

Computación Gráfica 2283

Introducción

El presente informe documentará el desarrollo de una aplicación en C# que permite realizar cálculos de áreas y perímetros de diversas figuras geométricas mediante el uso de formularios. Esta aplicación, implementada bajo el entorno de desarrollo .NET Framework y utilizando formularios de Windows Forms, tiene como propósito familiarizarse con el entorno de VS asi como repasar fundamentos claves de programación orientada a objetos y del uso de varios formularios y subformularios

El proyecto incluye formularios dedicados a cada tipo de figura geométrica, como cuadrados, rectángulos, círculos y triángulos, donde el usuario puede ingresar los parámetros correspondientes, por ejemplo, lados, radio o base y altura. A partir de estos datos, la aplicación realiza los cálculos necesarios y muestra los resultados de manera clara y dinámica. Además, se incorpora un diseño basado en principios de programación estructurada y orientada a objetos, haciendo uso de patrones como el Singleton para optimizar el manejo de ventanas y evitar redundancias en la ejecución.

Requerimientos

Entrada de datos

La entrada de datos para cada formulario está dada por solo los inputs de cada dato que se necesita de la figura y en la parte de las clases se guarda en variables flotantes

Formularios

txtNombreDelInput /*Objeto Tipo TextBox*/

Clases

mNombredelDato /*Objeto Tipo Flotante*/

Salida de datos

La salida de datos está dada por cajas de texto las cuales están inhabilitadas para editar y bloqueadas para poder mover las cajas

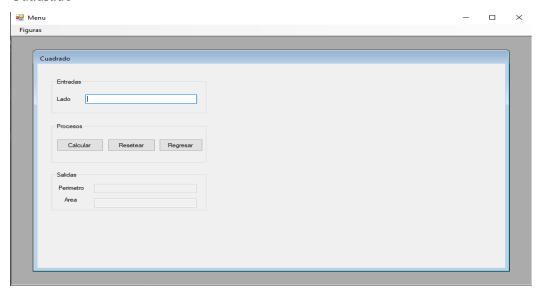
Formularios

txtNombredelOutput /*Objeto Tipo TextBox*/

Diseños de la solución

Dentro de los diseños de la solución, cada figura cuenta con sus respectivos textbox para las entradas de datos y va a depender de cada figura, también se aplica un patrón de diseño singleton en los formularios de cada figura para que se cree una única instancia de esta forma evitando formularios repetidos, dentro de la clase del formulario se encuentra los métodos de cada botón y su método para inicializar datos.

Cuadrado



```
public partial class FrmSquare : Form

{

Square square = new Square();

private static FrmSquare instance;

1 referencia
public FrmSquare()
{

InitializeComponent();
}

2 referencias
public static FrmSquare GetInstance() ...

2 referencias
public void InitializeData() ...

1 referencia
private void btnExit_Click(object sender, EventArgs e)...

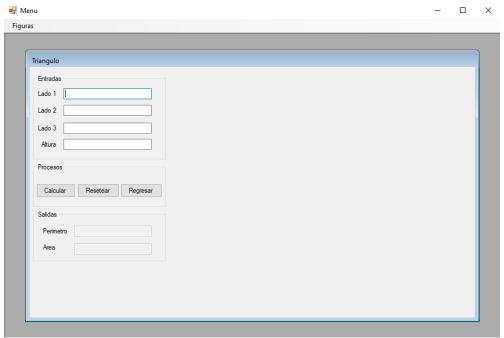
1 referencia
private void btnReset_Click(object sender, EventArgs e)...

1 referencia
private void btnCalculate_Click(object sender, EventArgs e)...

1 referencia
private void btnCalculate_Click(object sender, EventArgs e)...

1 referencia
private void FrmSquare_Load(object sender, EventArgs e)...
```

