UNIVERSIDAD DEL VALLE DE GUATEMALA

CC 2008 - Estructura de Datos

Sección 10

Ing. Michaelle Alexander Pérez Riz

HDT 2

Jonathan Zacarías-231104

June Herrera Watanabe -231038

GUATEMALA, 1 de febrero de 2024

Requisitos funcionales

El sistema proporciona una calculadora que evalúa expresiones en notación postfix. La calculadora utiliza una pila implementada mediante la interfaz Stack y la clase StackArrayList. La evaluación de expresiones postfix se lleva a cabo mediante la clase CalculadoraPostfix.

Métodos , atributos , visibilidad y parámetros:

Stack Interface

Métodos:

push(E item): Agrega un elemento a la pila.

pop(): E: Elimina y devuelve el elemento superior de la pila.

peek(): E: Devuelve el elemento superior de la pila sin eliminarlo.

empty(): boolean: Verifica si la pila está vacía.

size(): int: Retorna el tamaño de la pila.

StackArrayList Class:

Propósito y Funcionalidad:

Implementa una pila utilizando un ArrayList. Sigue la interfaz estándar de una pila.

Atributos:

data (tipo: ArrayList<E>, visibilidad: protected): Estructura de datos subyacente para almacenar elementos.

Métodos:

StackArrayList(): Constructor para inicializar el ArrayList.

push(E item): Agrega un elemento a la pila.

pop(): E: Elimina y devuelve el elemento superior de la pila.

peek(): E: Devuelve el elemento superior de la pila sin eliminarlo.

size(): int: Devuelve el número de elementos en la pila.

empty(): boolean: Verifica si la pila está vacía.

CalculadoraADT Interface:

Propósito y Funcionalidad:

Define las operaciones de una calculadora para expresiones en notación postfix.

Métodos:

evaluarExpresion(String expresion): int: Evalúa una expresión en notación postfix y devuelve el resultado.

CalculadoraPostfix Class:

Propósito y Funcionalidad:

Implementa la interfaz CalculadoraADT para evaluar expresiones en notación postfix.

Atributos:

pila (tipo: Stack<Integer>, visibilidad: private): Pila utilizada para evaluar las expresiones.

Métodos:

CalculadoraPostfix(Stack<Integer> pila): Constructor que recibe una pila como parámetro.

evaluarExpresion(String expresion): int: Evalúa una expresión en notación postfix y devuelve el resultado.

esOperando(String token): boolean: Verifica si el token es un operando válido.

esOperador(String token): boolean: Verifica si el token es un operador válido.

evaluarOperacion(String operador): void: Realiza la operación correspondiente según el operador.

Main Class:

Propósito y Funcionalidad:

Clase principal que utiliza la calculadora para evaluar expresiones postfix desde un archivo.

Métodos:

main(String[] args): Método principal que lee expresiones desde un archivo, las evalúa y muestra los resultados.

TestCalculadora Class:

Propósito y Funcionalidad:

Contiene pruebas unitarias para validar la funcionalidad de la clase CalculadoraPostfix.

Atributos:

calculadora (tipo: CalculadoraADT, visibilidad: private): Instancia de la calculadora utilizada en las pruebas.

Métodos:

setUp(): void: Inicializa la calculadora antes de ejecutar cada prueba.

testExpresionSimple(): void: Prueba la evaluación de una expresión postfix simple.

testExpresionCompleja(): void: Prueba la evaluación de una expresión postfix más compleja.

testExpresionInvalida(): void: Prueba el manejo de expresiones con caracteres no válidos.

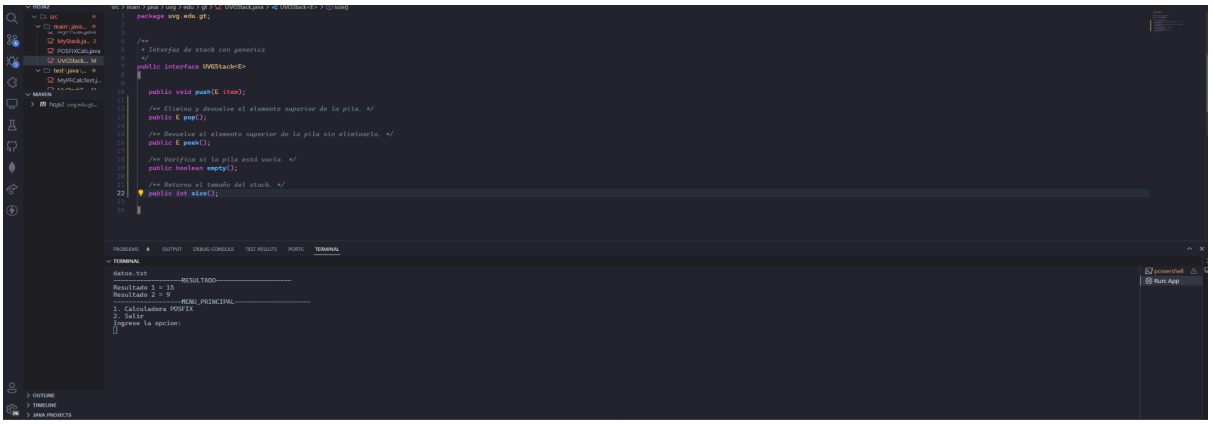
testDivisionPorCero(): void: Prueba el manejo de divisiones por cero.

testOperandosInsuficientes(): void: Prueba el manejo de expresiones con operadores insuficientes.

