PROYECTO III

Evaluación de expresiones matemáticas

Instituto Tecnológico de Costa Rica
Escuela de Ingeniería en Computadores
Algoritmos y Estructuras de Datos II (CE 1103)
Verano 2024-2025

Objetivos

General

- Implementar una calculadora utilizando árboles de expresión binarios para evaluar expresiones de cualquier longitud.

Específicos

- Implementar la arquitectura cliente/servidor utilizando sockets TCP.
- Implementar árboles de expresión binarios.
- Almacenar y leer información de archivos con elementos separados por comas (csv).

Descripción general

Este proyecto consiste en construir una calculadora que evalúa expresiones de longitud arbitraria. Con ese fin se utilizará un árbol de expresión binaria. La calculadora realizará operaciones algebraicas simples (+, -, *, /, %, **), así como operaciones lógicas (and, or, not, xor) de cualquier longitud, colocando la expresión en un árbol de expresiones binarias y luego evaluando el árbol de expresiones.

Un árbol de expresión binaria es un árbol binario, que tiene como máximo dos hijos. Recuerde que existen dos tipos de nodos en un árbol binario, los nodos hoja que no tienen hijos y los nodos internos que tienen uno o más hijos (y forman el cuerpo de el árbol). En un árbol de expresión binaria, los nodos internos contendrán los operadores de la expresión (+,

-, *, /, %, etc). Los nodos hoja contendrán los operandos de la expresión (en nuestro caso, valores enteros). Un árbol de expresión binaria se muestra en la Figura 1.

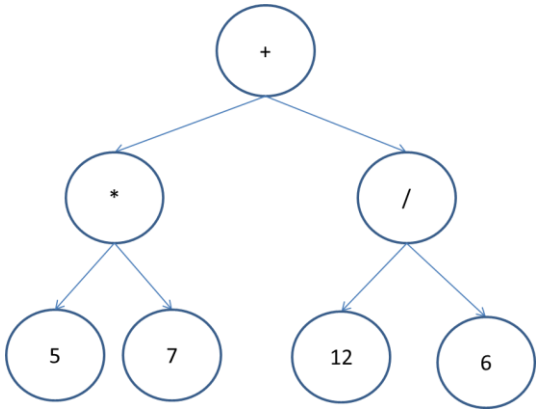

Figura 1. Árbol de expresión binaria.

El árbol en la Figura 1 representa la expresión (5 * 7) + (12/6) y el resultado de su evaluación es 37. Mientras que la expresión representada por el árbol de la Figura 2 es (5 * (10 - 15)) + 7 y su resultado es -18.

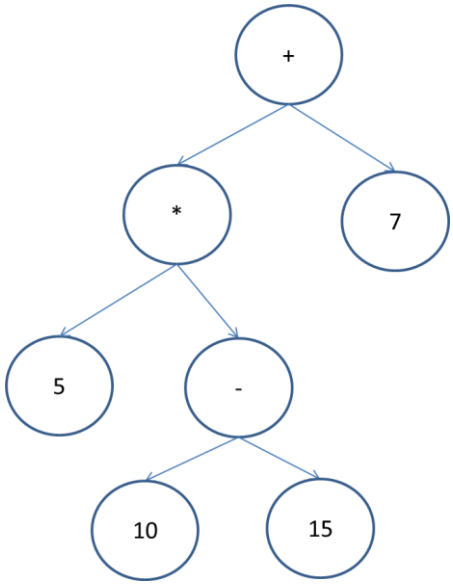

Figura 2. Árbol de la expresión (5 * (10 - 15)) + 7.

El proceso para evaluar árboles de expresión es recursivo. Primero evalúa el subárbol izquierdo, luego se evalúa el subárbol derecho y finalmente combina las dos soluciones usando el operador en el nodo.

El algoritmo para construir un árbol de expresión requiere utilizar la notación postfija (también conocida como notación polaca inversa, una versión modificada de una notación matemática inventada por matemáticos polacos a principios del siglo XX). La notación de sufijo se usa ampliamente en los círculos de computación porque las expresiones anotadas en notación de sufijo son completamente inequívocas sin tener que recurrir a paréntesis.

Requerimientos

A continuación, se resumen los requerimientos del proyecto. Esta misma tabla será la rúbrica de evaluación.

ID	DESCRIPCIÓN	PUNTOS
001	Implementar una arquitectura cliente/servidor utilizando sockets TCP que soporte n clientes. Las expresiones se ingresan en el cliente y las envía al servidor, donde se evalúan utilizando árboles de expresión. El servidor evalúa las expresiones y envía el resultado de vuelta al cliente, que despliega el valor calculado en un campo para ese fin. El servidor debe ser capaz de evaluar varias operaciones de clientes distintos al mismo tiempo.	35
002	El cliente cuenta con una interfaz gráfica sencilla que permita el ingreso de expresiones matemáticas de cualquier longitud que usan los operadores +, -, *, /,%, **, (,), and (&), or (|), xor (^) y not (~).	15
003	El servidor debe mantener el registro de las operaciones realizadas por cada cliente. Este registro debe contener la expresión, el resultado y la fecha en que realizó su evaluación en un archivo .csv.	10
004	El cliente puede consultar la lista de expresiones que mandó a evaluar, cuando recibe el resultado lo despliega en una tabla que contiene una columna para la fecha, la expresión y su resultado.	15
005	Planificación y administración del proyecto: se utilizará Azure DevOps para la administración del proyecto. Debe incluir: • De cada uno de los requerimientos presentes en esta misma tabla, se debe descomponer cada requerimiento en forma de historias de usuario. De cada requerimiento pueden extraerse varias historias de usuario. • Agrupar las historias de usuario en 3 sprints, de forma que se note un desarrollo incremental. • Asignación de los user stories entre las personas integrantes del grupo. El formato de los user stories es: Yo como <USUARIO> quiero <ACCION> para así <BENEFICIO>. Por ejemplo: Yo como usuario de la aplicación quiero ingresar una expresión matemática en la calculadora para obtener su valor numérico evaluado.	25
	Total	**100**

Aspectos operativos

- El trabajo se realizará en grupos de dos personas.
- El proyecto debe ser desarrollado en Java o C#.
- El uso de Git y Github es obligatorio.
- La fecha de entrega será según lo especificado en el TEC Digital.
- Se entrega en el TEC digital un link al repositorio donde vive el proyecto.
- Los estudiantes pueden seguir trabajando en el código hasta 15 minutos antes de la cita revisión oficial.

Evaluación

- Los proyectos que no cumplan con los siguientes requisitos no serán revisados:
    - Toda la solución debe estar integrada
- Aún cuando el código y la documentación tienen sus notas por separado, se aplican las siguientes restricciones
    - Si no se utiliza un manejador de código se obtiene una nota de 0.
- Cada estudiante tendrá 15 minutos para exponer su trabajo al profesor y defenderlo, es responsabilidad de los estudiantes mostrar todo el trabajo realizado, por lo que se recomienda tener todo listo antes de entrar a la defensa.
- Cada grupo es responsable de llevar los equipos requeridos para la revisión, si no cuentan con estos deberán avisar al menos 2 días antes de la revisión a el profesor para coordinar el préstamo de estos.
- Durante la revisión únicamente podrán participar los miembros del grupo, asistentes, otros profesores y el coordinador del área.