Triangulo



```
7 referencias
public partial class FrmTriangle : Form
{
    Triangle triangle = new Triangle();
    private static FrmTriangle instance;

1 referencia
public FrmTriangle()
{
    InitializeComponent();
}

1 referencia
public static FrmTriangle GetInstance() ...

2 referencias
public void InitializeData()...

1 referencia
private void FrmTriangle_Load(object sender, EventArgs e)...

1 referencia
private void btnCalculate_Click(object sender, EventArgs e)...

1 referencia
private void btnReset_Click(object sender, EventArgs e)...

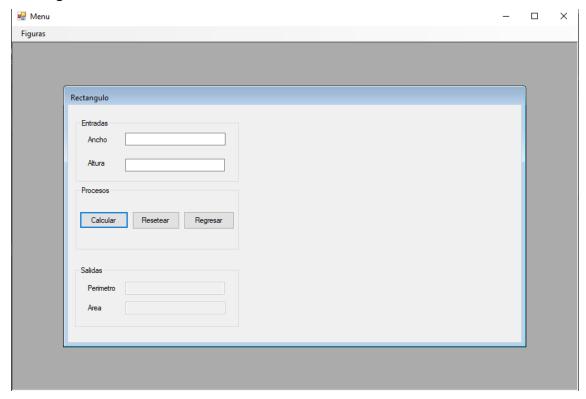
1 referencia
private void btnReset_Click(object sender, EventArgs e)...

1 referencia
private void btnReset_Click(object sender, EventArgs e)...

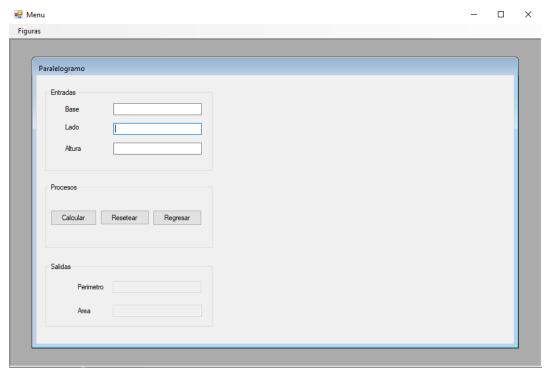
1 referencia
private void btnExit_Click(object sender, EventArgs e)...

1 referencia
```

