

TAREA 5

CAPACITACIÓN C#

 **Ezequiel Remus**
ezequielremus@gmail.com

30 de marzo de 2023

ENUNCIADO

Keywords First keyword · Second keyword · More

Tarea 5: Parte 1

1. Modificar la Clase Profesor

1.1. Enunciado

En la carpeta Clases modificar la Clase Profesor con los métodos realizados en la tarea parte uno de la clase anterior. Debería quedar la clase con todos los métodos para agregarlo, modificarlo y eliminarlo de la base de datos

1.2. Solución

Lo primero que tenemos que hacer es agregar las funciones en la clase Profesor.cs. En este caso, también agregue las funciones GetProfesorPorId() y GetTodosLosProfesor().

Teniendo minimizadas las Funciones el archivo queda así

```
1  /// <summary> Agrega en la base de datos los datos pasados correspondientes a un ...  
2  1 referencia  
3  public int AgregarEnDb()  
4  
5  /// <summary> Funcion que actualiza los datos de un profesor en la base de datos  
6  0 referencias  
7  public int ActualizarEnDb(Profesor profesorAActualizar)  
8  
9  /// <summary> Elimina los datos de un profesor en la base de datos  
10 1 referencia  
11 public int EliminarEnDb(int idProfesorEliminar)  
12  
13 /// <summary> Hace un Get de los datos del profesor segun el id Correspondiente  
14 0 referencias  
15 public Profesor GetProfesorPorId(int profesorId)  
16  
17 /// <summary> Toma a todos los profesores existentes en la base de datos  
18 1 referencia  
19 public List<Profesor> GetTodosLosProfesores()
```

Figura 1: Métodos de la clase Profesor

2. ProfesoresForm

2.1. Enunciado

Crear el formulario ProfesoresForm.

2.2. Solución

En la carpeta **Vistas** agregamos el formulario **ProfesoresForm()**. Quedando El árbol como sigue:

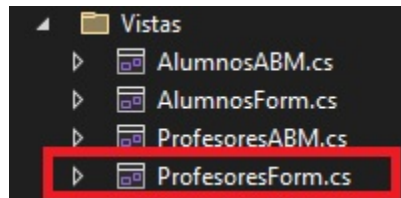


Figura 2: Estructura de la carpeta Vistas

3. Agregar un evento al botón Profesores

3.1. Enunciado

Agregar un evento al botón **Profesores** del formulario **FormInicio** que abra el formulario del punto anterior

3.2. Solución

Ahora, enlazamos el botón del **FormInicio** Con el **ProfesoresForm** mediante el método *Show*

```
1 referencia
private void button2_Click(object sender, EventArgs e)
{
    ProfesoresForm profesoresForm = new ProfesoresForm();
    profesoresForm.Show();
}
```

Figura 3: Evento en FormInicio

4. Agregar al formulario ProfesoresForm

4.1. Enunciado

Agregar al formulario **ProfesoresForm**

1. Un *botón* **Agregar Profesor**
2. Un *label* con el texto **Id Profesor**
3. Agregar un *Textbox* con el (name) **txtIdProfesor**
4. Un *botón* **Modificar Profesor**
5. Un *botón* **Eliminar Profesor**
6. Agregar un control *GridView* con el (name) **gvProfesores**

4.2. Solución

Agregando todos los elementos indicados anteriormente, el diseño del formulario nos queda

