

Proyecto II:

Evaluación de expresiones matemáticas

Samuel Marín Soto
Escuela de Ingeniería en Computadores
Instituto Tecnológico de Costa Rica
Cartago, Costa Rica
Email: samarin@estudiantec.cr
Carné: 2023073212

Jose Isaac Corrales Cascante
Escuela de Ingeniería en Computadores
Instituto Tecnológico de Costa Rica
Cartago, Costa Rica
Email: jocorrales@estudiantec.cr
Carné: 202317258

I. DESCRIPCION DEL PROBLEMA

El problema en cuestión se centra en el desarrollo de una calculadora de expresiones que aproveche la potencia de los árboles de expresiones, implementando una arquitectura cliente-servidor a través de sockets. Además, se busca integrar una interfaz de usuario amigable para que los usuarios puedan interactuar con la calculadora de manera eficiente. Para hacer que esta calculadora sea aún más versátil, se plantea la inclusión de una cámara capaz de leer datos escritos a mano, los cuales serán posteriormente procesados digitalmente.

El desafío principal de este problema es la coordinación efectiva entre el cliente y el servidor a través de la red utilizando sockets para transmitir las expresiones matemáticas y recibir los resultados. Además, se necesita desarrollar una lógica sólida para la construcción y evaluación de árboles de expresiones, lo que implica el manejo de operadores matemáticos y operandos de manera adecuada.

La interfaz de usuario es otro componente crucial, ya que debe ser intuitiva y permitir a los usuarios ingresar expresiones, visualizar los resultados y, posiblemente, interactuar con la cámara para capturar expresiones escritas a mano.

La introducción de la cámara agrega un nivel adicional de complejidad al problema, ya que implica la captura de datos en formato de imagen y su procesamiento mediante técnicas de reconocimiento de escritura a mano o reconocimiento óptico de caracteres (OCR).

En resumen, el desarrollo de esta calculadora de expresiones con árboles de expresiones, arquitectura cliente-servidor con sockets, interfaz de usuario y funcionalidad de lectura de escritura a mano presenta un conjunto diverso de desafíos técnicos y ofrece una solución potencialmente útil para usuarios que deseen realizar cálculos matemáticos de manera conveniente y versátil.

II. DIAGRAMAS DE CLASES

III. ESTRUCTURAS DE DATOS DESARROLLADAS

A. Árbol de Expresión

Un árbol de expresión, en el contexto de la computación y las matemáticas, es una estructura de datos utilizada para representar expresiones matemáticas o lógicas de manera

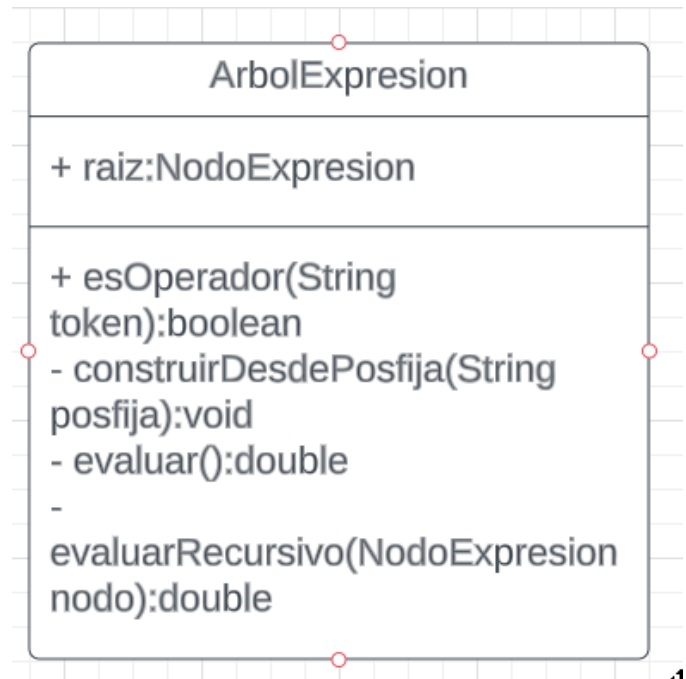


Fig. 1. Árbol expresion

jerárquica. En este tipo de árbol, cada nodo puede representar un operador o un operando de la expresión. Los nodos internos suelen representar operadores, como suma (+), resta (-), multiplicación (*), división (/), etc., mientras que los nodos hoja representan operandos, como números o variables.

La estructura del árbol refleja la jerarquía de las operaciones en la expresión. Los nodos superiores representan operaciones de mayor prioridad, mientras que los nodos más bajos representan operaciones de menor prioridad. Los árboles de expresión son útiles para evaluar expresiones de manera eficiente y también para simplificar y transformar expresiones algebraicas.

