ODUOR MACPHALEN LOWELL

SCT222-0399/2023

DATA STRUCTURES & ALGORITHMS

Convert the following expression into an expression tree using a stack and show the process;

a + b / c * d - e

Working

Token	Operator Stack	Postfix
a		a
+	+	a
b	+	ab
/	+/	ab
c	+/	abc
*	+*	abc/
d	+*	abc/d
-	-	abc/d*+
e	-	abc/d*+e
		abc/d*+e-

['a']

['a','b']

['a','b','c']

['a','(b/c)']

['a','(b/c)',d]

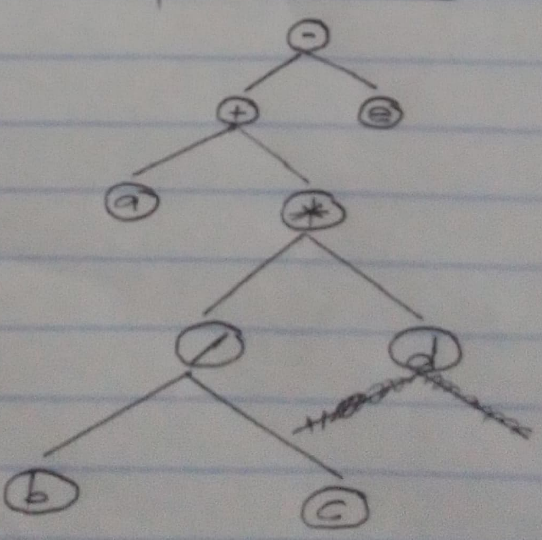
['a','((b/c)*d)']

['(a+((b/d)*d))']

['(a+((b/d)*d))','e']

['((a+((b/d)*d))-e)']

Expression tree



Convert the expression into postfix

Hand method

a + b / c * d - e

a + bc / * d - e

a + bc / d * - e

abc / d * + - e

abc / d * + e -

Given the following prefix expression convert it into an infix using a stack, show workings;

+a*bc

Workings

Token	Stack	Infix
c	c	
b	cb	
*		b*c
a	(b*c)a	
+		a+(b*c)

a+(b*c)