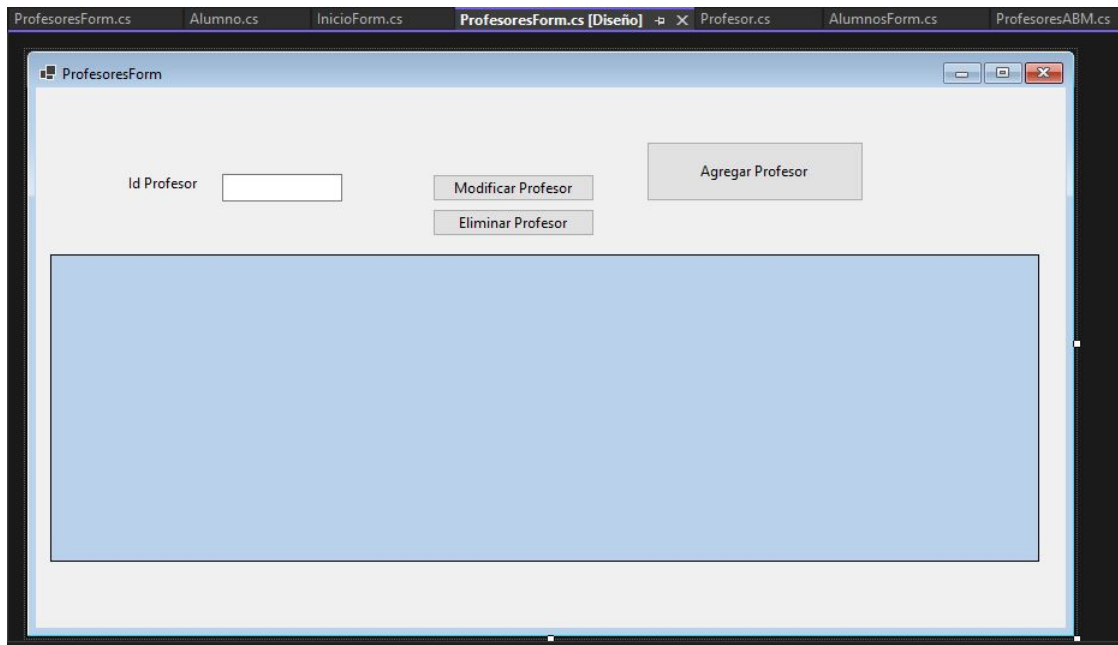


Figura 4: Diseño del Formulario

5. Agregar un Formulario ProfesoresABM

5.1. Enunciado

Agregar un Formulario **ProfesoresABM** con los *label* y *textbox* con los datos del profesor

5.2. Solución

Teniendo en cuenta los atributos de la clase Profesor, creamos el Formulario **ProfesoresABM**. Este formulario sera

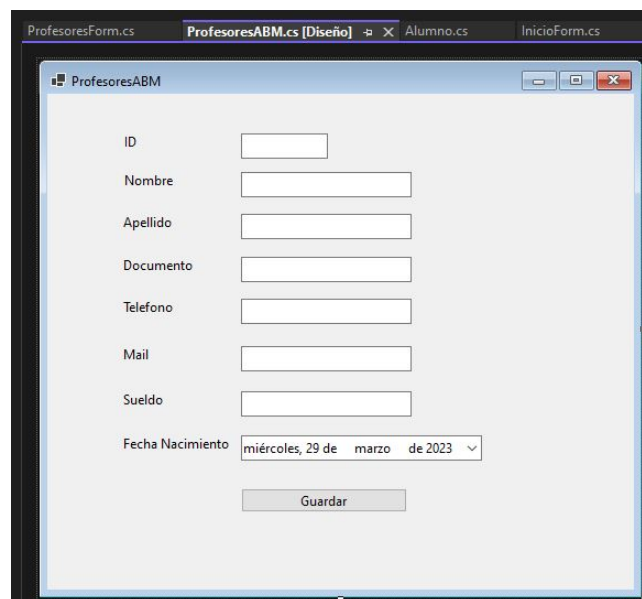


Figura 5: Diseño del Formulario

ejecutado a partir del Botón **Agregar Profesor** el cual esta en el formulario **ProfesoresForm**.

```
1 referencia
private void BtnAgregarProfesor_Click(object sender, EventArgs e)
{
    ProfesoresABM profesorABM = new ProfesoresABM();
    profesorABM.Show();
}
```

Figura 6: Agrego el evento

6. Agregar un botón Guardar

6.1. Enunciado

Agregar un *botón Guardar*, agregarle un método al evento Click del botón que instancie un profesor y tomando los datos del formulario lo agregue en la DB.

