

UNIVERSIDAD NACIONAL DE INGENIERÍA

FACULTAD DE ELECTROTECNIA Y COMPUTACIÓN

Algoritmización y Estructuras de Datos

Profesor:

Adilson G. López

Grupo: 2M1 - CO

Desarrolladores:

Gabriel A. Ortiz — 2020 - 0325U

Marcel E. Díaz — 2020 - 1384U

Índice

1. Introducción	2
1.1. Explicación del Laboratorio	2
1.1.1. Registro de Créditos	2
1.1.2. Registro de Proyectos	2
1.2. Lenguaje e IDE de Desarrollo	3
1.2.1. Lenguaje	3
1.2.2. IDE	3
2. Desarrollo	4
2.1. Clases	4
2.1.1. Crédito	4
2.1.2. Proyecto	5
2.2. MDI Principal	6
2.2.1. Diseño de Interfaz	6
2.2.2. Eventos	7
2.3. Presentación	8
2.3.1. Diseño de Interfaz	8
2.4. Registro de Créditos	9
2.4.1. Diseño de Interfaz	9
2.4.2. Variable globales	9
2.4.3. Métodos	10
2.4.4. Eventos	13
2.5. Registro de Proyectos	16
2.5.1. Diseño de Interfaz	16
2.5.2. Variable globales	16
2.5.3. Métodos	17
2.5.4. Eventos	20
3. Repositorio	23
4. Conclusión	24
4.1. Ejecución del Programa	24
4.1.1. Formulario Principal	24
4.1.2. Formulario Crediticio	24
4.1.3. Formulario de Proyectos	25
4.1.4. Pantalla de Presentación	25
4.1.5. Pantalla de Reportes	26
4.1.6. Formulario de Recursos de Actividades	26

1. Introducción

1.1. Explicación del Laboratorio

1.1.1. Registro de Créditos

Una casa comercial necesita almacenar información de sus deudores, ordenados de menor a mayor por su deuda total.

Escribir un programa que permita:

1. Listar los datos de todos los clientes
2. Eliminar un cliente dado el Id
3. Insertar un Cliente (Ordenado por su deuda)
4. Modificar un cliente dado el Id (Respetar Orden)
5. Listar los datos de un cliente determinado
6. Lista de clientes con deudas pendientes
7. Lista de clientes sin deudas

1.1.2. Registro de Proyectos

Se requiere llevar el control de entregas de proyecto de fin de curso por persona de la asignatura de AED, ordenado por su fecha de entrega.

Escribir un programa que permita:

1. Listar todos los registros
2. Eliminar un proyecto dado el Carnet
3. Insertar una nueva entrega (Ordenado por su fecha de entrega)
4. Modificar datos de un proyecto dado el carné (Respetar el orden según la fecha)
5. Listar los datos de un proyecto en específico de acuerdo al carné del estudiante
6. Listar los proyectos que se entregaron después de su fecha límite de entrega

1.2. Lenguaje e IDE de Desarrollo

1.2.1. Lenguaje

El lenguaje que se utilizará es «C#», es un lenguaje de programación multi-paradigma desarrollado y estandarizado por la empresa Microsoft como parte de su plataforma.

Paralelo a ello el programa se utilizará «Windows Forms», es una biblioteca de clases gráfica (GUI) gratuita y de código abierto incluida como parte de Microsoft .NET.

1.2.2. IDE

«Visual Studio Community. Un completo IDE extensible y gratuito para crear aplicaciones modernas para Windows, Android e iOS, además de aplicaciones web y servicios en la nube.» -Microsoft. Microsoft Visual Studio es un entorno de desarrollo integrado compatible con múltiples lenguajes de programación al igual que entornos de desarrollo web.

