Reverse Polish Notation (RPN)

you will learn how to use a ***stack*** *ADT to evaluate a mathematical expression written in* ***reverse Polish notation*** *(RPN= postfix notation).*

(A+B)*C	AB+C*
infix	postfix

Notice that the **postfix** expression **HAS NO AMBIGUITY**, so it does not need **parentheses** to ensure that A+B is computed before the multiplication of C.

Try converting these example infix expressions to postfix form.

(A+B)*(C+D)/(E-F) (AB+)*(CD+)/(EF-) (AB+CD+*)/(EF-) AB+CD+*EF-/	A+B+C+D	((A+B)*C+D)/E	A+(B-(C+D*E))

We can use a stack to evaluate a postfix expression in the following algorithm.

> Scanning from left to right when we encounter an operand we push its value onto a stack. When we encounter a binary (two operand) operator we pop two values off the stack, perform the indicated operation and then push the result back onto the stack. When we are finished we can pop the stack to get the final result.

Enter values for three variables A=1, B=2 and C=3.

AB+C*	AB+C*	AB+C*	AB+C*	AB+C*
1	2 1	3	3 3	9
push A	push B	pop B pop A push A+B	push C	pop C pop A+B push (A+B)*C