6.2. Solución

Al boton Guardar del Formulario **ProfesorABM** le agregamos el siguiente evento para poder agregar un Profesor.

```
1 referencia
private void BtnGuardarProfesor_Click(object sender, EventArgs e)
{
    Profesor profesorAGuardar = new Profesor();
    profesorAGuardar.Nombre = txtNombre.Text;
    profesorAGuardar.Apellido = txtApellido.Text;
    profesorAGuardar.Documento = txtDocumento.Text;
    profesorAGuardar.Telefono = txtTelefono.Text;
    profesorAGuardar.Mail = txtMail.Text;
    profesorAGuardar.Sueldo = Convert.ToDecimal(txtSueldo.Text);
    profesorAGuardar.FechaNacimiento = txtFechaNacimiento.Value;

    int resultado = profesorAGuardar.AgregarEnDb();

    if (resultado == 1)
    {
        MessageBox.Show("Profesor Agregado con exito");
        this.Close();
    }
    else
    {
        MessageBox.Show("Error guardando profesor");
    }
}
```

Figura 7: Agrego el evento

Pero, ¿Realmente funciona?

Corramos la aplicación y agreguemos a un profesor. En esta instancia debemos agregar al Profesor Carlo Rovelli

En la segunda parte podremos visualizar los cambios realizados en la base de datos en la aplicación

Figura 8: Agrego un profesor

7. Botón Eliminar Profesor

7.1. Enunciado

En el Formulario **ProfesoresForm** en el **botón Eliminar Profesor** agregar un evento que instancie un Profesor Auxiliar, y llame al método **EliminarProfesor** pasandole como Id, el Id que se encuentra en la textbox **txtIdProfesor**.

7.2. Solución

Agregamos en el botón **Eliminar Profesor** ubicado en el formulario **ProfesoresForm** el siguiente método

```
1 referencia
private void BtnEliminarProfesor_Click(object sender, EventArgs e)
{
    int idProfesorAEliminar = Convert.ToInt32(this.txtIdProfesor.Text);

    Profesor profesorAuxiliar = new Profesor();

    int r = profesorAuxiliar.EliminarEnDb(idProfesorAEliminar);

    if (r == 1)
    {
        MessageBox.Show("Profesor eliminado");
    }
    else
    {
        MessageBox.Show("No se pudo eliminar profesor");
    }
}
```

Figura 9: Agrego el método para eliminar Profesor de la Db

Como Carmen Nuñez esta dos veces, vamos a eliminarla, ya que para eso hicimos el método

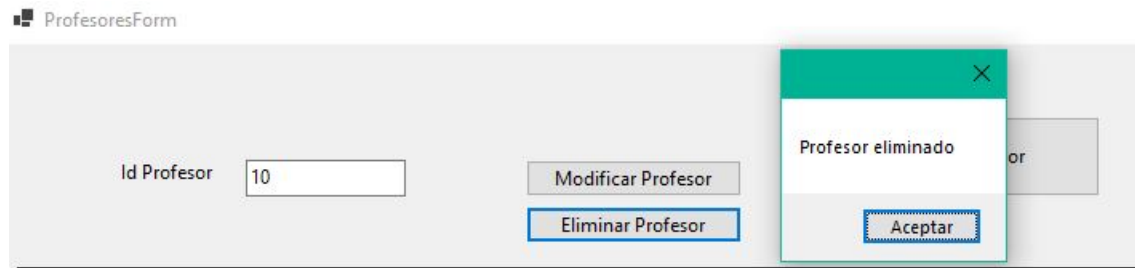


Figura 10: Elimino a Carmen

Tarea 5: Parte 2

8. Agregar en la clase Profesor el método GetProfesorPorId()

8.1. Enunciado

Agregar en la clase Profesor el método **GetProfesorPorId()** que reciba como parámetro un Id de profesor y devuelva una clase Profesor con todos los datos cargados de la base de datos

