INGENIERÍA EN SISTEMAS COMPUTACIONALES

Materia:
Tópicos Avanzados de Programación.

Semestre:
4°. Semestre.

Producto Académico:
Reporte de Práctica: “Práctica 3 U1 Cap. 2’’.

Presenta(n):
Alejandro Zurita Pérez

Docente:
M.T.I. Dionisio Pérez Pérez.

Medellín de Bravo, Ver. SEP. – DIC. 20

1. INTRODUCCION

En esta serie de ejercicios, exploraremos el desarrollo de aplicaciones de escritorio utilizando Windows Forms en C#. Windows Forms es una tecnología de interfaz de usuario que permite crear aplicaciones de escritorio visualmente atractivas y funcionales en el entorno de desarrollo de Visual Studio.

A lo largo de estos ejercicios, nos centraremos en aplicar los conceptos fundamentales de programación en C# para crear diversas aplicaciones que aborden diferentes problemas y escenarios comunes.

Ejercicio 1: Cálculo del Perímetro de un Polígono Regular

En este primer ejercicio, crearemos una aplicación que calculará el perímetro de cualquier polígono regular dado el número de lados y la longitud de uno de sus lados. A través de una interfaz de usuario intuitiva, el usuario podrá ingresar el número de lados y la longitud de un lado, y la aplicación calculará y mostrará el perímetro correspondiente.

Ejercicio 2: Generación de Errores Deliberados

En el segundo ejercicio, exploraremos cómo se comporta el compilador cuando se genera un error a propósito en una aplicación de Windows Forms. A través de la introducción deliberada de un error en el código, observaremos cómo el compilador detecta y maneja este error durante la compilación del proyecto.

Ejercicio 3: Conversión de Grados a Radianes

En el tercer ejercicio, crearemos una aplicación que convierta una medida dada en grados a su equivalente en radianes. Utilizando controles de entrada de texto y un botón de conversión, el usuario podrá ingresar una cantidad en grados, y la aplicación calculará y mostrará el equivalente en radianes.

Ejercicio 4: Conversión de Grados Celsius a Fahrenheit

En el cuarto ejercicio, nos centraremos en la conversión de temperaturas de grados Celsius a grados Fahrenheit. Mediante una interfaz simple, el usuario podrá ingresar una temperatura en grados Celsius, y la aplicación calculará y mostrará la temperatura equivalente en grados Fahrenheit.

Ejercicio 5: Conversión entre Dólares y Euros

Finalmente, en el quinto ejercicio, desarrollaremos una aplicación que facilite la conversión entre dólares y euros. A través de la entrada de la cantidad de dinero en dólares y el tipo de cambio del día, la aplicación calculará y mostrará la cantidad equivalente en euros.

A lo largo de estos ejercicios, pondremos en práctica los principios de diseño de interfaz de usuario, manejo de eventos, entrada y salida de datos, y lógica de programación en C# para crear aplicaciones funcionales y útiles utilizando Windows Forms.

2. OBJETIVOS (COMPETENCIAS)

Es esta práctica se alcanzarán las siguientes competencias:

- Desarrollo de Aplicaciones de Escritorio: Aprender a desarrollar aplicaciones de escritorio utilizando Windows Forms en C#, lo que permitirá a los estudiantes familiarizarse con el entorno de desarrollo de Visual Studio y adquirir habilidades en el diseño y desarrollo de interfaces de usuario.

- Manejo de Controles y Eventos: Practicar el manejo de controles como botones, cuadros de texto y etiquetas, así como el manejo de eventos asociados a estos controles. Esto ayudará a los estudiantes a comprender cómo interactuar con los usuarios y responder a sus acciones dentro de la aplicación.

- Validación de Entrada de Datos: Aprender a validar la entrada de datos del usuario para garantizar que los valores ingresados sean válidos y cumplan con ciertos criterios. Esto incluye la verificación de números válidos, rangos aceptables y formatos adecuados.

- Cálculos y Conversiones: Practicar el desarrollo de lógica de programación para realizar cálculos matemáticos y conversiones entre diferentes unidades de medida. Esto ayudará a los estudiantes a fortalecer sus habilidades en la manipulación de datos numéricos y algoritmos básicos.

