Departamento de Computación

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

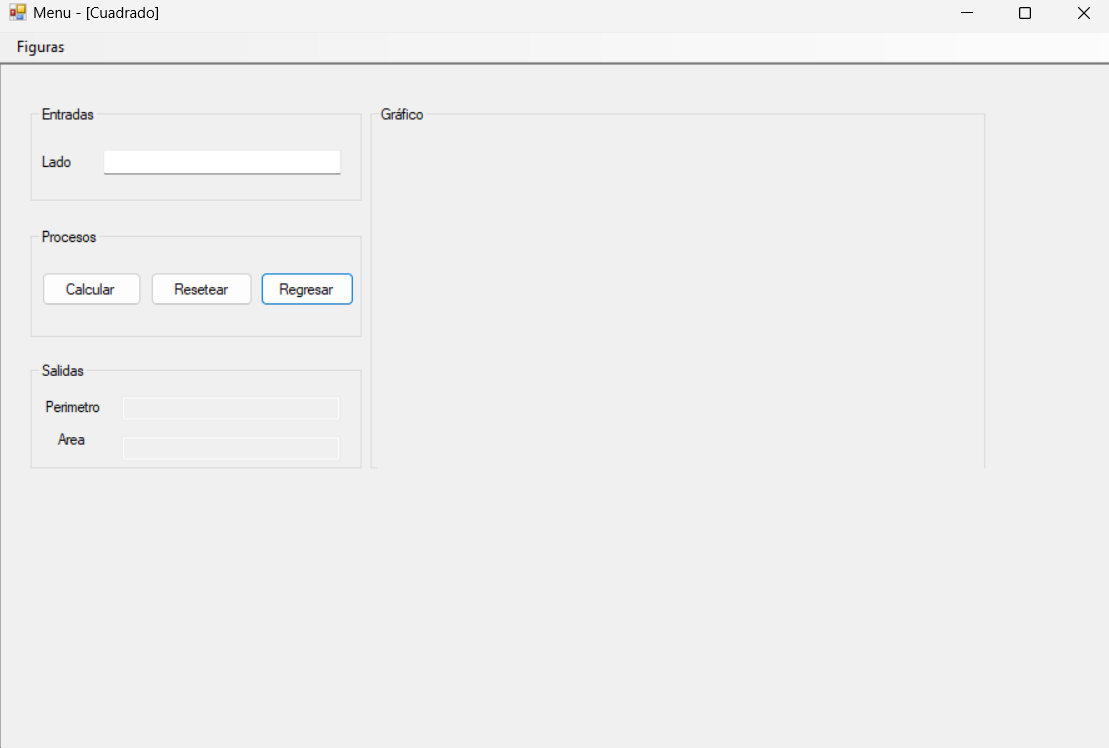
**Formularios**

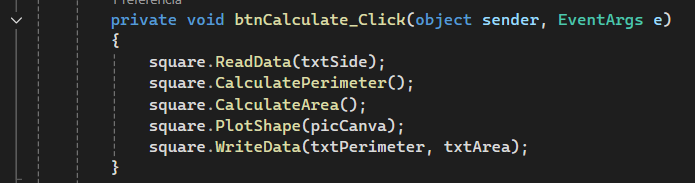
txtNombredelOutput /\*Objeto Tipo TextBox\*/

# **Diseños de la solución**

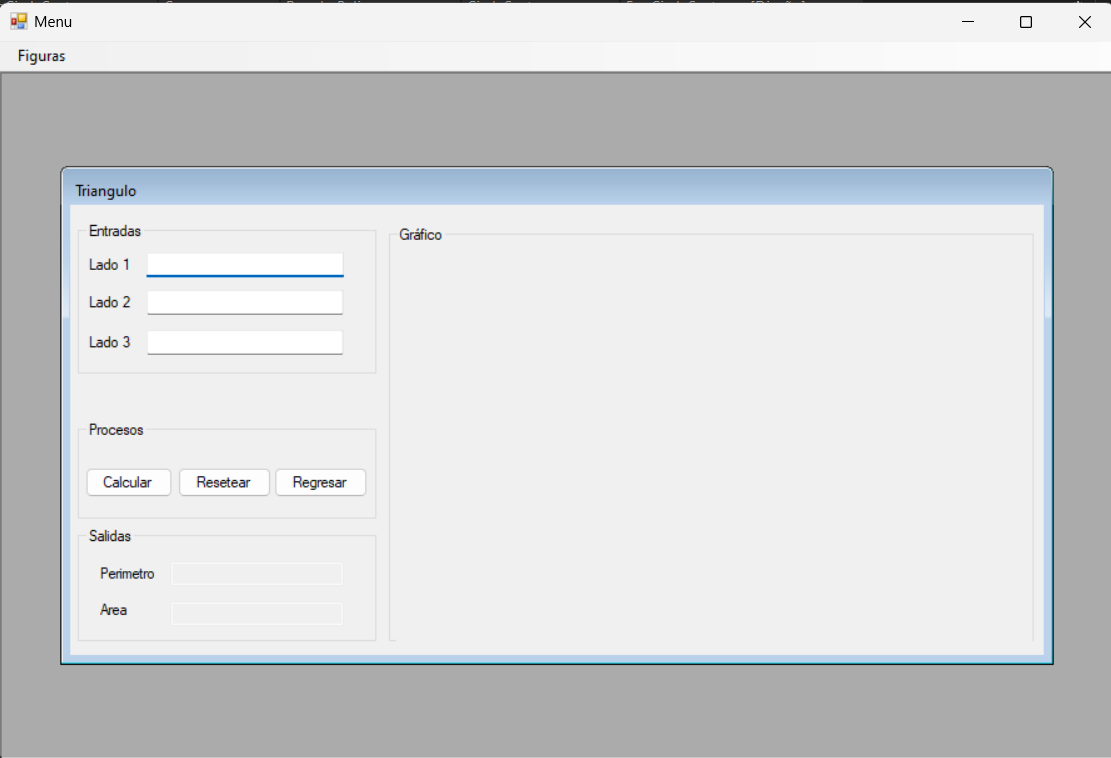
Dentro de los diseños de la solución, cada figura cuenta con sus respectivos textbox para las entradas de datos y va a depender de cada figura, también se aplica un patrón de diseño singleton en los formularios de cada figura para que se cree una única instancia de esta forma evitando formularios repetidos,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