2. Desarrollo

2.1. Clases

2.1.1. Crédito

Esta clase se emplea en el formulario correspondiente al registro de crédito, en la cual se almacenará la información de los estudiantes a ingresar.

```
● ● ●
1  namespace AEDHub.Models
2  {
3      class DEBTOR : PERSON
4      {
5          public int Id { get; set; }
6          public double Debt { get; set; }
7          public bool isPay { get; set; }
8      }
9 }
```

```
● ● ●
1  using System;
2
3  namespace AEDHub.Models
4  {
5      class PERSON
6      {
7          public string IdCard { get; set; }
8          public string Name { get; set; }
9          public string LastName { get; set; }
10         public string Phone { get; set; }
11         public string Address { get; set; }
12         public string Sex { get; set; }
13         public DateTime Birth{ get; set; }
14     }
15 }
```

2.1.2. Proyecto

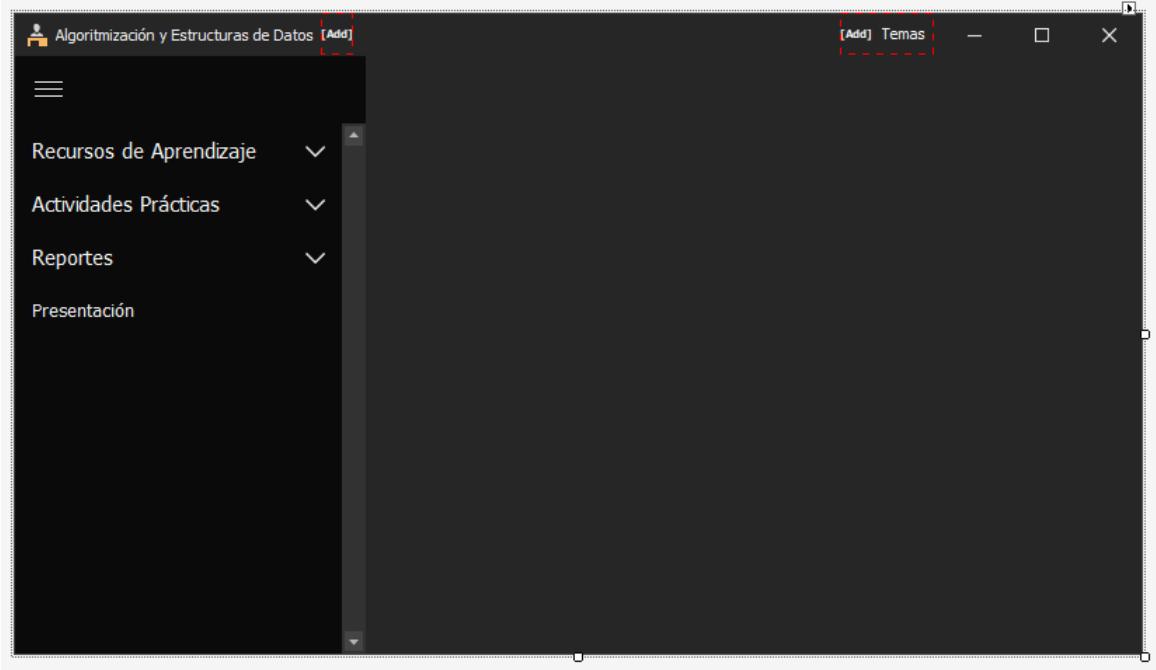
Esta clase se emplea en el formulario correspondiente al registro de proyectos, en la cual se almacenará la información de los estudiantes a ingresar.

```
● ● ●
1  using System;
2
3  namespace AEDHub.Models
4  {
5      class PROJECT : STUDENT
6      {
7          public string ProjectName { get; set; }
8          public DateTime DateSent { get; set; }
9          public DateTime DateLimit { get; set; }
10         public int Score { get; set; }
11         public bool isSentLate { get; set; }
12     }
13 }
```

```
● ● ●
1  namespace AEDHub.Models
2  {
3      class STUDENT : PERSON
4      {
5          public int FirstPartial { get; set; }
6          public int SecondPartial { get; set; }
7          public int Systematic { get; set; }
8          public double FinalNote { get; set; }
9      }
10 }
```

2.2. MDI Principal

2.2.1. Diseño de Interfaz



2.2.2. Eventos

1. Botón Registro de Crédito:

```
1  private void AceCreditRecord_Click(object sender, EventArgs e)
2  {
3      PanelContainer.Controls.Clear();
4      PanelContainer.Controls.Add(new CreditRecord() { Dock = DockStyle.Fill });
5 }
```