- Manejo de Mensajes y Diálogos: Aprender a mostrar mensajes informativos, de error o de confirmación utilizando cuadros de diálogo como MessageBox. Esto permitirá a los estudiantes comunicarse de manera efectiva con los usuarios y proporcionar retroalimentación adecuada sobre las acciones realizadas en la aplicación.

- Resolución de Problemas: Desarrollar habilidades para identificar, analizar y resolver problemas de programación a través de la implementación de soluciones prácticas en las aplicaciones desarrolladas. Esto fomentará la capacidad de los estudiantes para enfrentar desafíos y encontrar soluciones efectivas en entornos de desarrollo reales.

- Práctica de Buenas Prácticas de Codificación: Promover el uso de buenas prácticas de codificación, incluyendo la legibilidad del código, la modularidad, el uso adecuado de nombres de variables y la documentación de código. Esto ayudará a los estudiantes a escribir código limpio, estructurado y fácil de mantener.

Al abordar estos objetivos, será para estar más preparados para desarrollar aplicaciones de escritorio funcionales y efectivas utilizando Windows Forms en C#, así como para enfrentar desafíos más avanzados en el campo de la programación.

3. FUNDAMENTO

Desarrollo de Aplicaciones de Escritorio: Aprender a desarrollar aplicaciones de escritorio utilizando Windows Forms en C#, lo que permitirá a los estudiantes familiarizarse con el entorno de desarrollo de Visual Studio y adquirir habilidades en el diseño y desarrollo de interfaces de usuario.

Manejo de Controles y Eventos: Practicar el manejo de controles como botones, cuadros de texto y etiquetas, así como el manejo de eventos asociados a estos controles. Esto ayudará a los estudiantes a comprender cómo interactuar con los usuarios y responder a sus acciones dentro de la aplicación.

Validación de Entrada de Datos: Aprender a validar la entrada de datos del usuario para garantizar que los valores ingresados sean válidos y cumplan con ciertos criterios. Esto incluye la verificación de números válidos, rangos aceptables y formatos adecuados.

Cálculos y Conversiones: Practicar el desarrollo de lógica de programación para realizar cálculos matemáticos y conversiones entre diferentes unidades de medida. Esto ayudará a los estudiantes a fortalecer sus habilidades en la manipulación de datos numéricos y algoritmos básicos.

Manejo de Mensajes y Diálogos: Aprender a mostrar mensajes informativos, de error o de confirmación utilizando cuadros de diálogo como MessageBox. Esto permitirá a los estudiantes comunicarse de manera efectiva con los usuarios y proporcionar retroalimentación adecuada sobre las acciones realizadas en la aplicación.

Resolución de Problemas: Desarrollar habilidades para identificar, analizar y resolver problemas de programación a través de la implementación de soluciones prácticas en las aplicaciones desarrolladas. Esto fomentará la capacidad de los estudiantes para enfrentar desafíos y encontrar soluciones efectivas en entornos de desarrollo reales.

Práctica de Buenas Prácticas de Codificación: Promover el uso de buenas prácticas de codificación, incluyendo la legibilidad del código, la modularidad, el uso adecuado de nombres de variables y la documentación de código. Esto ayudará a los estudiantes a escribir código limpio, estructurado y fácil de mantener.

Al abordar estos objetivos, será con el fin de estar mejor preparados para desarrollar aplicaciones de escritorio funcionales y efectivas utilizando Windows Forms en C#, así como para enfrentar desafíos más avanzados en el campo de la programación.

4. EQUIPAMIENTO Y MATERIAL DE APOYO

Computadora o Portátil: Se requiere una computadora o portátil funcional con un entorno de desarrollo instalado para practicar la escritura y ejecución de código en C#. Se recomienda utilizar un entorno de desarrollo integrado (IDE) como Visual Studio Community 2022 que proporcionan herramientas avanzadas para la programación en C#.

Acceso a Internet: El acceso a Internet es útil para acceder a recursos en línea, como documentación oficial de C#, tutoriales, foros de discusión y plataformas de aprendizaje en línea. Estos recursos pueden ayudar a los estudiantes a profundizar su comprensión de los conceptos y resolver problemas específicos durante las prácticas de programación.

