Proyecto II – Evaluación de expresiones matemáticas
Instituto Tecnológico de Costa Rica
Área Académica Ingeniería en Computadores
Algoritmos y Estructuras de Datos I (CE1103)
Segundo Semestre 2023
Valor 20%

TEC | Tecnológico de Costa Rica

OBJETIVO GENERAL

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

OBJETIVOS ESPECÍFICOS

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

DESCRIPCIÓN DEL PROBLEMA

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.

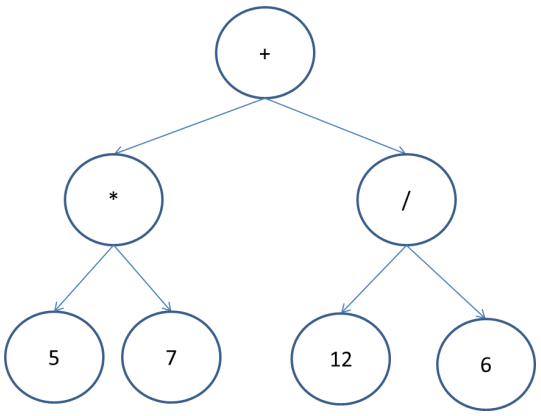

Figura 1. Árbol de expresión binaria.

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.

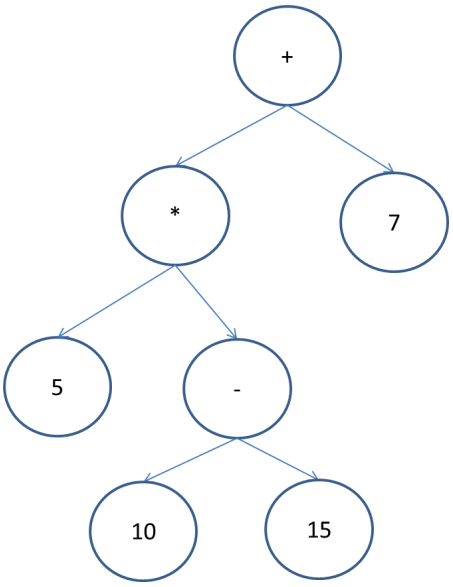

Figura 2. Árbol de la expresión (5 * (10 - 15)) + 7.

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.

Reconocimiento expresión matemática impresa

La aplicación debe tener un modo en el cual active una cámara web y evalúe una expresión matemática que esté impresa en una hoja. Para facilitar el reconocimiento del texto, la hoja será blanca y la expresión se colocará en fuente Arial, color negro. Se recomienda el uso de la biblioteca OpenCV y Tesseract OCR.

Requerimientos

1. Implementar una arquitectura cliente/servidor utilizando sockets TCP que soporte *n* clientes.
2. El cliente cuenta con una interfaz web sencilla que permita:
    a. El ingreso de expresiones matemáticas de cualquier longitud que usan los operadores +, -, *, /,%, **, (,), and (&), or (|), xor (^) y not (~).
    b. Habilitar la cámara web para evalúar una expresión matemática impresa en una hoja.
3. Las expresiones se ingresan en el cliente y las envía al servidor.
4. El servidor evalúa las expresiones y envía el resultado de vuelta al cliente, que despliega el valor calculado en un campo para ese fin.
5. El usuario puede ingresar expresiones como 5 * 3 / 8 + (95 % 5 – 10). Considere el orden de evaluación de los operadores con base en el mismo orden que utiliza Java.
6. 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.
7. 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.
8. El servidor debe ser capaz de evaluar varias operaciones de clientes distintos al mismo tiempo.

Documentación requerida

1. Internamente, el código se debe documentar utilizando Javadoc. Se debe generar el HTML.
2. La documentación externa deberá desarrollarse en Latex utilizando la plantilla llamada IEEE Conference Template, que se puede descargar desde este link: https://www.overleaf.com/gallery/tagged/ieee-official/page/1. Para el desarrollo en Latex se recomienda el uso de Overleaf. La documentación externa se hará en un documento que incluya lo siguiente (deberá entregarse un PDF):
    a. Breve descripción del problema.
    b. Diagrama de clases.
    c. Descripción de las estructuras de datos desarrolladas.
    d. Problemas encontrados en forma de bugs de *github*: En esta sección se detalla cualquier problema que no se ha podido solucionar en el trabajo.
3. **Planificación y administración del proyecto:** se utilizará Azure DevOps para la administración del proyecto. Debe incluir:
    a. Lista de features e historias de usuario identificados de la especificación.
    b. Plan de iteraciones que agrupen cada bloque de historias de usuario de forma que se vea un desarrollo incremental
    c. Descomposición de cada user story en tareas.
    d. Asignación de tareas entre las personas integrantes del grupo.

Aspectos operativos y evaluación:

1. **Fecha de entrega: De acuerdo con el cronograma del curso y lo establecido en el TEC Digital**
2. El proyecto tiene un valor de 20% de la nota del curso.
3. El trabajo es **en grupos de dos personas.**
4. Deben entregar en el TEC Digital un documento con el link del repositorio de GitHub, Azure DevOps y el PDF de la documentación. Para ambas herramientas deben dar acceso al correo del profesor.
5. Es obligatorio utilizar un Git y GitHUB para el control de versiones del código fuente y evidenciar el uso de Commits frecuentes.
6. Es obligatorio integrar toda la solución.
7. La presentación funcional tendrá un valor de 80% y la documentación externa 20%.
8. De las notas mencionadas en el punto anterior se calculará la Nota Final del Proyecto.
9. Se evaluará que la documentación sea coherente, acorde a la dificultad/tamaño del proyecto y el trabajo realizado. Se recomienda que realicen la documentación conforme se implementa el código.
10. La nota de la documentación externa es proporcional a la completitud del proyecto.
11. Las citas de revisión oficiales serán determinadas por el profesor durante las lecciones o mediante algún medio electrónico.
12. Los estudiantes pueden seguir trabajando en el código hasta 15 minutos antes de la primera cita de revisión oficial.
13. Aún cuando el código y la documentación externa tienen sus notas por separado, se aplican las siguientes restricciones
    a. Si no se entrega documentación externa, automáticamente se obtiene una nota de cero en la nota final del proyecto.
    b. Si no se utiliza un manejador de código se obtiene una nota de cero en la nota final del proyecto.
    c. Si la documentación externa no se entrega en la fecha indicada se obtiene una nota de cero en la nota final del proyecto.
    d. Sí el código no compila se obtendrá una nota de cero en la nota final del proyecto, por lo cual se recomienda realizar la defensa con un código funcional.

    e. El código debe ser desarrollado en Java (Windows), en caso contrario se obtendrá una nota de cero en la nota final del proyecto.
    f. Si alguna persona integrante del proyecto no se presenta a la revisión se asignará una nota de cero en la nota final del proyecto.
14. La revisión de la documentación podrá ser realizada antes, durante o después de la revisión del proyecto.
15. Cada estudiante tendrá como máximo 30 minutos para exponer su trabajo al profesor y realizar la defensa de éste, es responsabilidad de los estudiantes mostrar todo el trabajo realizado, por lo cual se recomienda tener todo listo antes de ingresar a la defensa.
16. Cada excepción o error que salga durante la ejecución del proyecto y que se considere debió haber sido contemplada durante el desarrollo del proyecto, se castigará con 2 puntos de la nota final del proyecto.
17. Cada estudiante 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 al profesor para coordinar el préstamo de estos.
18. Durante la revisión únicamente podrán participar el estudiante, asistentes, otros profesores y el coordinador del área.