2. Botón Registro de Proyectos:

```
1  private void AceProjectsRecord_Click(object sender, EventArgs e)
2  {
3      PanelContainer.Controls.Clear();
4      PanelContainer.Controls.Add(new ProjectsRecord() { Dock = DockStyle.Fill });
5 }
```

3. Botón Presentación:

```
1  private void AcePresentation_Click(object sender, EventArgs e)
2  {
3      PanelContainer.Controls.Clear();
4      PanelContainer.Controls.Add(new Presentation());
5 }
```

2.3. Presentación

2.3.1. Diseño de Interfaz



2.4. Registro de Créditos

2.4.1. Diseño de Interfaz

The screenshot shows a software window divided into two main sections: 'Datos del Cliente' (Client Data) on the left and 'Detalles del Préstamo' (Loan Details) on the right.

Datos del Cliente:

- Cédula: [Text Input]
- Nombres: [Text Input]
- Apellidos: [Text Input]
- Dirección: [Text Input]
- Teléfono: [Text Input]

Detalles del Préstamo:

- Icono de moneda: [Image]
- Id: [Text Input]
- Estado: [Text Input]
- Deuda: [Text Input]

At the bottom are two buttons: a red 'X' labeled 'Cerrar' (Close) and a green checkmark labeled 'Guardar' (Save).

Below the main form is a navigation bar with three radio buttons and a 'Nuevo Préstamo' button:

- Todos
- Solo deudas vigentes
- Solo deudas pagadas

Nuevo Préstamo

At the bottom of the interface is a table with columns: Cédula, Nombre, Apellido, Dirección, Teléfono, and Id de Préstamo. The first row contains 'string' for all columns. The second row also contains 'string' for all columns. To the right of the table are two buttons: 'Retrieve Details' and 'Run Designer'.

2.4.2. Variable globales

The code editor displays the following global variable declarations:

```
1 DEBTOR[] database;
2 int n = 0;
```

2.4.3. Métodos

1. Invertir Posición:

```
● ● ●  
1 private void InvertPosition(int pos1, int pos2)  
2 {  
3     var temp = database[pos1];  
4     database[pos1] = database[pos2];  
5     database[pos2] = temp;  
6 }
```

2. Método de Ordenación Burbuja Menor:

```
● ● ●  
1 private void MinorBubble()  
2 {  
3     for (int i = 1; i < n; i++)  
4         for (int j = n - 1; j > 0; j--)  
5             if (database[j - 1].Id > database[j].Id)  
6                 InvertPosition(j, j - 1);  
7 }
```

3. Cargar Deudor:

```
● ● ●  
1 private void LoadDebtor(DEBTOR debtor)  
2 {  
3     txtAddress.Text = debtor.Address;  
4     txtDebt.Text = debtor.Debt.ToString();  
5     txtId.Text = debtor.Id.ToString();  
6     txtIdCard.Text = debtor.IdCard;  
7     txtLastNames.Text = debtor.LastName;  
8     txtNames.Text = debtor.Name;  
9     txtPhone.Text = debtor.Phone;  
10 }
```

4. Cargar Datos a la Tabla:

```
1 private void LoadDataTable(char filter)
2 {
3     if (database == null)
4         return;
5
6     switch (filter)
7     {
8         case 'P':
9             gcDebts.DataSource = database.Where(x => x.isPay).ToList();
10            gvDebts.RefreshData();
11            break;
12        case 'N':
13            gcDebts.DataSource = database.Where(x => !x.isPay).ToList();
14            break;
15        case 'A':
16        default:
17            gcDebts.DataSource = database.ToList();
18            break;
19    }
20 }
```

