MANUAL DE DESARROLLADOR

PRGRAMA METODO DE INTEGRACIÓN POR TRAPECIOS

Logotipo

Descripción generada automáticamente

Carlos Isaac Bimbela López 21310116

Salvador Ignacio Rodríguez Flores 22110376

Programación Avanzada

3P

Proyecto final

Este manual de desarrollador proporciona una guía detallada sobre el código del programa de integración por trapecios. A continuación, se presentan los componentes y funcionalidades clave del código, junto con instrucciones para desarrolladores.

* Introducción

El presente manual proporciona una guía detallada para el desarrollo y comprensión del código del programa de integración por trapecios. Este programa, implementado en lenguaje C++, permite calcular una aproximación numérica de la integral de una función utilizando el método de los trapecios.

Objetivo del manual:

El objetivo de este manual es proporcionar a los desarrolladores una visión completa del código del programa de integración por trapecios, explicando su estructura, componentes clave y funcionalidades. Con esta información, los desarrolladores podrán comprender y modificar el código según sus necesidades y realizar mejoras en el programa.

Audiencia objetivo:

Este manual está dirigido a desarrolladores de software que tengan conocimientos básicos de programación en C++ y deseen entender y trabajar con el programa de integración por trapecios. También puede ser útil para estudiantes y entusiastas de la programación interesados en aprender sobre métodos numéricos de integración.

* Descripción del Código

El código proporcionado implementa una clase llamada "TrapeciosIntegracion" en lenguaje C++. Esta clase permite calcular una aproximación numérica de la integral de una función utilizando el método de los trapecios.

Componentes clave:

Variables miembros:

- a: Representa el límite inferior de integración.

- b: Representa el límite superior de integración.

- n: Indica el número de trapecios a utilizar en la aproximación de la integral.

Captura de pantalla de computadora

Descripción generada automáticamente

- funcion: Es un objeto de tipo function<double(double)> que representa la función a integrar. Permite al usuario definir la función que se desea integrar.

Texto

Descripción generada automáticamente

Constructor:

- TrapeciosIntegracion: Inicializa los valores de los límites de integración (`a` y `b`), así como el número de trapecios (`n`). El constructor puede ser modificado para adaptarse a requisitos específicos del usuario.

Funciones miembro:

- calcularFuncion: Solicita al usuario una expresión matemática que contiene la función a evaluar y un valor para evaluarla en ese punto. Utiliza un objeto llamado ExprTkEvaluator para evaluar la expresión y devuelve el resultado de la evaluación.

Captura de pantalla de computadora

Descripción generada automáticamente

- calcularIntegral: Realiza el cálculo de la integral utilizando el método de los trapecios. Divide el intervalo de integración en trapecios y suma los valores de la función en los puntos intermedios. Finalmente, aplica la fórmula del trapecio para obtener el resultado de la integral.

Captura de pantalla de computadora

Descripción generada automáticamente

Uso del programa:

El programa permite al usuario ingresar una expresión matemática que define la función a integrar y un valor para evaluarla utilizando la función calcularFuncion(). Además, proporciona la funcionalidad principal de calcular la integral de la función definida utilizando el método de los trapecios a través de la función calcularIntegral(). Los resultados se imprimen en la consola.

Nota: El código incluye comentarios para facilitar la comprensión de su funcionamiento. Los comentarios precedidos por //#include indican que se trata de líneas de código comentadas que pueden ser descomentadas si se desea utilizar una biblioteca llamada "muparser-master" para evaluación de expresiones matemáticas.

Texto

Descripción generada automáticamente

Captura de pantalla de computadora

Descripción generada automáticamente

Configuración del Entorno de Desarrollo

Para desarrollar y trabajar con el código proporcionado, se requiere la configuración de un entorno de desarrollo adecuado. A continuación, se detalla una configuración recomendada:

Instalación de un IDE (Entorno de Desarrollo Integrado): Recomendamos utilizar un IDE para facilitar el desarrollo y la edición del código. Algunas opciones populares para C++ son:

Visual Studio Code (VS Code): Un IDE liviano y altamente personalizable con una amplia variedad de extensiones para admitir el desarrollo en C++. Se puede descargar desde el sitio web oficial de Visual Studio Code.

