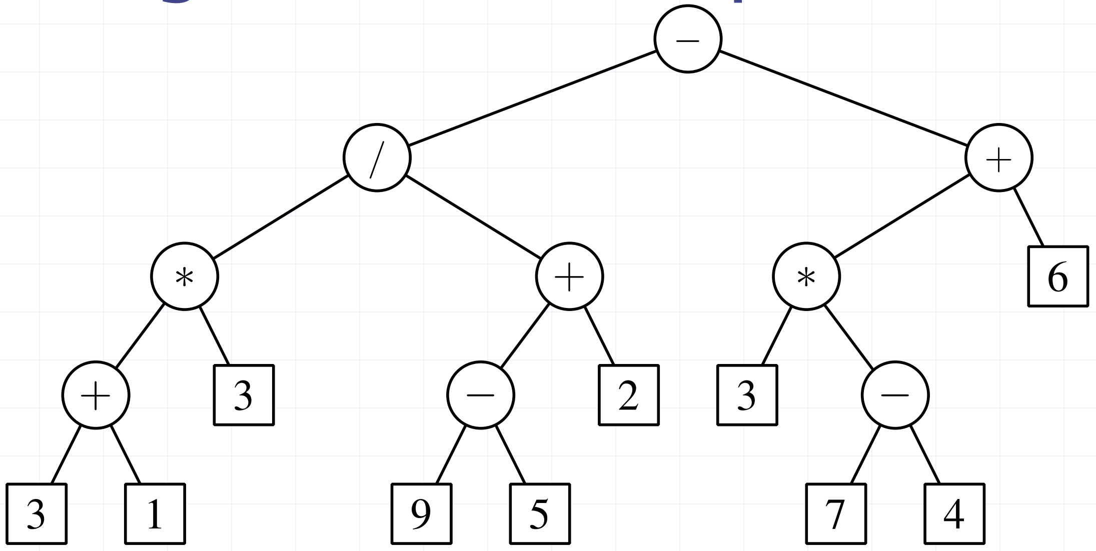
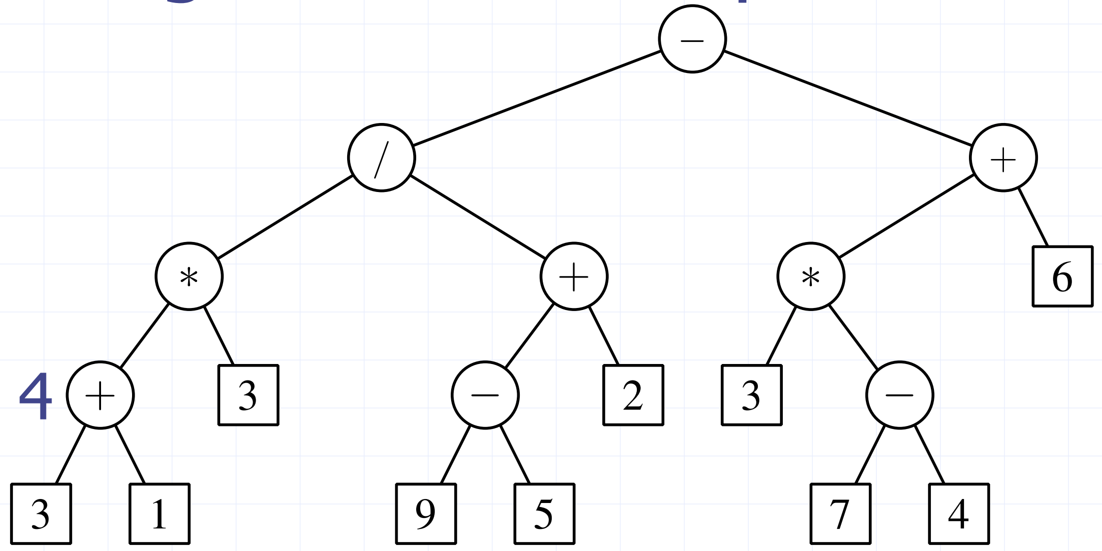
Application of Postorder Traversal