5. Validar Formulario:

```
1 private bool ValidateForm()
2 {
3     try
4     {
5         if (string.IsNullOrEmpty(txtAddress.Text) ||
6             string.IsNullOrEmpty(txtDebt.Text) ||
7             string.IsNullOrEmpty(txtId.Text) ||
8             string.IsNullOrEmpty(txtIdCard.Text) ||
9             string.IsNullOrEmpty(txtLastNames.Text) ||
10            string.IsNullOrEmpty(txtNames.Text) ||
11            string.IsNullOrEmpty(txtPhone.Text) ||
12            string.IsNullOrEmpty(cbStatus.Text))
13             throw new Exception("Debe llenar todos los campos para continuar");
14
15         return true;
16     }
17     catch (Exception ex)
18     {
19         XtraMessageBox.Show(ex.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error
20     );
21         return false;
22     }
23 }
```

6. Limpiar Formulario:

```
1  private void CleanForm()
2  {
3      txtAddress.Text = txtDebt.Text = txtId.Text = txtIdCard.Text = txtLastNames.Text = txtNames.Text = txtPhone.Text = string.Empty;
4  }
5
```

7. Insertar o Actualizar Deudor:

```
1  private void InsertOrUpdateDebtor(int index)
2  {
3      database[index] = new DEBTOR()
4      {
5          Id = int.Parse(txtId.Text),
6          IdCard = txtIdCard.Text,
7          Address = txtAddress.Text,
8          Debt = Double.Parse(txtDebt.Text),
9          LastName = txtLastNames.Text,
10         Name = txtNames.Text,
11         Phone = txtPhone.Text,
12         isPay = cbStatus.EditValue.Equals("Pagada")
13     };
14 }
```

2.4.4. Eventos

1. Botón Editar:

```
● ● ●
1  private void BbiEdit_ItemClick(object sender, DevExpress.XtraBars.ItemClickEventArgs e)
2  {
3      try
4      {
5          int id =(int) gvDebts.GetFocusedRowCellValue("Id");
6          for (int i = 0; i < n; i++)
7          {
8              if (database[i].Id == id)
9              {
10                  LoadDebtor(database[i]);
11                  break;
12              }
13          }
14      lcInput.Visible = true;
15  }
16  catch
17  {
18      XtraMessageBox.Show("Ha ocurrido un error inesperado", "Error", MessageBoxButtons.OK
19      , MessageBoxIcon.Error);
20  }
21 }
```

2. Botón Nuevo:

```
● ● ●
1  private void BbiNew_ItemClick(object sender, DevExpress.XtraBars.ItemClickEventArgs e)
2  {
3      CleanForm();
4      lcInput.Visible = true;
5  }
```

3. Botón Eliminar:

```
1  private void BbiDelete_ItemClick(object sender, DevExpress.XtraBars.ItemClickEventArgs e)
2  {
3      try
4      {
5          int id = (int)gvDebts.GetFocusedRowCellValue("Id");
6
7          for (int i = 0; i < n; i++)
8              if (database[i].Id == id)
9              {
10                  for (int j = i; j < n - 1; j++)
11                      database[j] = database[j + 1];
12
13                  break;
14              }
15
16          n--;
17
18          if (n == 0)
19              database = null;
20          else
21              Array.Resize(ref database, n);
22
23          gcDebts.DataSource = null;
24          gcDebts.DataSource = database.ToList();
25      }
26      catch
27      {
28          XtraMessageBox.Show("No hay un registro que editar, intentelo de nuevo", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
29      }
30 }
```

4. Botón Nuevo Proyecto:

```
1  private void BtnNew_Click(object sender, EventArgs e)
2  {
3      lcInput.Visible = true;
4 }
```

5. Botón Cerrar:

```
1  private void BtnClose_Click(object sender, EventArgs e)
2  {
3      CleanForm();
4      lcInput.Visible = false;
5 }
```

6. Botón Guardar:

```
1  private void BtnSave_Click(object sender, EventArgs e)
2  {
3      try
4      {
5          if (!ValidateForm())
6              return;
7
8          Array.Resize(ref database, n + 1);
9
10         InsertOrUpdateDebtor(n);
11         n++;
12         gcDebts.DataSource = database.ToList();
13         MinorBubble();
14         CleanForm();
15         lcInput.Visible = false;
16     }
17     catch
18     {
19         XtraMessageBox.Show("Ha ocurrido un error inesperado", "Error", MessageBoxButtons.OK
20             , MessageBoxIcon.Error);
21     }
}
```

