

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. Clases	5
2.4. Variables globales	6
2.5. Eventos	7
2.5.1. Botón Registrar	7
2.5.2. Botón Eliminar	7
2.5.3. Cargar Formulario	8
2.5.4. Botón Establecer	8
2.5.5. Botón Imprimir	9
2.5.6. Botón Actualizar	9
2.6. Métodos	10
2.6.1. Añadir Nuevo Registro	10
2.6.2. Limpiar	10
2.6.3. Eliminar Registro	11
2.6.4. Obtener índice por Carnet	11
2.6.5. Nuevo Registro	12
2.6.6. Imprimir información	12
2.6.7. Actualizar Registro	13
2.6.8. Verificar Espacio	13
3. Repositorio	14
4. Conclusión	15

1. Introducción

En este laboratorio se estará tratando registro de datos por medio de arreglos, con un sencillo CRUD que lleva los siguientes elemntos:

1. Carnet
2. Nombre
3. Apellido
4. Sexo
5. Dia de Nacimiento
6. Mes de Nacimiento
7. Año de Nacimiento

Las herramientas con las que se estará trabajando son:

1. TextBox
2. ListBox
3. ComboBox
4. Labels
5. Buttons

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

Herramientas	Propiedades			Events
	Name	Text	DropDownStyle	
Label	label1	Cantidad de elementos a ingresar		
	label2	Carnet		
	label3	Nombres		
	label4	Apellidos		
	label5	Sexo		
	label6	Día		
	label7	Mes		
	label9	Año		
GroupBox	groupbox1	Datos personales		
	groupbox2	Fecha de Nacimiento		
Button	btnAdd	Registrar		btnAdd_Click
	btnUpdate	Actualizar		btnUpdate_Click
	btnDelete	Eliminar		btnDelete_Click
	btnShow	Imprimir		btnShow_Click
	btnSet	Establecer		btnSet_Click
ComboBox	cbSex		DropDownList	
	cbDay		DropDownList	
	cbMonth		DropDownList	
TextBox	txtElements			
	txtCarnet			
	txtName			
	txtLastName			
	txtAge			
ListBox	lbConsole			

2.2. Interfaz gráfica

The image displays two screenshots of a graphical user interface (GUI) for a student database application, titled "RA5".

Top Screenshot:

- Form Fields:**
 - Quantity:** "Cantidad de elementos a ingresar" with a value of 5 and an "Establecer" button.
 - Datos personales:**
 - Camet:** Input field with value "2020-0325U".
 - Nombres:** Input field with value "Gabriel".
 - Apellidos:** Input field with value "Ortiz".
 - Sexo:** Dropdown menu with value "Hombr".
 - Fecha de nacimiento:**
 - Día:** Dropdown menu with value "25".
 - Mes:** Dropdown menu with value "Noviem".
 - Año:** Input field.
- Table:**

Camet	Nombre	Apellidos	Nacimiento
2020-0325U	Gabriel	Ortiz	25-Noviembre-2002
- Buttons:** "Registrar", "Actualizar", "Eliminar", and "Imprimir" (highlighted with a blue border).

Bottom Screenshot:

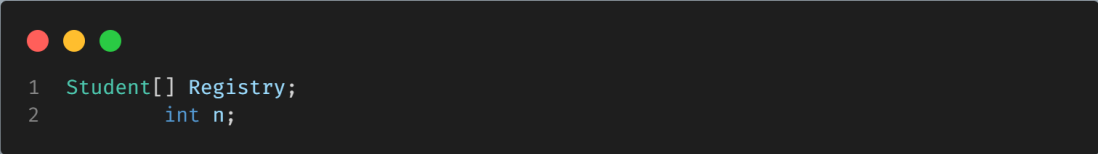
- The same form is shown, but with a confirmation dialog box overlaid on the table.
- Dialog Box:**
 - Title:** "El estudiante con el carnet 2020-0325U se ha eliminado".
 - Buttons:** "OK" (highlighted with a blue border).
- The "Eliminar" button in the background form is also highlighted with a blue border.

