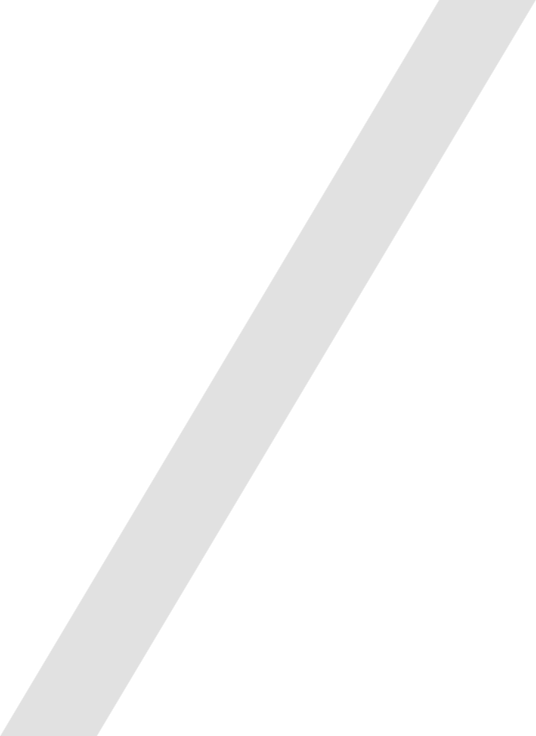
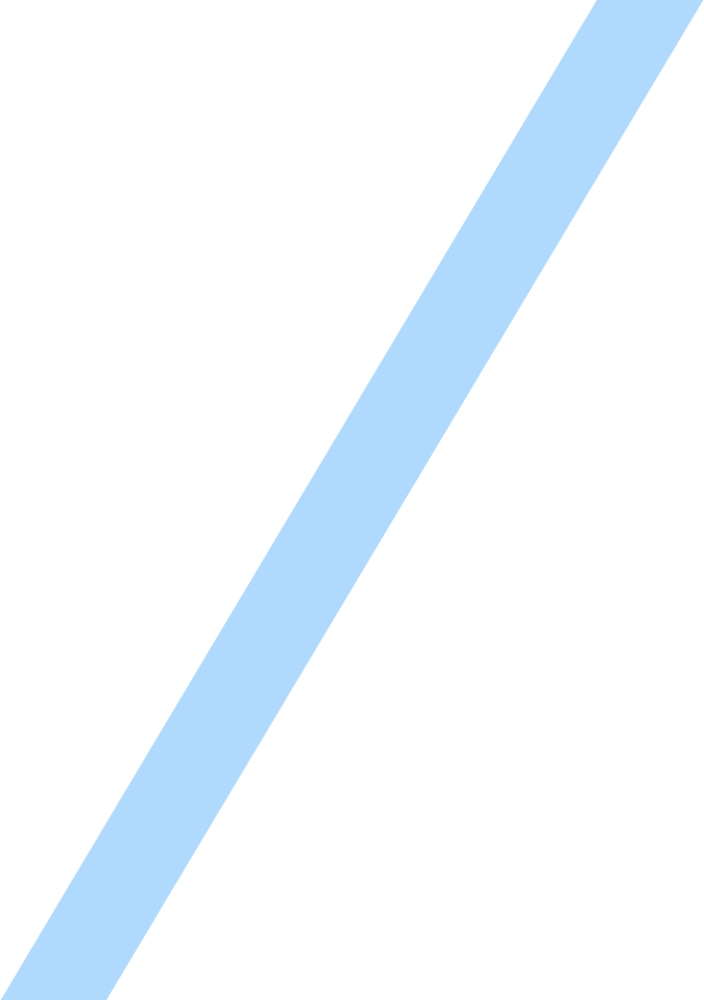
|  |
| --- |
| Software Newton Raphson |

|  |
| --- |
| Manual de desarrollo |



|  |  |
| --- | --- |
| David Brizuela Martínez - 22110337  Edna Erandeni Carrasco González - 22110389 |  |

Manual de Desarrollo - Solucionador de Ecuaciones

**Introducción**

El solucionador de ecuaciones es un programa que utiliza el método de Newton-Raphson para encontrar una solución aproximada a una ecuación proporcionada por el usuario. El programa se desarrolló utilizando el lenguaje de programación C++ y hace uso de la biblioteca SymEngine para el análisis simbólico y cálculos matemáticos.

**Requisitos del sistema**

* Compilador de C++ compatible con C++11 o superior.
* Biblioteca SymEngine (versión X o superior).