7. Botones radiales de filtro:

```
1  private void RgFilter_EditValueChanged(object sender, EventArgs e)
2  {
3      try
4      {
5          LoadDataTable((char)(rgFilter.EditValue));
6      }
7      catch
8      {
9          XtraMessageBox.Show("Ha ocurrido un error inesperado", "Error", MessageBoxButtons.OK
10             , MessageBoxIcon.Error);
11     }
12 }
```

8. Abrir PopUp:

```
1  private void GvDebts_PopupMenuShowing(object sender, DevExpress.XtraGrid.Views.Grid.PopupMenu
2  ShowingEventArgs e)
3  {
4      pmActions.ShowPopup(MousePosition);
5 }
```

2.5. Registro de Proyectos

2.5.1. Diseño de Interfaz

Datos del Estudiante

Carnet: [Text Box]

Nombres: [Text Box]

Apellidos: [Text Box]

Datos del Proyecto

Nombre: [Text Box]

Fecha de envío: [Text Box]

Fecha límite: [Text Box]

Nota: [Text Box]

X Cerrar ✓ Guardar

Mostrar solo tardíos + Nuevo Proyecto

Carnet:	Nombre	Apellido	Nombre del Pr...	Fecha de Envío	Fecha Límite d...	Calificación
string	string	string	string	ncProyecto		
string	string	string	string	Retrieve Details	Run Designer	

2.5.2. Variable globales

```
PROJECT[] database;
int n = 0;
```

2.5.3. Métodos

1. Invertir Posición:

```
● ● ●  
1 private void InvertPosition(int pos1, int pos2)  
2 {  
3     var temp = database[pos1];  
4     database[pos1] = database[pos2];  
5     database[pos2] = temp;  
6 }
```

2. Método de Ordenación Burbuja Menor:

```
● ● ●  
1 private void MinorBubble()  
2 {  
3     for (int i = 1; i < n; i++)  
4         for (int j = n - 1; j > 0; j--)  
5             if (database[j - 1].DateSent > database[j].DateSent)  
6                 InvertPosition(j, j - 1);  
7 }
```

3. Cargar Proyectos:

```
● ● ●  
1 private void LoadProject(PROJECT project)  
2 {  
3     txtIdCard.Text = project.IdCard;  
4     txtName.Text = project.ProjectName;  
5     txtScore.Text = project.Score.ToString();  
6     txtStudentLastName.Text = project.LastName;  
7     txtStudentName.Text = project.Name;  
8     dtLimit.DateTime = project.DateLimit;  
9     dtSent.DateTime = project.DateSent;  
10 }
```

4. Validar Formulario:

```
1  private bool ValidateForm()
2  {
3      try
4      {
5          if (string.IsNullOrEmpty(txtIdCard.Text) ||
6              string.IsNullOrEmpty(txtName.Text) ||
7              string.IsNullOrEmpty(txtStudentLastName.Text) ||
8              string.IsNullOrEmpty(txtStudentName.Text) ||
9              string.IsNullOrEmpty(dtLimit.Text) ||
10             string.IsNullOrEmpty(dtSent.Text) ||
11             string.IsNullOrEmpty(txtScore.Text))
12             throw new Exception("Por favor, rellene todos los campos para continuar");
13
14         if (int.Parse(txtScore.Text) > 100 || int.Parse(txtScore.Text) < 0)
15             throw new Exception("Por favor, ingrese una nota válida");
16
17         return true;
18     }
19     catch(Exception ex)
20     {
21         XtraMessageBox.Show(ex.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error
22     );
23     }
24 }
```

5. Limpiar Formulario:

```
1  private void CleanForm()
2  {
3      txtIdCard.Text = txtName.Text = txtStudentLastName.Text = txtStudentName.Text = dtLimit.T
ext = dtSent.Text = txtScore.Text = string.Empty;
4  }
```