1) *Pila*: Una pila es una estructura de datos lineal que sigue el principio de LIFO (Last In, First Out), lo que significa que el último elemento que se inserta en la pila es el primero en ser retirado. Una pila es como una pila de libros o platos,

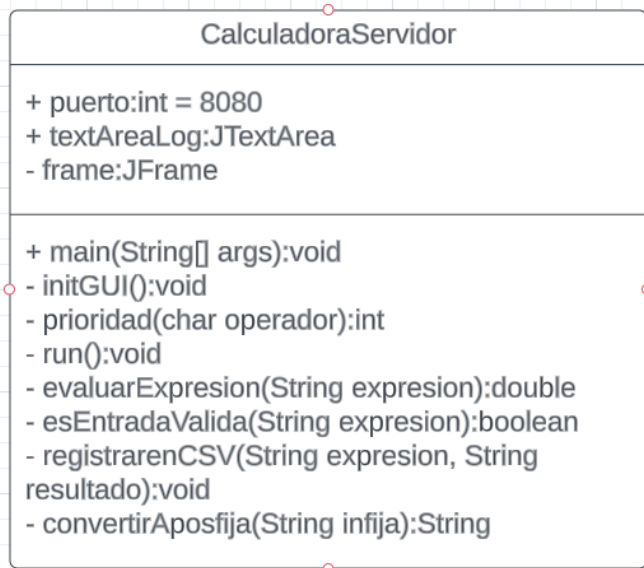


Fig. 2. Calculadora servidor

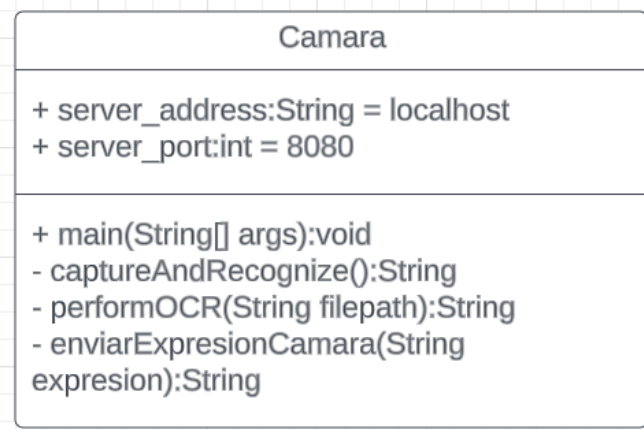


Fig. 3. Camara

donde solo puedes agregar o quitar elementos desde la parte superior.

Las operaciones principales en una pila son "push" (agregar un elemento a la parte superior) y "pop" (quitar el elemento superior). Además, se suele tener una operación llamada "peek" para ver el elemento superior sin eliminarlo. Las pilas son útiles en una variedad de aplicaciones, como la implementación de algoritmos de procesamiento de datos, evaluación de expresiones, seguimiento de la navegación en un programa, y más.

IV. BUGS DE GITHUB

Se encontraron varios bugs durante el desarrollo del proyecto. Por ejemplo, a la hora de escanear una foto con la camara, no se detectaba bien la tecla a presionar, entonces en vez de capturar la imagen en pantalla (lo cual se hacia con la

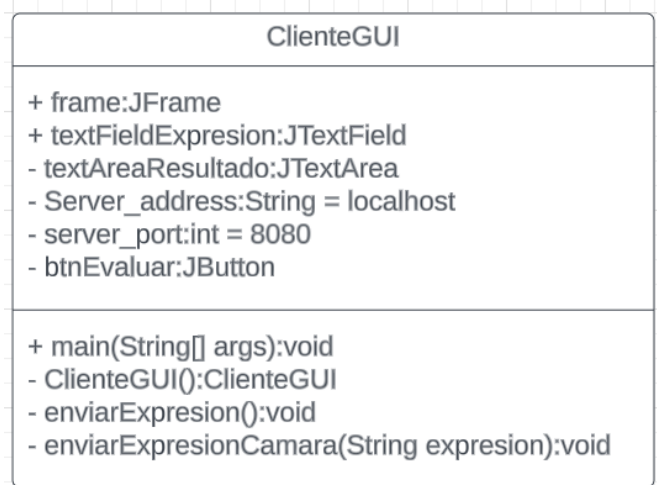


Fig. 4. Cliente gui

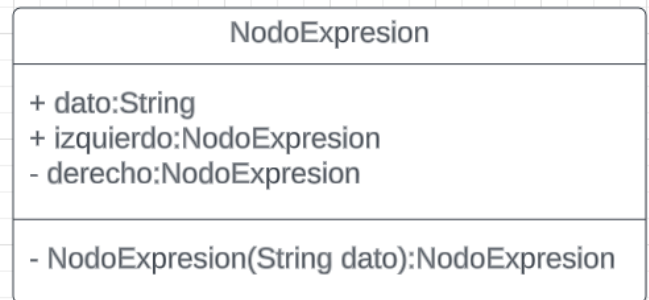


Fig. 5. Nodo expresion

tecla c), se cancelaba el seleccionamiento (lo cual se hace con la tecla ESC). Esto llevaba a otro bug, que es que cuando uno intentaba escanear y no se ponía la tecla correcta, la función que se suponía que retornaría el texto escaneado retornaba un mensaje de error, el cual se enviaba a la función que convertía la expresión en un árbol de expresión para calcularlo, y entonces intentaba resolver una hilera de caracteres. Hay muchos más bugs que se resolvieron pero ese es uno de los principales.