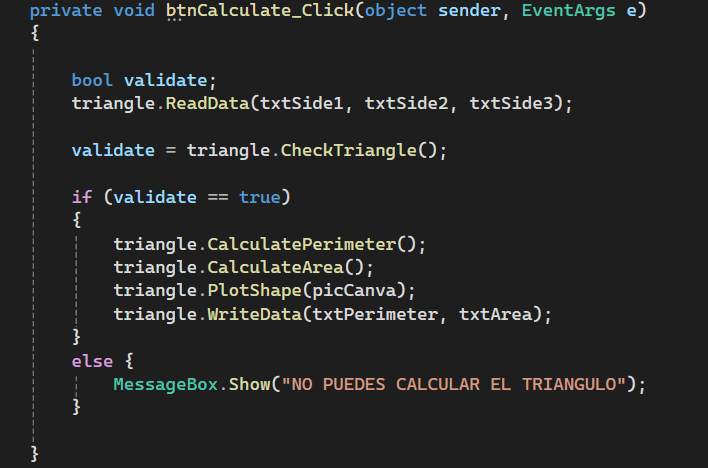
## Cuadrado



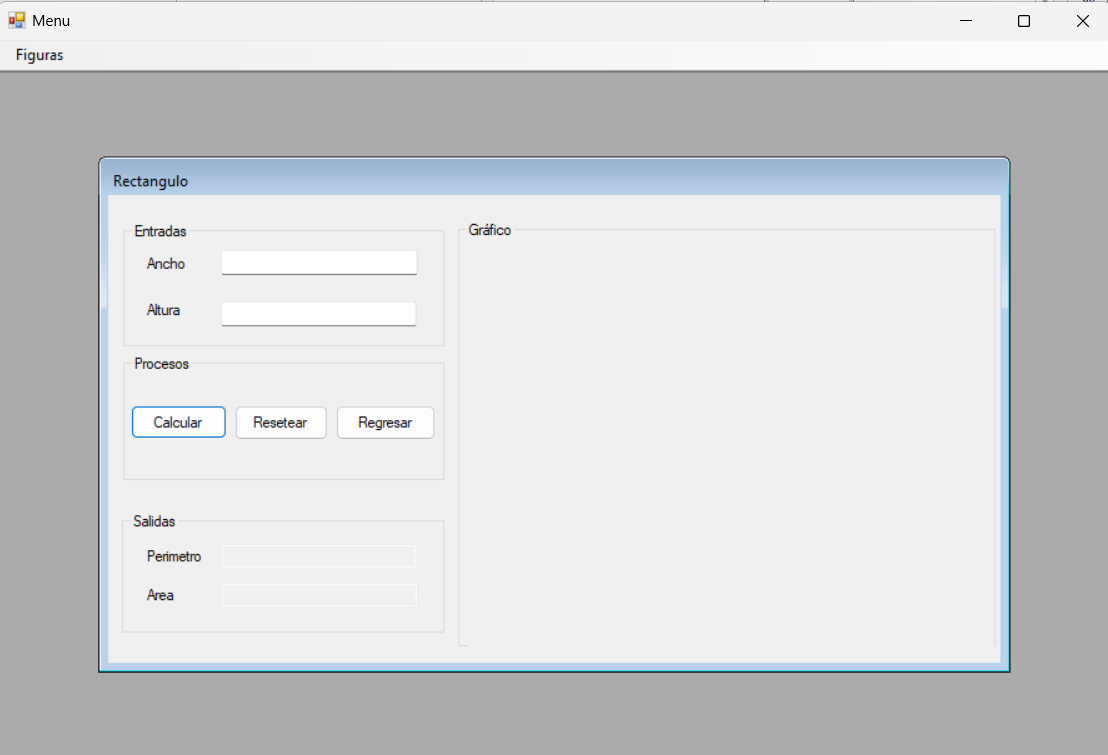


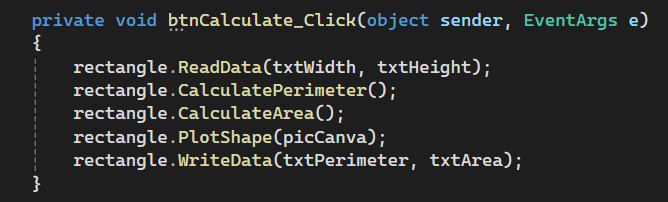
## Triangulo



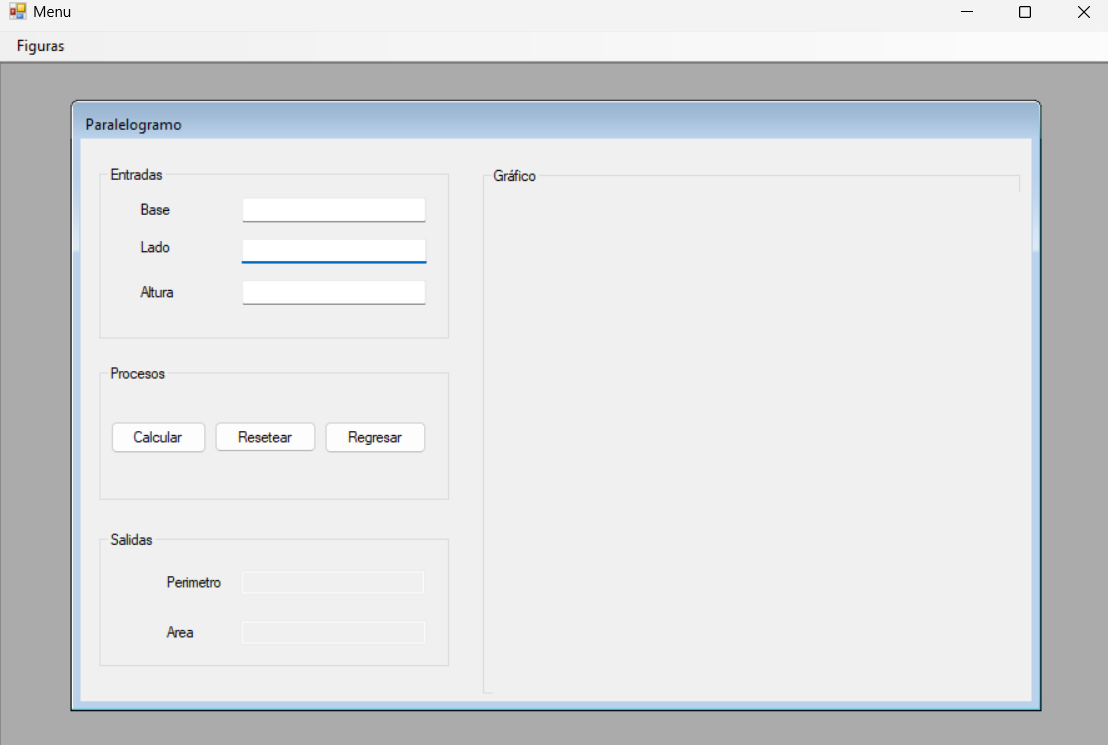


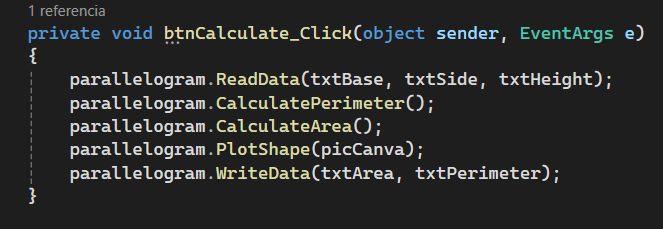
## Rectángulo



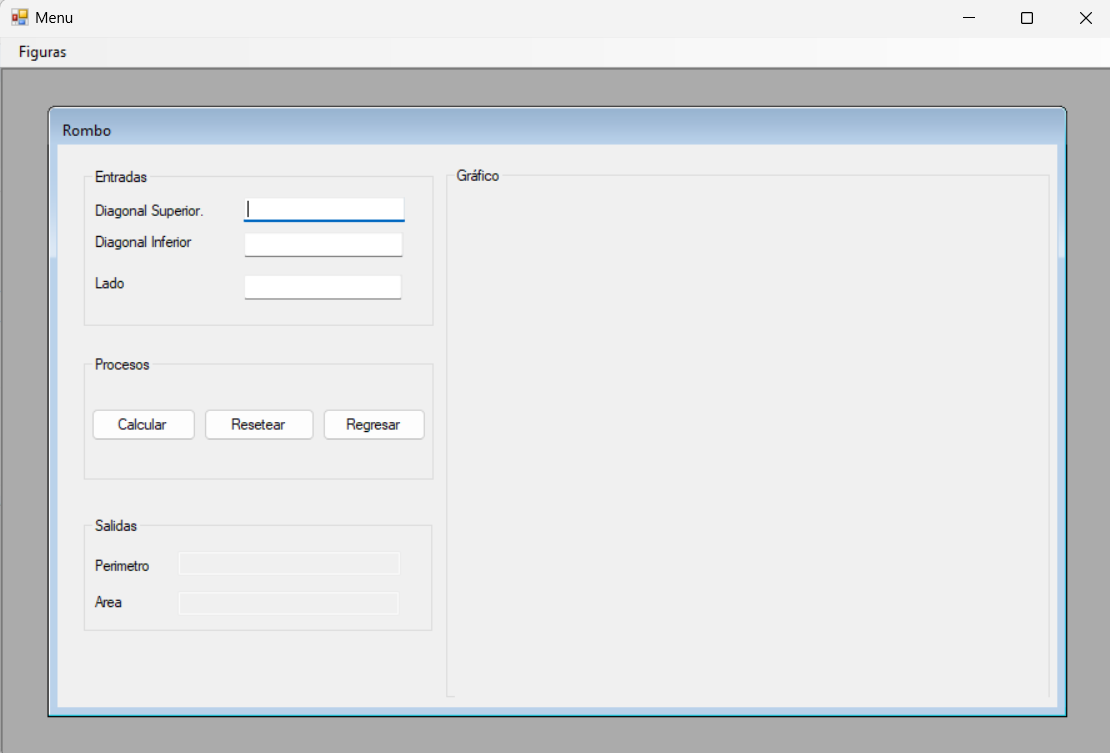


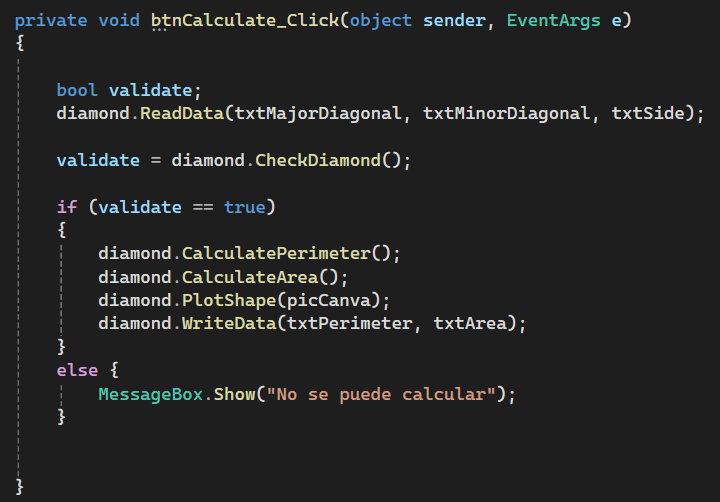
## Paralelogramo



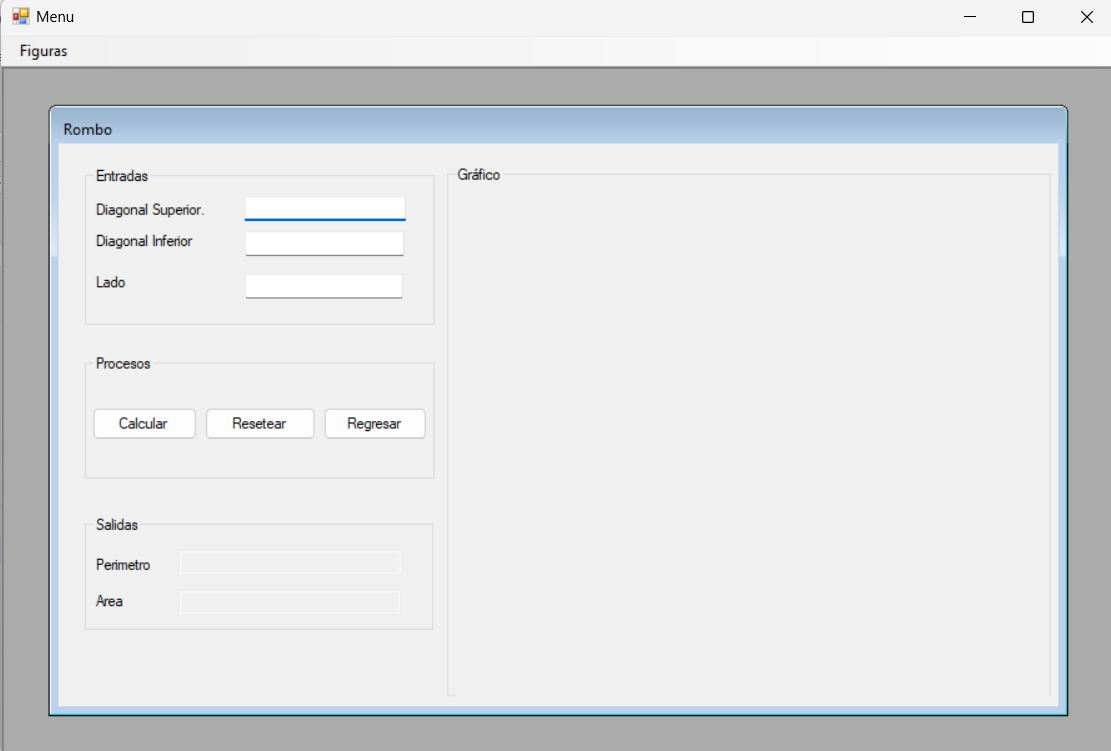


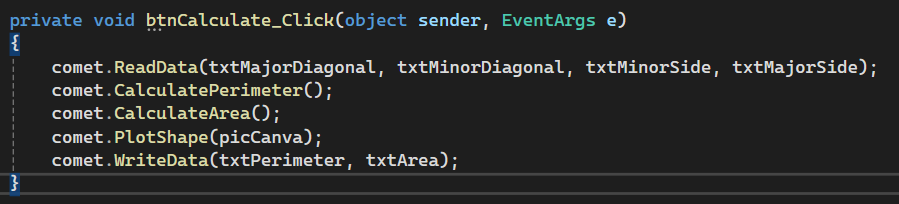
## Rombo



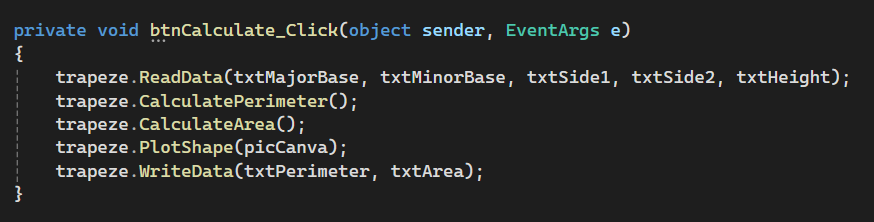


## Cometa

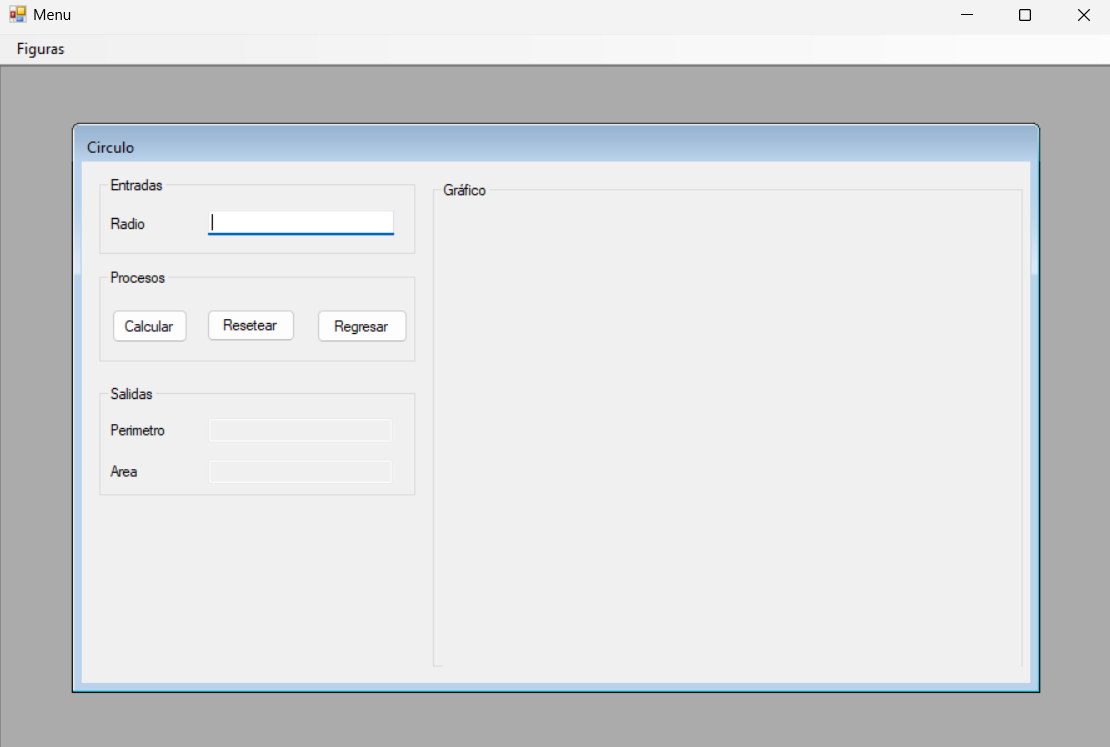


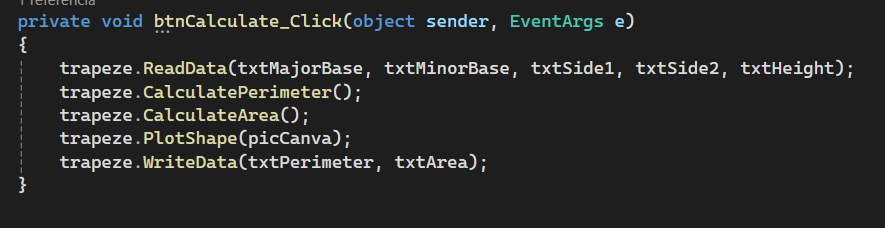


## Trapecio

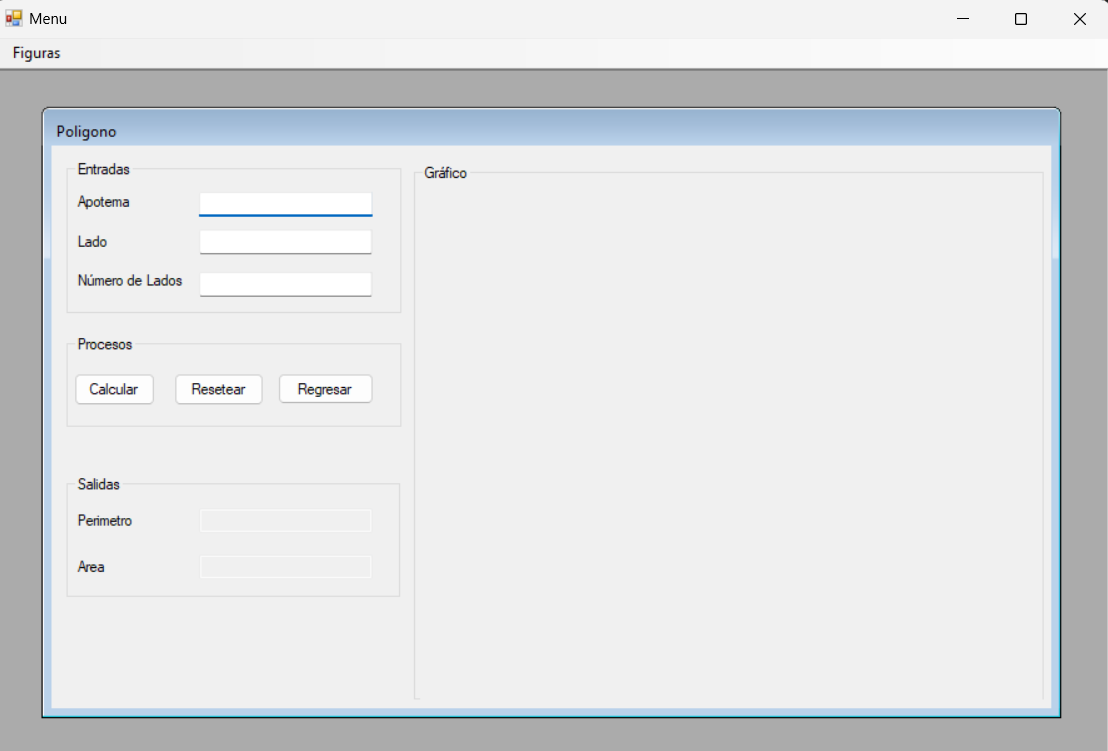