Eclipse: Un IDE robusto y ampliamente utilizado para el desarrollo en C++. Proporciona características avanzadas y herramientas de depuración. Se puede descargar desde el sitio web oficial de Eclipse C++ IDE.

Configuración del compilador C++: Asegúrate de tener un compilador C++ instalado en tu sistema. La configuración varía según el sistema operativo:

Windows: Puedes utilizar MinGW, que proporciona un entorno de desarrollo C++ para Windows. Descarga el instalador de MinGW y sigue las instrucciones de instalación.

MacOS: Xcode, el IDE oficial de Apple, incluye el compilador C++. Puedes instalarlo desde la App Store o a través del sitio web oficial de Apple.

Linux: La mayoría de las distribuciones de Linux vienen con el compilador C++ instalado por defecto. Si no lo tienes, puedes instalarlo a través del gestor de paquetes de tu distribución (por ejemplo, GCC para Ubuntu).

Inclusión de bibliotecas: El código muestra una línea comentada `//#include "muparser-master"`. Si deseas utilizar la biblioteca "muparser" para evaluar expresiones matemáticas, debes seguir los siguientes pasos:

Descarga la biblioteca "muparser" desde su sitio web oficial o desde un repositorio confiable.

Extrae los archivos de la biblioteca en un directorio de tu elección.

Descomenta la línea //#include "muparser-master" en el código para incluir la biblioteca.

Configura la configuración del compilador para que incluya la ruta de la biblioteca durante la compilación.

Dependencias adicionales: El código hace uso de la biblioteca `ExprTkEvaluator`, que no se proporciona en el código. Asegúrate de tener acceso a esta biblioteca o de incluirla en tu proyecto si deseas utilizarla.

Una vez que hayas configurado el entorno de desarrollo con un IDE y un compilador C++, podrás abrir el archivo de código fuente en el IDE, compilarlo y ejecutarlo para probar y trabajar con el programa de integración por trapecios.

Estructura del Código

La estructura del código proporcionado se basa en la implementación de una clase llamada `TrapeciosIntegracion`, que se utiliza para realizar la integración numérica utilizando el método de los trapecios. A continuación, se describe la estructura y los componentes clave del código:

1. Inclusión de bibliotecas:

- #include <iostream>: Incluye la biblioteca estándar de C++ que proporciona funcionalidades de entrada y salida, como la lectura desde la entrada estándar y la impresión en la salida estándar.

- #include <cmath>: Incluye la biblioteca estándar de C++ que proporciona funciones matemáticas, como `sin`, `cos`, `exp`, entre otras.

2. Declaración de la clase `TrapeciosIntegracion`:

- La clase `TrapeciosIntegracion` encapsula la lógica relacionada con el cálculo de la integral utilizando el método de los trapecios.

- Tiene cuatro miembros de datos privados:

- `double a`: representa el límite inferior de integración.

- `double b`: representa el límite superior de integración.

- `int n`: representa el número de trapecios a utilizar en la aproximación de la integral.

- `function<double(double)> funcion`: es un objeto de función que representa la función a integrar.

- La clase tiene un constructor público que recibe los valores de `a`, `b` y `n` y los asigna a los miembros de datos correspondientes.

3. Método `calcularFuncion()`:

- Este método se utiliza para evaluar una expresión matemática ingresada por el usuario utilizando un objeto `ExprTkEvaluator` (que se asume que es parte de la biblioteca "muparser").

- Solicita al usuario que ingrese la expresión matemática y el valor de `x`.

- Utiliza el evaluador para calcular el resultado de la expresión y lo imprime en la salida estándar.

- Devuelve el resultado calculado.

4. Método `calcularIntegral()`:

- Este método se utiliza para calcular la integral numérica utilizando el método de los trapecios.

- Calcula el ancho del trapecio (`h`) dividiendo la diferencia entre `b` y `a` por `n`.

- Inicializa una variable `suma` en 0 para almacenar la suma de los valores de la función en los puntos intermedios.

- Utiliza un bucle `for` para iterar desde 1 hasta `n-1` e incrementar `i` en cada iteración.

- En cada iteración, calcula el valor de `x` utilizando la fórmula `x = a + i \* h` y suma el valor de la función evaluada en `x` a `suma`.

- Utiliza la fórmula del trapecio para calcular la integral aproximada y la almacena en la variable `integral`.

- Devuelve el valor de la integral calculada.