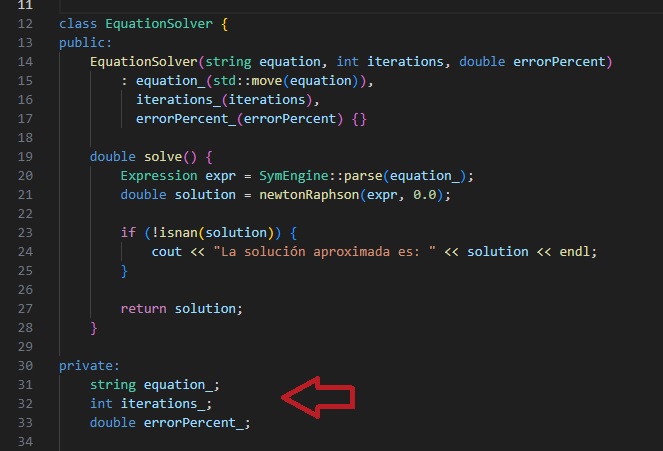
**Configuración del entorno desarrollado (IDE)**

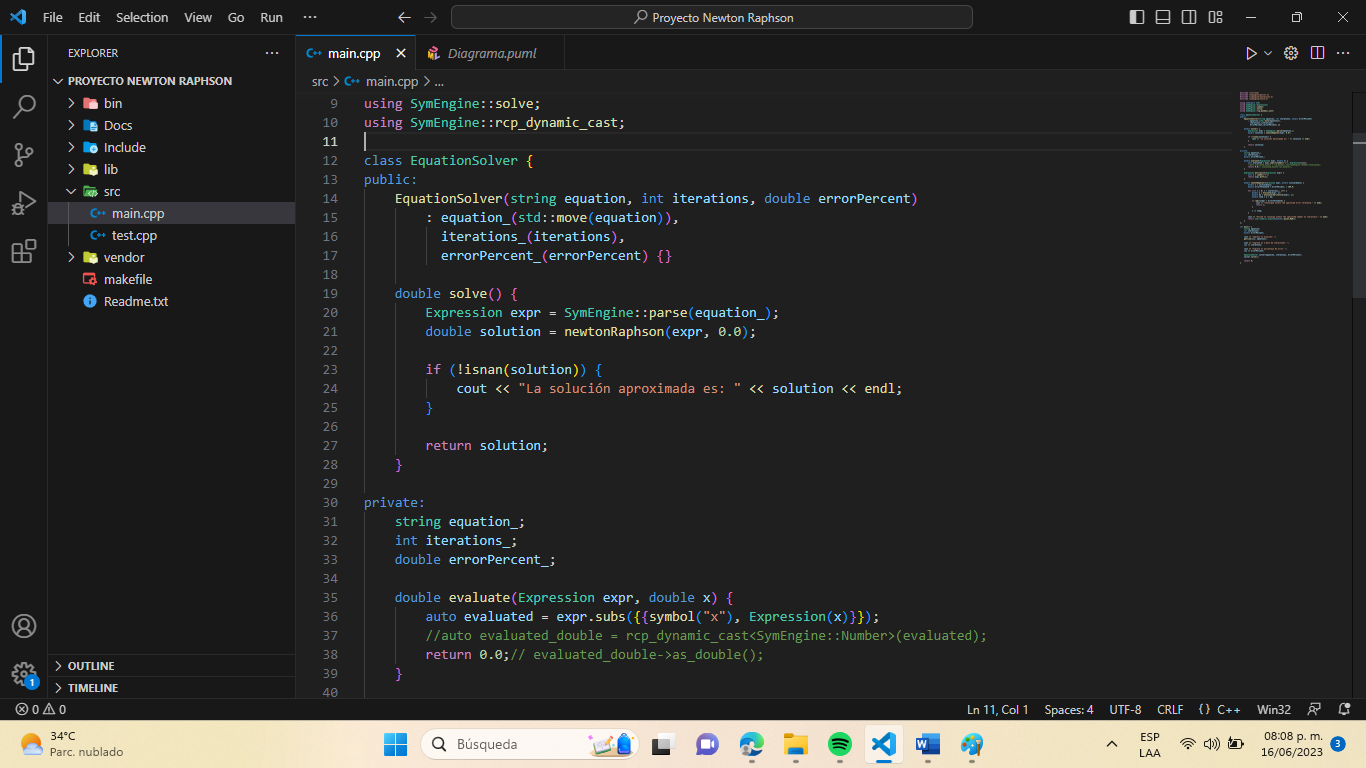
1. Instala el compilador de C++ en tu sistema si aún no lo tienes.
2. Descarga la biblioteca SymEngine desde [https://github.com/symengine/symengine] e instálala siguiendo las instrucciones proporcionadas.
3. Configura tu entorno de desarrollo para que incluya las rutas de inclusión y las bibliotecas necesarias de SymEngine.

