la lista, quitar el nombre seleccionado, ordenarla ascendentemente y limpiar toda la lista de empleados.



2. Crear una aplicación un diccionario (Clave / Valor) para almacenar el teléfono (Clave) y el nombre (Valor) de los clientes, para ser mostrados un **ListView**, programar opciones para: Agregar, Buscar y Quitar elementos del diccionario.



- 3. Crear una aplicación, para almacenar estructuras **Empleado** (NIT, nombre y sueldo) en un diccionario con clave (NIT) y con la estructura Empleado como valor. Según los siguientes requerimientos:
  - Botón **Agregar**: validar que los textbox: NIT, nombre y sueldo no se encuentren vacíos, si el NIT no se encuentra en el diccionario, será agregado al diccionario y al DataGridView, además actualizar el total de la planilla, en caso contrario un mensaje de error.
  - Botón **Buscar**: si el NIT se encuentra, deberá mostrar un MessageBox con los datos del empleado o en caso contrario un mensaje de error.
  - Botón **Quitar**: si el NIT se encuentra, deberá quitarlo del diccionario y del DataGridView o en caso contrario un mensaje de error.