6. Funcionalidades del Programa

El código proporciona funcionalidades relacionadas con el cálculo de la integral numérica utilizando el método de los trapecios.

A continuación, se describen las principales funcionalidades del código:

1. Cálculo de una función:

- El método `calcularFuncion()` permite al usuario ingresar una expresión matemática que contiene una variable `x`.

- Utiliza un objeto `ExprTkEvaluator` (que se asume que es parte de la biblioteca "muparser") para evaluar la expresión con un valor específico de `x`.

- Solicita al usuario que ingrese la expresión matemática y el valor de `x`.

- Después de evaluar la expresión, imprime el resultado en la salida estándar.

2. Cálculo de la integral numérica:

- El método `calcularIntegral()` realiza el cálculo de la integral utilizando el método de los trapecios.

- Toma como entrada los límites inferior (`a`) y superior (`b`) de integración, así como el número de trapecios (`n`) a utilizar en la aproximación.

- Calcula el ancho del trapecio dividiendo la diferencia entre `b` y `a` por `n`.

- Itera sobre los puntos intermedios dentro del intervalo de integración y suma los valores de la función en estos puntos.

- Utiliza la fórmula del trapecio para calcular la integral aproximada.

- Devuelve el valor de la integral calculada.

Estas funcionalidades permiten a los desarrolladores utilizar la clase `TrapeciosIntegracion` para aproximar numéricamente la integral de una función específica utilizando el método de los trapecios. Los métodos proporcionados facilitan el cálculo de la función y la integral, brindando una herramienta útil para aplicaciones que requieren cálculos de integración numérica.

7. Configuración de la Función a Integrar

La configuración de la función a integrar en el código se realiza a través del método `calcularFuncion()`, el cual permite al usuario ingresar una expresión matemática que contiene la variable `x`. A continuación, se explica cómo se configura la función a integrar:

1. Solicitar la expresión matemática:

- El código utiliza `std::getline()` para solicitar al usuario que ingrese la expresión matemática a evaluar.

- La expresión debe estar en formato matemático válido y puede incluir operadores, funciones y la variable `x` como la variable de entrada.

2. Solicitar el valor de `x`:

- Después de obtener la expresión matemática, el código solicita al usuario que ingrese el valor de `x` que se utilizará para evaluar la función.

- Utiliza `std::cin` para leer el valor ingresado por el usuario.

3. Evaluación de la expresión:

- Utiliza un objeto `ExprTkEvaluator` (que se asume que es parte de la biblioteca "muparser") llamado `evaluator` para evaluar la expresión matemática con el valor de `x` especificado.

- El método `evaluateExpression()` del evaluador se utiliza para realizar la evaluación y devuelve el resultado de la expresión como un valor double.

4. Impresión del resultado:

- Después de evaluar la expresión, el código imprime el resultado en la salida estándar utilizando `std::cout`.

- Muestra el resultado de la evaluación de la expresión matemática.

La configuración de la función a integrar permite al usuario proporcionar una expresión matemática personalizada y evaluarla para obtener el resultado correspondiente. Esto brinda flexibilidad al usuario para trabajar con diversas funciones y obtener los valores deseados en el proceso de integración numérica utilizando el método de los trapecios.

8. Configuración de los Parámetros de Integración

El código proporciona una forma flexible de configurar los parámetros de integración, lo cual es fundamental para obtener resultados precisos y adaptados a las necesidades del usuario. A continuación, se explica cómo se realiza la configuración de cada uno de los parámetros en el código:

1. Límites de integración:

Los límites de integración definen el rango en el cual se calculará la integral de la función. El código utiliza dos variables: `a` y `b`, que representan el límite inferior y superior de integración, respectivamente. Estos valores se establecen al momento de crear una instancia de la clase `TrapeciosIntegracion` mediante su constructor.

Por ejemplo:

```

TrapeciosIntegracion integracion(0.0, 1.0, 100);

```

En este caso, se configura un objeto `integracion` con un límite inferior de integración de 0.0 y un límite superior de integración de 1.0.

2. Número de trapecios

El número de trapecios define la cantidad de subdivisiones que se utilizarán para aproximar la integral. Cuanto mayor sea este número, mayor será la precisión del resultado. En el código, se utiliza la variable `n` para representar el número de trapecios. Al igual que los límites de integración, este valor se establece al crear una instancia de la clase `TrapeciosIntegracion`.