6. Insertar o Actualizar Proyecto:

```
1  private void InsertOrUpdateProject(int index)
2  {
3      database[index] = new PROJECT()
4      {
5          ProjectName = txtName.Text,
6          Score = int.Parse(txtScore.Text),
7          DateLimit = dtLimit.DateTime,
8          DateSent = dtSent.DateTime,
9          IdCard = txtIdCard.Text,
10         Name = txtStudentName.Text,
11         LastName = txtStudentLastName.Text,
12         isSentlate = dtlimit.DateTime < dtSent.DateTime
13     };
14 }
```

2.5.4. Eventos

1. Botón Editar:

```
1  private void BbiEdit_ItemClick(object sender, DevExpress.XtraBars.ItemClickEventArgs e)
2  {
3      try
4      {
5          string idCard = gvProjects.GetFocusedRowCellValue("IdCard").ToString();
6
7          for (int i = 0; i < n; i++)
8              if (database[i].IdCard == idCard)
9              {
10                  LoadProject(database[i]);
11                  break;
12              }
13
14          lcInput.Visible = true;
15      }
16      catch
17      {
18          XtraMessageBox.Show("No hay un registro que editar, intentelo de nuevo", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
19      }
20 }
```

2. Botón Nuevo:

```
1  private void BbiNew_ItemClick(object sender, DevExpress.XtraBars.ItemClickEventArgs e)
2  {
3      CleanForm();
4      lcInput.Visible = true;
5 }
```

3. Botón Eliminar:

```
1  private void BbiDelete_ItemClick(object sender, DevExpress.XtraBars.ItemClickEventArgs e)
2  {
3      try
4      {
5          string idCard = gvProjects.GetFocusedRowCellValue("IdCard").ToString();
6
7          for (int i = 0; i < n; i++)
8              if (database[i].IdCard == idCard)
9              {
10                  for(int j = i; j < n - 1; j++)
11                      database[j] = database[j + 1];
12
13                  break;
14              }
15
16          n--;
17
18          if (n == 0)
19              database = null;
20          else
21              Array.Resize(ref database, n);
22
23          gcProjects.DataSource = null;
24          gcProjects.DataSource = database.ToList();
25      }
26      catch
27      {
28          XtraMessageBox.Show("No hay un registro que editar, intentelo de nuevo", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
29      }
30 }
```

4. Botón Nuevo Proyecto:

```
1  private void BtnNew_Click(object sender, EventArgs e)
2  {
3      CleanForm();
4      lcInput.Visible = true;
5 }
```

5. Botón Cerrar:

```
1  private void BtnClose_Click(object sender, EventArgs e)
2  {
3      CleanForm();
4      lcInput.Visible = false;
5 }
```

6. Botón Guardar:

```
1  private void BtnSave_Click(object sender, EventArgs e)
2  {
3      if (!ValidateForm())
4          return;
5
6      Array.Resize(ref database, n + 1);
7
8      InsertOrUpdateProject(n);
9      CleanForm();
10     n++;
11     MinorBubble();
12     gcProjects.DataSource = database.ToList();
13     lcInput.Visible = false;
14 }
```

7. Checkbox Tardanzas:

```
1  private void CeOnlyLate_CheckedChanged(object sender, EventArgs e)
2  {
3      try
4      {
5          gcProjects.DataSource = ceOnlyLate.Checked ? database.Where(x => x.isSentLate).ToList()
6          : database.ToList();
7          gcProjects.RefreshDataSource();
8      }
9      catch
10      {
11          XtraMessageBox.Show("Ha ocurrido un error al obtener los proyectos entregados tarde",
12          "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);
13      }
14 }
```

8. Abrir PopUp:

```
1  private void GvProjects_PopupMenuShowing(object sender, DevExpress.XtraGrid.Views.Grid.PopupM
2  enuShowingEventArgs e)
3  {
4      pmActions.ShowPopup(MousePosition);
5 }
```