Libros y Recursos de Aprendizaje: Se recomienda utilizar libros de texto y otros recursos de aprendizaje dedicados a la programación en C#. Estos recursos proporcionan una guía estructurada para aprender los fundamentos del lenguaje, así como ejemplos de código y ejercicios prácticos para reforzar el aprendizaje.

Comunidad de Desarrolladores: Unirse a comunidades en línea de desarrolladores de C# puede ser beneficioso para compartir conocimientos, hacer preguntas, obtener ayuda con problemas de codificación y colaborar en proyectos. Plataformas como Stack Overflow, Reddit (r/csharp), y foros de desarrolladores de Microsoft son excelentes lugares para interactuar con otros programadores de C#.

Práctica y Ejercicios: La práctica regular es fundamental para mejorar las habilidades de programación en C#. Los estudiantes deben dedicar tiempo a resolver problemas y completar ejercicios de programación para reforzar los conceptos aprendidos y desarrollar su capacidad para resolver problemas de manera independiente.

Al disponer del equipamiento y material de apoyo adecuados, se puede maximizar el aprendizaje y dominar los fundamentos de la programación en C#. Es importante dedicar tiempo y esfuerzo a practicar regularmente y explorar una variedad de recursos para obtener una comprensión completa del lenguaje y sus aplicaciones.

5. DESARROLLO DE LA PRACTICA

Ejercicio 1: Cálculo del Perímetro de un Polígono Regular

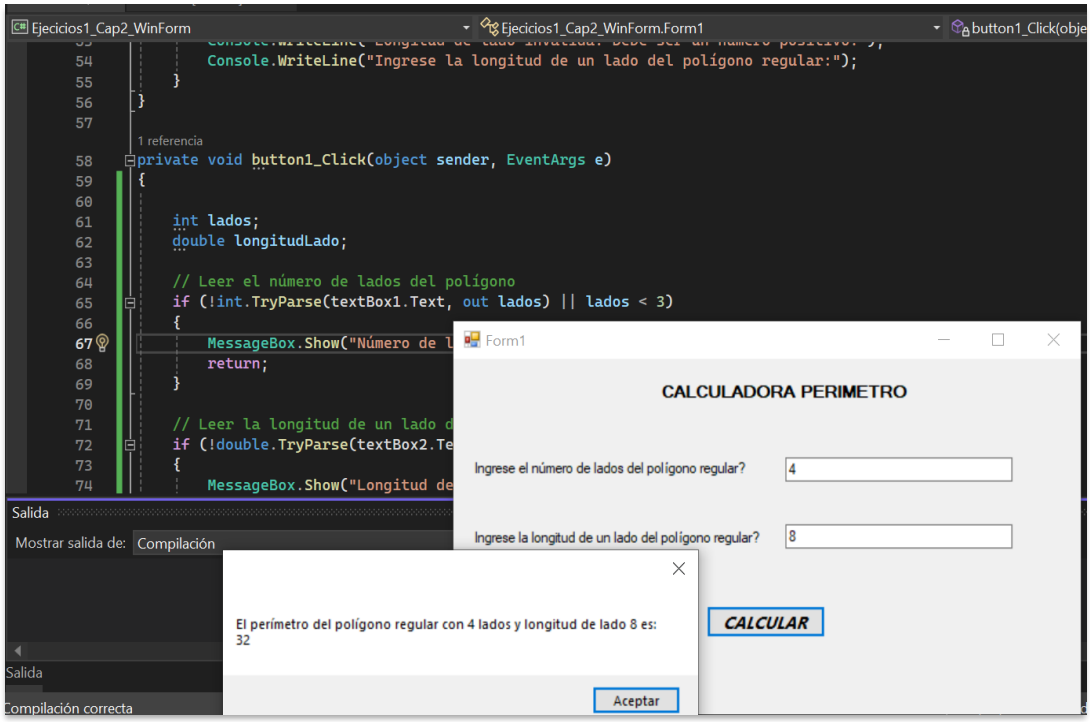
Interfaz de Usuario: Se diseñó una interfaz de usuario simple que consta de etiquetas para indicar al usuario qué información debe ingresar y cuadros de texto para que el usuario ingrese el número de lados del polígono y la longitud de un lado.

