

Actividad |3| Menú de Áreas de Figuras Geométricas Aplicación 3.

Desarrollo de Aplicaciones Móviles III.

Ingeniería en Desarrollo de Software.



TUTOR: Sandra Luz Lara Devora.

ALUMNO: Ramón Ernesto Valdez Felix.

FECHA: 06/08/2025.

Introducción.	3
Descripción.	3
Justificación.	4
Desarrollo.	5
Codificación.	5
Prueba de la aplicación.	13
Conclusion.	18
Referencias.	19

Introducción.

En esta actividad final de la materia de Desarrollo de Aplicaciones Móviles III, donde tenemos que realizar la documentación de la creación de una aplicación la cual debe contar con un menú de opciones para calcular distintas áreas de figuras geométricas. En el ámbito de la programación, la resolución de problemas prácticos es una de las tareas más comunes. Una excelente manera de familiarizarse con un lenguaje como Swift es a través de la creación de herramientas útiles y funcionales. Con este objetivo en mente, se ha desarrollado una aplicación que simplifica el cálculo de áreas para diversas figuras geométricas. Esta herramienta está diseñada para ser intuitiva y fácil de usar, permitiendo a los usuarios seleccionar la figura de su interés, como un cuadrado, rectángulo, triángulo o círculo, e introducir las medidas necesarias para obtener el resultado de manera rápida y precisa. Esta aplicación no solo demuestra la capacidad de Swift para manejar la interacción con el usuario a través de menús de opciones, sino que también refuerza conceptos fundamentales de lógica y matemáticas aplicadas.

Descripción.

Como proyecto final para el curso de Desarrollo de Aplicaciones Móviles III, nos enfocamos en la documentación en la cual se ha desarrollado una aplicación en Swift diseñada para ser una herramienta educativa y funcional. Esta aplicación, centrada en la Programación Orientada a Objetos (POO), permite a los usuarios calcular de forma interactiva el área de diversas figuras geométricas a través de un menú sencillo y claro.

- La interfaz de la aplicación, construida para ser intuitiva, guía al usuario paso a paso:
- Un menú principal muestra las figuras disponibles: cuadrado, rectángulo, triángulo y círculo.
- Al seleccionar una opción, la aplicación solicita las dimensiones necesarias (lados, base, altura o radio).

Finalmente, presenta el resultado del área de forma precisa, facilitando la comprensión y el uso.

Este proyecto demuestra no solo la habilidad para desarrollar aplicaciones funcionales en el entorno de Swift, sino que también solidifica la comprensión de conceptos clave como la encapsulación, el manejo de la entrada del usuario y la aplicación de fórmulas matemáticas de manera práctica.

Justificación.

Este proyecto se justifica como la culminación práctica del curso de Desarrollo de Aplicaciones Móviles III, aplicando los conocimientos teóricos en un entorno real. El desarrollo de esta aplicación de cálculo de áreas no solo cumple con el requisito de crear una herramienta funcional en Swift, sino que también sirve como una demostración clara de la Programación Orientada a Objetos (POO). La estructura del código, dividida en clases para cada figura geométrica, ilustra perfectamente conceptos como la encapsulación, el reuso de código y la modularidad, facilitando así su mantenimiento y escalabilidad. La aplicación, al guiar al usuario de manera intuitiva, valida nuestra capacidad para diseñar interfaces amigables y manejar la interacción del usuario de manera efectiva. En esencia, este proyecto es una prueba tangible de las habilidades adquiridas en el curso, traduciendo la teoría en una solución práctica y bien estructurada.

Entre otros puntos adicionales a utilizar en la justificación para la realización de la documentación de esta actividad que son los siguientes:

- PDF de esta actividad en el portafolio GitHub.
- Descarga el script genera de swift y subirlo como parte de la actividad a GitHub
- Anexa link de GitHub en documento.
- Utilizar la herramienta del entorno de trabajo online Replit o XCode que se usarán con lenguaje de programación: Swift.

Desarrollo.

En esta segunda actividad de la materia continuamos con el desarrollo de una aplicación de control de inventario, aquí tomaremos en cuenta que utilizaremos la consola de programación online Repli con el cual utilizaremos el lenguaje de programación Swift. Esto será documentado en los puntos siguientes de este documento donde nuestro propósito central es garantizar una interpretación de manera correcta del desglose y uso del script de la aplicación a crear.

Link: GitHub.

Link: Script.

Codificación.

En este punto de la actividad realizaremos un breve descripción de los bloques de líneas del script de la aplicación de inventario que permitirá el registro, almacenaje de los registros, ver los artículos y hacer consultas individuales de artículos o productos.

Codificación de app.

El objetivo principal de este código es calcular y mostrar el área de diferentes figuras geométricas: un cuadrado, un rectángulo y, parcialmente, un triángulo. El diseño del código utiliza la Programación Orientada a Objetos (POO).

Desglose por Clase

class Cuadrado

private var side: Double = 0.0: Declara una variable llamada side (lado), de tipo Double, que almacenará la longitud del lado del cuadrado. El modificador private significa que esta variable solo puede ser accedida desde dentro de la clase Square, lo que ayuda a mantener los datos seguros.

init(): Este es el constructor de la clase. Se ejecuta automáticamente cada vez que se crea un nuevo

objeto Square.

`print(...)`: Muestra un mensaje en la consola para el usuario.