2.3. Clases

```
1 namespace AED
2 {
3     class Date
4     {
5         public int Day { get; set; }
6         public string Month { get; set; }
7         public int Age { get; set; }
8     }
9 }
```

```
1 namespace AED
2 {
3     class Student
4     {
5         public string IdCard { get; set; }
6         public string Name { get; set; }
7         public string LastName { get; set; }
8         public string Sex { get; set; }
9         public Date Birth { get; set; }
10    }
11 }
```

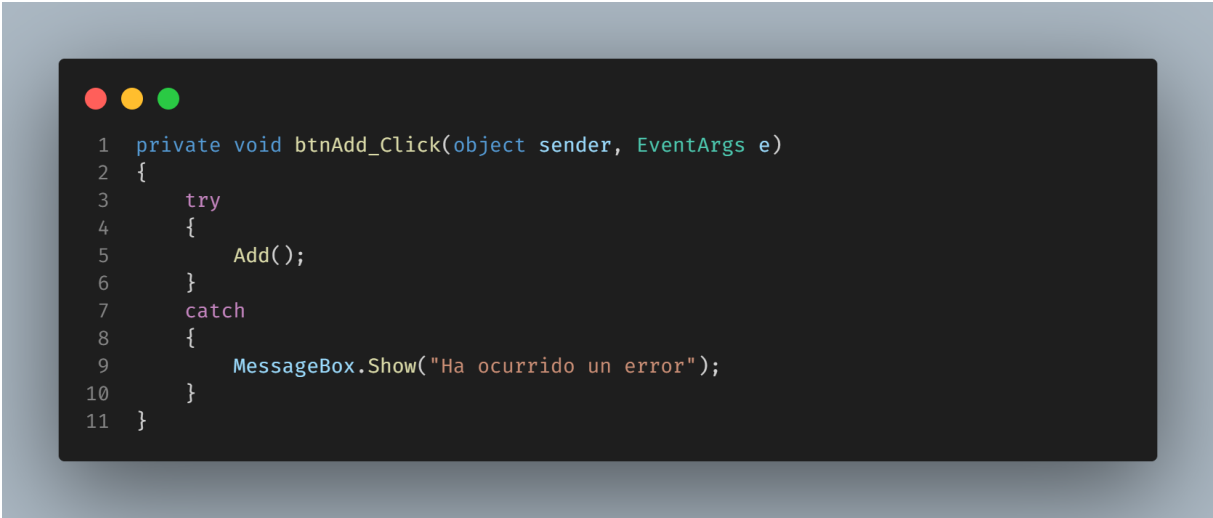
2.4. Variables globales



```
1 Student[] Registry;  
2     int n;
```

