UNIVERSIDAD NACIONAL DE INGENIERÍA FACULTAD DE ELECTROTECNIA Y COMPUTACIÓN

REPORTE DE LABORATORIO Nº2

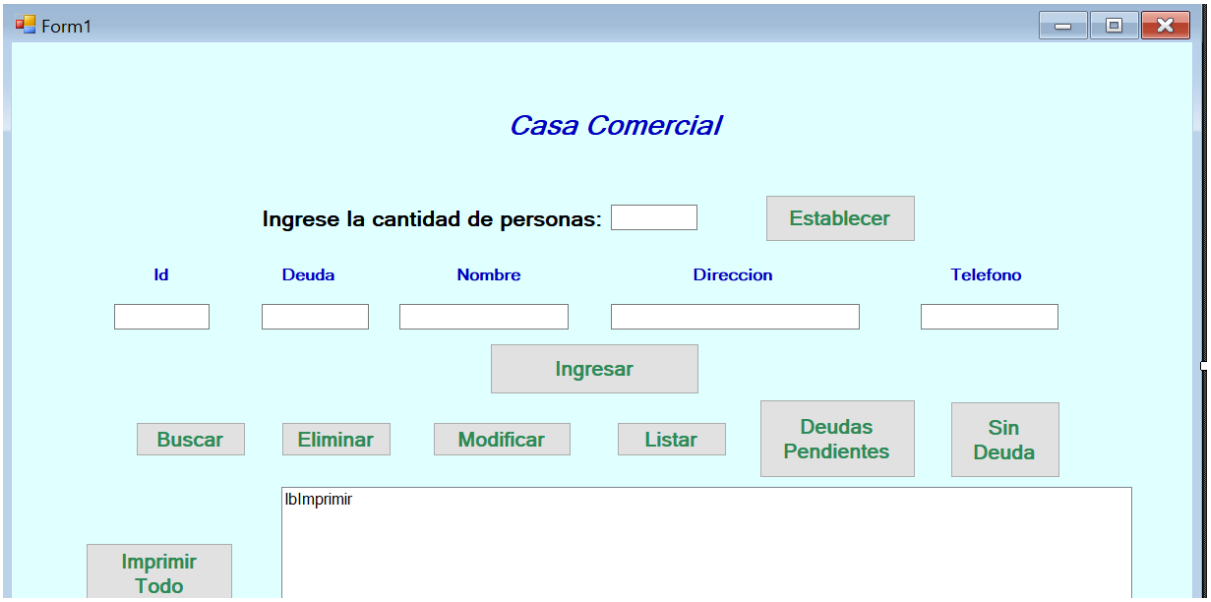
ALGORITMIZACIÓN Y ESTRUCTURA DE DATOS

PROF. ADILSON GONZALEZ

NOMBRE:

- **ALANIS DEL CARMEN TOBIAS SOTELO 2021-0148U**
- **MADELING REGINA CABRERA ROJAS 2021-0257U**

FECHA:
30/08/2023


Form1
Casa Comercial
Ingrese la cantidad de personas:
Establecer
Id
Deuda
Nombre
Direccion
Telefono
Ingresar
Buscar
Eliminar
Modificar
Listar
Deudas Pendientes
Sin Deuda
lblImprimir
Imprimir Todo

```
internal class CasaComercial
{
    9 referencias
    public int Id { get; set; }
    10 referencias
    public int Deuda { get; set; }
    8 referencias
    public string Nombre { get; set; }
    8 referencias
    public string Direccion { get; set; }
    8 referencias
    public int Telefono { get; set; }
    2 referencias
    public string EdD (int a)
    {
        if (a == 0)
            return "Pagada";
        else
            return "No pagada";
    }
    2 referencias
    public string Imprimir()
    {
        return Id + "\t" + Deuda + "\t" + Nombre + "\t" + Direccion + "\t" + Telefono;
    }

}
```

```
public partial class Form1 : Form
{
    1 referencia
    public Form1()
    {
        InitializeComponent();
    }
    //Declaraciones Globales
    CasaComercial[] Instancia;

    int Tam, N = 0, i, Pos, x, x2;

    2 referencias
    void Limpiar()
    {
        txtId.Clear();
        txtDeuda.Clear();
        txtNombre.Clear();
        txtDireccion.Clear();
        txtTelefono.Clear();
        txtId.Focus();
    }
```

```
public int Buscar(int x)
{
    i = 0;
    while (i < N && Instancia[i].Id < x)
    {
        i = i + 1;
    }
    if (i >= N || Instancia[i].Id < x)
    {
        Pos = -i;
    }
    else
    {
        Pos = i;
    }
    return Pos;
}
1 referencia
void Eliminar(int x)
{
    if (N > 0)
    {
        Pos = Buscar(x);
        MessageBox.Show(x + Pos.ToString());
        if (Pos <= -1)
        {
            MessageBox.Show("La persona con el Id: " + x + " no se encuentra registrada");
        }
        else
        {
            for (i = Pos; i < N - 1; i++)
```

```
void Eliminar(int x)
{
    if (N > 0)
    {
        Pos = Buscar(x);
        MessageBox.Show(x + Pos.ToString());
        if (Pos <= -1)
        {
            MessageBox.Show("La persona con el Id: " + x + " no se encuentra registrada");
        }
        else
        {
            for (i = Pos; i < N - 1; i++)
            {
                Instancia[i].Id = Instancia[i + 1].Id;
                Instancia[i].Deuda = Instancia[i + 1].Deuda;
                Instancia[i].Nombre = Instancia[i + 1].Nombre;
                Instancia[i].Direccion = Instancia[i + 1].Direccion;
                Instancia[i].Telefono = Instancia[i + 1].Telefono;
            }
            N = N - 1;
            MessageBox.Show("La persona con el Id: " + x + " se ha eliminado");
        }
    }
    else
    {
        MessageBox.Show("No hay registros");
    }
}
```

```
void Insertar(int x)
{
    if (N > 0)
    {
        if (N <= Tam - 1)
        {
            Pos = Buscar(x);
            if (Pos > 0)
            {
                MessageBox.Show("El elemento ya existe");
            }
            else
            {
                Pos = Pos * -1;
                for (i = N; i >= Pos +1; i--)
                {
                    Instancia[i] = new CasaComercial();
                    Instancia[i].Id = Instancia[i - 1].Id;
                    Instancia[i].Deuda = Instancia[i - 1].Deuda;
                    Instancia[i].Nombre = Instancia[i - 1].Nombre;
                    Instancia[i].Direccion = Instancia[i - 1].Direccion;
                    Instancia[i].Telefono = Instancia[i - 1].Telefono;

                }

                Instancia[Pos] = new CasaComercial();
                Instancia[Pos].Id = int.Parse(txtId.Text);
                Instancia[Pos].Deuda = int.Parse(txtDeuda.Text);
                Instancia[Pos].Nombre = txtNombre.Text;
                Instancia[Pos].Direccion = txtDireccion.Text;
                Instancia[Pos].Telefono = int.Parse(txtTelefono.Text);
```

```
            }
        }
        else
        {
            MessageBox.Show("No hay espacio");
        }
    }
    else
    {

        Instancia[N] = new CasaComercial();
        Instancia[N].Id = int.Parse(txtId.Text);
        Instancia[N].Deuda = int.Parse(txtDeuda.Text);
        Instancia[N].Nombre = txtNombre.Text;
        Instancia[N].Direccion = txtDireccion.Text;
        Instancia[N].Telefono = int.Parse(txtTelefono.Text);
        N = N + 1;
        MessageBox.Show("Persona Insertada");
    }
}
```