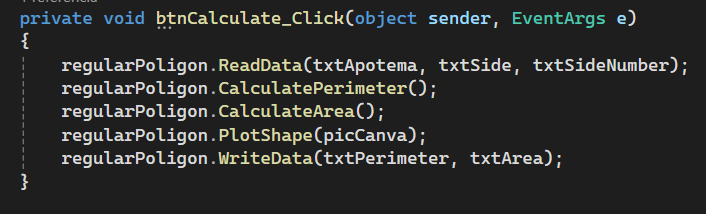
## Circulo



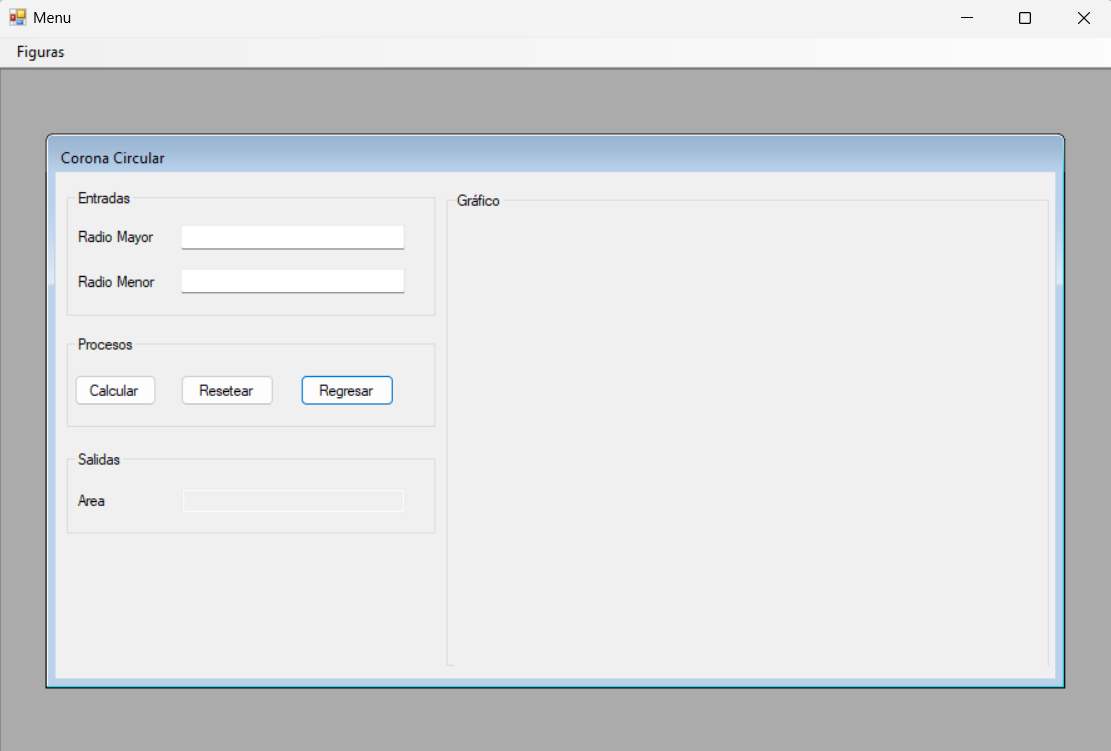


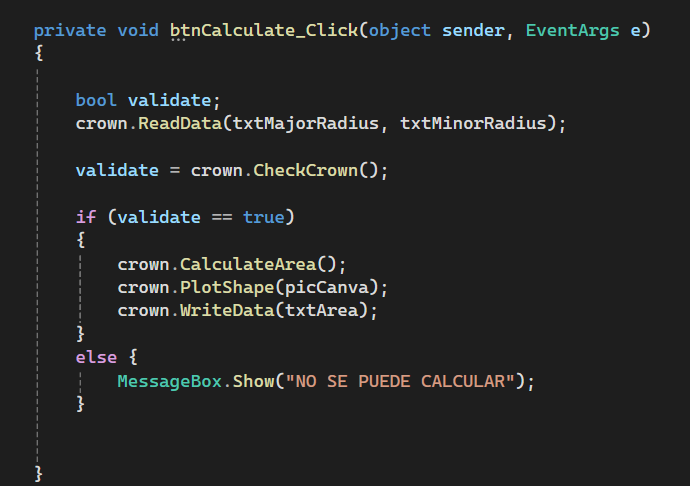
## Polígono Regular



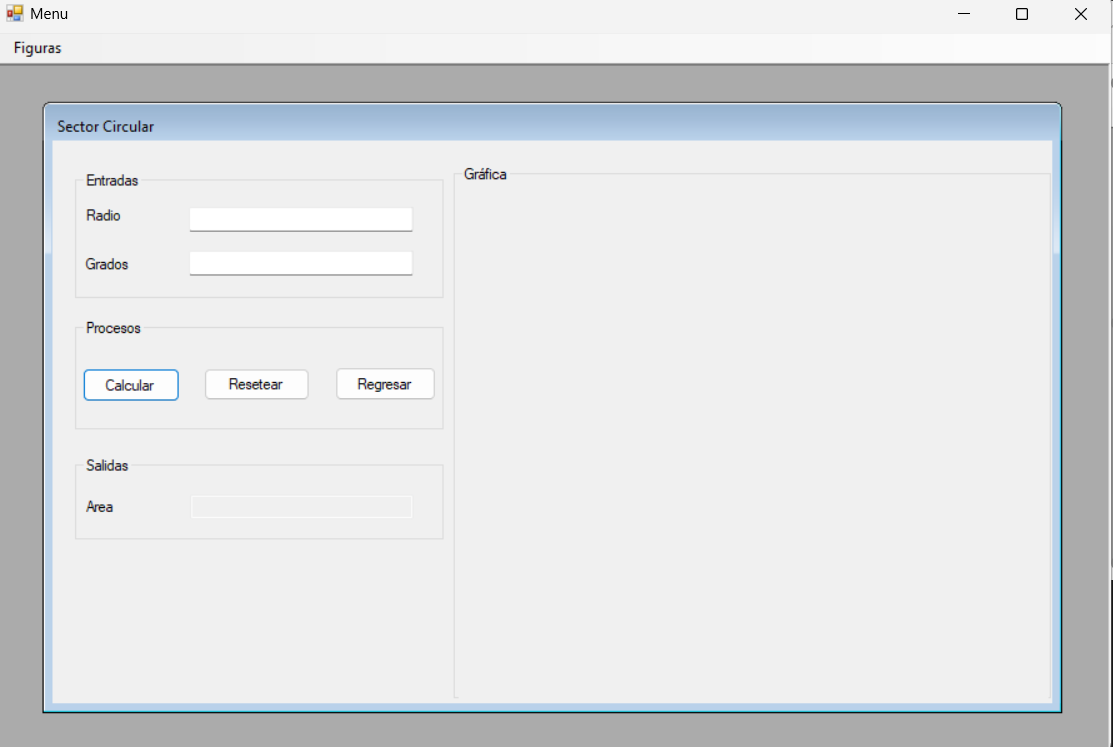


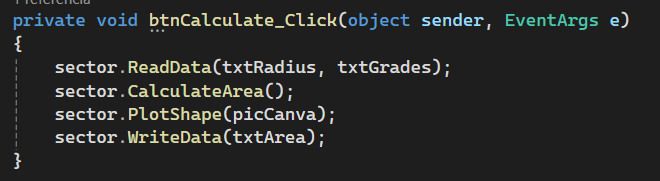
## Corona Circular





## Sector Circular

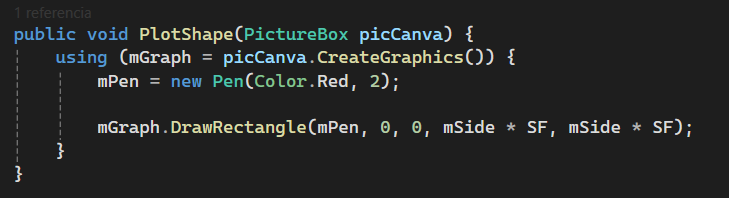




# **Implementación**

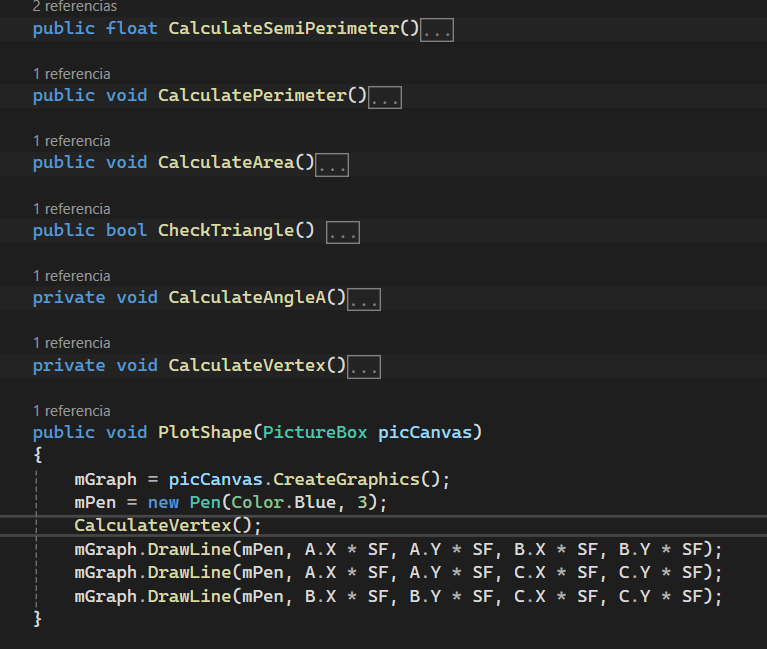
Dentro de la implementación de la solución, se encuentran las clases de cada figura, se tiene un constructor público para inicializar las variables de la clase, cada clase tiene su método para recibir los respectivos datos