GitHub: https://github.com/skemono/HDT2

Funcionamiento al swapear clases

Nuestra clase integrada en un proyecto ajeno:



Clase ajena integrada en nuestro proyecto:

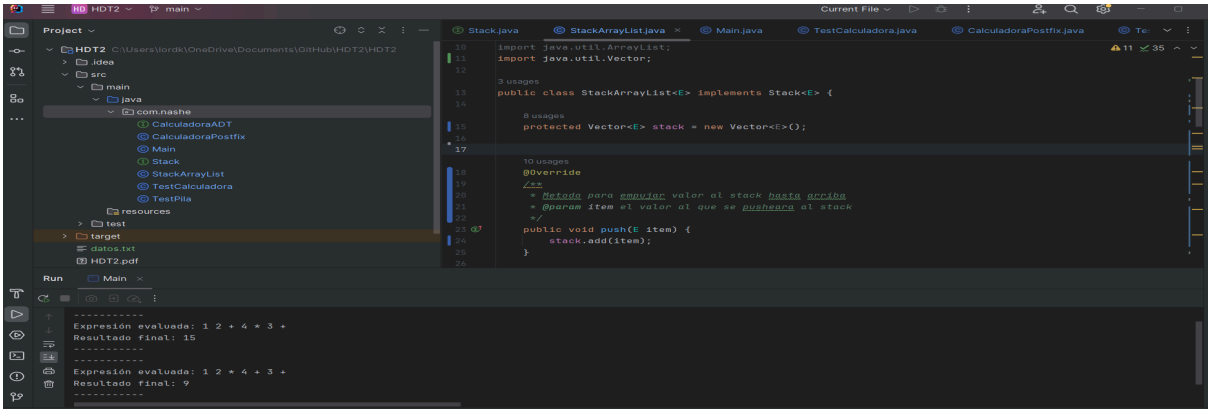


Diagrama de Secuencia UML

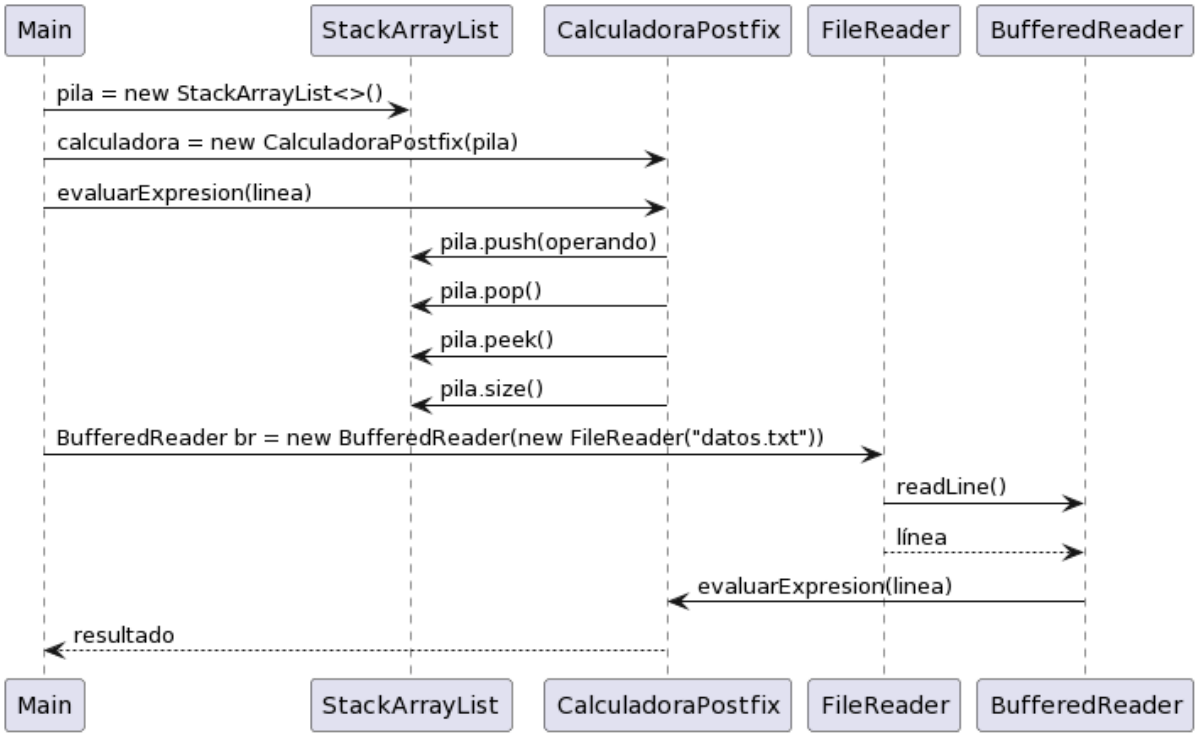


Diagrama de Clases UML