- Evaluating Arithmetic Expressions



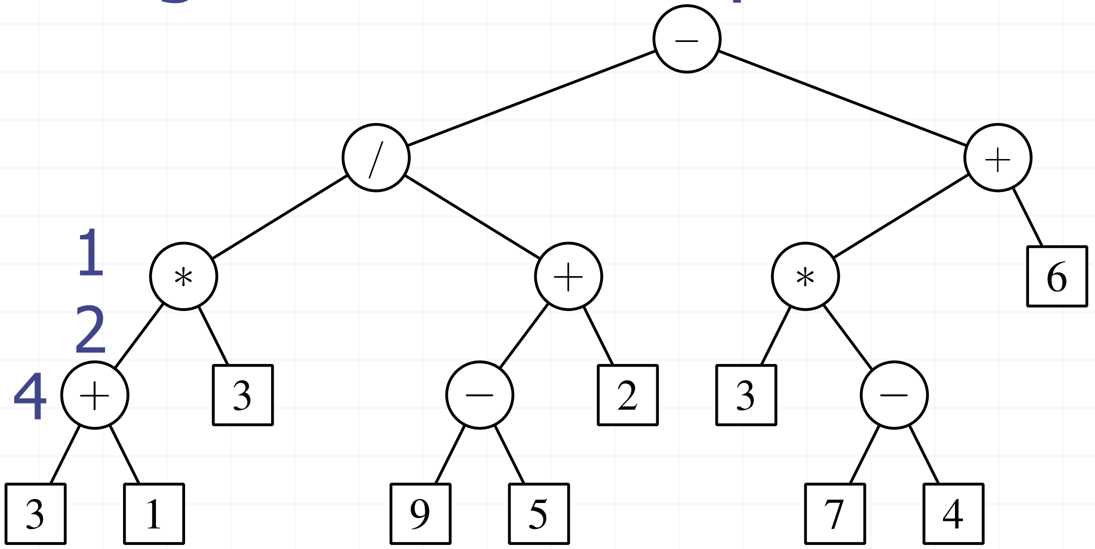
Application of Postorder Traversal

- Evaluating Arithmetic Expressions



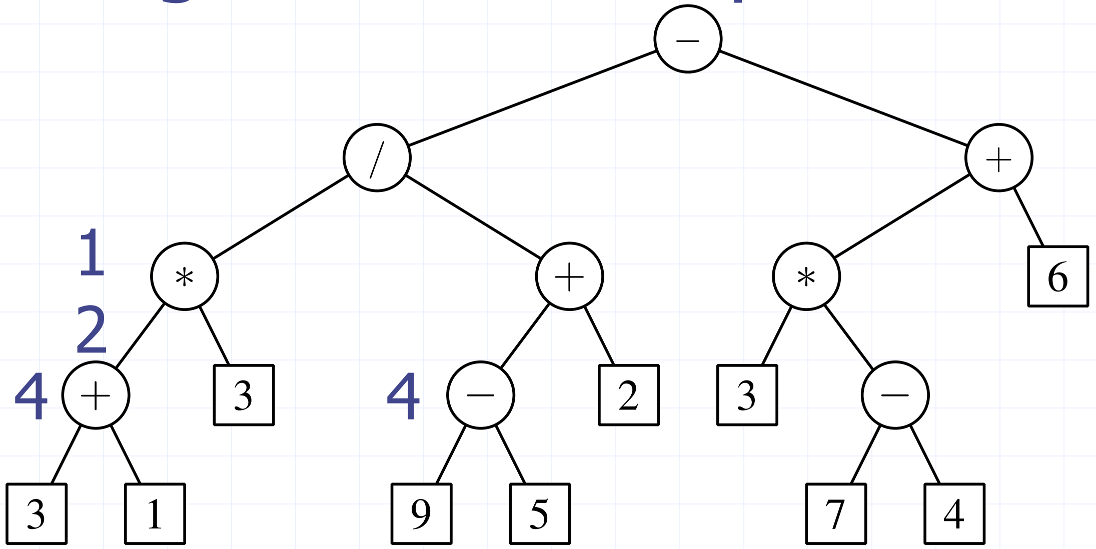
Application of Postorder Traversal

- Evaluating Arithmetic Expressions



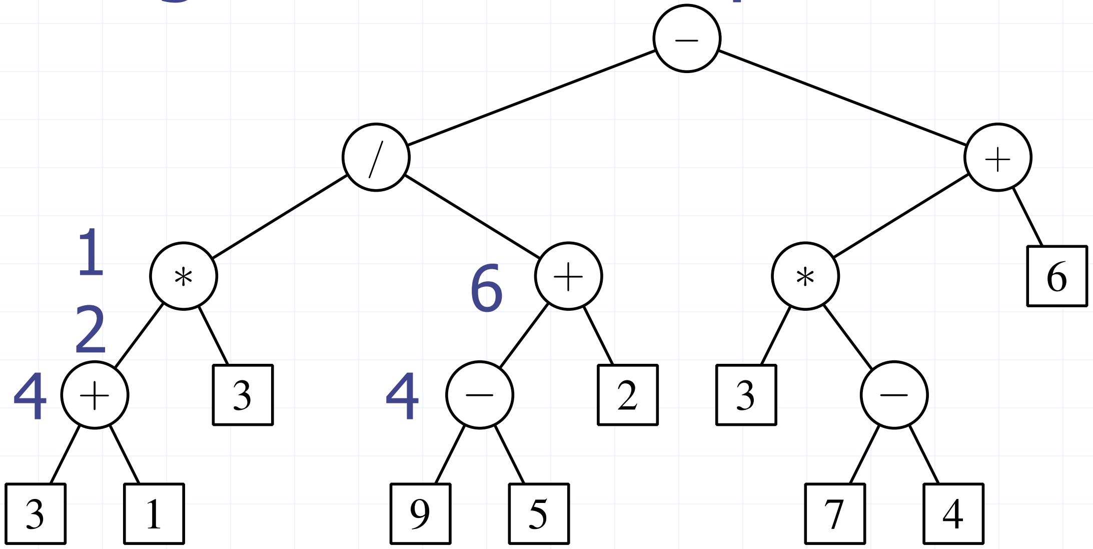
Application of Postorder Traversal

- Evaluating Arithmetic Expressions