de la figura, asignarlos a las variables, realizar los cálculos del perímetro y del área y el método para escribir dentro del formulario, se añadió la función para dibujar cada figura, en cada uno va a estar en captura los métodos.

## Cuadrado

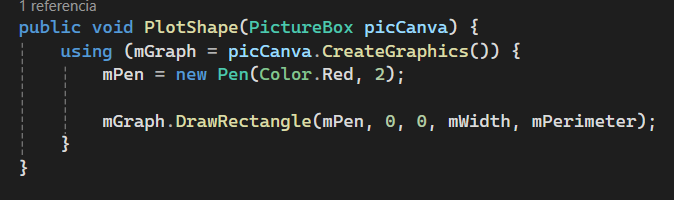


## Triangulo

Todos los métodos que se muestran sirven para poder dibujar el triangulo.

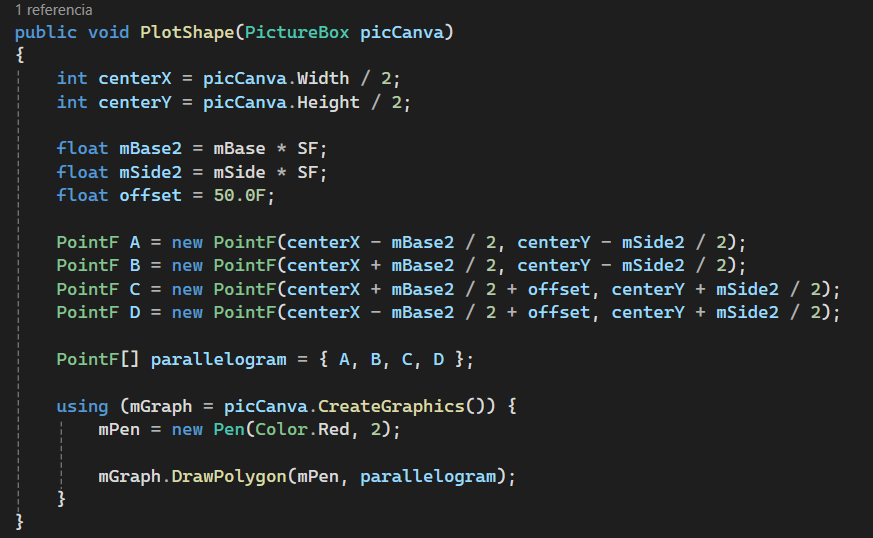


## Rectángulo



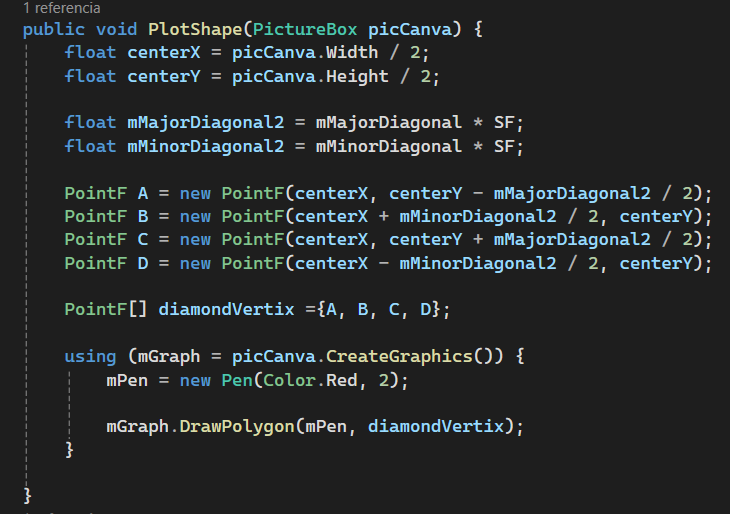
## Paralelogramo

Se definen puntos para los vértices para luego crear un arreglo de puntos flotantes y luego mandar a dibujar con el método DrawPolygon.