Manejo de Eventos: Se implementaron eventos asociados a los cuadros de texto y al botón de cálculo para capturar la entrada del usuario y realizar los cálculos correspondientes.

Validación de Entrada: Se incluyó la validación de la entrada del usuario para garantizar que el número de lados sea un entero mayor o igual a 3 y que la longitud de un lado sea un número positivo.

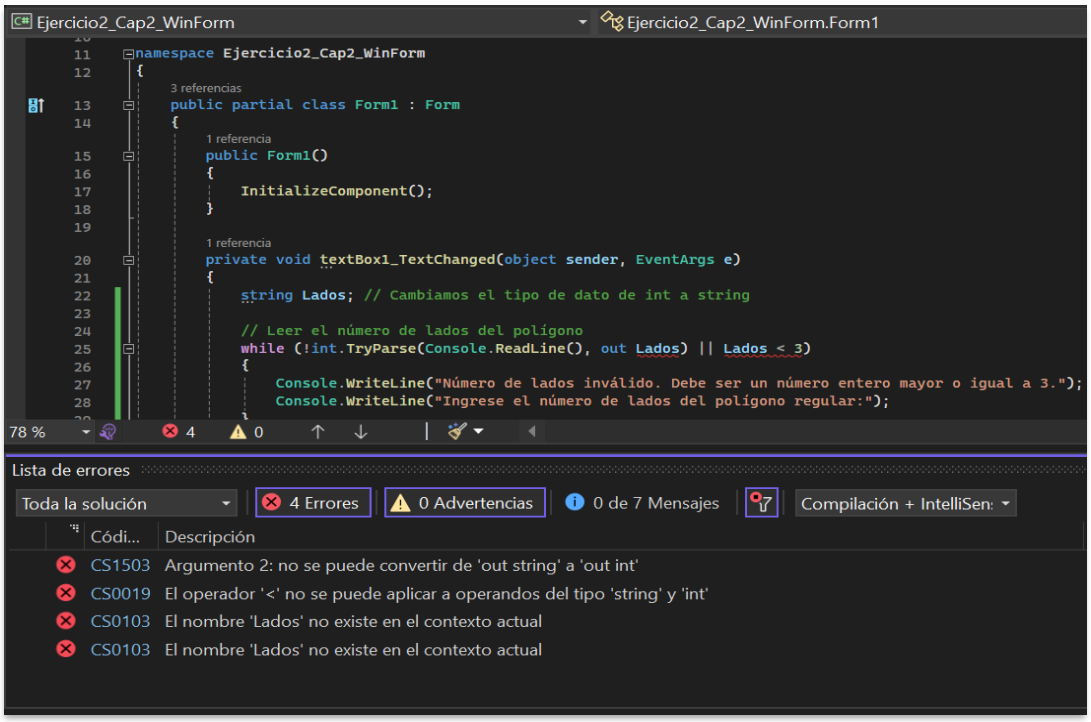
Cálculo del Perímetro: Se realizó el cálculo del perímetro del polígono regular utilizando la fórmula P = n * s, donde "n" es el número de lados y "s" es la longitud de un lado.

Presentación del Resultado: Se mostró el resultado del cálculo del perímetro en un cuadro de mensaje utilizando MessageBox.Show().



Ejercicio 2: Generación de Errores Deliberados

En este ejercicio, se introdujo deliberadamente un error en el código para observar cómo el compilador detecta y maneja este error durante la compilación del proyecto. No se implementó ninguna interfaz de usuario, ya que el objetivo era puramente académico.