Por ejemplo:

```

TrapeciosIntegracion integracion(0.0, 1.0, 100);

```

En este caso, se configura un objeto `integracion` con un número de trapecios igual a 100.

3. Función a integrar:

El código permite al usuario configurar la función que se desea integrar. Utiliza el tipo `std::function<double(double)>` y la variable `funcion` para almacenar la función a integrar. Este tipo de dato permite flexibilidad al aceptar diferentes tipos de funciones.

Para configurar la función a integrar, se deben seguir los siguientes pasos:

- Crear una función que tome un parámetro `double` y devuelva un resultado `double` de acuerdo a la expresión matemática deseada.

- Asignar la función creada a la variable `funcion` de la instancia de la clase `TrapeciosIntegracion`.

Por ejemplo:

```

double funcionEjemplo(double x) {

return x \* x;

}

// Configuración de la función a integrar

TrapeciosIntegracion integracion(0.0, 1.0, 100);

integracion.funcion = funcionEjemplo;

```

En este caso, se define la función `funcionEjemplo` que calcula el cuadrado de `x`, y luego se asigna esta función al miembro `funcion` del objeto `integracion`.

La flexibilidad en la configuración de los parámetros de integración permite al desarrollador adaptar el programa a diferentes funciones y rangos de integración. Esto facilita la exploración de distintos escenarios y cálculos personalizados.

9. Compilación y Ejecución del Programa

Para poder utilizar y ejecutar el código del programa de integración por trapecios, es necesario realizar los siguientes pasos de compilación y ejecución:

1. Configurar el entorno de desarrollo:

Asegúrate de tener un entorno de desarrollo instalado en tu sistema, como un compilador de C++ y un entorno de desarrollo integrado (IDE) o una línea de comandos para compilar y ejecutar el código. Algunas opciones populares incluyen GCC, Clang y Microsoft Visual Studio.

2. Crear un archivo fuente:

Crea un nuevo archivo con extensión `.cpp` y copia el código del programa de integración por trapecios en él. Guarda el archivo en una ubicación conveniente en tu sistema.

3. Incluir bibliotecas necesarias:

El código utiliza las bibliotecas `<iostream>` y `<cmath>`, que proporcionan funciones y utilidades para entrada/salida y operaciones matemáticas. Asegúrate de que estas bibliotecas estén incluidas en el código.

4. Opcional: Configurar dependencias adicionales:

Si el código requiere bibliotecas adicionales, como "muparser", asegúrate de tenerlas instaladas en tu sistema y de incluir las declaraciones de inclusión correspondientes en el código. En el código proporcionado, la inclusión de "muparser-master" está comentada (`//#include "muparser-master"`), lo que indica que no se está utilizando actualmente.

5. Compilar el código:

Abre una línea de comandos o tu IDE y navega hasta la ubicación del archivo fuente. Utiliza el comando de compilación correspondiente para compilar el código. Por ejemplo, si estás utilizando GCC, puedes ejecutar el siguiente comando en la línea de comandos:

```

g++ archivo\_fuente.cpp -o programa

```

Esto compilará el archivo fuente y generará un archivo ejecutable llamado "programa" (puedes cambiar el nombre según tus preferencias).

6. Ejecutar el programa:

Después de la compilación exitosa, ejecuta el programa utilizando el comando correspondiente en la línea de comandos o utilizando la función de ejecución de tu IDE. Por ejemplo, si estás en la línea de comandos, puedes ejecutar:

```

./programa

```

Esto ejecutará el programa de integración por trapecios y mostrará los mensajes en la consola.

7. Interactuar con el programa:

Durante la ejecución del programa, se te solicitará ingresar la expresión matemática a evaluar y el valor de 'x' correspondiente. Sigue las instrucciones proporcionadas en la consola para proporcionar la entrada requerida.

8. Observar los resultados:

El programa calculará la evaluación de la función y mostrará el resultado en la consola. También calculará la integral utilizando el método de los trapecios y mostrará el resultado correspondiente.

Siguiendo estos pasos, podrás compilar y ejecutar el programa de integración por trapecios en tu entorno de desarrollo, interactuar con él y obtener los resultados deseados.

Asegúrate de entender y personalizar los parámetros de integración según tus necesidades antes de ejecutar el programa.