CALCULADORA OPERATIVA

DOCUMENTACIÓN DE PROYECTO

Integrantes

Santiago Palacio Cardenas Cc. 1045076775

Ramón Steven Castrillón Martínez Cc. 1021802908

Profesora

Diana Margot Margot Lopez Herrera

Curso

Técnicas de Programación y Laboratorio

Universidad de Antioquia

Facultad de Ingeniería

Medellín

2024-2

Enfoque del proyecto

El propósito de este proyecto es desarrollar una calculadora operativa en Java que permita realizar operaciones aritméticas básicas utilizando diversos operadores, como aritméticos, relacionales, booleanos, de bits, incrementales, combinados, condicionales, y de cadenas que, además de cumplir con los requisitos asignados, resultara sencilla y comprensible para el usuario. Para ello, se diseñó un programa que lee los datos ingresados por el usuario, los valida y realiza las operaciones correspondientes, de manera intuitiva mediante el uso de menús e instrucciones de uso que facilitaran su comprensión.

Durante el desarrollo, se aplicaron principios de programación orientada a objetos, como la sobrecarga de métodos, el principio de responsabilidad única y, adicionalmente, se aplicó también el uso de interfaces. Se mantuvo el código limpio y bien estructurado, con un enfoque en la simplicidad y la claridad.

Explicacion del codigo

Para asegurar el buen funcionamiento del código, fue necesario importar las clases BufferedReader, IOException e InputStreamReader del paquete IO.

Paquetes Utilizados

Para el desarrollo del programa fue necesario la creación de 4 paquetes, lo cuales son Interface, Operation, Util y calculadoraOperativa.

Interface

Contiene interfaces que definen una serie de métodos para realizar operaciones específicas. Cada interfaz establece un conjunto de acciones que las clases encargadas de la implementación deben llevar a cabo.

Clases que contiene:

ArithmeticInterface: Define métodos para operaciones matemáticas básicas y avanzadas, como suma, resta, multiplicación, división, módulo, potencia, raíces y cálculos con factoriales. También incluye métodos para comparar números y determinar si son positivos o negativos.

BinaryInterface: Define métodos para operaciones con números binarios, como convertir entre binario y decimal, y realizar operaciones bit a bit (AND, OR, XOR, NOT) y desplazamientos de bits (izquierda y derecha).

ChainsInterface: Define métodos para manipular cadenas de texto, incluyendo concatenación, longitud, extracción de subcadenas, comprobación de contenidos, conversión a mayúsculas o minúsculas, comparación de igualdad y reemplazo de texto.

Operation

Este paquete contiene clases que implementan las operaciones definidas en las interfaces pertenecientes al paquete Interface.

Clases que contiene:

ArithmeticOperations: Implementa las operaciones matemáticas definidas en ArithmeticInterface.

Métodos que emplea:

- **add(double a, double b):** Suma dos números.
- **subtract(double a, double b):** Resta el segundo número del primero.
- **multiply(double a, double b):** Multiplica dos números.
- **divide(double a, double b):** Divide el primer número por el segundo. Devuelve NaN si el divisor es 0.
- **module(double a, double b):** Calcula el residuo de la división del primer número por el segundo. Devuelve NaN si el divisor es 0.
- **power(double a, double b):** Eleva el primer número a la potencia del segundo número.
- **squareRoot(double a):** Calcula la raíz cuadrada del número.
- **cubeRoot(double a):** Calcula la raíz cúbica del número.
- **absoluteValue(double a):** Devuelve el valor absoluto del número.
- **factorial(double a):** Calcula el factorial del número (solo funciona correctamente con números enteros no negativos).
- **increaseAdd(double a, double b, double c):** Aumenta el primer número sumando el segundo número c veces.

- **increaseSubtract(double a, double b, double c):** Aumenta el primer número restando el segundo número c veces.
- **increaseMultiply(double a, double b, double c):** Aumenta el primer número multiplicándolo por el segundo número c veces.
- **increaseDivide(double a, double b, double c):** Aumenta el primer número dividiéndolo por el segundo número c veces.
- **increaseModule(double a, double b, double c):** Aumenta el primer número aplicando el módulo con el segundo número c veces.
- **positiveOrNegative(double a):** Determina si el número es positivo o no positivo (cero o negativo).
- **mayor(double a, double b):** Compara si el primer número es mayor que el segundo.
- **menor(double a, double b):** Compara si el primer número es menor que el segundo.
- **mayorIgual(double a, double b):** Compara si el primer número es mayor o igual que el segundo.
- **menorIgual(double a, double b):** Compara si el primer número es menor o igual que el segundo.