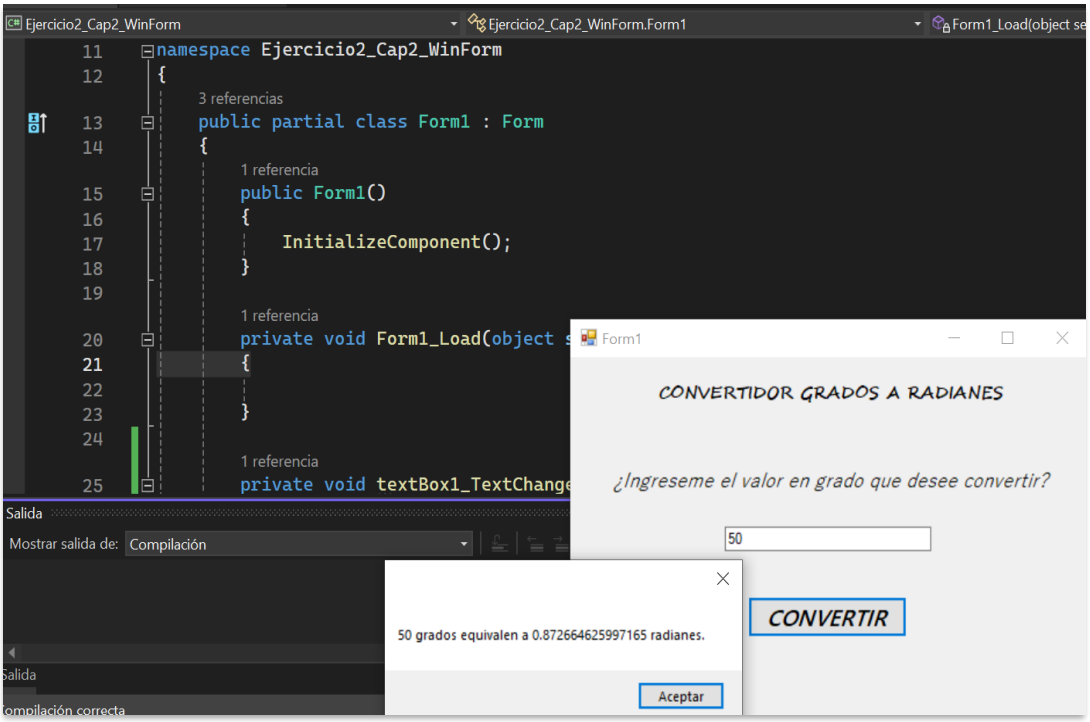
Ejercicio 3: Conversión de Grados a Radianes

Interfaz de Usuario: Se diseñó una interfaz de usuario con un cuadro de texto para que el usuario ingrese la cantidad de grados a convertir y un botón de conversión.

Manejo de Eventos: Se implementó un evento asociado al botón de conversión para capturar la entrada del usuario y realizar los cálculos correspondientes.

Cálculo de la Conversión: Se realizó el cálculo de la conversión de grados a radianes utilizando la fórmula radianes = (grados * PI) / 180, donde "PI" es el valor de pi (π).

Presentación del Resultado: Se mostró el resultado de la conversión en un cuadro de mensaje utilizando MessageBox.Show().



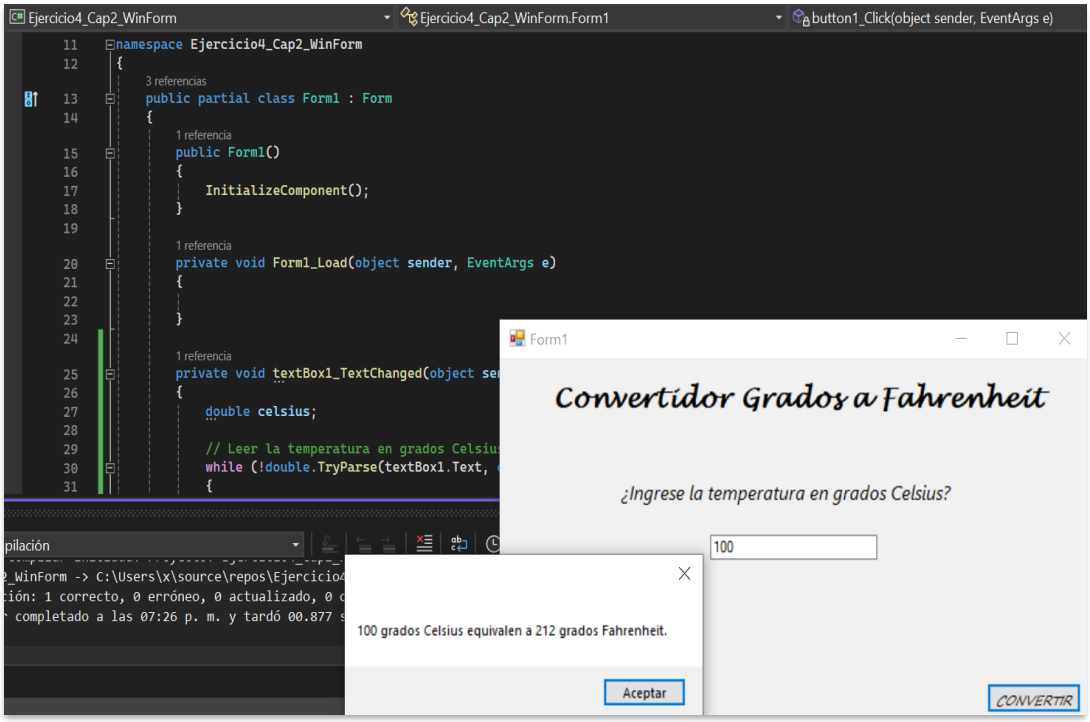
Ejercicio 4: Conversión de Grados Celsius a Fahrenheit