`self.side = getInput(...)`: Aquí el programa le pide al usuario que ingrese la longitud del lado. La función `getInput` (que no se muestra en tu código, pero se puede inferir) se encarga de leer lo que el usuario escribe. El valor ingresado se guarda en la variable `side`.

`func calculateArea() -> Double`: Esta es una función que calcula el área del cuadrado.

`return side * side`: Retorna el resultado de multiplicar el lado por sí mismo.

`func displayArea()`: Esta función es responsable de mostrar el área calculada.

`let area = calculateArea()`: Llama a la función `calculateArea()` para obtener el valor del área y lo guarda en una constante llamada `area`.

`print(...)`: Muestra el resultado final al usuario, formateando el número para que tenga solo dos decimales (`.2f`).

`class Rectangulo`

Esta clase funciona de manera muy similar a `Square`, pero está adaptada para un rectángulo.

`private var base: Double = 0.0` y `private var height: Double = 0.0`: Declara dos variables para la base y la altura del rectángulo.

`init()`: El constructor solicita al usuario la base y la altura, y los almacena en las variables correspondientes.

`func calculateArea() -> Double`: Calcula el área del rectángulo multiplicando la base por la altura (`base * height`).

`func displayArea()`: Muestra el área del rectángulo en la consola, también formateada a dos decimales.

```
class Triangulo
```

El fragmento de código solo muestra el inicio de la clase `Triangle`. Se puede inferir que su propósito es calcular el área de un triángulo.

`private var base: Double = 0.0` y `private var height: Double = 0.0`: De manera similar a `Rectangle`, se declaran variables para la base y la altura.

Se esperaría que esta clase también tenga un constructor (`init()`), una función para calcular el área (`calculateArea()`, que en este caso sería $\text{base} * \text{height} / 2$), y una función para mostrar el resultado (`displayArea()`).

```

1  import Foundation
2
3  // Define la estructura para cada figura geométrica, encapsulando la lógica, para calcular su área. Esto hace el código más modular y fácil de
4  // mantener.
5
6  class Square {
7      private var side: Double = 0.0
8
9      init() {
10         print("\n--- Calcular Área del Cuadrado ---")
11         self.side = getInput(prompt: "Ingresa la longitud del lado:")
12     }
13
14     func calculateArea() -> Double {
15         return side * side
16     }
17
18     func displayArea() {
19         let area = calculateArea()
20         print("El área del cuadrado es: \(String(format: "%.2f", area))")
21     }
22 }
23
24 class Rectangle {
25     private var base: Double = 0.0
26     private var height: Double = 0.0
27
28     init() {
29         print("\n--- Calcular Área del Rectángulo ---")
30         self.base = getInput(prompt: "Ingresa la longitud de la base:")
31         self.height = getInput(prompt: "Ingresa la altura:")
32     }
33
34     func calculateArea() -> Double {
35         return base * height
36     }
37
38     func displayArea() {
39         let area = calculateArea()
40         print("El área del rectángulo es: \(String(format: "%.2f", area))")
41     }
42 }
43
44 class Triangle {
45     private var base: Double = 0.0
46     private var height: Double = 0.0

```

class Triangulo

init(): Este es el constructor de la clase Triangle.

Muestra un mensaje para indicar que se calculará el área de un triángulo.

Usa la función getInput (que veremos más adelante) para solicitar y almacenar los valores de la base y la altura del triángulo.

func calculateArea() -> Double: Esta función calcula el área usando la fórmula matemática del triángulo: $(base * altura) / 2$. El resultado se devuelve como un Double.

`func displayArea()`: Esta función llama a `calculateArea()` para obtener el área y luego la imprime en la consola, formateada a dos decimales para una mejor presentación.

`class Cirlo`

`private var radius: Double = 0.0`: Declara una variable `radius` (radio) de tipo `Double` para guardar el radio del círculo.

`init()`: El constructor de la clase `Circle`.

Muestra un mensaje en la consola.

Utiliza `getInput` para pedir al usuario que ingrese el radio del círculo y lo guarda.

`func calculateArea() -> Double`: Esta función calcula el área de un círculo.

`return Double.pi * radius * radius`: Utiliza la constante `Double.pi` (que representa el valor de π , aproximadamente 3.14159...) y la fórmula πr^2 para calcular el área.

`func displayArea()`: Obtiene el área con `calculateArea()` y la imprime en la consola, también con un formato de dos decimales.

`func getInput(prompt: String) -> Double`

Esta es una función auxiliar muy útil y bien diseñada. Su propósito es manejar toda la interacción con el usuario para obtener una entrada válida.

`while true`: Crea un bucle infinito que no terminará hasta que se devuelva un valor válido.

`print(prompt)`: Muestra el mensaje que se le pasa a la función (por ejemplo, "Ingresa el radio:").

`guard let inputString = readLine(), let value = Double(inputString), value > 0 else`: Esta es una de las

partes más importantes.

`readLine()`: Lee una línea de texto que el usuario ingresa en la consola.

`guard let ... else`: Es una forma segura de verificar varias condiciones a la vez. El código solo continuará si se cumplen todas las condiciones, de lo contrario, pasará al bloque `else`.

Se verifica que la entrada (`inputString`) no sea nula, que se pueda convertir a un número (`Double(inputString)`), y que ese número sea mayor que cero (`value > 0`).