8.2. Solución

```
3 referencias
public Profesor GetProfesorPorId(int profesorId){
    try {
        string select = s"select * from Profesores where id={profesorId}";
        SqlCommand command = new SqlCommand(select);
        DataTable dt = ac.execDT(command);
        if (dt.Rows.Count <= 0) {
            //no se encuentra pedido para actualizar estado
            return null;
        }
        Profesor profesorADevovlerConDatosDelaBaseDeDatos = new Profesor();
        foreach (DataRow dr in dt.Rows) {
            profesorADevovlerConDatosDelaBaseDeDatos.Id = Convert.ToInt32(dr["Id"]);
            profesorADevovlerConDatosDelaBaseDeDatos.Nombre = dr["Nombre"].ToString();
            profesorADevovlerConDatosDelaBaseDeDatos.Apellido = dr["Apellido"].ToString();
            profesorADevovlerConDatosDelaBaseDeDatos.Documento = dr["Documento"].ToString();
            profesorADevovlerConDatosDelaBaseDeDatos.Telefono = dr["Telefono"].ToString();
            profesorADevovlerConDatosDelaBaseDeDatos.Mail = dr["Mail"].ToString();
            profesorADevovlerConDatosDelaBaseDeDatos.Sueldo = Convert.ToDecimal(dr["Sueldo"]);
            profesorADevovlerConDatosDelaBaseDeDatos.FechaNacimiento = Convert.ToDateTime(dr["FechaNacimiento"]);
        }
        return profesorADevovlerConDatosDelaBaseDeDatos;
    }
    catch (Exception ex) {
        return null;
    }
    finally {
        ac.DesConectar();
    }
}
```

Figura 11: Código del Método

9. Agregar en la clase Profesor el Método GetTodosLosProfesores()

9.1. Enunciado

Agregar en la clase Profesor el Método **GetTodosLosProfesores()**, que devuelva una lista de Profesores con todos los que hay en la base de dato

9.2. Solución

```
2 Referencias
public List<Profesor> GetTodosLosProfesores() {
    try {
        string select = "select * from Profesores ";
        SqlCommand command = new SqlCommand(select);
        DataTable dt = ac.execDT(command);

        if (dt.Rows.Count <= 0) {
            //no se encuentra pedido para actualizar estado
            return null;
        }

        List<Profesor> profesorADevovlerConDatosDeLaBaseDeDatos = new List<Profesor>();
        foreach (DataRow dr in dt.Rows) {
            Profesor ProfesorAuxiliar = new Profesor();
            ProfesorAuxiliar.Id = Convert.ToInt32(dr["Id"]);
            ProfesorAuxiliar.Nombre = dr["Nombre"].ToString();
            ProfesorAuxiliar.Apellido = dr["Apellido"].ToString();
            ProfesorAuxiliar.Documento = dr["Documento"].ToString();
            ProfesorAuxiliar.Telefono = dr["Telefono"].ToString();
            ProfesorAuxiliar.Mail = dr["Mail"].ToString();
            ProfesorAuxiliar.Sueldo = Convert.ToDecimal(dr["Sueldo"]);
            ProfesorAuxiliar.FechaNacimiento = Convert.ToDateTime(dr["FechaNacimiento"]);
            //agrego a la lista
            profesorADevovlerConDatosDeLaBaseDeDatos.Add(ProfesorAuxiliar);
        }
        return profesorADevovlerConDatosDeLaBaseDeDatos;
    }
    catch (Exception ex) {
        return null;
    }
    finally {
        ac.DesConectar();
    }
}
```

Figura 12: Código del Método

10. Crear El método CargarProfesores()

10.1. Enunciado

En el formulario ProfesoresForm Crear El método **CargarProfesores()** que cargue la info de los profesores y la muestre en la grilla *gvProfesores* creada en la parte 1 de la tarea.