Interfaz de Usuario: Se diseñó una interfaz de usuario similar al ejercicio anterior, pero esta vez para que el usuario ingrese la temperatura en grados Celsius.

Manejo de Eventos: Se implementó un evento asociado al botón de conversión para capturar la entrada del usuario y realizar los cálculos correspondientes.

Cálculo de la Conversión: Se realizó el cálculo de la conversión de grados Celsius a Fahrenheit utilizando la fórmula fahrenheit = (celsius * 9/5) + 32.

Presentación del Resultado: Se mostró el resultado de la conversión en un cuadro de mensaje utilizando MessageBox.Show().



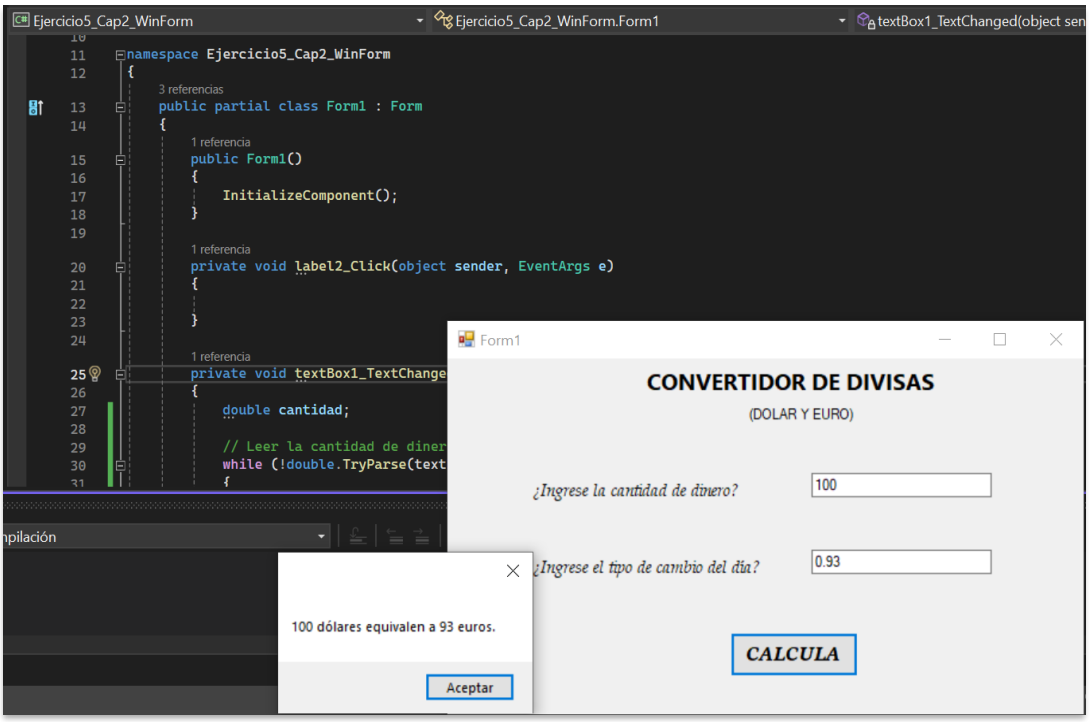
Ejercicio 5: Conversión entre Dólares y Euros

Interfaz de Usuario: Se diseñó una interfaz de usuario con dos cuadros de texto para que el usuario ingrese la cantidad de dinero en dólares y el tipo de cambio del día, junto con un botón de conversión.