`print("Error: ...")`: Si alguna de las condiciones del `guard` falla, se imprime un mensaje de error y el bucle `while` se repite con la palabra clave `continúe`.

`return value`: Cuando el usuario ingresa un número válido, se devuelve el `value` y el bucle termina.

```

47 init() {
48     print("\n--- Calcular Área del Triángulo ---")
49     self.base = getInput(prompt: "Ingresa la longitud de la base:")
50     self.height = getInput(prompt: "Ingresa la altura:")
51 }
52
53 func calculateArea() -> Double {
54     return (base * height) / 2
55 }
56
57 func displayArea() {
58     let area = calculateArea()
59     print("El área del triángulo es: \(String(format: "%.2f", area))")
60 }
61 }
62
63 class Circle {
64     private var radius: Double = 0.0
65
66     init() {
67         print("\n--- Calcular Área del Círculo ---")
68         self.radius = getInput(prompt: "Ingresa el radio:")
69     }
70
71     func calculateArea() -> Double {
72         return Double.pi * radius * radius
73     }
74
75     func displayArea() {
76         let area = calculateArea()
77         print("El área del círculo es: \(String(format: "%.2f", area))")
78     }
79 }
80
81 // Estas funciones se encargan de la interacción con el usuario, como leer la entrada y validarla, lo que simplifica el código de las clases.
82
83 func getInput(prompt: String) -> Double {
84     while true {
85         print(prompt)
86         guard let inputString = readLine(), let value = Double(inputString), value > 0 else {
87             print("Error: Entrada no válida. Por favor, ingresa un número positivo.")
88             continue
89         }
90         return value
91     }
92 }
93
94 // Este método se encarga de la interacción con el usuario, como leer la entrada y validarla, lo que simplifica el código de las clases.

```

class AreaCalculatorApp

func run(): Este es el método principal que inicia y mantiene la aplicación en funcionamiento.

var running = true: Declara una variable booleana para controlar el ciclo de vida del programa.

Mientras el running sea true, el programa continuará ejecutándose.

while running: Inicia un bucle infinito que se ejecuta mientras la aplicación está activa.

displayMenu(): Llama a la función que muestra las opciones al usuario.

`if let choice = readLine():` Lee la entrada del usuario desde la consola. Si la entrada es válida, la guarda en la constante `choice`.

`switch choice:` Esta estructura evalúa la opción que el usuario ingresó y ejecuta el código correspondiente.

`case "1" a "4":` Para cada opción de figura, se crea una nueva instancia de la clase correspondiente (por ejemplo, `let square = Square()`). El constructor de cada clase se encarga de pedirle al usuario los datos necesarios. Luego, se llama al método `displayArea()` para mostrar el resultado.

`case "5":` Cambia el valor de `running` a `false`, lo que provoca que el bucle `while` termine y la aplicación finalice.

`default:` Maneja cualquier opción que no sea válida, mostrando un mensaje de error al usuario.

`private func displayMenu():` Esta función simplemente imprime el menú de opciones en la consola para guiar al usuario. El uso de `private` asegura que esta función solo pueda ser llamada desde dentro de la clase `AreaCalculatorApp`.

Iniciar la aplicación

`let app = AreaCalculatorApp():` Crea una nueva instancia de la clase `AreaCalculatorApp`.

`app.run():` Llama al método `run()` para iniciar la ejecución del programa.

```

94 // AreaCalculatorApp gestiona el flujo principal de la aplicación, mostrando el menú y creando instancias de las clases de figuras según la
    elección del usuario.
95 // Esto desacopla la lógica del menú de la lógica de cálculo de áreas.
96
97 class AreaCalculatorApp {
98     func run() {
99         var running = true
100         while running {
101             displayMenu()
102             if let choice = readLine() {
103                 switch choice {
104                     case "1":
105                         let square = Square()
106                         square.displayArea()
107                     case "2":
108                         let rectangle = Rectangle()
109                         rectangle.displayArea()
110                     case "3":
111                         let triangle = Triangle()
112                         triangle.displayArea()
113                     case "4":
114                         let circle = Circle()
115                         circle.displayArea()
116                     case "5":
117                         running = false
118                         print("Saliendo de la aplicación. ¡Hasta pronto!")
119                     default:
120                         print("Opción no válida. Por favor, elige un número del 1 al 5.")
121                 }
122             } else {
123                 print("Error al leer la entrada. Por favor, intenta de nuevo.")
124             }
125         }
126     }
127
128     private func displayMenu() {
129         print("\n--- Menú de Cálculo de Áreas ---")
130         print("1. Área del cuadrado")
131         print("2. Área del rectángulo")
132         print("3. Área del triángulo")
133         print("4. Área del círculo")
134         print("5. Salir")
135         print("Elige una opción:")
136     }
137 }
138
139 // Iniciar la aplicación
140 let app = AreaCalculatorApp()
141 app.run()
  
