Polish Notation vs. Infix Notation

By: Thiviyan Sithganesan

Infix Notation (the notation that most people are familiar with using) places the mathematical operator between two numbers.

Example:

$$2 + 3$$

Prefix Notation (otherwise known as **Polish Notation**) places the mathematical operator before the numbers rather than between two numbers.

Example:

$$+ 2 3$$

Although the difference may appear to be of no interest, there are significant benefits to using Polish notation rather than infix notation.

For instance, simple mathematical operators (e.g. '+') must be binary in infix notation which is inefficient when adding multiple numbers (e.g. 1 + 2 + 3 + 4). However, using prefix notation can accomplish this using only a single instance of the operator (e.g. + 1 2 3 4). Comparatively, infix notation uses n-1 more characters (not including spaces) than Polish notation (for n > 2), which could be significant in large programs involving many mathematical operations.

Another upside to using Polish notation is clarity in evaluating algebraic expressions. Evidence of this is the need for an "order of operations" when using infix notation (i.e. BEDMAS/PEDMAS) which adds an unnecessary layer of difficulty to these problems. Meanwhile, using Polish notation eliminates the need for parentheses all together because the order of operations is well-defined.

Infix Notation:

$$\begin{aligned} & 1+3\times(6\div 2) \\ = \quad & 1+3\times 3 \\ = \quad & 1+9 \\ = \quad & 10 \end{aligned}$$

When using infix notation, BEDMAS suggests that we should evaluate the brackets first and then the product and finally the summation to arrive at the answer, which is 10. This non-linear system can become confusing when there are many expressions and operators involved which is why some people struggle with this concept.

Polish Notation:

$$\begin{aligned} & +1\times 3\div 6\ 2 \\ = \quad & +1\times 3\ 3 \\ = \quad & +1\ 9 \\ = \quad & 10 \end{aligned}$$

Contrary to infix notation, Polish notation is read from left-to-right and each operator is evaluated step-by-step to arrive at the solution in a systematic manner. For the above example, the addition operator cannot be evaluated until the multiplication operator is evaluated, and the multiplication operator cannot be evaluated until the quotient operator is evaluated.

Regardless of whether I have convinced anyone that Polish notation can be more efficient that infix notation, there are some programming languages that use Polish notation by default (e.g. variants of Lisp) and others that use post-fix notation (e.g. Forth) so it might be worth your while to familiarize yourself with these different kinds of notations as a programmer. Though Polish notation may appear complex at first, it becomes second-nature with a bit of practice. Eventually, you may find yourself wondering why we even bother using infix notation.