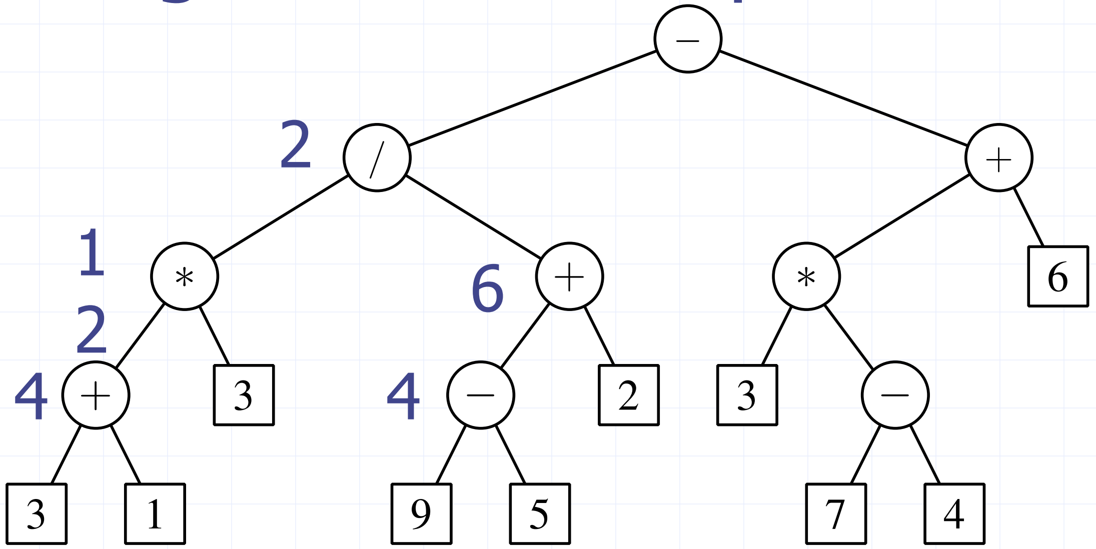
Application of Postorder Traversal

- Evaluating Arithmetic Expressions



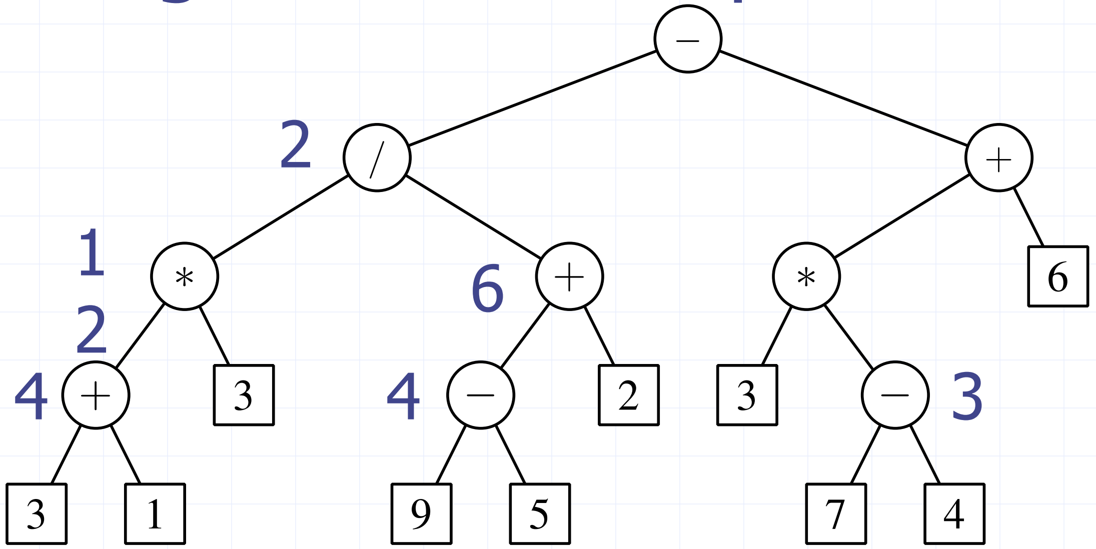
Application of Postorder Traversal

- Evaluating Arithmetic Expressions



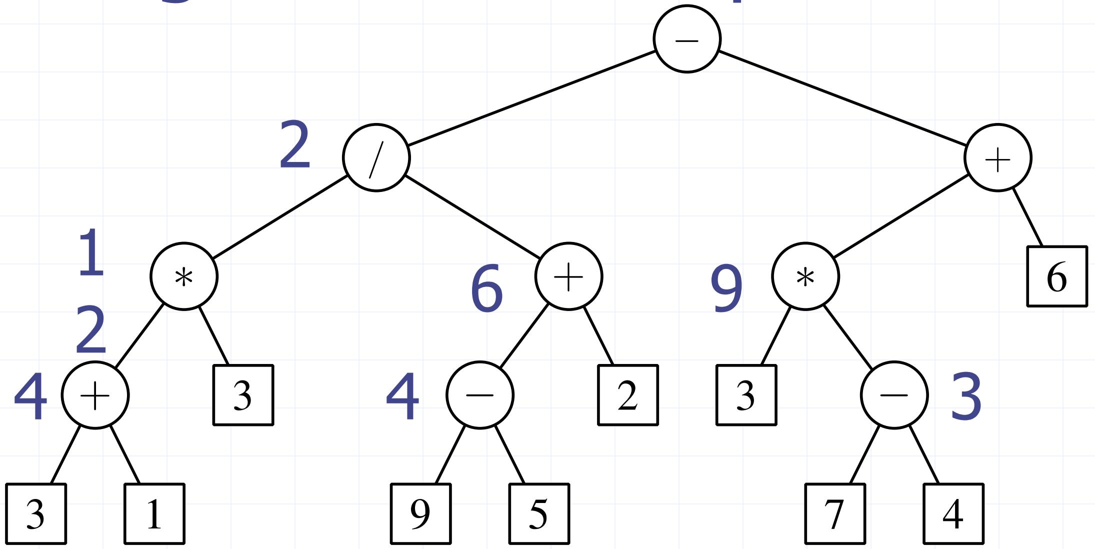
Application of Postorder Traversal

- Evaluating Arithmetic Expressions



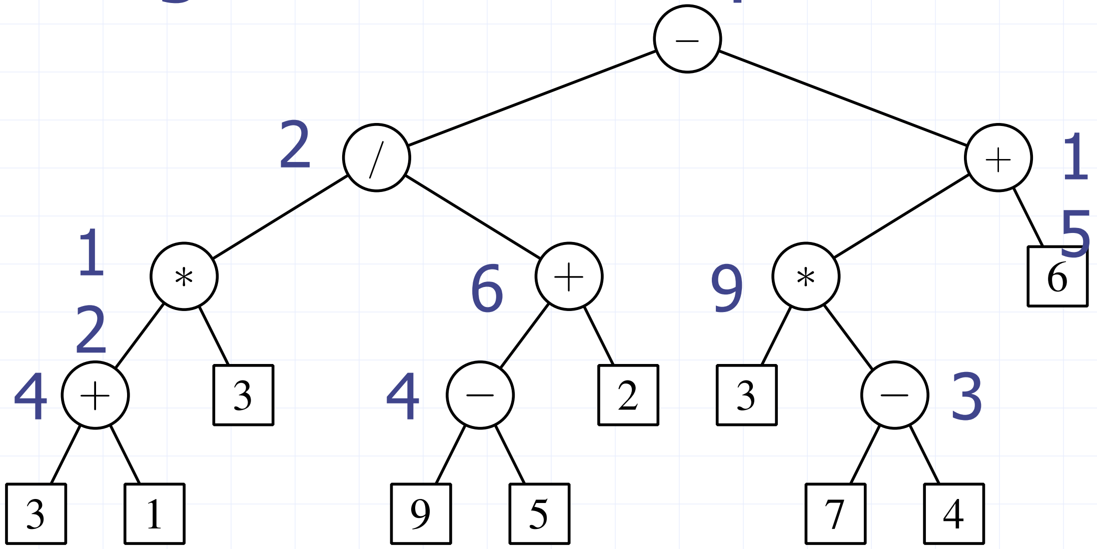