```

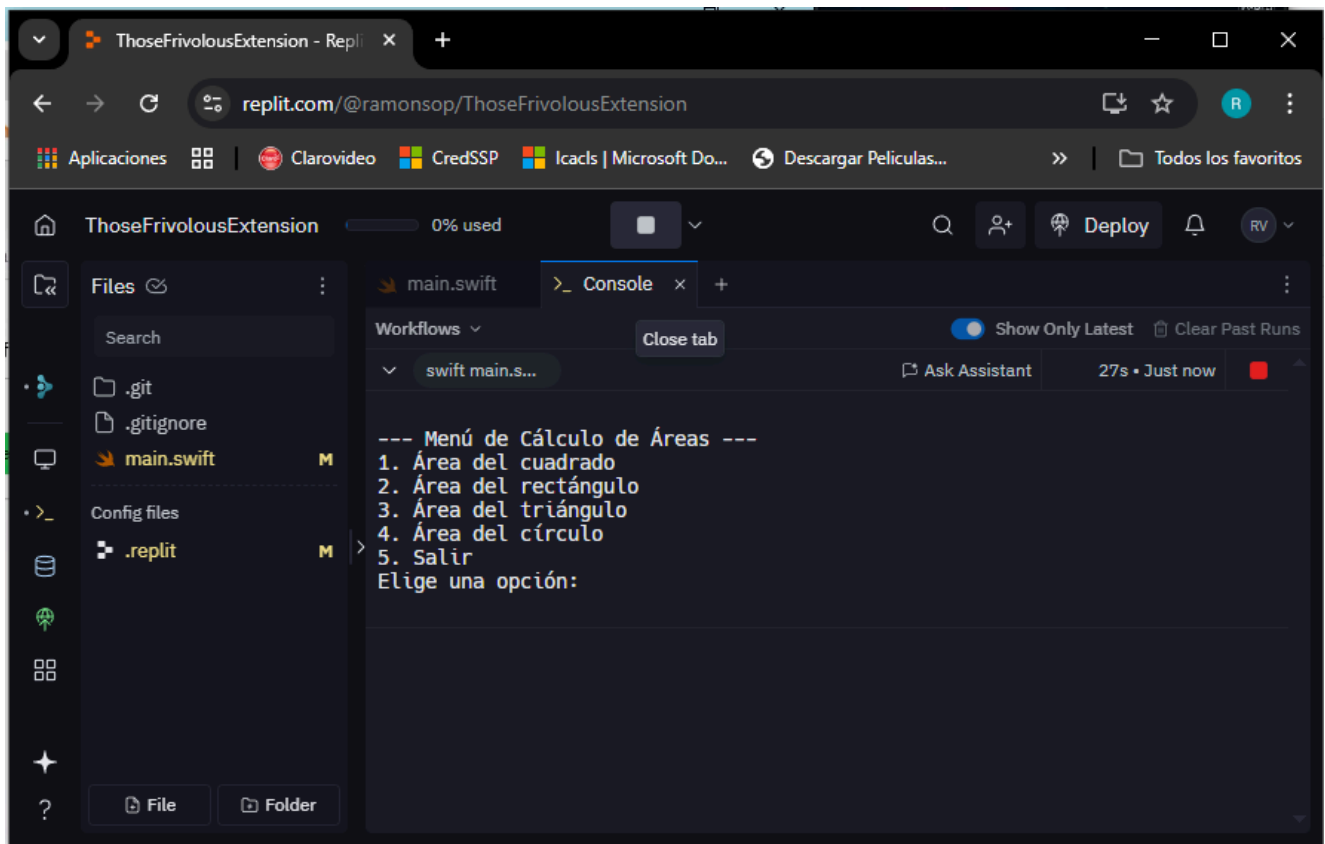
Prueba de la aplicación.

En este punto realizaremos la ejecución del código de la aplicación que mostrará el menú principal del cálculo de área de figuras geométricas en el cual podremos introducir el valor o valores para poder obtener la respuesta del área de esa figura seleccionada.

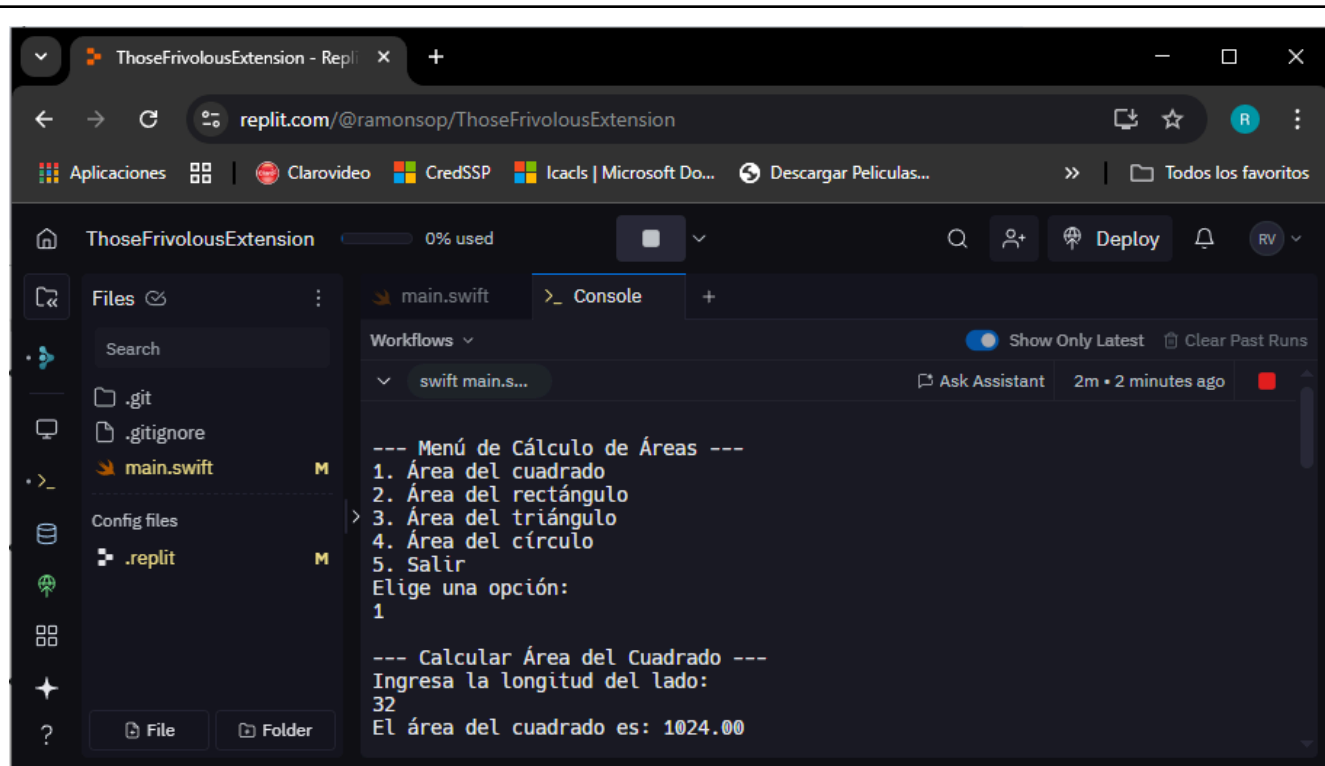
Pruebas de ejecución de la app:

