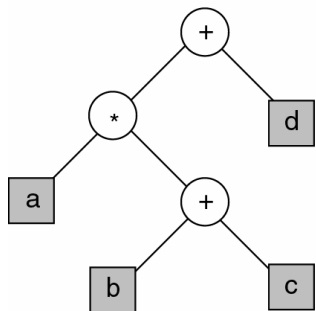


CST8130: Data Structures

Expression Trees (pages 322-324): Defining in a Binary Tree

- leaf nodes: operands
- root and internal nodes: operators
- subtrees → subexpressions (subroot is an operator)

a * (b + c) + d



Expression Trees: Infix Traversal

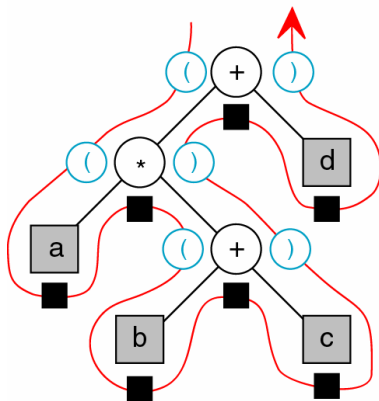
- start subexpression: open parenthesis
- finish subexpression: close parenthesis

```

InFix(BinaryNode* pbnSubRoot)
{
if (pbnSubRoot != NULL) {
if (pbnSubRoot->token is operand)
display (pbnSubRoot->token)
else {
display (open parenthesis)
InFix (pbnSubRoot->pbnLeft)
display (pbnSubRoot->token)
InFix (pbnSubRoot->pbnRight)
display (close parenthesis)
}
}
}

```

((a * (b + c)) + d)

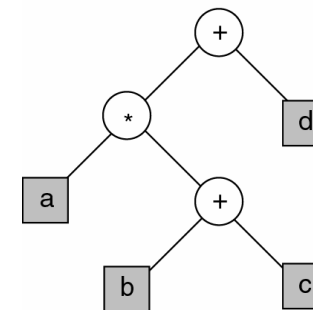


Expression Trees: Postfix Traversal

```

PostFix(BinaryNode* pbnSubRoot)
{
if (pbn != NULL) {
PostFix (pbnSubRoot->pbnLeft)
PostFix (pbnSubRoot->pbnRight)
display (pbnSubRoot->token)
}
}

```



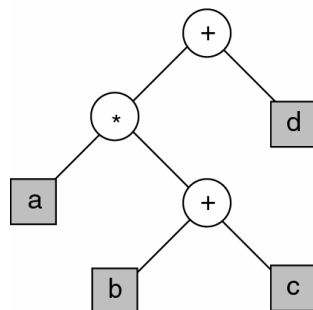
Output:

Expression Trees: Prefix Traversal

```

PreFix(BinaryNode* pbnSubRoot)
{
if (pbn != NULL) {
display (pbnSubRoot->token)
PreFix (pbnSubRoot->pbnLeft)
PreFix (pbnSubRoot->pbnRight)
}
}

```



Output: