TRABALHO 2

A execução de expressões aritméticas exige um processamento ordenado das operações parciais, ordenação essa derivada inicialmente da precedência estabelecida entre operadores, mas também de alterações explícitas de precedência representadas por parêntesis ou outros símbolos.

Observe as expressões abaixo e avalie a ordem de resolução:

5 + 3 * 4 – 2

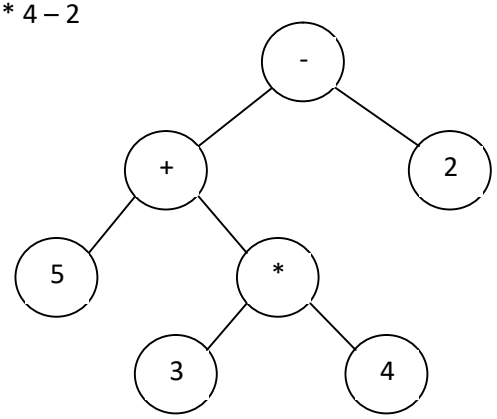
5 – 3 + 4 / 2

(5 + 3) * 4 / 2

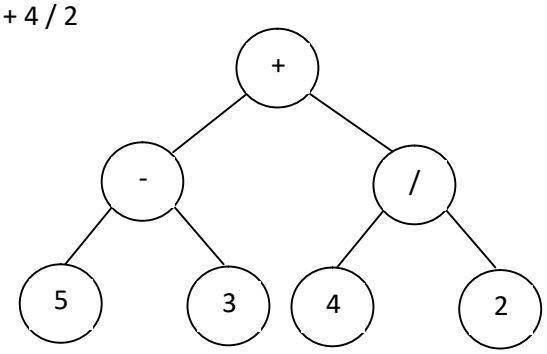
Uma das aplicações de árvores binárias é a representação não ambígua de expressões, seja aritméticas, algébricas, relacionais ou lógicas. Cada nó armazenará um operando ou um operador. Cada nó que armazena operador indica uma operação a ser feita entre o resultado obtido com a resolução da expressão representada na subárvore da esquerda e o resultado obtido com a resolução da expressão representada na subárvore da direita.

Em uma árvore binária que representa uma expressão, a distribuição das subárvores já reflete a ordenação conforme a precedência de resolução, dispensando parêntesis. As árvores a seguir refletem as expressões apresentadas anteriormente.

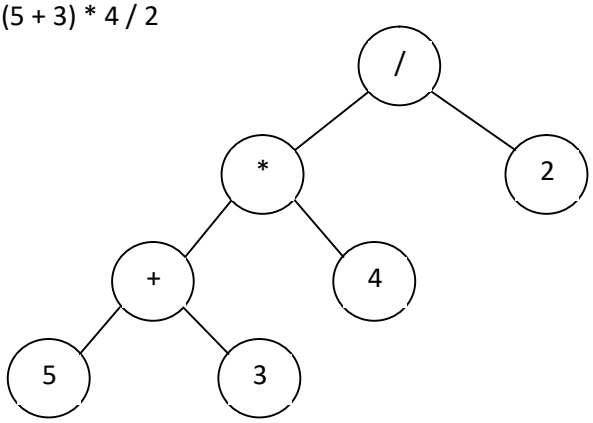
5 + 3 * 4 – 2



5 – 3 + 4 / 2



(5 + 3) * 4 / 2



O trabalho consiste em implementar operações para construir e manipular árvores que representam expressões aritméticas. O programa deve ter funções específicas para as seguintes operações:

1) Dada uma string que representa uma expressão, como “(5 + 3) * 4 / 2” , por exemplo, construir em memória a árvore equivalente. Nessa operação, algumas simplificações podem ser consideradas: todos os números serão de

apenas um dígito, as expressões não terão operadores unários de sinal (‘+’ e ‘–‘ para indicar positivo e negativo), e pode-se assumir que as expressões informadas sempre estarão corretas/completas. Por outro lado, é indispensável considerar a existência de parêntesis, inclusive aninhados, e a leitura de expressões de diversos tamanhos.

2) Com uma árvore já construída, calcular e exibir o resultado da expressão representada.

3) A partir de uma árvore já construída, gerar uma string contendo a expressão aritmética, na notação infixada (a normal, a mesma da entrada), que está representada na árvore. Essa string deve apresentar parêntesis apenas onde são realmente necessários para representar a mesma ordem de execução da árvore.

ATENÇÃO, essa string da operação 3) deve ser gerada a partir da árvore, e não simplesmente exibir a string informada na operação descrita em 1), usada na construção. Note que essa string gerada a partir da árvore pode inclusive ser diferente da usada na construção.

O programa deve ter ainda funções para exibir a árvore quando solicitado, e para eliminar a árvore antes do encerramento da execução. Essas funções podem ser genéricas, aplicáveis a qualquer árvore binária.

Orientações de entrega:

O código fonte (arquivo “.c” funcional, que possa ser executado para testar) deve ser enviado via Moodle conforme data e horário especificados na atividade. O trabalho representa 30% da segunda nota para quem fizer e apresentar. O código deve ser desenvolvido na linguagem de programação C, em grupos de até 2 (DUAS, o que significa que são DUAS no máximo!) pessoas. Casos com grupos com mais de dois integrantes, e/ou soluções iguais entre dois grupos, serão considerados trabalhos entregues e receberão a nota 0 (zero).

O conteúdo entregue será apresentado, em horário a combinar. Na apresentação, poderá ser solicitado que cada membro do grupo apresente algum código ou trecho de código individualmente, conforme ESCOLHA DO PROFESSOR. A entrega dos exercícios será desconsiderada para aqueles alunos que não apresentarem o código do exercício definido pelo professor. Portanto, é importante que todos os membros do grupo tenham domínio sobre todo o conteúdo entregue.