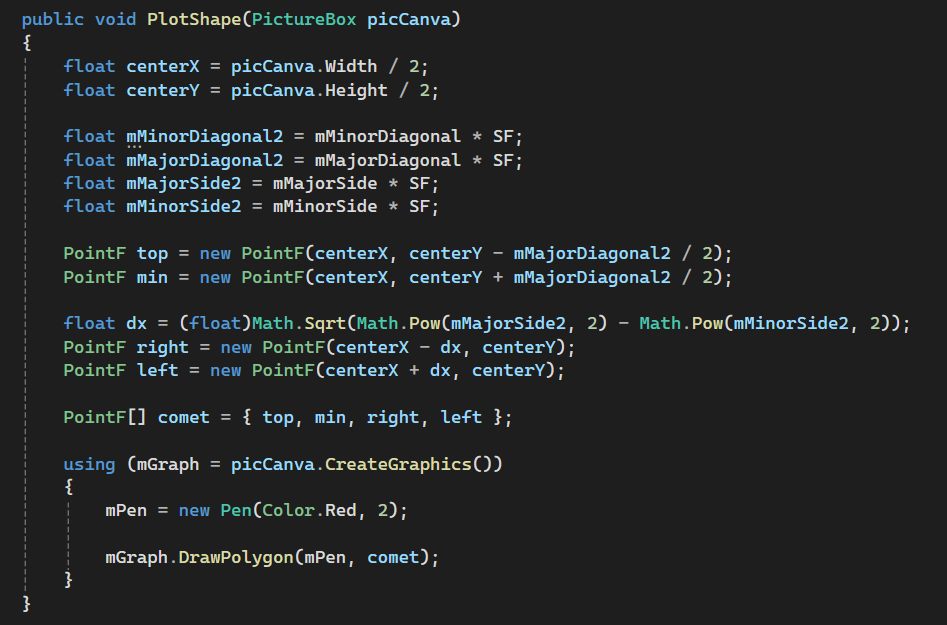
## Rombo

Al igual que la anterior figura se definen los vértices del rombo para mandar a dibujar con el mismo método de la biblioteca Graphics.



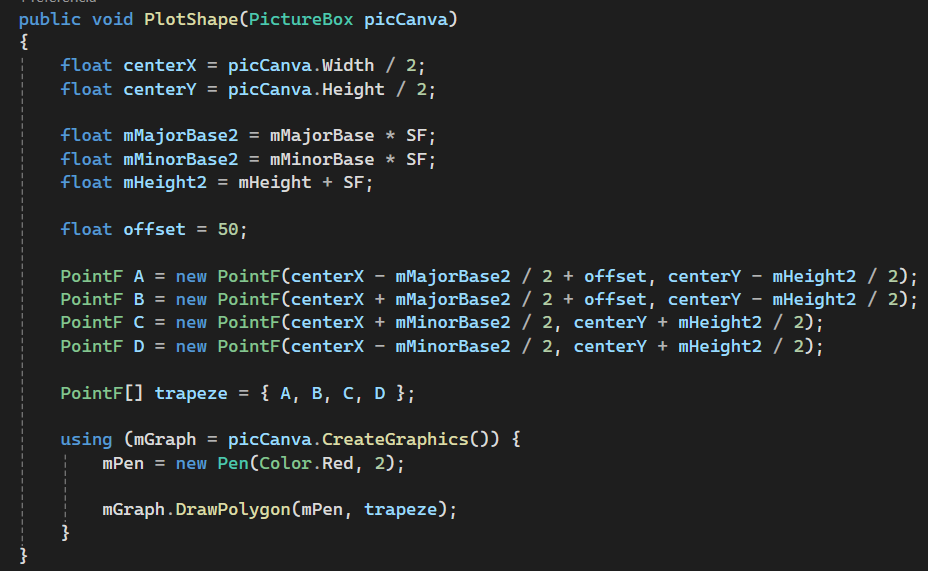
## Cometa

Con las diagonales y los dos lados más geometría básica se definen los vértices de la cometa para que se dibuje y con ayuda del teorema de Pitágoras se calcula la hipotenusa que sirve como un lado de la cometa



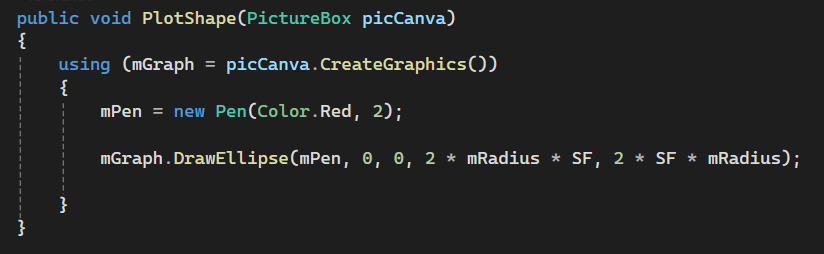
## Trapecio

Al igual que las anteriores figuras se definen puntos para los vértices y como nueva variable se tiene el offset que sirve como para desplazar la parte de debajo de la figura para formar el trapecio.



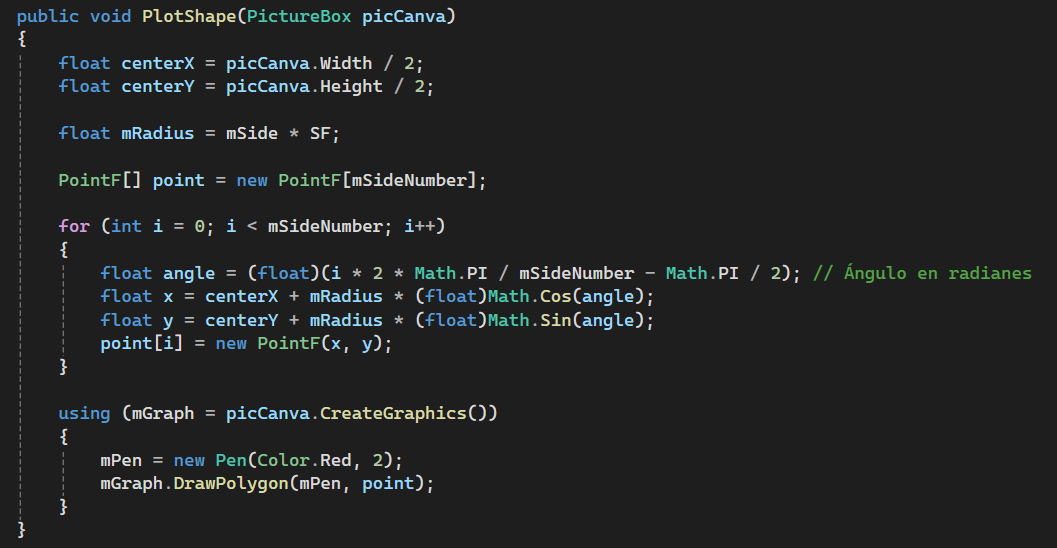
## Circulo

Se dibuja normalmente el circulo teniendo en cuenta el dato del radio



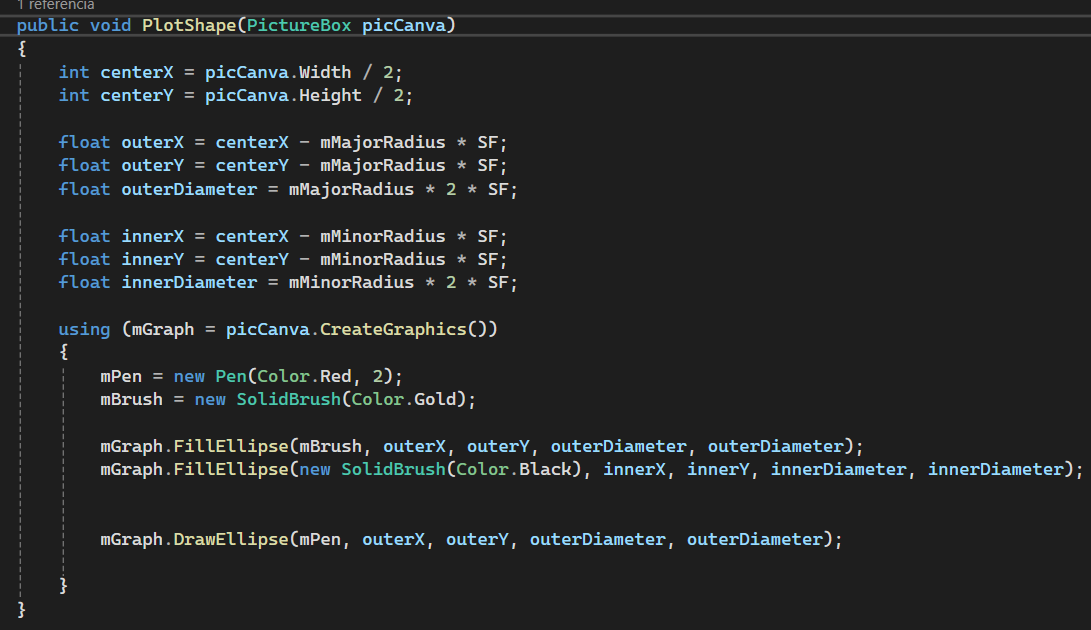
## Polígono Regular

Se toma como radio, la longitud del lado del polígono, y el numero de lados se le toma para aplicarlo en el bucle for para definir la figura del polígono regular.



