Seminar 07

Thursday, June 2, 2022 3:05 PM

# Binary trees

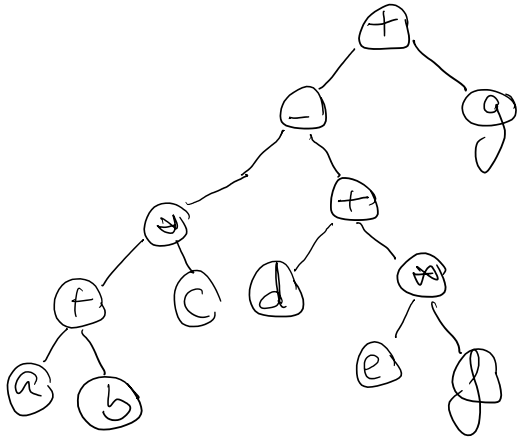
1) Create the binary tree for an arithmetic expression:

$(a+b)*c-(d+e*f)+g$

operands: a, b, c, d, e, f, g

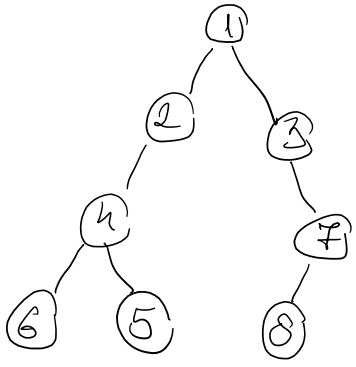
operators: +, -, *, /

$ab+c*def*+-g+$ ⇒ postfix notation




- Preorder: 1 2 4 6 5 3 7 8
- Inorder: 6 4 5 2 1 3 8 7
- Postorder: 6 5 4 2 8 7 3 1
- (DFS)

Level-order: 1 2 3 4 7 6 5 8
(BFS)




# Implementation

## Stack (operations)

init, push, pop, isEmpty, Top

Node

info: TElem (operand/operator)
left: ↑Node
right: ↑Node

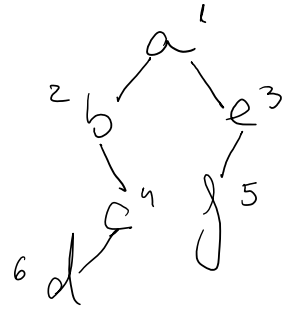
BT

root: ↑Node

subalgorithm build_tree(postE, bT) is:
    init(S)
    for e in postE execute:
        if e is operand then:
            allocate(newNode)
            [newNode].left ← NIL
            [newNode].right ← NIL
            [newNode].info ← e
            push(S, newNode)
        else:
            allocate(newNode)
            [newNode].right ← pop(S)
            [newNode].left ← pop(S)

```
        [newNode].info ← e
        push(S, newNode)
    endif
endfor
bT.root ← pop(S)
endsubalgorithm
```

2) Generate a table with information of a binary tree.




| | info | indexLeft | indexRight |
|---|---|---|---|
| 1 | a | 2 | 3 |
| 2 | b | -1 | 4 |
| 3 | e | 5 | -1 |
| 4 | c | 6 | -1 |
| 5 | f | - | - |
| 6 | d | - | - |

Dynamic array (operations)
init, addLast(a,e), addPos(a,e,p), removeLast, removePos(a,p), ...

Lucie
info:
indexLeft
indexRight

subalgorithm table(tree, table) is:
    init(g)

init (array)
push (q, tree.root)
counter ← 1

while ¬isEmpty(q) execute.
    e ← pop(q)
    allocate(l)
    l.info ← e.info
    l.indexLeft