En esta pantalla nos muestra el menú de la aplicación que se creó para la actividad, utilizamos los mismos valores de ingresados por el docente en la clase para validar que los resultados del código

fueran correctos.



En esta pantalla se accede a la primera opción del menú para calcular el área de un cuadrado, introduciendo el valor y obteniendo el resultado del número ingresado del área del cuadrado. Anexamos las imágenes de evidencia.



```
ThoseFrivolousExtension - Repl | x +
replit.com/@ramonsop/ThoseFrivolousExtension
Aplicaciones | Clarovideo | CredSSP | Icacls | Microsoft Do... | Descargar Peliculas... | Todos los favoritos

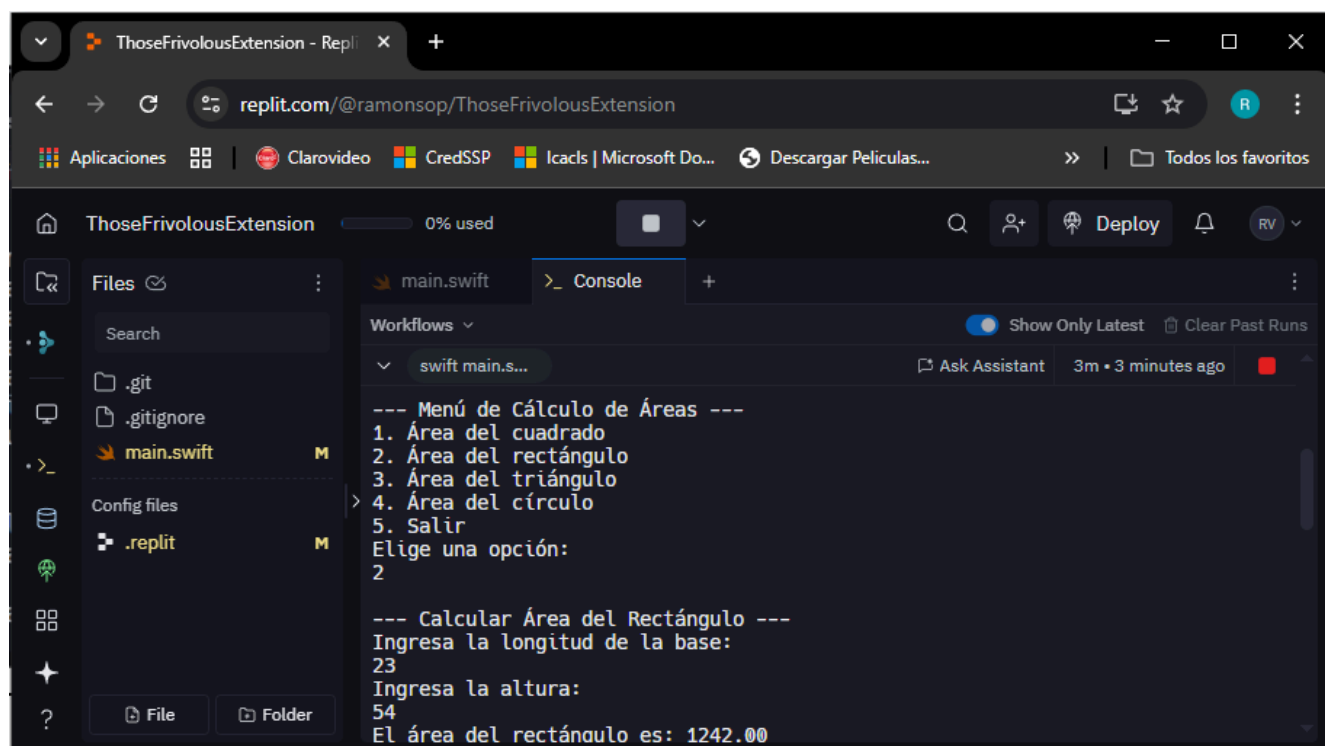
ThoseFrivolousExtension 0% used
Files | main.swift | Console
Search
.git
.gitignore
main.swift
Config files
.replit
File Folder

Workflows
swift main.s... Ask Assistant 2m • 2 minutes ago

--- Menú de Cálculo de Áreas ---
1. Área del cuadrado
2. Área del rectángulo
3. Área del triángulo
4. Área del círculo
5. Salir
Elige una opción:
1

--- Calcular Área del Cuadrado ---
Ingresa la longitud del lado:
32
El área del cuadrado es: 1024.00
```

