3. Stacks and Queues :: Example Application :: Evaluating Arithmetic Expressions

How would we write a program that would evaluate **arithmetic expressions** such as:

3 + 4
3 - 4 * 5
(2.2 + 3.3) * (5.5 - (7.7 * 1.3))

The approach is to use a stack data structure, and in particular, two stacks: one for the numbers (known as **operands**) and one for the **operators**. The former we shall call the operand stack and the latter the operator stack.

3. Stacks and Queues :: Example Application :: Evaluating Arithmetic Expressions

Consider evaluating $3 + 4$. The steps are:

1. Create the operand and operator stacks. Both are empty at the beginning.
2. Scan the first operand (3) and push it onto the operand stack.
3. Scan the first operator (+) and push it onto the operator stack.
4. Scan the next operand (4) and push it onto the operand stack.
5. The end of the expression has been reached. Evaluate "the top".
   a. Pop the top number from the operand stack. Call this $right = 4$.
   b. Pop the top number from the operand stack. Call this $left = 3$.
   c. Pop the top operator from the operator stack. Call this $op = +$.
   d. Perform the operation specified by op on $left$ and $right$; push the result onto the operand stack.
6. The result of evaluating the expression (7) is the number on top of the operand stack.