10.2. Solución

```
/// <summary>
/// Toma una lista de profesores mediante el metodo
/// GetTodosLosProfesores y se lo pasa al DataSource del GridView
/// </summary>
1 referencia
private void CargarProfesores()
{
    Profesor profesorAuxiliar = new Profesor();
    List<Profesor> profesorEnlaDb = profesorAuxiliar.GetTodosLosProfesores();
    gvProfesores.DataSource = profesorEnlaDb;
}
```

Figura 13: Código del Método

11. Agregar un nuevo constructor

11.1. Enunciado

En el formulario ProfesoresABM agregar un nuevo constructor que reciba como parámetro *idProfesorAModificar*

11.2. Solución

```
1 referencia
public ProfesoresABM()
{
    InitializeComponent();
    //EstoyModificanto = true;
}

1 referencia
public ProfesoresABM(int idProfesorAModificar)
{
    InitializeComponent();
    //EstoyModificanto = true;
    CargarDatosProfesorParaModificar(idProfesorAModificar);
}
```

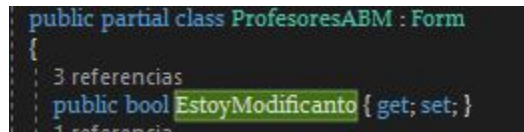
Figura 14: Código de ambos constructores

12. Nuevo atributo

12.1. Enunciado

En el formulario ProfesoresABM agregar un nuevo atributo a la clase de tipo **bool** llamado *EstoyModificando*

12.2. Solución



```
public partial class ProfesoresABM : Form
{
    3 referencias
    public bool EstoyModificando { get; set; }
    1 referencia
}
```

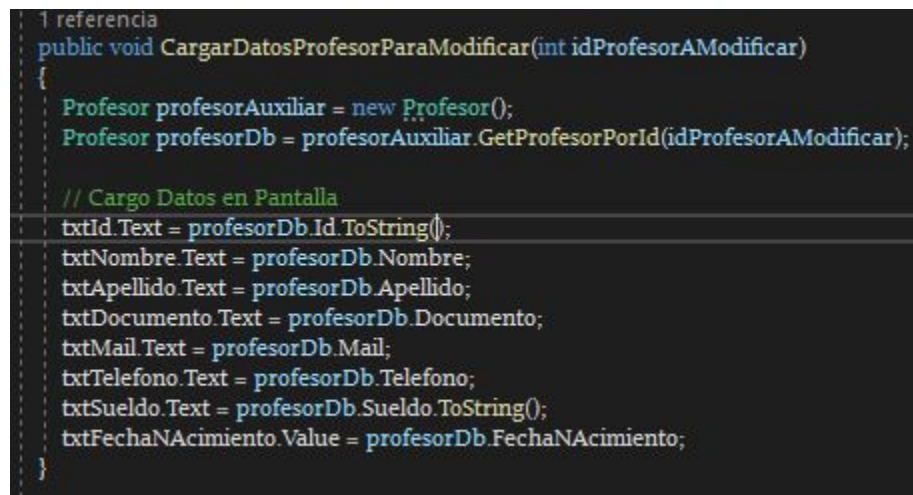
Figura 15: Agrego el atributo

13. Agregar el método CargarDatosProfesorParaModificar

13.1. Enunciado

En el formulario ProfesoresABM agregar el método **CargarDatosProfesorParaModificar()** que reciba como parámetro un id de profesor a modificar, lo busque en la base de datos y cargue sus datos en el formulario