En estas pantalla se accede a la segunda opción del menú para calcular el área del rectángulo, introduciendo los valores largo sobre ancho y obteniendo el resultado de los valores ingresado para saber área del rectángulo. Anexamos las imágenes de evidencia.



```
ThoseFrivolousExtension - Repl | x +
replit.com/@ramonsop/ThoseFrivolousExtension
Aplicaciones | Clarovideo | CredSSP | Icacls | Microsoft Do... | Descargar Peliculas... | Todos los favoritos

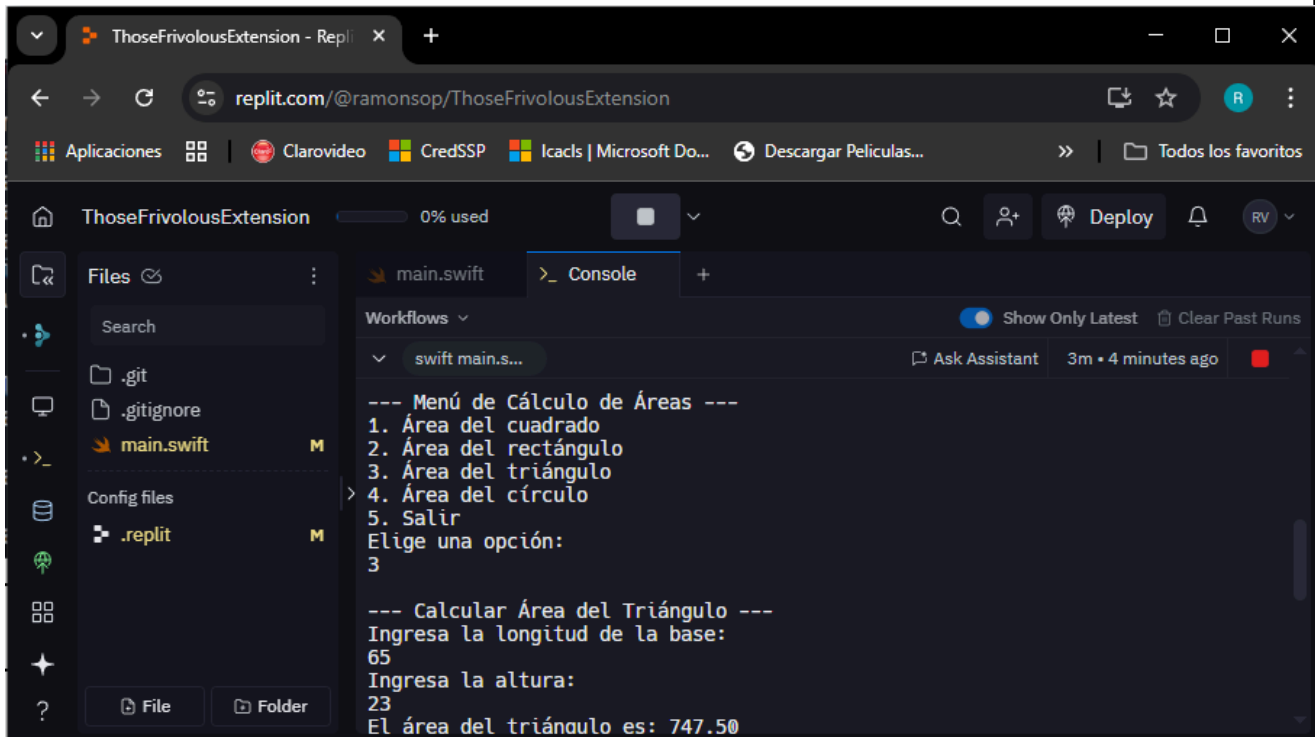
ThoseFrivolousExtension 0% used
Files | main.swift | Console
Search
.git
.gitignore
main.swift
Config files
.replit
File Folder

Workflows
swift main.s... Ask Assistant 3m • 3 minutes ago

--- Menú de Cálculo de Áreas ---
1. Área del cuadrado
2. Área del rectángulo
3. Área del triángulo
4. Área del círculo
5. Salir
Elige una opción:
2

--- Calcular Área del Rectángulo ---
Ingresa la longitud de la base:
23
Ingresa la altura:
54
El área del rectángulo es: 1242.00
```

En estas pantalla se accede a la tercera opción del menú para calcular el área del triángulo, introduciendolos valores, obteniendo el resultado de los valores ingresado para saber área del triángulo. Anexamos las imágenes de evidencia.

