

This is a paragraph with `inline raw text` that now has a background. It also works with longer snippets like `code with spaces`.

1. Lecture 6

1.1. Table of Contents

1.2. Evaluating Mathematical Expressions

1.2.1. Evaluation

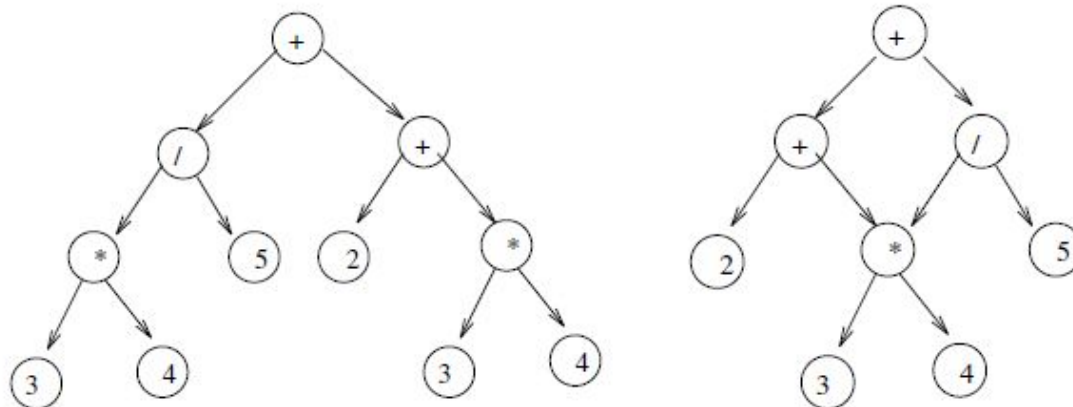


Figure 5.17: Expression $2 + 3 * 4 + (3 * 4)/5$ as a tree and a DAG

1.2.2. Infix Notation

$A + ((B \cdot C) - (D/E^F) \cdot G) \cdot H$

$(a+b)$ left subtree \cdot operator \cdot right subtree

Readable by humans

1.2.3. Postfix Notation

Readable by computers/calculators. Used in languages like LISP

$(ab+)$ left subtree \cdot right subtree \cdot operator

1.2.3.1. Advantages of Postfix

- No parenthesis
- No operator precedence
- No associativity

Infix	Postfix
$A + ((B \cdot C) - (D/E^F) \cdot G) \cdot H$	$ABC \cdot DEF^{\wedge}/G \cdot -H \cdot +$

Postfix notation was initially called RPN (Reverse Polish Notation) named after the Polish scientist Jan Lucasiewicz. It was first used in a calculator in 1938

1.2.3.2. Rules for Postfix Evaluation

1. Operand -> Postfix
2. Open Bracket -> Push to Stack
3. Closed Bracket -> Pop everything one by one up to and including an opening bracket and put in postfix
4. Operator -> Follow the chart, then push the current operator to stack

What to do?	Precedence	Associativity
Pop	$\text{tos} > \text{op}$	-
Pop	$\text{tos} = \text{op}$	left -> right
Push	$\text{tos} = \text{op}$	right -> left
Push	$\text{tos} < \text{op}$	-

Column1	Column2	Column3
A	(empty)	A
+	+	A
(+(A
B	+(AB
.	+(.	AB
C	+(.	ABC
-	+(-	ABC .
(+(-(ABC .
D	+(-(ABC . D
/	+(-(/	ABC . D
E	+(-(/	ABC . DE
^	+(-(/ ^	ABC . DE
F	+(-(/ ^	ABC . DEF
)	+(-	ABC . DEF ^ /
.	+(-.	ABC . DEF ^ /
G	+(-.	ABC . DEF ^ / G
)	+	ABC . DEF ^ / G .
H	+	ABC . DEF ^ / G . H

End of expression -> Pop stack

Final Expression: `ABC . DEF^/G . -H . +`

Another way of doing it is traversing the tree top down and adding an element only when you encounter it for the last time

1.2.3.3. Puzzle

Q: Convert to Postfix: `(56 + 12) . 3 - 4`

A: `56 12 + 3 . 4 -`

Q: A^B^C

Character	Stack	Postfix
A	(empty)	A
^	^	A
B	^	AB
^	^^	AB
C	^^	ABC

End of expression -> Pop stack

A: $ABC ^^$

1.2.3.4. Evaluating Postfix

Postfix Evaluation is done using a stack, scanning the expression left -> right

- Operand -> Push to Stack
- Operator:
 - Pop the top element (y)
 - Pop next element (x)
 - Calculate $z = x \text{ operator } y$
 - Push z to stack

1.3. Queue

First-In-First-Out (FIFO) Data Structure

1.3.1. Queue Operations

- Insert (enqueue)
- Delete (dequeue)