13.2. Solución



```
1 referencia
public void CargarDatosProfesorParaModificar(int idProfesorAModificar)
{
    Profesor profesorAuxiliar = new Profesor();
    Profesor profesorDb = profesorAuxiliar.GetProfesorPorId(idProfesorAModificar);

    // Cargo Datos en Pantalla
    txtId.Text = profesorDb.Id.ToString();
    txtNombre.Text = profesorDb.Nombre;
    txtApellido.Text = profesorDb.Apellido;
    txtDocumento.Text = profesorDb.Documento;
    txtMail.Text = profesorDb.Mail;
    txtTelefono.Text = profesorDb.Telefono;
    txtSueldo.Text = profesorDb.Sueldo.ToString();
    txtFechaNacimiento.Value = profesorDb.FechaNacimiento;
}
```

Figura 16: Código del Método

14. Llamar al método `CargarDatosProfesorParaModificar()`

14.1. Enunciado

Desde el constructor del punto 4, llamar al método `CargarDatosProfesorParaModificar()` para que se carguen los datos

14.2. Solución

```
3 referencias
public bool EstoyModificante { get; set; }
1 referencia
public ProfesoresABM()
{
    InitializeComponent();
    EstoyModificante = false;
}

1 referencia
public ProfesoresABM(int idProfesorAModificar)
{
    InitializeComponent();
    EstoyModificante = true;
    CargarDatosProfesorParaModificar(idProfesorAModificar);
}
```

Figura 17: Constructores del Formulario

15. En el botón Modificar Profesor agregar un evento

15.1. Enunciado

En el Formulario `ProfesoresForm` en el botón *Modificar Profesor* agregar un evento que lea el Id que se encuentra en el textbox `TXTIDPROFESOR`, instancie la clase **ProfesoresABM** y le pase como parámetro el Id del profesor a modificar.

15.2. Solución

```
1 referencia
private void btnModificarProfesor_Click(object sender, EventArgs e)
{
    int IdProfesorAModificar = Convert.ToInt32(txtIdProfesor.Text);
    ProfesoresABM profesoresABMModoModificación = new ProfesoresABM(IdProfesorAModificar);
    profesoresABMModoModificación.Show();
}
```

Figura 18: Código para el botón Modificar Profesor

Luego, Modificamos en `ProfesoresABM` el botón `guardarProfesor` ya que este Formulario sera utilizado para que agregue o modifique un profesor.

El código del para dicho botón queda como sigue

```
1 referencia
private void BtnGuardarProfesor_Click(object sender, EventArgs e)
{
    if (EstoyModificante == true)
    {
        ModificarProfesor();
    }
    else
    {
        AgregarProfesor();
    }
}
```

Figura 19: Código del botón Guardar en ProfesoresABM

Donde los métodos que utiliza este método para realizar la bifurcación dependiendo del botón que son las siguientes

```
1 referencia
private void AgregarProfesor()
{
    Profesor profesorAGuardar = new Profesor();
    profesorAGuardar.Nombre = txtNombre.Text;
    profesorAGuardar.Apellido = txtApellido.Text;
    profesorAGuardar.Documento = txtDocumento.Text;
    profesorAGuardar.Telefono = txtTelefono.Text;
    profesorAGuardar.Mail = txtMail.Text;
    profesorAGuardar.Sueldo = Convert.ToDecimal(txtSueldo.Text);
    profesorAGuardar.FechaNacimiento = txtFechaNacimiento.Value;

    int resultado = profesorAGuardar.AgregarEnDb();

    if (resultado == 1)
    {
        MessageBox.Show("Profesor Agregado con exito");
        this.Close();
    }
    else
    {
        MessageBox.Show("Error guardando profesor");
    }
}
```

```
1 referencia
private void ModificarProfesor()
{
    Profesor profesorAModificar = new Profesor();

    profesorAModificar.Id = Convert.ToInt32(txtId.Text);
    profesorAModificar.Nombre = txtNombre.Text;
    profesorAModificar.Apellido = txtApellido.Text;
    profesorAModificar.Documento = txtDocumento.Text;
    profesorAModificar.Telefono = txtTelefono.Text;
    profesorAModificar.Mail = txtMail.Text;
    profesorAModificar.Sueldo = Convert.ToDecimal(txtSueldo.Text);
    profesorAModificar.FechaNacimiento = txtFechaNacimiento.Value;

    Profesor profesorAux = new Profesor();
    int respuesta = profesorAux.ActualizarEnDb(profesorAModificar);
    if (respuesta != -1)
    {
        MessageBox.Show("Profesor Modificado con exito");
        this.Close();
    }
    else
    {
        MessageBox.Show("Error Modificando Profesor");
    }
}
```