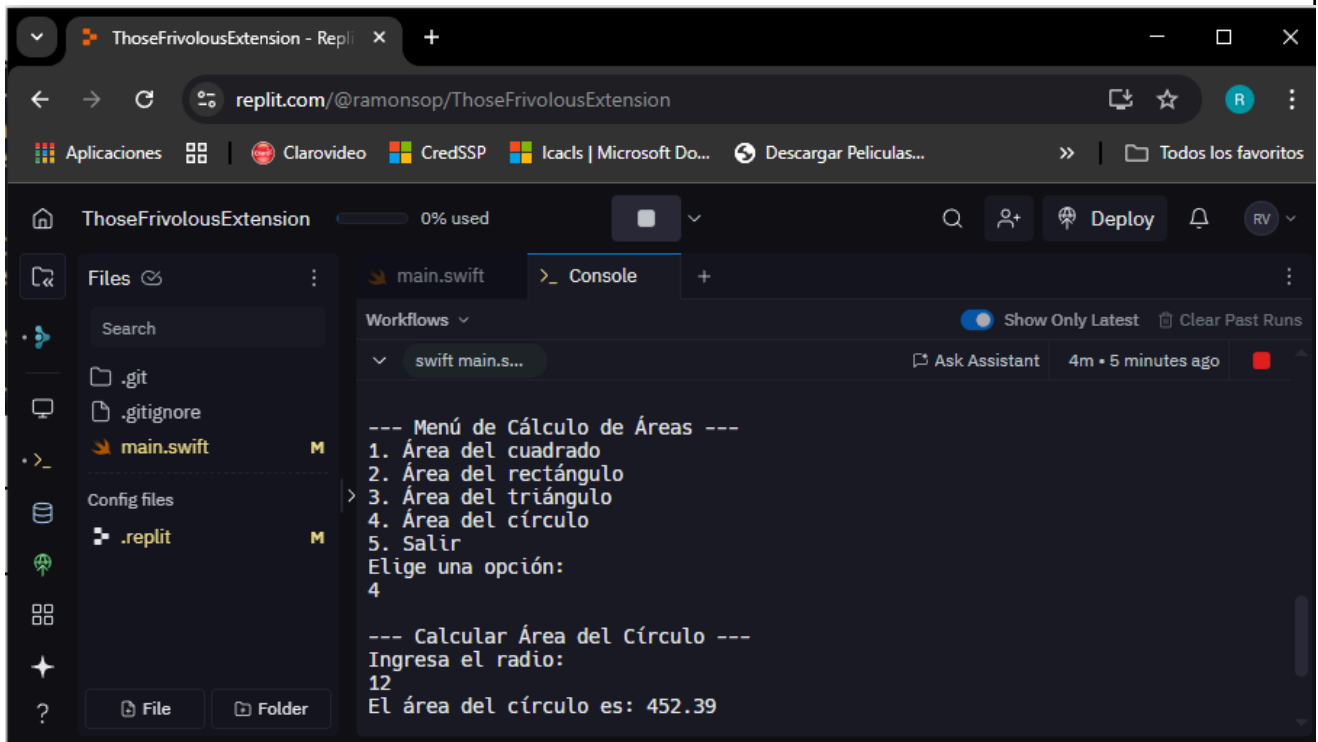


The screenshot shows a Replit web interface for a project named 'ThoseFrvilousExtension'. The left sidebar displays the file explorer with files like '.git', '.gitignore', 'main.swift', and '.replit'. The main area shows the 'main.swift' file with a Swift program. The console output shows the program's execution, including a menu for calculating areas, user input for base and height, and the final calculated area of 747.50.

```
--- Menú de Cálculo de Áreas ---
1. Área del cuadrado
2. Área del rectángulo
3. Área del triángulo
4. Área del círculo
5. Salir
Elige una opción:
3

--- Calcular Área del Triángulo ---
Ingresa la longitud de la base:
65
Ingresa la altura:
23
El área del triángulo es: 747.50
```

En estas pantalla se accede a la tercera opción del menú para calcular el radio de un círculo, introduciendolos valor, obteniendo el resultado de los valor ingresado para saber radio del círculo. Anexamos las imágenes de evidencia.