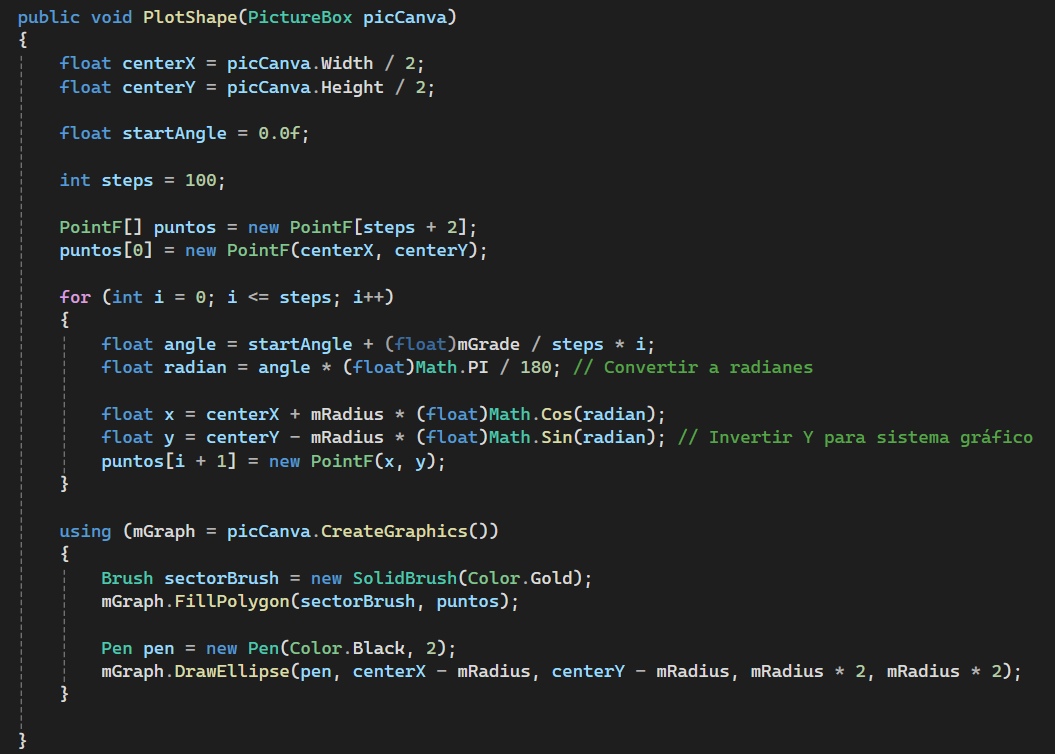
## Corona Circular.

Se dibujan dos círculos con datos del radio mayor y menor, se rellena el interior de estos dos.



## Sector Circular

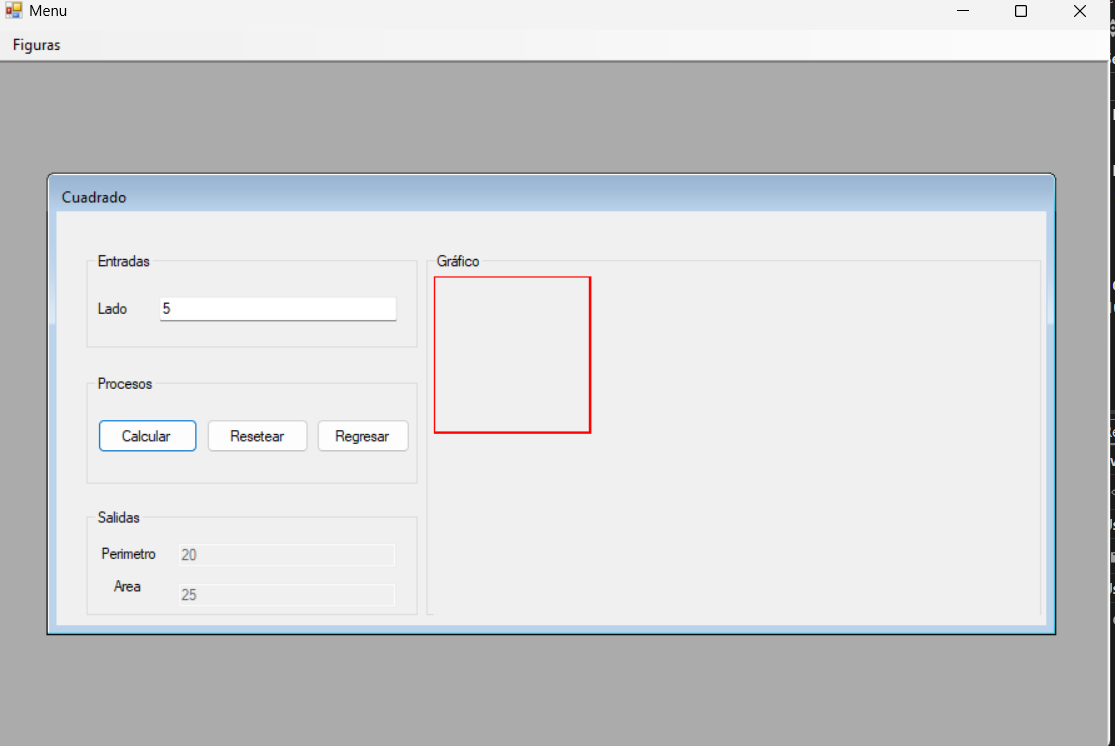
Con el for se calcula la parte del sector que debe pintarse dentro del circulo que se dibuja.