Manejo de errores y excepciones:

- divide(double a, double b) y module(double a, double b): Verifican si el divisor b es 0 antes de realizar la operación. Si b es 0, el método devuelve Double.NaN (Not a Number) para indicar que la operación no es válida.
- increaseAdd(double a, double b, double c), increaseSubtract(double a, double b, double c), increaseMultiply(double a, double b, double c), increaseDivide(double a, double b, double c) y increaseModule(double a, double b, double c): Incluyen un bucle que solicita al usuario ingresar un valor mayor a 0 para c si no es válido.

Además, increaseDivide y increaseModule verifican si el divisor b es 0 antes de realizar la operación, devolviendo Double.NaN si es así.

BinaryOperations: Implementa las operaciones con números binarios definidas en BinaryInterface.

Métodos que emplea:

- **decimalToBinary(int decimal):** Convierte un número decimal a binario.
- **binaryToDecimal(String binary):** Convierte un número binario en formato de cadena a decimal.
- **bitwiseAnd(int a, int b):** Realiza una operación AND bit a bit entre dos números.
- **bitwiseOr(int a, int b):** Realiza una operación OR bit a bit entre dos números.
- **bitwiseXor(int a, int b):** Realiza una operación XOR bit a bit entre dos números.
- **bitwiseNot(int a):** Realiza una operación NOT bit a bit en un número.
- **leftShift(int a, int positions):** Desplaza los bits de un número a la izquierda un número específico de posiciones.
- **rightShift(int a, int positions):** Desplaza los bits de un número a la derecha un número específico de posiciones.

ChainsOperations: Implementa las manipulaciones de cadenas de texto definidas en ChainsInterface.

Métodos que emplea:

- **concatenate(String a, String b):** Une dos cadenas de texto.
- **length(String a):** Devuelve la longitud de una cadena de texto.

- **substring(String a, int start, int end):** Extrae una subcadena de una cadena de texto desde el índice start hasta el índice end.
- **contains(String a, String b):** Verifica si una cadena de texto contiene otra cadena.
- **toUpperCase(String a):** Convierte todos los caracteres de una cadena de texto a mayúsculas.
- **toLowerCase(String a):** Convierte todos los caracteres de una cadena de texto a minúsculas.
- **equals(String a, String b):** Compara dos cadenas de texto para ver si son iguales.
- **replace(String a, String target, String replacement):** Reemplaza todas las ocurrencias de una subcadena en una cadena de texto con otra subcadena.

Util

Permite manejar la entrada de datos desde el teclado.

Clases que contiene:

Lectura: proporciona métodos para leer diferentes tipos de datos desde el teclado usando un BufferedReader.

Métodos que emplea:

- **tecladoLinea(BufferedReader bIn, String label):** Muestra un mensaje (label) al usuario y lee una línea de texto ingresada por el usuario.
- **tecladoDouble(BufferedReader bIn, String label):** Muestra un mensaje y lee un número decimal (double) ingresado por el usuario. Si el usuario no ingresa un número válido, muestra un mensaje de error y solicita el dato nuevamente.
- **tecladoInt(BufferedReader bIn, String label):** Muestra un mensaje y lee un número entero (int) ingresado por el usuario. Si el usuario no ingresa un número válido, muestra un mensaje de error y solicita el dato nuevamente.
- **tecladoFloat(BufferedReader bIn, String label):** Muestra un mensaje y lee un número flotante (float) ingresado por el usuario. Si el usuario no ingresa un número válido, muestra un mensaje de error y solicita el dato nuevamente.
- **tecladoBoolean(BufferedReader bIn, String label):** Muestra un mensaje y lee un valor booleano (true/false) ingresado por el usuario. Si el usuario no ingresa un valor válido, muestra un mensaje de error y solicita el dato nuevamente.