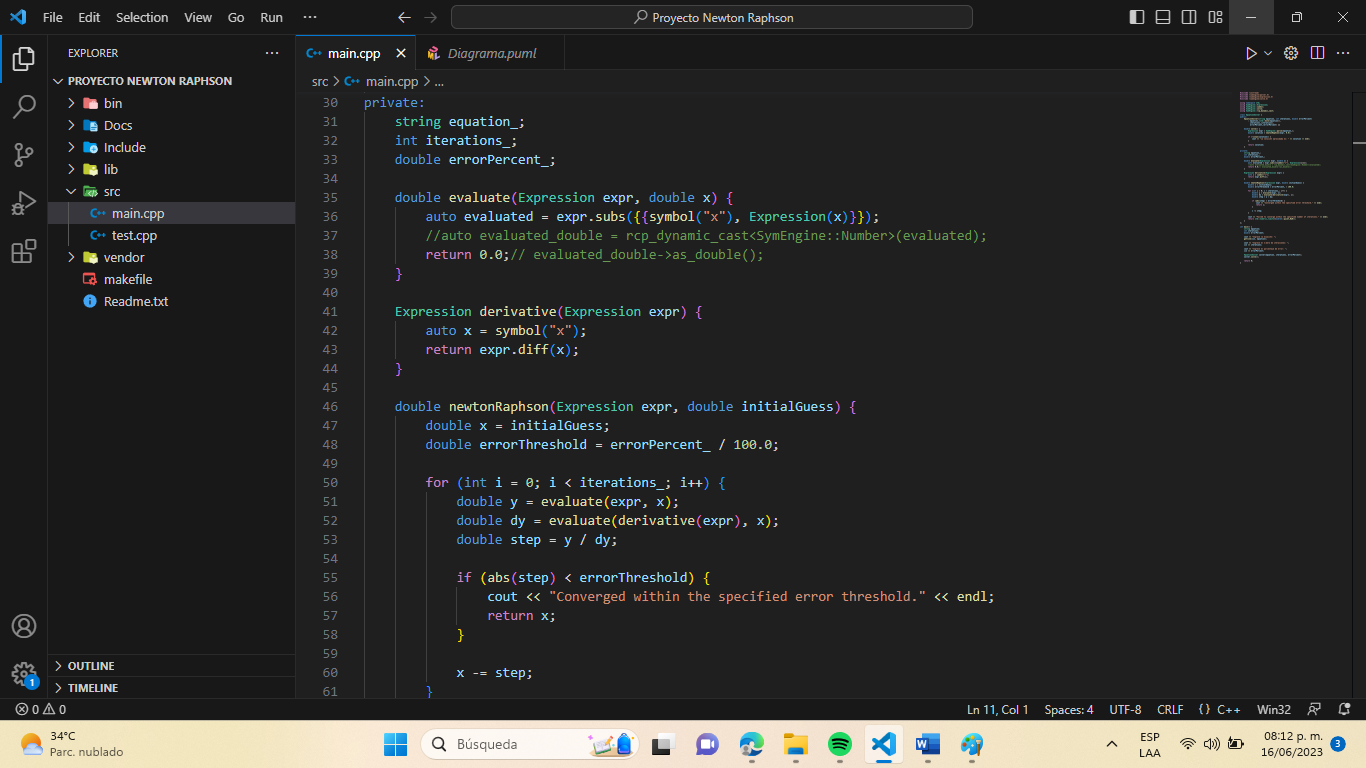
**Estructura del código**

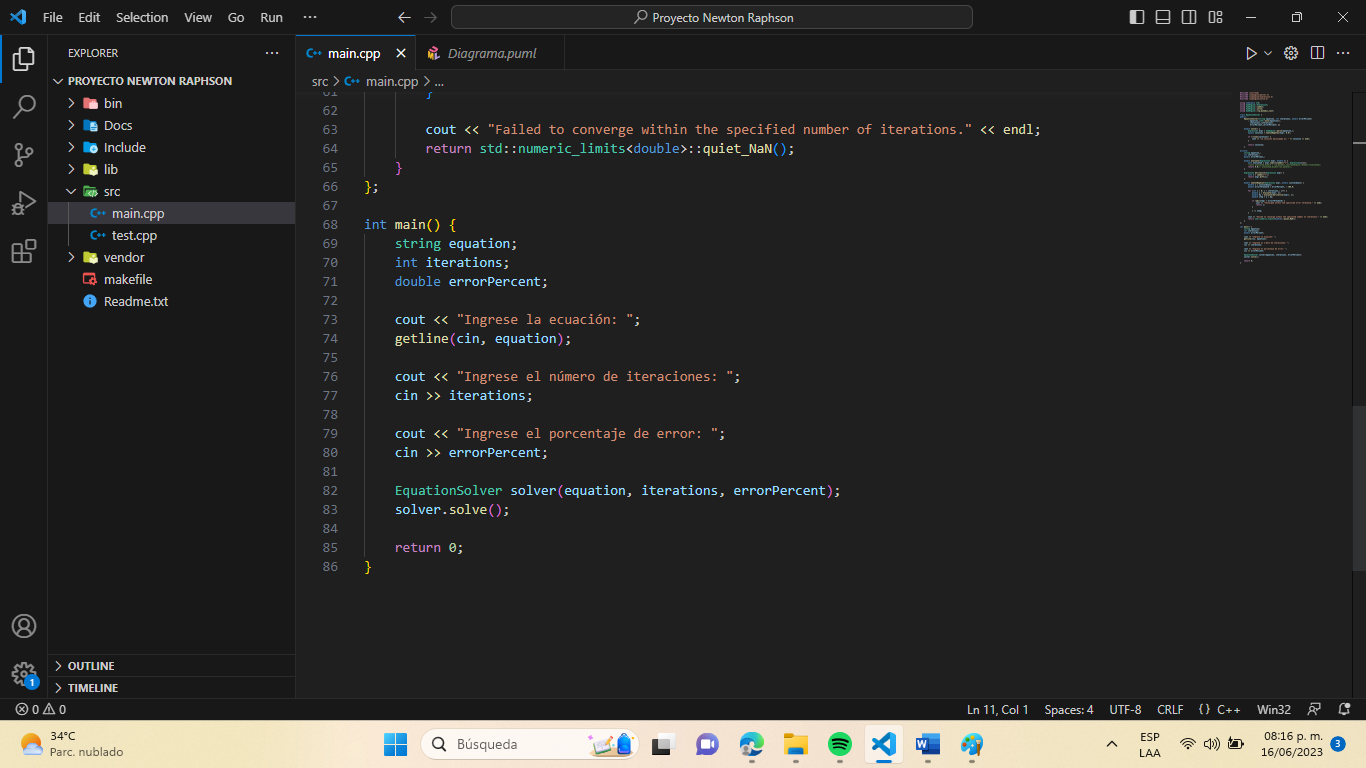
El código fuente del programa se organiza en una clase principal llamada “EquationSolver” y una función “main ()” que maneja la interacción con el usuario.

* **EquationSolver:** Esta clase encapsula la lógica principal del solucionador de ecuaciones. Proporciona métodos para analizar, resolver y encontrar soluciones aproximadas a las ecuaciones proporcionadas.
* Atributos:
* equation\_: Almacena la ecuación proporcionada por el usuario.
* iterations\_: Número de iteraciones máximo permitido para el método de Newton-Raphson.
* errorPercent\_: Porcentaje de error máximo permitido para el método de Newton-Raphson.



* Métodos:
* solve (): Resuelve la ecuación utilizando el método de Newton-Raphson y devuelve una solución aproximada.
* evaluate (expr, x): Evalúa la expresión ‘expr’ sustituyendo la variable ‘x’ por un valor dado.
* derivative (expr): Calcula la derivada de la expresión ‘expr’ con respecto a la variable ‘x’.
* newtonRaphson (expr, initialGuess): Implementa el método de Newton-Raphson para encontrar una solución aproximada.



* **main ():** Interactúa con el usuario para obtener la ecuación, el número de iteraciones y el porcentaje de error. Luego, crea una instancia de EquationSolver y llama al método solve () para obtener la solución aproximada.

**Estándares de codificación**

* Utiliza el estilo de codificación [estándar de tu elección].
* Utiliza nombres de variables y funciones descriptivas.
* Incluye comentarios adecuados para explicar la lógica del código y su funcionalidad.

**Pruebas y depuración**

* Realiza pruebas exhaustivas para verificar el correcto funcionamiento del solucionador de ecuaciones.
* Asegúrate de manejar adecuadamente situaciones de error y validaciones de entrada del usuario.
* Utiliza herramientas de depuración para solucionar problemas y errores durante el desarrollo.

**Flujo de trabajo y control de versiones**

* Utiliza un sistema de control de versiones (por ejemplo, Git) para gestionar el código fuente y las versiones del proyecto.
* Sigue un flujo de trabajo de desarrollo adecuado, como ramas de características, control de calidad y fusiones regulares.

**Procedimientos de despliegue**

* Si deseas desplegar el programa en un entorno de producción, asegúrate de cumplir con los requisitos de configuración y las dependencias necesarias.
* Realiza pruebas adicionales antes de implementar en un entorno de producción para garantizar un funcionamiento correcto y sin errores.