3. Repositorio

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

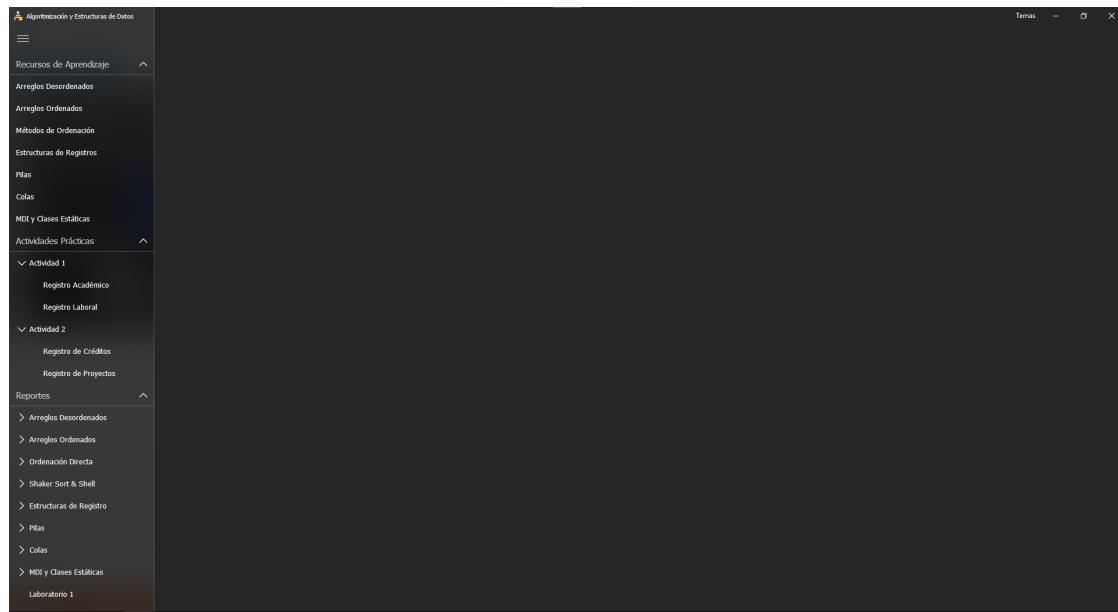
Nota: El repositorio es privado, si necesita acceder solicite ser colaborador a cualquiera de los autores de este documento.

File	Description	Time Ago
Models	Incorporación de los recursos de aprendizaje e implementación de herramientas	13 hours ago
Modules	-	12 hours ago
Properties	Incorporación de los recursos de aprendizaje e implementación de herramientas	13 hours ago
Reports	-	12 hours ago
Resources	Incorporación de los recursos de aprendizaje e implementación de herramientas	13 hours ago
AEDHub.csproj	-	12 hours ago
App.config	Add project files.	2 days ago
MdiMain.Designer.cs	-	12 hours ago
MdiMain.cs	-	12 hours ago
MdiMain.resx	-	12 hours ago
PdfReader.Designer.cs	Incorporación de los recursos de aprendizaje e implementación de herramientas	13 hours ago
PdfReader.cs	-	12 hours ago
PdfReader.resx	Incorporación de los recursos de aprendizaje e implementación de herramientas	13 hours ago
Presentation.Designer.cs	-	12 hours ago
Presentation.cs	-	12 hours ago
Presentation.resx	Incorporación de los recursos de aprendizaje e implementación de herramientas	13 hours ago
Program.cs	Add project files.	2 days ago