Manejo de Eventos: Se implementó un evento asociado al botón de conversión para capturar la entrada del usuario y realizar los cálculos correspondientes.

Cálculo de la Conversión: Se realizó el cálculo de la conversión de dólares a euros multiplicando la cantidad de dólares por el tipo de cambio ingresado.

Presentación del Resultado: Se mostró el resultado de la conversión en un cuadro de mensaje utilizando MessageBox.Show().



6. RESULTADOS Y CONCLUSIONES

6.1 Resultados

Durante el desarrollo de esta práctica, se obtuvieron los siguientes resultados significativos:

1. **Ejercicio 1:**
   - Se desarrolló una aplicación de Windows Forms que calcula el perímetro de un polígono regular dado el número de lados y la longitud de un lado.
   - La aplicación permite al usuario ingresar los valores requeridos y muestra el resultado del cálculo en un cuadro de mensaje.
2. **Ejercicio 2:**
   - Se introdujo deliberadamente un error en el código para observar cómo el compilador detecta y maneja este error durante la compilación del proyecto.
   - Se observó el comportamiento del compilador y se identificaron las acciones necesarias para corregir el error.
3. **Ejercicio 3:**
   - Se desarrolló una aplicación de Windows Forms que convierte grados a radianes.
   - La aplicación permite al usuario ingresar la cantidad de grados y muestra el resultado de la conversión en un cuadro de mensaje.
4. **Ejercicio 4:**
   - Se desarrolló una aplicación de Windows Forms que convierte grados Celsius a Fahrenheit.
   - La aplicación permite al usuario ingresar la temperatura en grados Celsius y muestra el resultado de la conversión en un cuadro de mensaje.
5. **Ejercicio 5:**
   - Se desarrolló una aplicación de Windows Forms que convierte dólares a euros utilizando el tipo de cambio del día.
   - La aplicación permite al usuario ingresar la cantidad de dólares y el tipo de cambio, y muestra el resultado de la conversión en un cuadro de mensaje.

6.2 Conclusiones

En conclusión, Los ejercicios proporcionaron una oportunidad para familiarizarse con el desarrollo de aplicaciones de escritorio utilizando Windows Forms en C#. Se adquirió experiencia en el diseño de interfaces de usuario y la implementación de lógica de programación para resolver problemas específicos.

Se comprendió la importancia del manejo de eventos y la validación de entrada de datos para garantizar la funcionalidad y la usabilidad de las aplicaciones desarrolladas. La implementación adecuada de estos elementos permitió crear aplicaciones interactivas y robustas.

Se observó el proceso de detección y corrección de errores durante la compilación del proyecto. La introducción deliberada de errores en el código proporcionó una comprensión más profunda de cómo funciona el compilador y cómo se pueden solucionar los errores de programación.

Se demostró la capacidad de realizar cálculos matemáticos y conversiones de unidades utilizando C# y Windows Forms. Estos ejercicios ayudaron a fortalecer las habilidades en la manipulación de datos numéricos y el desarrollo de algoritmos básicos.

Se enfatizó la importancia de la presentación de mensajes y la comunicación efectiva con el usuario a través de cuadros de diálogo. Proporcionar retroalimentación clara y oportuna mejoró la experiencia del usuario y la usabilidad de las aplicaciones desarrolladas.

En resumen, los ejercicios proporcionaron una experiencia práctica valiosa en el desarrollo de aplicaciones de escritorio utilizando Windows Forms en C#, y contribuyeron al desarrollo de habilidades fundamentales en programación y desarrollo de software.

7. ANEXOS

Sin anexos.

8. REFERENCIAS