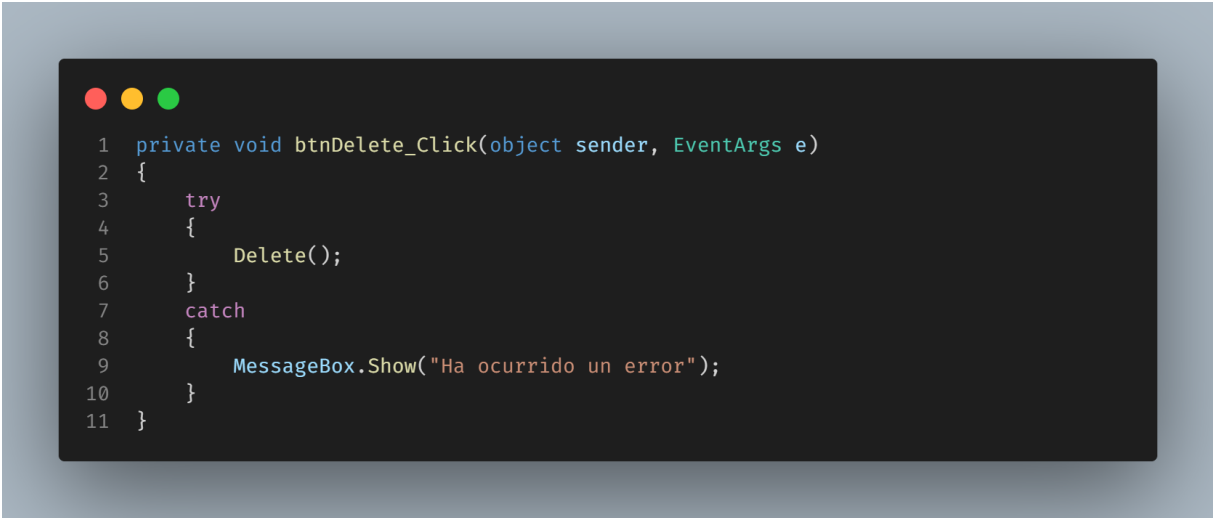
2.5. Eventos

2.5.1. Botón Registrar



```
1 private void btnAdd_Click(object sender, EventArgs e)
2 {
3     try
4     {
5         Add();
6     }
7     catch
8     {
9         MessageBox.Show("Ha ocurrido un error");
10    }
11 }
```

2.5.2. Botón Eliminar



```
1 private void btnDelete_Click(object sender, EventArgs e)
2 {
3     try
4     {
5         Delete();
6     }
7     catch
8     {
9         MessageBox.Show("Ha ocurrido un error");
10    }
11 }
```

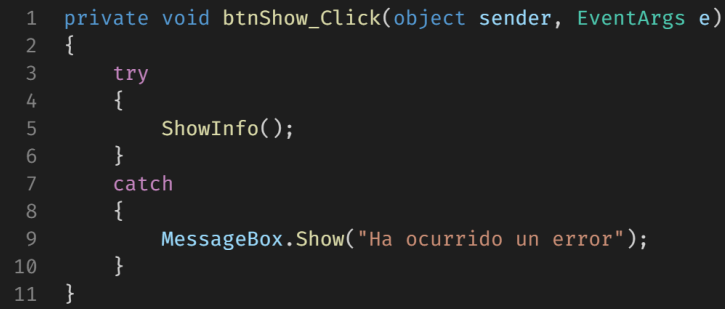

2.5.3. Cargar Formulario

```
1 private void RA5_Load(object sender, EventArgs e)
2 {
3     for(int i = 1; i ≤ 30; i++)
4         cbDay.Items.Add(i);
5
6
7 }
```

2.5.4. Botón Establecer

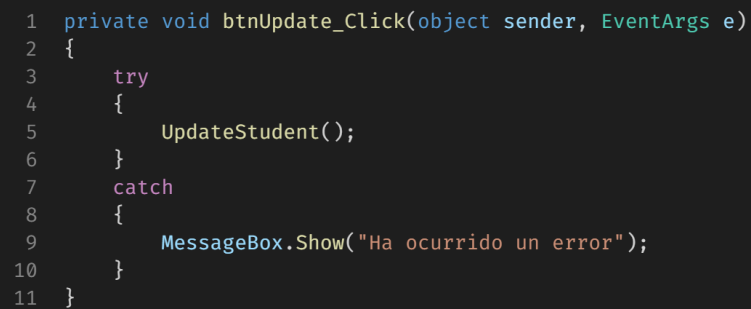
```
1 private void btnSet_Click(object sender, EventArgs e)
2 {
3     try
4     {
5         int tam;
6         if (int.TryParse(txtElements.Text, out tam))
7             Registry = new Student[tam];
8         else
9             Registry = null;
10        n = 0;
11    }
12    catch
13    {
14        MessageBox.Show("Ha ocurrido un error");
15    }
16 }
```

2.5.5. Botón Imprimir



```
1 private void btnShow_Click(object sender, EventArgs e)
2 {
3     try
4     {
5         ShowInfo();
6     }
7     catch
8     {
9         MessageBox.Show("Ha ocurrido un error");
10    }
11 }
```