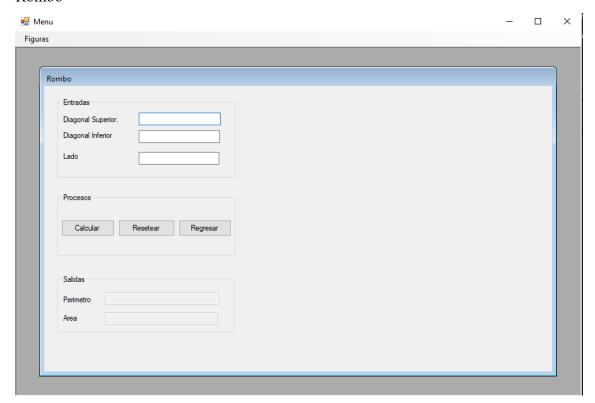
Rectángulo



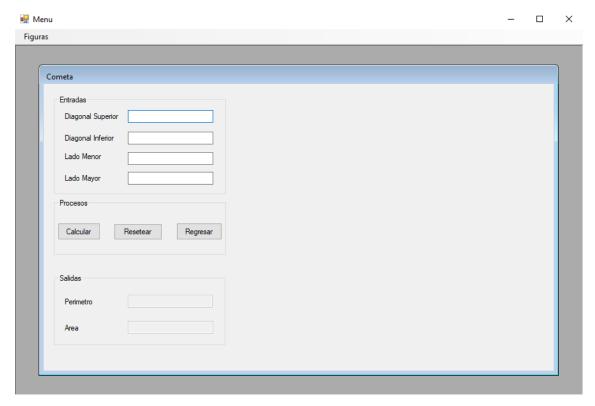
Paralelogramo



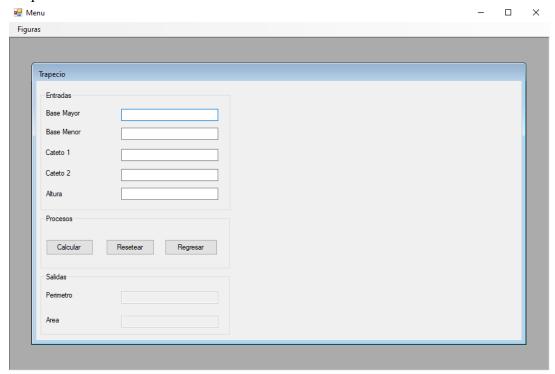
Rombo



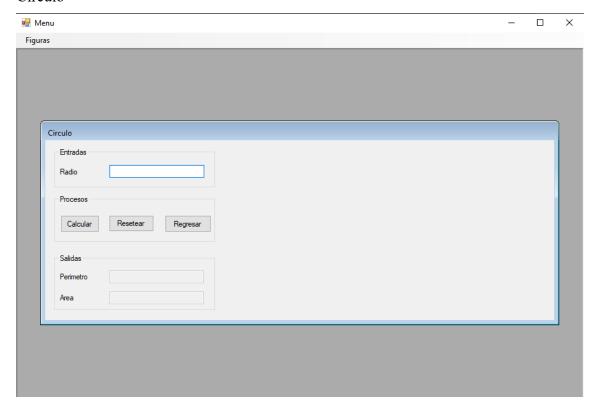
Cometa



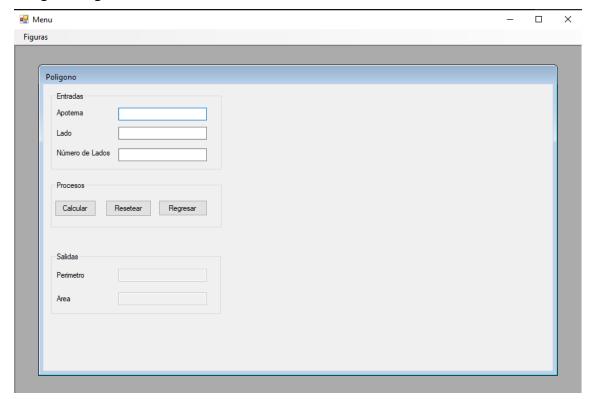
Trapecio



Circulo



Polígono Regular



```
namespace Practica_6

{
    7 referencias
    public partial class FrmRegularPoligon : Form
    {
        RegularPoligon regularPoligon = new RegularPoligon();
        private static FrmRegularPoligon instance;
        1 referencia
        public FrmRegularPoligon()...

1 referencia
    public static FrmRegularPoligon GetInstance() ...

2 referencias
    public void InitializeData()...
1 referencia
    private void FrmRegularPoligon_Load(object sender, EventArgs e)...

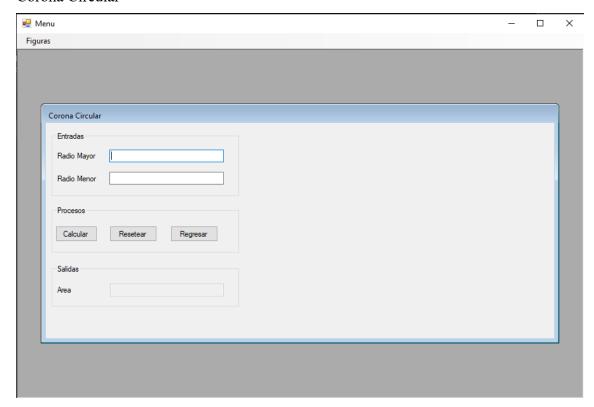
1 referencia
    private void btnReset_Click(object sender, EventArgs e)...

1 referencia
    private void btnExit_Click(object sender, EventArgs e)...

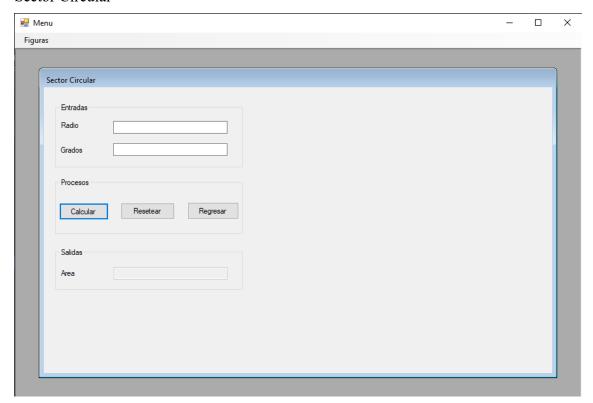
1 referencia
    private void btnCalculate_Click(object sender, EventArgs e)...

1 referencia
    private void btnCalculate_Click(object sender, EventArgs e)...
```