Figura 20: Métodos utilizados por btn

Veamos como trabaja esto!

Primero veamos como se ven en el GridView los profesores adquiridos desde la base de datos

The screenshot shows a Windows application window titled "ProfesoresForm". It contains a form with fields for "Id Profesor", "Modificar Profesor", "Eliminar Profesor", and "Agregar Profesor". Below the form is a GridView displaying a table of professors. The table has the following data:

	Id	Nombre	Apellido	NombreCompleto	Documento	Telefono	Mail	Sueldo
▶	1	Carmen	Núñez	Carmen Núñez	20232654	1122334455	carmen@iafe.u...	1000,00
	12	Rodrigo	Moreiras	Rodrigo Moreiras	20232654	1165658989	rodrigo.moreira...	1000,00
	1002	Juan Pablo	Paz	Juan Pablo Paz	13212835	1145763353	paz@df.uba.ar	150000,00
	1004	Carlo	Rovelli	Carlo Rovelli	102564389	33 (0) 491 26 9644	rovelli@cpt.uni...	300000,00

Figura 21: Profesores dentro de la base de datos

Notamos que Aparecen Carlo Rovelli agregado al principio y se elimino una Carmen, la que estaba en el Id=10.¹

Luego, modifiquemos el documento del profesor Juan Pablo Paz Pongámosle un numero de Documento diferente

The screenshot shows a Windows application window titled "ProfesoresABM". It contains a form for modifying a professor's data. The form fields are: ID (1002), Nombre (Juan Pablo), Apellido (Paz), Documento (11558864), Telefono (1145763353), Mail (paz@df.uba.ar), Sueldo (150000,00), and Fecha Nacimiento (domingo, 5 de abril de 1959). A "Guardar" button is at the bottom. A modal dialog box is open in the center, displaying the message "Profesor Modificado con exito" and an "Aceptar" button. In the background, a table of professors is visible, showing the same data as in Figure 21.

Figura 22: Modifico un profesor

¹UN PROBLEMA QUE VEMOS ACÁ, ES QUE EL ID AUTOINCREMENTADO SE AGREGA DE A PARES Y EN EL ORDEN DE LOS MILES. ESTE ERROR TODAVÍA NO LO PUDE IDENTIFICAR.

Al parecer se modifico con exito. Refresquemos para verificar

The screenshot shows a Windows application window titled "ProfesoresForm". It contains a text input field labeled "Id Profesor", a "Modificar Profesor" button, an "Agregar Profesor" button, and an "Eliminar Profesor" button. Below these is a table with the following data:

	Id	Nombre	Apellido	NombreCompleto	Documento	Telefono	Mail	Sueldo
▶	1	Carmen	Nuñez	Carmen Nuñez	20232654	1122334455	carmen@iafe.u...	1000,00
	12	Rodrigo	Moreiras	Rodrigo Moreiras	20232654	1165658989	rodrigo.moreira...	1000,00
	1002	Juan Pablo	Paz	Juan Pablo Paz	11558864	1145763353	paz@df.uba.ar	150000,00
	1005	Carlo	Rovelli	Carlo Rovelli	10251346	33 (0) 491 26 9644	rovelli@cpt.uni...	300000,00

Below the table is a large blue rectangular area, likely a placeholder for a new record or a confirmation message. At the bottom of the window is a horizontal scrollbar.

Figura 23: Verificación