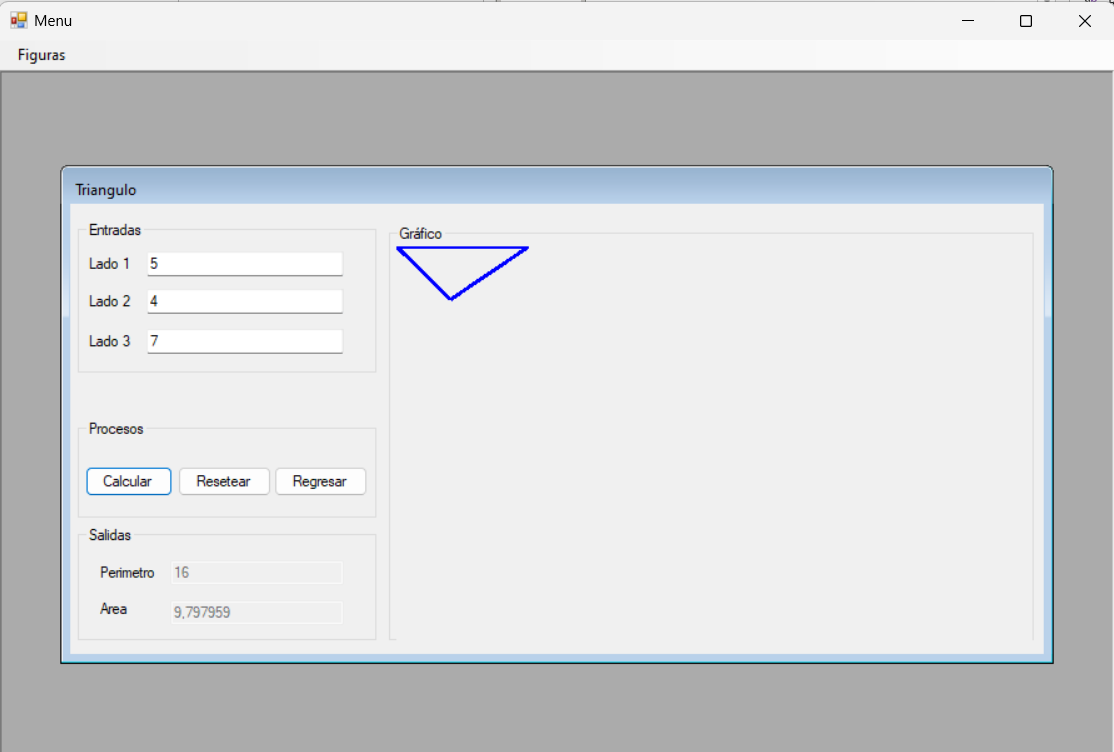
# **Pruebas y Resultados**

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

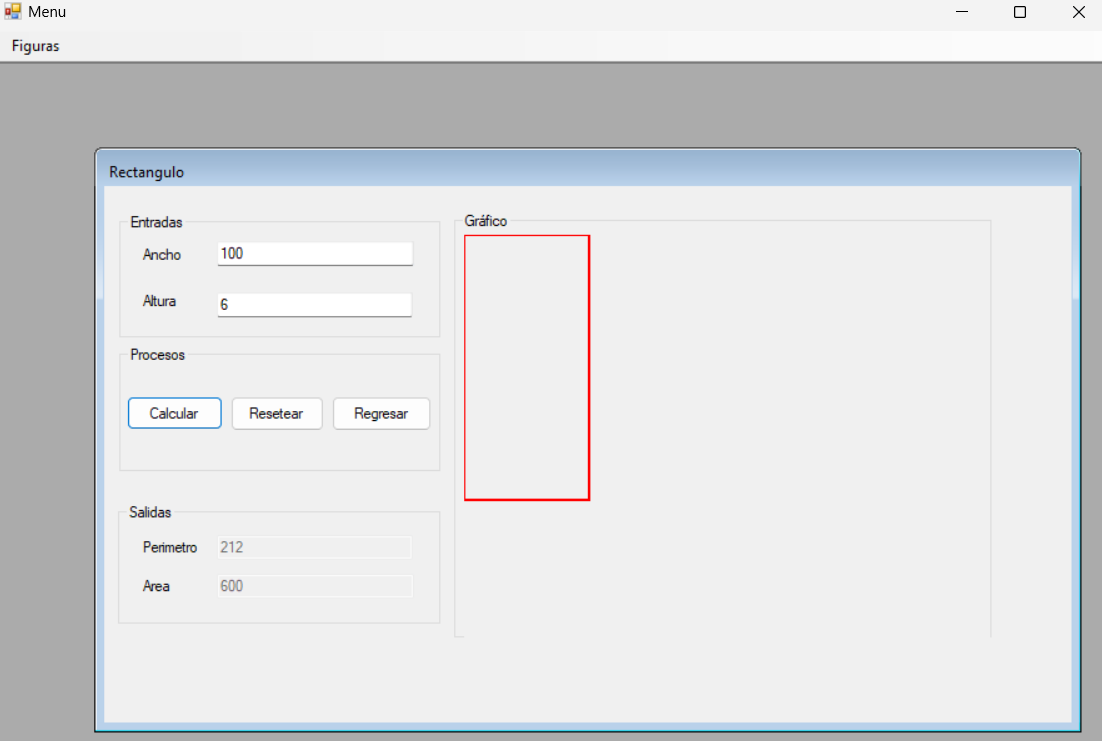
## Cuadrado



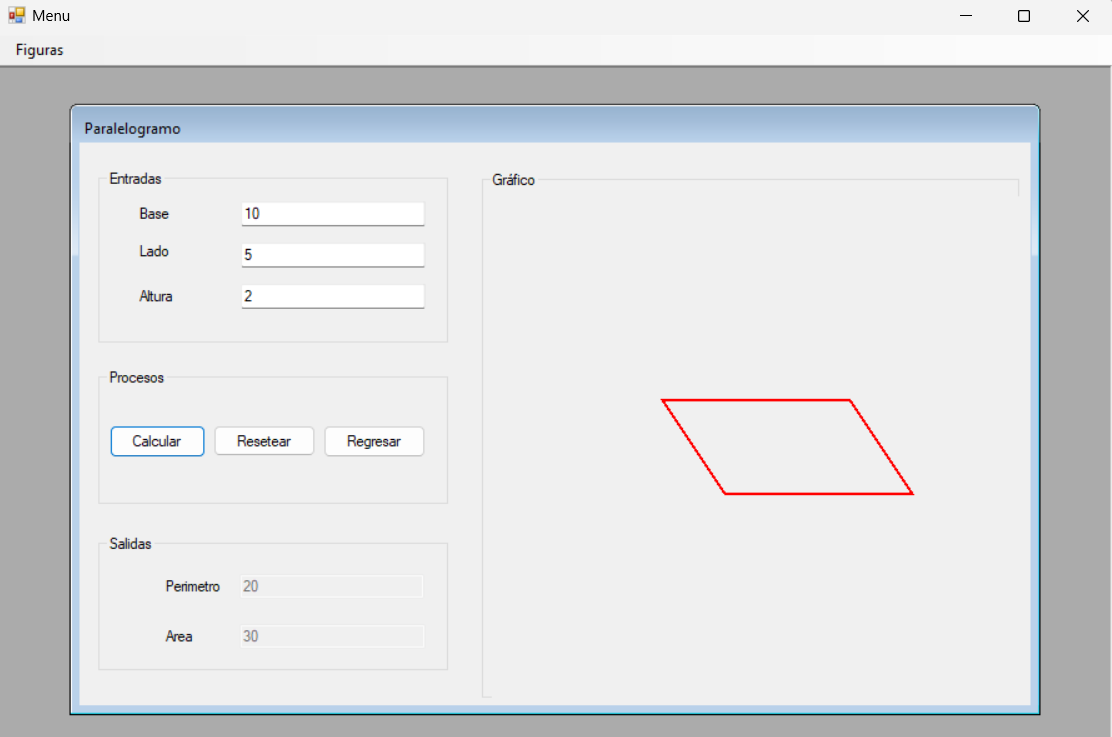
## Triangulo



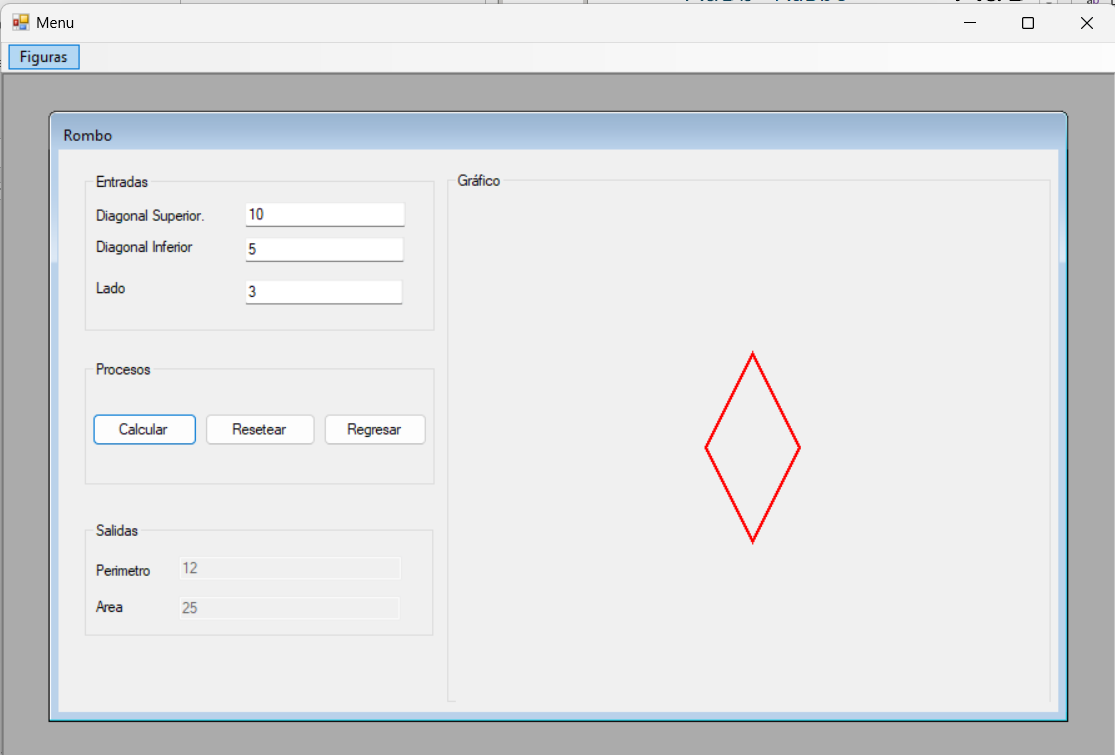
## Rectángulo



## Paralelogramo

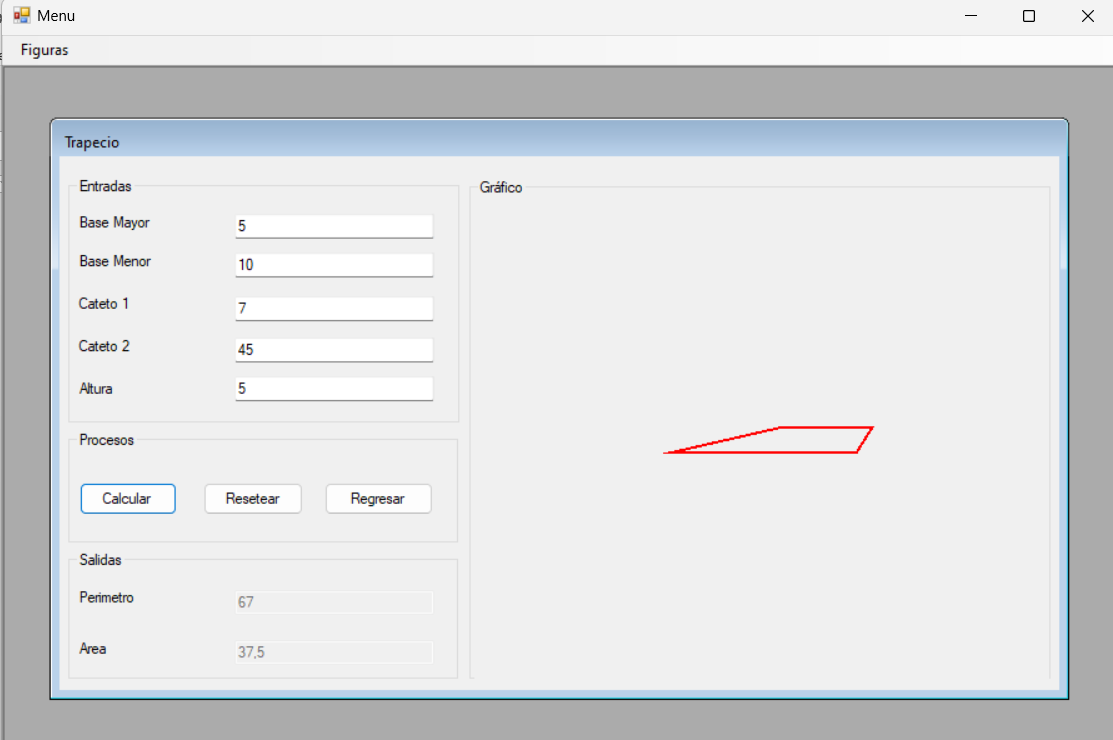