Corona Circular



Sector Circular



Implementación

Dentro de la implementación de la solución, se encuentran las clases de cada figura, se tiene un constructor público para inicializar las variables de la clase, cada clase tiene su método para recibir los respectivos datos de la figura, asignarlos a las variables, realizar los cálculos del perímetro y del área y por último el método para escribir dentro del formulario

Cuadrado

```
vising System;
using System.Collections.Generic;
using System.Ling;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

value namespace Practica_6
{
    private float mSide;
    private float mPerimeter;
    private float mArea;

    referencia
    public Square() ...

    1 referencia
    public void ReadData(TextBox txtSide) ...

    1 referencia
    public void CalculatePerimeter() ...

    1 referencia
    public void CalculateArea() ...

    1 referencia
    public void WriteData(TextBox txtPerimeter, TextBox txtArea) ...
}
```

Triangulo

Rectángulo

Paralelogramo

Rombo

Cometa

```
namespace Practica_6
{
    internal class Comet
    {
        private float mMajorDiagonal;
        private float mMajorDiagonal;
        private float mMinorDiagonal;
        private float mMajorSide;
        private float mMajorSide;
        private float mMajorSide;
        private float mArea;
        1 referencia
        public Comet()...

        1 referencia
        public void ReadData(TextBox txtMajorDiagonal, TextBox txtMinorDiagonal, TextBo
```

Trapecio

Circulo

```
namespace Practica_6
{
    3 referencias
internal class Circle
    {
        private float mRadius;
        private float mPerimeter;
        private float mArea;
        private const float pi = 3.1416F;
        1 referencia
        public Circle()
        1 referencia
        public void ReadData(TextBox txtRadius)...
        public void CalculatePerimeter()...
        public void CalculateArea()
        1 referencia
        public void WriteData(TextBox txtPerimeter, TextBox txtArea)
```

Polígono Regular

```
namespace Practica_6
{

3 referencias
internal class RegularPoligon
{

private float mSide;
private int mSideNumber;
private float mPerimeter;
private float mArea;

1 referencia
public RegularPoligon()...

1 referencia
public void ReadData(TextBox txtApotema, TextBox txtSide, TextBox txtSideNumber)...

1 referencia
public void CalculatePerimeter()...

1 referencia
public void CalculateArea()...

1 referencia
public void WriteData(TextBox txtPerimeter, TextBox txtArea)...

1 referencia
public void WriteData(TextBox txtPerimeter, TextBox txtArea)...
```

Corona Circular

```
namespace Practica_6
{
    3 referencias
    internal class Crown
{
        private float mMajorRadius;
        private float mMinorRadius;
        private float mArea;
        private const float pi = 3.1416F;

        1 referencia
        public Crown()...

        1 referencia
        public void ReadData(TextBox txtMajorRadius, TextBox txtMinorRadius)...

        1 referencia
        public void CalculateArea()...

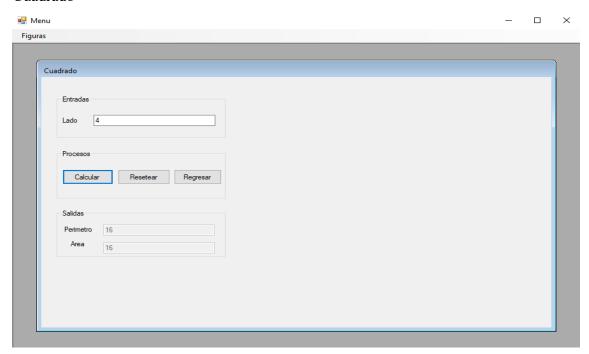
        1 referencia
        public void WriteData(TextBox txtArea)...
```