1 referencia

```
private void btnImprimir_Click(object sender, EventArgs e)
{
    lbImprimir.Items.Clear();
    for (i = 0; i < N; i++)
    {
        string E = Instancia[i].EdD(Instancia[i].Deuda);
        lbImprimir.Items.Add(Instancia[i].Imprimir() + "\t" + E);

    }
}
```

```
1 referencia
private void Listar_Click(object sender, EventArgs e)
{
    lbImprimir.Items.Clear();
    string x = txtId.Text;


    if (N < 0)
    {
        Pos = Buscar(int.Parse(x));
        if (Pos <= -1)
        {
            MessageBox.Show("Persona no registrada");
        }
        else
        {
            for (i = Pos; i < N - 1; i++)
            {
                string E = Instancia[i].EdD(Instancia[i].Deuda);
                lbImprimir.Items.Add(Instancia[i].Imprimir() + "\t" + E);
            }
        }
    }
    else
    {
        MessageBox.Show("Persona no registrada");
    }
```

```
1 referencia
private void btnEstablecer_Click(object sender, EventArgs e)
{
    Tam = int.Parse(txtEstablecer.Text);
    Instancia = new CasaComercial[Tam];
    N = 0;

}

1 referencia
private void brnBuscar_Click(object sender, EventArgs e)
{
    if (N > 0)
    {
        x2 = int.Parse(txtId.Text);
        Pos = Buscar(x2);
        if (Pos <= -1)
        {
            MessageBox.Show("La persona con el Id: " + x2 + " no esta registrada");
        }
        else
        {
            txtDeuda.Text = Convert.ToString(Instancia[Pos].Deuda);
            txtNombre.Text = Instancia[Pos].Nombre;
            txtDireccion.Text = Instancia[Pos].Direccion;
            txtTelefono.Text = Convert.ToString(Instancia[Pos].Telefono);
        }
    }
    else
    {
        MessageBox.Show("No hay registros");
    }
```

```
1 referencia
private void btnEliminar_Click(object sender, EventArgs e)
{
    x = int.Parse(txtId.Text);
    Eliminar(x);
    Limpiar();
}

1 referencia
private void btnInsertar_Click(object sender, EventArgs e)
{
    x = int.Parse(txtId.Text);
    Insertar(x);
    Limpiar();
}
```