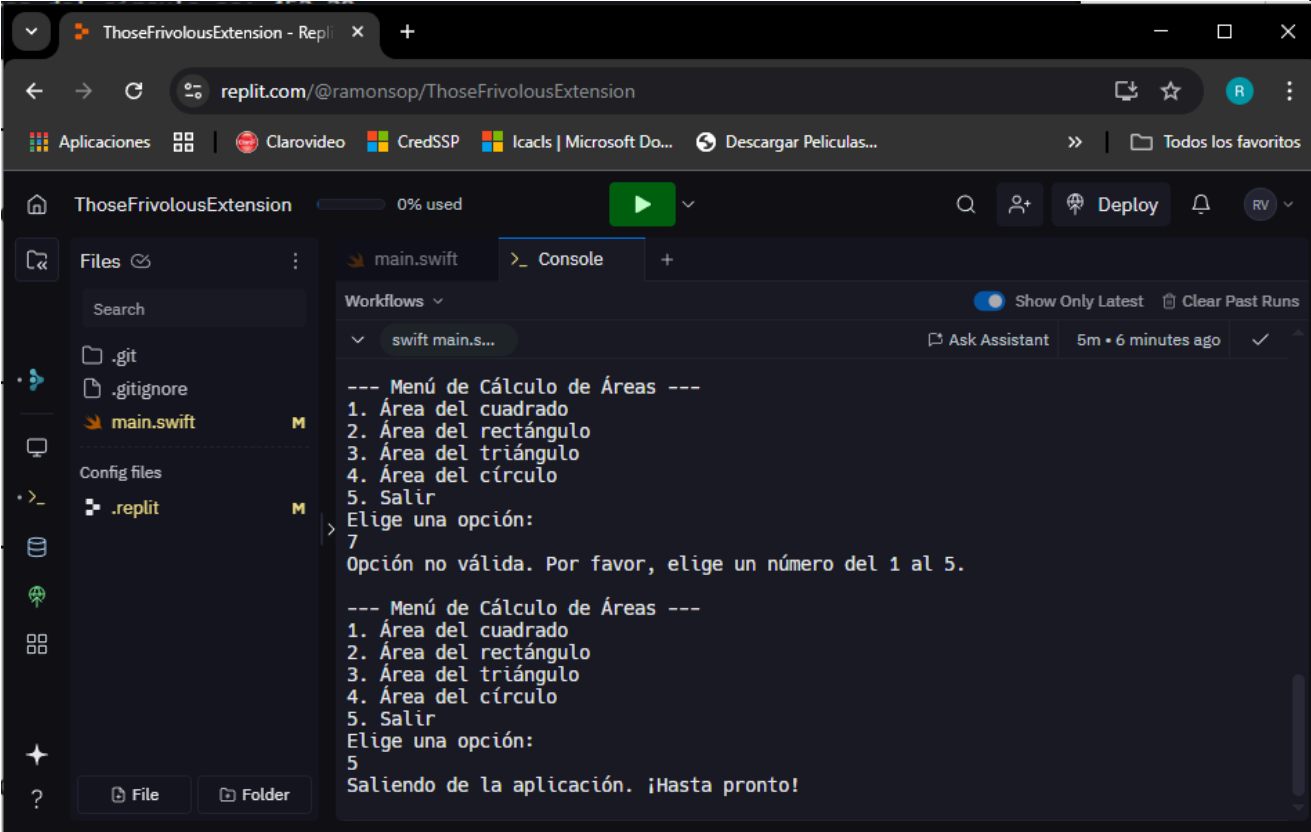


The screenshot shows a web browser window with the URL `replit.com/@ramonsop/ThoseFrivolousExtension`. The browser's address bar and tabs are visible at the top. Below the browser, the Replit interface is shown. On the left, a file explorer displays the project structure, including `.git`, `.gitignore`, `main.swift`, and `.replit`. The main area is a terminal window titled `main.swift` with a `_ Console` tab selected. The terminal output shows a menu for calculating areas, followed by the selection of option 4 (Area del círculo). The user is prompted to enter the radius, and the value 12 is entered. The final output is "El área del círculo es: 452.39".

```
--- Menú de Cálculo de Áreas ---
1. Área del cuadrado
2. Área del rectángulo
3. Área del triángulo
4. Área del círculo
5. Salir
Elige una opción:
4

--- Calcular Área del Círculo ---
Ingresa el radio:
12
El área del círculo es: 452.39
```

En la última pantalla se introduce un número no existente en el menú para generar un error, después introducimos la opción final del menú 5 que nos permite salir de la aplicación. Anexamos las imágenes de evidencia.



```
--- Menú de Cálculo de Áreas ---
1. Área del cuadrado
2. Área del rectángulo
3. Área del triángulo
4. Área del círculo
5. Salir
Elige una opción:
7
Opción no válida. Por favor, elige un número del 1 al 5.

--- Menú de Cálculo de Áreas ---
1. Área del cuadrado
2. Área del rectángulo
3. Área del triángulo
4. Área del círculo
5. Salir
Elige una opción:
5
Saliendo de la aplicación. ¡Hasta pronto!
```

Conclusion.

En conclusión: El desarrollo de una aplicación realizada en Swift que calcula áreas y radios de figuras geométricas no es solo un ejercicio de programación, es una demostración práctica de cómo la lógica y las matemáticas se aplican para resolver problemas del mundo real. Esta actividad refuerza la comprensión de conceptos fundamentales de geometría y su traducción a un lenguaje de programación, lo que es esencial en campos como la ingeniería, el diseño gráfico y la arquitectura.

En la vida cotidiana, esta aplicación puede ser una herramienta útil para estudiantes, profesores y profesionales que necesitan realizar cálculos rápidos y precisos. A nivel profesional, la capacidad de crear una interfaz de usuario (menú) intuitiva y de implementar funciones matemáticas precisas es una habilidad altamente valorada. Demuestra competencia en el desarrollo de software y la habilidad para crear soluciones eficientes y accesibles para los usuarios, lo que es crucial para cualquier desarrollador. Se me hizo muy fácil el poder realizar esta actividad debido a que la consola online de programación

que utilizamos con el lenguaje swift es muy intuitiva y aporta mucho apoyo al desarrollador con el autoayuda o autocompletar al estar desarrollando de alguna aplicación.

Referencias.

Gemini - chat to supercharge your ideas. (n.d.). Gemini. Retrieved January 9, 2025, from <https://gemini.google.com/>

Ingeniería en desarrollo de software. (n.d.). Edu.Mx. Retrieved January 9, 2025, from <https://umi.edu.mx/coppel/IDS/login/index.php>