2.5.6. Botón Actualizar



```
1 private void btnUpdate_Click(object sender, EventArgs e)
2 {
3     try
4     {
5         UpdateStudent();
6     }
7     catch
8     {
9         MessageBox.Show("Ha ocurrido un error");
10    }
11 }
```

2.6. Métodos

2.6.1. Añadir Nuevo Registro

```
1 private void Add()  
2 {  
3     if (!VerificateSpace())  
4         return;  
5  
6     RegisterStudent(n);  
7  
8     Clean();  
9     n++;  
10 }
```

2.6.2. Limpiar

```
1 private void Clean()  
2 {  
3     txtIdCard.Clear();  
4     txtName.Clear();  
5     txtLastName.Clear();  
6     txtAge.Clear();  
7 }
```

2.6.3. Eliminar Registro

```
1 private void Delete()
2 {
3     string id = txtIdCard.Text;
4     int index = GetIndexStudent(id);
5
6     if (index == -1)
7     {
8         MessageBox.Show(id + " no está registrado");
9         return;
10    }
11
12    for(int k = index; k < n; k++)
13        Registry[k] = Registry[k + 1];
14
15    Clean();
16    ShowInfo();
17    n--;
18    MessageBox.Show("El estudiante con el carnet " + id + " se ha eliminado");
19 }
```

2.6.4. Obtener índice por Carnet

```
1 private int GetIndexStudent(string id)
2 {
3     int i;
4
5     for (i = 0; i < n && id != Registry[i].IdCard; i++) ;
6
7     if (id == Registry[i].IdCard)
8         return i;
9     else
10        return -1;
11 }
12 }
```

2.6.5. Nuevo Registro

```
1 private void RegisterStudent(int index)
2 {
3     Registry[index] = new Student()
4     {
5         IdCard = txtIdCard.Text,
6         Name = txtName.Text,
7         LastName = txtLastName.Text,
8         Sex = cbSex.Text,
9         Birth = new Date()
10        {
11            Day = int.Parse(cbDay.Text),
12            Month = cbMonth.Text,
13            Age = int.Parse(txtAge.Text)
14        }
15    };
16 }
```

2.6.6. Imprimir información

```
1 private void ShowInfo()
2 {
3     lbConsole.Items.Clear();
4     lbConsole.Items.Add("Carnet\t\tNombre\t\tApellidos\t\tNacimiento");
5
6     foreach(Student item in Registry)
7     {
8         if(item != null)
9             lbConsole.Items.Add($"{item.IdCard}\t\t{item.Name}\t\t{item.LastName}
10 \t\t{item.Birth.Day}-{item.Birth.Month}-{item.Birth.Age}");
11     }
12 }
```


2.6.7. Actualizar Registro

```
1 private void UpdateStudent()
2 {
3     string id = txtIdCard.Text;
4     int index = GetIndexStudent(id);
5
6     if (index ≥ n)
7     {
8         MessageBox.Show(id + " no está registrado");
9         return;
10    }
11
12    RegisterStudent(index);
13    Clean();
14    MessageBox.Show($"El estudiante con carnet {id} se ha actualizado");
15 }
```

2.6.8. Verificar Espacio

```
1 private bool VerificateSpace()
2 {
3     if (n > Registry.Length - 1)
4     {
5         MessageBox.Show("No hay espacio");
6         return false;
7     }
8     return true;
9 }
```

3. Repositorio

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

4. Conclusión

Se ha logrado una mejor comprensión del manejo de datos con arreglos y mejor apreciación a tecnologías como listas que prescinden de ciertas complicaciones de código. El laboratorio se ha concluido con un total de 208 líneas en el .cs del formulario principal y 24 líneas entre las clases de estudiante y fecha. Los diseños de los formularios es muy importante y la herramienta de GroupBox le da un nuevo diseño a estos ofreciendo una mejor experiencia al usuario. Se intentó realizar el menor número de líneas por cada función para una mejor optimización del código sin perjudicar al ejercicio y los objetivos principales del laboratorio.