EJERCICIO 2

```
public string Carnet { get; set; }
2 referencias
public string Nombre { get; set; }
2 referencias
public string Apellido { get; set; }
2 referencias
public string Proyecto { get; set; }
10 referencias
public DateTime FechaEntrega { get; set; }
2 referencias
public DateTime FechaLimite { get; set; }
2 referencias
public int Calificacion { get; set; }

1 referencia
public Entrega(string carnet, string nombre, string apellido, string proyecto, DateTime fechaEntrega, DateTime fechaLimite, int calificacion)
{
    Carnet = carnet;
    Nombre = nombre;
    Apellido = apellido;
    Proyecto = proyecto;
    FechaEntrega = fechaEntrega;
    FechaLimite = fechaLimite;
    Calificacion = calificacion;
}
```

```
namespace EntregaEstudiantes
{
    3 referencias
    public partial class Form1 : Form
    {

        1 referencia
        public Form1()
        {
            InitializeComponent();
        }
        // Declaraciones globales para los arreglos y el contador
        Entrega[] entregas;
        int N;
        int a;

        4 referencias
        void Imprimir()
        {
            dgElementos.DataSource = null;
            dgElementos.DataSource = entregas;
        }
```

```
private int BuscarPosicionInsercion(Entrega nuevaEntrega)
{
    int posicionInsercion = 0;
    while (posicionInsercion < N && nuevaEntrega.FechaEntrega > entregas[posicionInsercion].FechaEntrega)
    {
        posicionInsercion++;
    }

    // Si se encontró una coincidencia, devuelve -1 para indicar que el elemento ya existe
    if (posicionInsercion < N && nuevaEntrega.FechaEntrega == entregas[posicionInsercion].FechaEntrega)
    {
        return -1;
    }

    return posicionInsercion;
}
```

```
private void btnIngresar_Click(object sender, EventArgs e)
{
        // Crear una nueva entrega con los datos ingresados
        Entrega nuevaEntrega = new Entrega(txtCarnet.Text, txtNombre.Text, txtApellido.Text, txtProyecto.Text, DateTime.Parse(dtFechaLim.Text), DateTime.Parse(dtFechaEntrega.

        // Encontrar la posición donde se debe insertar la nueva entrega
        int posicionInsercion = 0;
        while (posicionInsercion < N && nuevaEntrega.FechaEntrega > entregas[posicionInsercion].FechaEntrega)
        {
            posicionInsercion++;
        }

        // Desplazar elementos para hacer espacio para la nueva entrega
        Array.Resize(ref entregas, N + 1);
        for (int i = N; i > posicionInsercion; i--)
        {
            entregas[i] = entregas[i - 1];
        }

        // Insertar la nueva entrega en la posición adecuada
        entregas[posicionInsercion] = nuevaEntrega;
        N++;

        // Actualizar la visualización y limpiar los campos
        Imprimir();
        Limpiar();
```

```
1 referencia
private void dgElementos_CellClick(object sender, DataGridViewCellEventArgs e)
{
    a = e.RowIndex;
    txtCarnet.Text = dgElementos.Rows[a].Cells[0].Value.ToString();
    txtNombre.Text = dgElementos.Rows[a].Cells[1].Value.ToString();
    txtApellido.Text = dgElementos.Rows[a].Cells[2].Value.ToString();
    txtProyecto.Text = dgElementos.Rows[a].Cells[3].Value.ToString();
    dtFechaLim.Text = dgElementos.Rows[a].Cells[4].Value.ToString();
    dtFechaEntrega.Text = dgElementos.Rows[a].Cells[5].Value.ToString();
    txtCalificacion.Text = dgElementos.Rows[a].Cells[6].Value.ToString();
}
```

```
1 referencia
private void btnEliminar_Click(object sender, EventArgs e)
{
    if (a >= 0 && a < N)
    {
        string carnetAEliminar = entregas[a].Carnet;

        // Encuentra la posición del elemento a eliminar en función del número de carnet
        int posicionEliminar = -1;
        for (int i = 0; i < N; i++)
        {
            if (entregas[i].Carnet == carnetAEliminar)
            {
                posicionEliminar = i;
                break; // Salir del bucle una vez que se encuentre la coincidencia
            }


        }

        if (posicionEliminar >= 0)
        {
            // Mueve los elementos posteriores un lugar hacia atrás
            for (int k = posicionEliminar; k < N - 1; k++)
            {
                entregas[k] = entregas[k + 1];
            }

            N--;
```

```
            N--;
            Array.Resize(ref entregas, N);
            Imprimir();
            Limpiar();
        }

    }
    else
    {
        MessageBox.Show("Selecciona una fila válida para eliminar.", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}

1 referencia
private void btnModificar_Click(object sender, EventArgs e)
{
    entregas[a].Carnet = txtCarnet.Text;
    entregas[a].Nombre = txtNombre.Text;
    entregas[a].Apellido = txtApellido.Text;
    entregas[a].Proyecto = txtProyecto.Text;
    entregas[a].FechaLimite = DateTime.Parse(dtFechaLim.Text);
    entregas[a].FechaEntrega = DateTime.Parse(dtFechaEntrega.Text);
    entregas[a].Calificacion = int.Parse(txtCalificacion.Text);
    Array.Resize(ref entregas, N);
    Imprimir();
    Limpiar();
}
```

```
private void btnListar_Click(object sender, EventArgs e)
{
    Limpiar();
    string carnetABuscar = txtCarnet.Text; // Obtén el número de carnet a buscar

    // Encuentra la posición del elemento a listar en función del número de carnet
    int posicionListar = -1;
    for (int i = 0; i < N; i++)
    {
        if (entregas[i].Carnet == carnetABuscar)
        {
            posicionListar = i;
            break; // Salir del bucle una vez que se encuentre la coincidencia
        }
    }

    if (posicionListar >= 0)
    {
        // Limpia el DataGridView antes de listar el elemento
        dgElementos.DataSource = null;

        // Crea un arreglo de una sola entrega con el elemento a listar
        Entrega[] entregaListada = new Entrega[] { entregas[posicionListar] };

        // Asigna el arreglo al DataGridView para mostrar el elemento
        dgElementos.DataSource = entregaListada;
    }
    else
    {
        MessageBox.Show("No se encontró la entrega con el número de carnet especificado.", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
    Limpiar();
}

1 referencia
private void btnListarTodo_Click(object sender, EventArgs e)
{
    Imprimir();
}

1 referencia
private void btnEntrgasTardias_Click(object sender, EventArgs e)
{
    // Obtén la fecha límite a partir del DateTimePicker
    DateTime fechaLimite = DateTime.Parse(dtFechaLim.Text);

    // Filtra las entregas tardías en un nuevo arreglo
    Entrega[] entregasTardias = entregas
        .Where(entrega => entrega.FechaEntrega > fechaLimite && entrega.FechaEntrega.Date != fechaLimite.Date)
        .ToArray();ss

    // Limpia el DataGridView antes de mostrar las entregas tardías
    dgElementos.DataSource = null;

    // Asigna el arreglo de entregas tardías al DataGridView
    dgElementos.DataSource = entregasTardias;
}
```