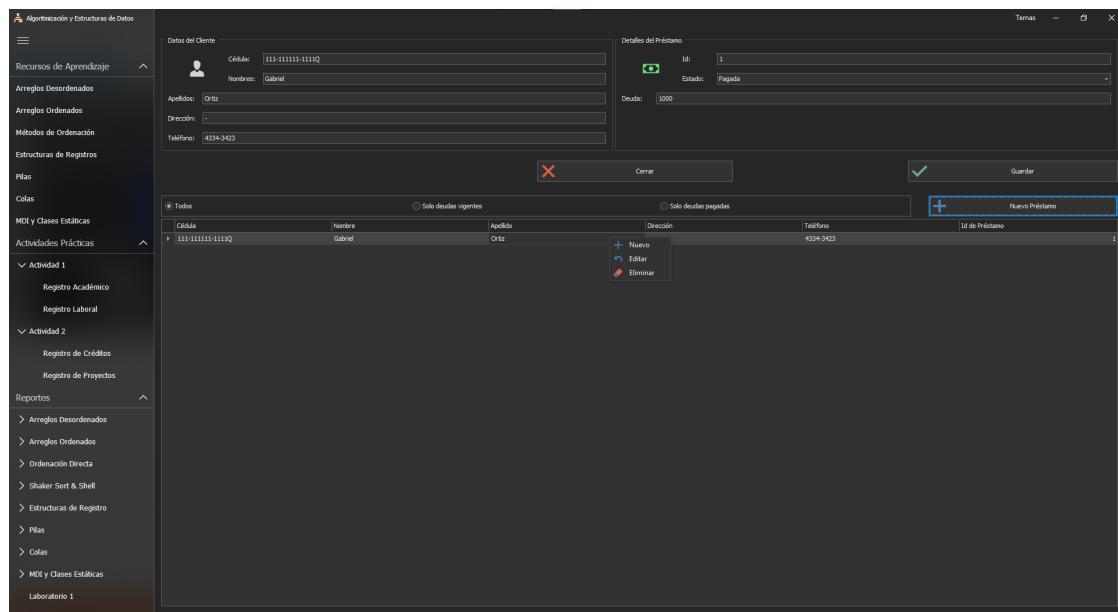
4. Conclusión

4.1. Ejecución del Programa

4.1.1. Formulario Principal



4.1.2. Formulario Crediticio

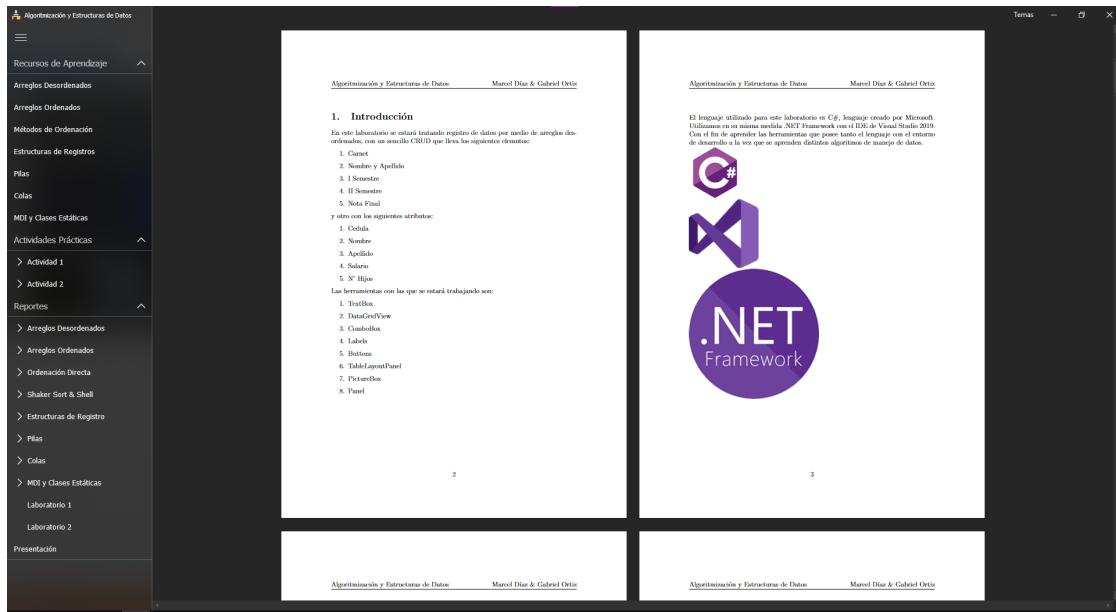


4.1.3. Formulario de Proyectos

4.1.4. Pantalla de Presentación



4.1.5. Pantalla de Reportes



4.1.6. Formulario de Recursos de Actividades