## Rombo

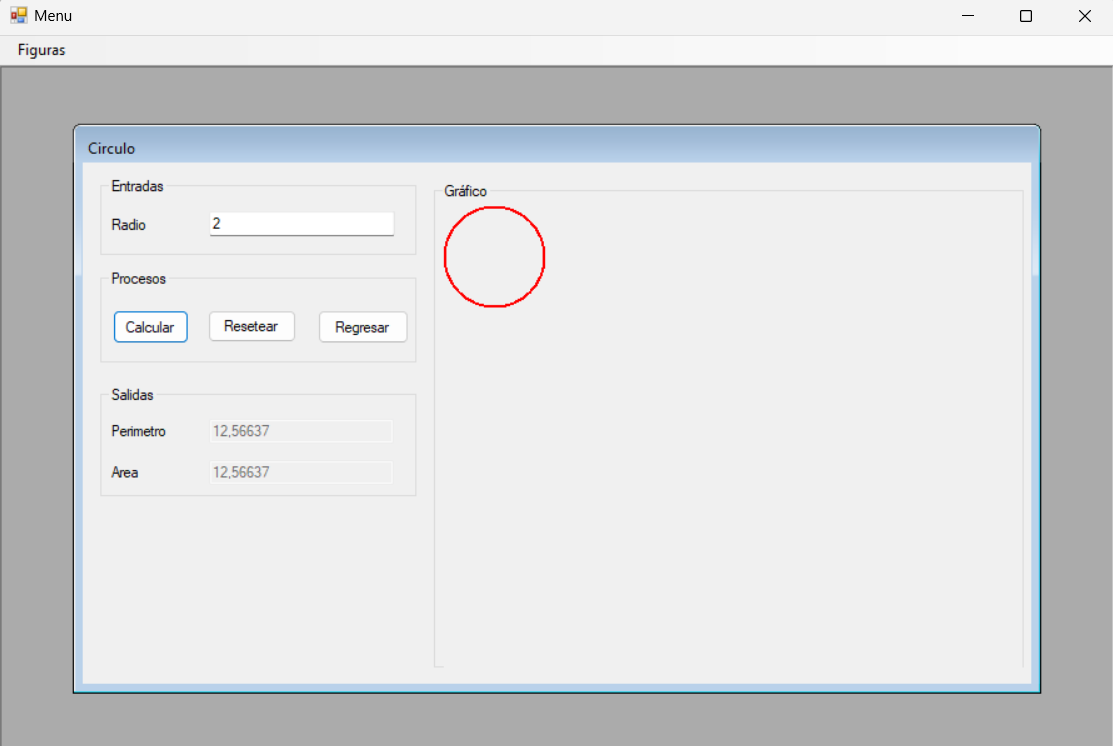


## Cometa

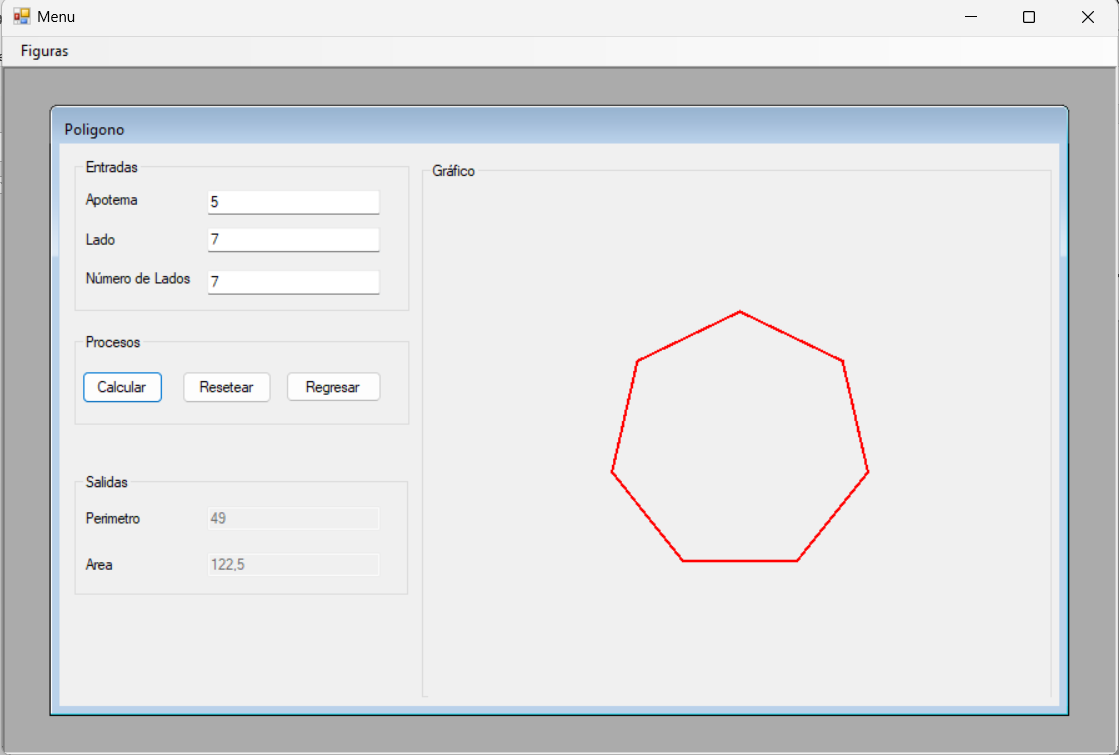
## Trapecio



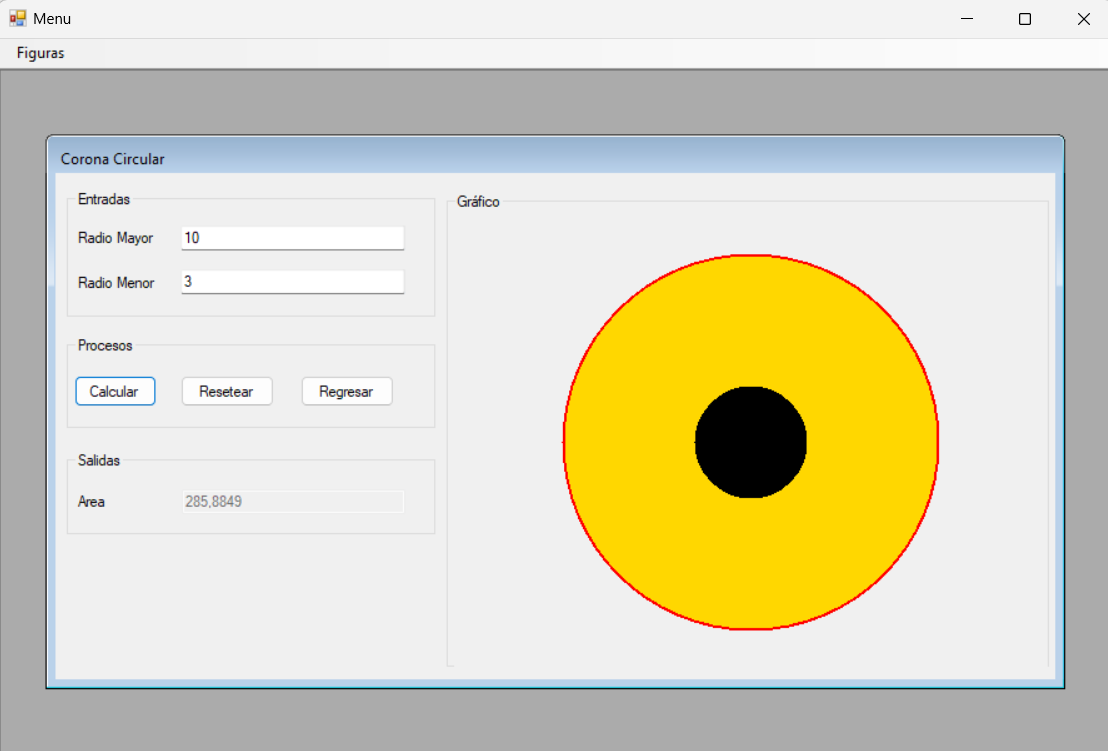
## Circulo



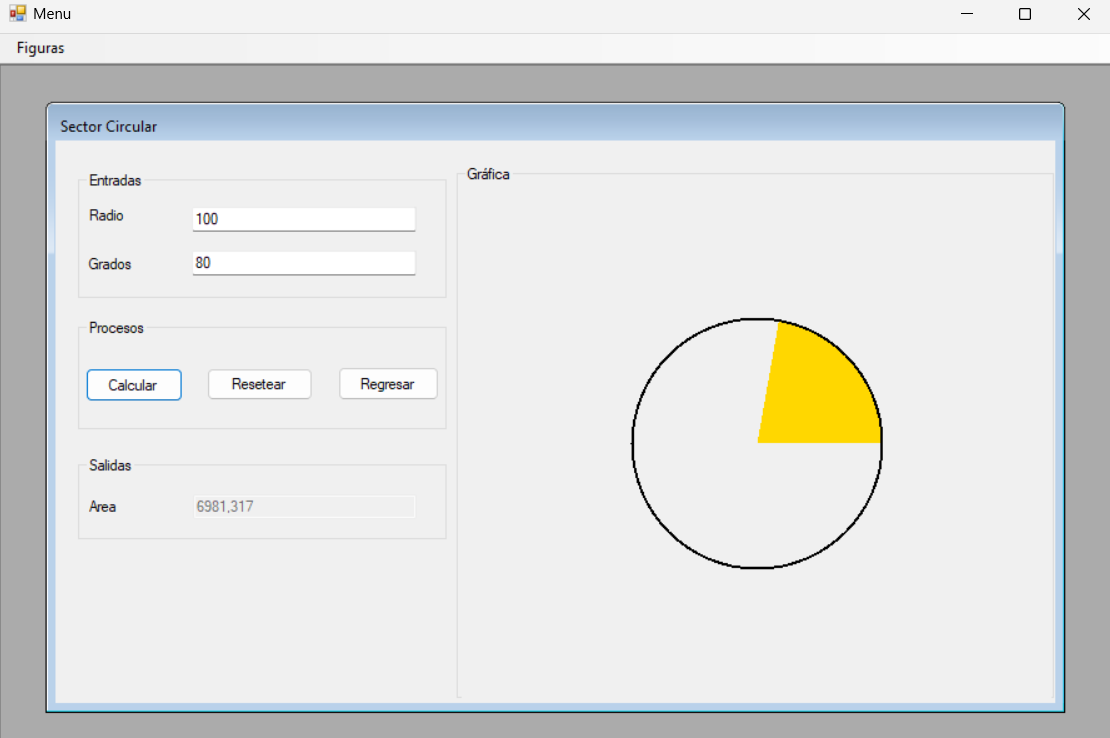
## Polígono Regular



## 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/>