Manejo de errores y excepciones:

- Cuatro métodos utilizan un bucle do-while para solicitar al usuario la entrada de datos. Si el usuario ingresa un valor que no puede ser convertido al tipo de dato correspondiente (double, int, float, boolean), el método captura una excepción (NumberFormatException para los números y IOException para los booleanos) y muestra un mensaje de error. Luego, vuelve a solicitar el dato hasta que el usuario ingrese un valor válido.

calculadoraOperativa

Contiene clases relacionadas con la interfaz y la funcionalidad principal de una calculadora.

Clases que contiene:

CalculadoraOperativa: Ejecuta el método main, que crea una instancia de UserInterface y llama al método Menu para iniciar la interfaz de usuario.

UserInterface: Maneja la interacción con el usuario y la navegación entre los diferentes menús de operaciones de la calculadora.

Métodos que emplea:

- **Menu():** Muestra el menú principal y procesa la opción seleccionada por el usuario.
- **mostrarMenuPrincipal():** Muestra las opciones del menú principal.
- **procesarOpcionMenuPrincipal(int opcion):** Ejecuta la opción seleccionada en el menú principal.
- **MenuBinary():** Muestra el menú de operaciones binarias y procesa la opción seleccionada.
- **mostrarMenuBinary():** Muestra las opciones del menú de operaciones binarias.
- **procesarOpcionMenuBinary(int opcion, BinaryOperations binaryOperations):** Ejecuta la opción seleccionada en el menú de operaciones binarias.
- **MenuArithmetic():** Muestra el menú de operaciones aritméticas y procesa la opción seleccionada.
- **mostrarMenuArithmetic():** Muestra las opciones del menú de operaciones aritméticas.

- **procesarOpcionMenuArithmetic(int opcion, ArithmeticOperations arithmeticOperations):** Ejecuta la opción seleccionada en el menú de operaciones aritméticas.
- **realizarOperacionesAritmeticas(ArithmeticOperations arithmeticOperations):** Ejecuta operaciones aritméticas básicas y avanzadas.
- **realizarOperacionesIncrementales(ArithmeticOperations arithmeticOperations):** Ejecuta operaciones de incremento y decremento.
- **evaluarPrioridadOperadores(ArithmeticOperations arithmeticOperations):** Evalúa expresiones matemáticas con prioridad de operadores.
- **mostrarOpcionesOperaciones():** Muestra las opciones de operaciones aritméticas con prioridad de operadores.
- **realizarOperacionSeleccionada(int opcion, ArithmeticOperations arithmeticOperations, double a, double b, double c, double d):** Ejecuta la operación seleccionada basada en la opción del usuario.
- **MenuChains():** Muestra el menú de operaciones con cadenas y procesa la opción seleccionada.
- **mostrarMenuChains():** Muestra las opciones del menú de operaciones con cadenas.
- **procesarOpcionMenuChains(int opcion, ChainsOperations chainsOperations):** Ejecuta la opción seleccionada en el menú de operaciones con cadenas.
- **evaluarOperacionesCondicionales(ArithmeticOperations arithmeticOperations):** Evalúa y muestra resultados de operaciones condicionales con números.

Manejo de errores y excepciones:

- Menu(), MenuBinary() y MenuArithmetic() capturan IOException y NumberFormatException. Además muestran un mensaje de error si hay un problema con la entrada/salida o si la entrada no es un número válido.
- MenuChains() captura IOException y muestra un mensaje de error si hay un problema con la entrada/salida.
- evaluarOperacionesCondicionales(ArithmeticOperations arithmeticOperations)**,** procesarOpcionMenuChains(int opcion, ChainsOperations chainsOperations)**,** evaluarPrioridadOperadores(ArithmeticOperations arithmeticOperations), procesarOpcionMenuArithmetic(int opcion, ArithmeticOperations arithmeticOperations)**,** realizarOperacionesIncrementales(ArithmeticOperations arithmeticOperations)**,** procesarOpcionMenuBinary(int opcion, BinaryOperations binaryOperations)**,** procesarOpcionMenuPrincipal(int opcion) y realizarOperacionesAritmeticas(ArithmeticOperations arithmeticOperations) lanzan IOException en caso de que se presente un error.

Rúbrica de autoevaluación

Funcionalidad 40 puntos: Excelente
Código 40 puntos: Excelente
PUR 10 puntos: Buena
POO 10 puntos: Excelente