Sector Circular

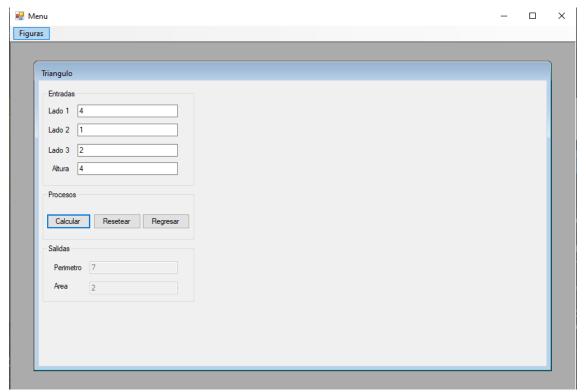
Pruebas y Resultados

Después de haber implementado las clases de cada figura en su respectivo formulario, se ingresaron valores en cada figura para comprobar que realizara los cálculos correctos del área y del perímetro, también se ve que los datos se muestran de manera correcta en los textbox correspondientes

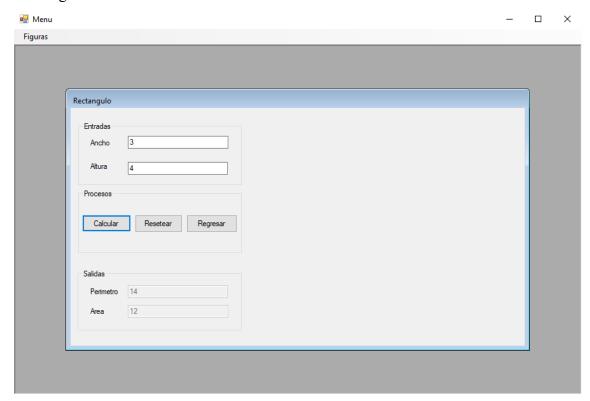
Cuadrado



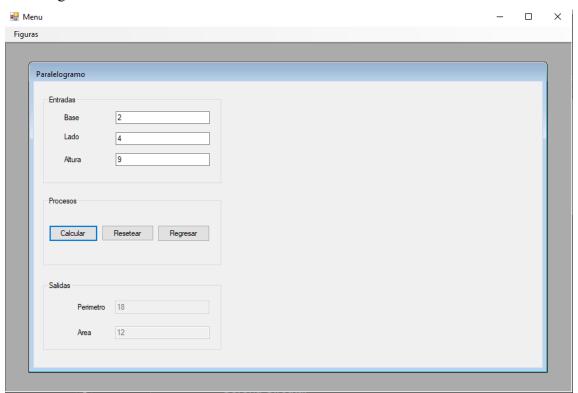
Triangulo



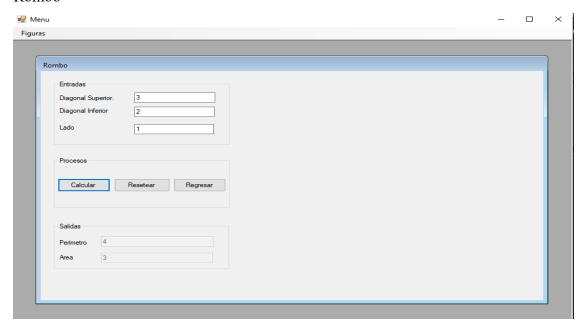
Rectángulo



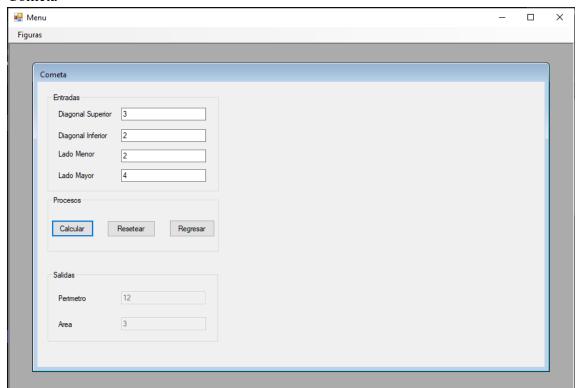
Paralelogramo



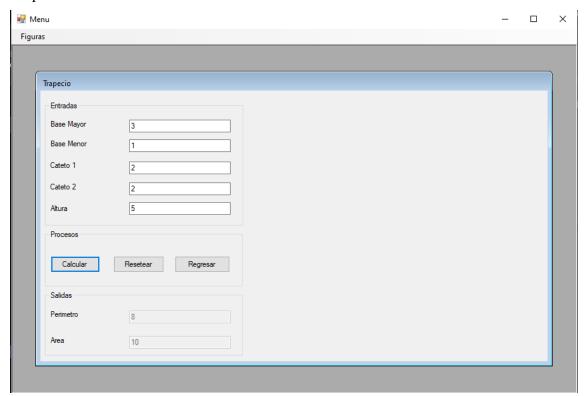
Rombo



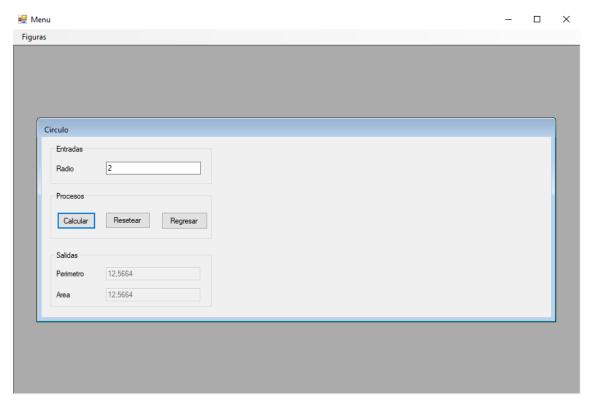
Cometa



Trapecio



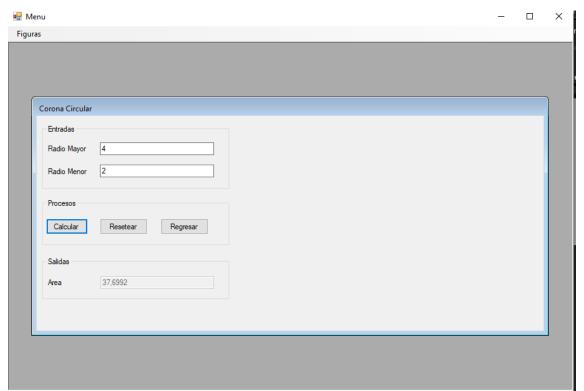
Circulo



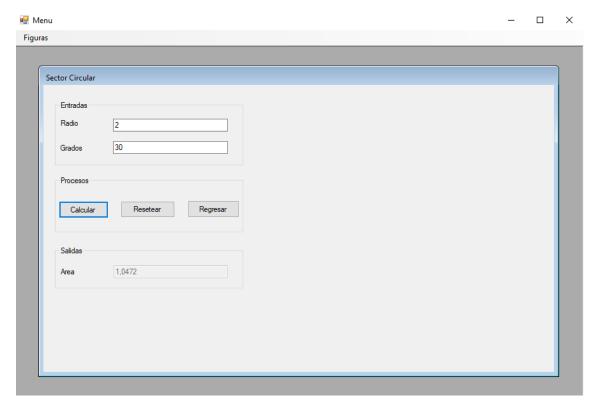
Polígono Regular



Corona Circular



Sector Circular



Conclusiones

- La aplicación desarrollada en C# demostró ser efectiva para realizar cálculos precisos de áreas y perímetros de diversas figuras geométricas, facilitando el aprendizaje y la resolución de problemas matemáticos.
- El uso de formularios de Windows Forms permitió crear una interfaz interactiva y fácil de usar, adecuada tanto para usuarios principiantes como para avanzados. La separación de formularios por figura geométrica simplificó la experiencia de navegación.
- La implementación del patrón Singleton garantizó que solo se abriera una instancia de cada formulario, mejorando el manejo de los recursos y evitando redundancias.

Recomendaciones

- Ampliar la aplicación para incluir cálculos de figuras más complejas como polígonos irregulares, elipses o incluso figuras tridimensionales como prismas y cilindros.
- Asegurarse de que los datos ingresados por los usuarios sean válidos mediante validaciones avanzadas, como restricciones para evitar conflictos con los cálculos, especialmente en figuras como triángulos
- Manejar la lógica de programación de mejor manera para que a la hora de refactorizarlo no nos de tantos problemas.

Bibliografía

Microsoft. (n.d.). *Windows Forms Overview*. Microsoft Learn. Recuperado el 11 de noviembre de 2024, de https://learn.microsoft.com/en-us/dotnet/desktop/winforms/