Application of Postorder Traversal

- Evaluating Arithmetic Expressions



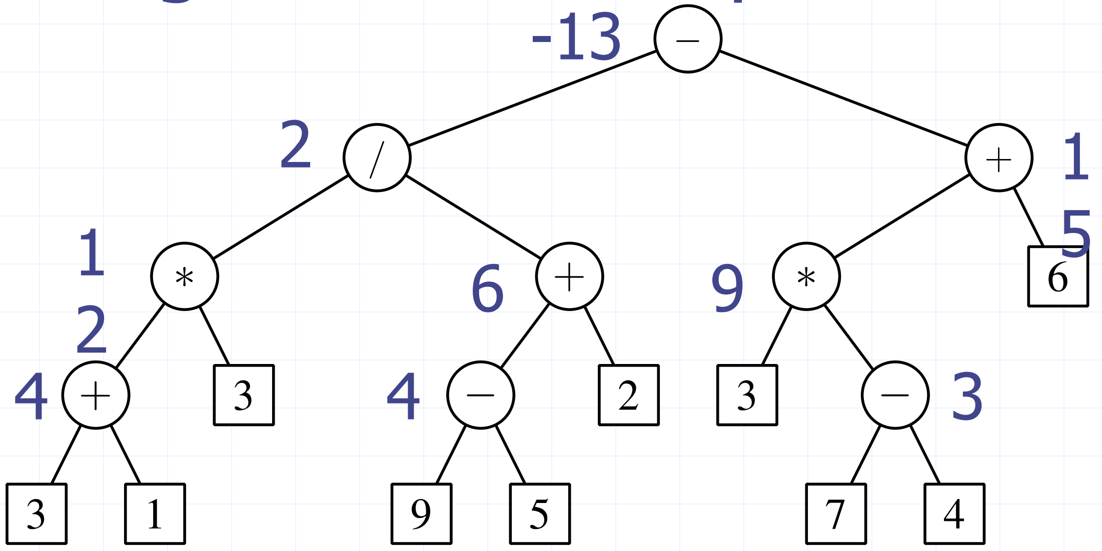
Application of Postorder Traversal

- Evaluating Arithmetic Expressions



Application of Postorder Traversal

- Evaluating Arithmetic Expressions

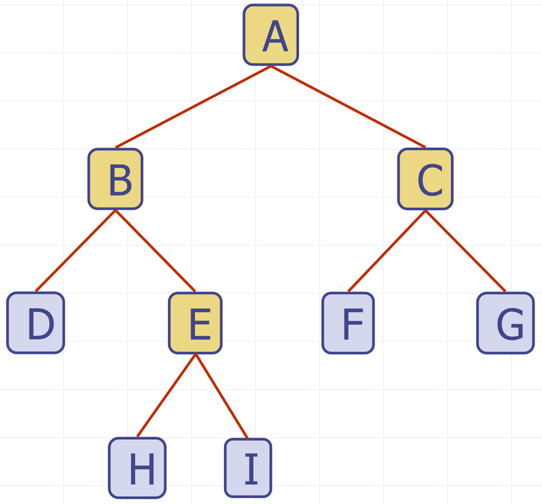


Inorder traversals

- Visit the node between the visit to the left and right subtree
- Algorithm inorder(p)
  - If p has a left child lc then
    - inorder(lc)
  - perform “visit” action for position p
  - If p has a right child rc then
    - inorder(rc)

Example - Inorder Traversal

- Inorder
  - d b h e i a f c g



Given Preorder and Inorder traversal, reconstruct the tree

Preorder

a b d e h i c f g

Inorder

$\underbrace{d\ b\ h\ e\ i}\ \boxed{a}\ \underbrace{f\ c\ g}$

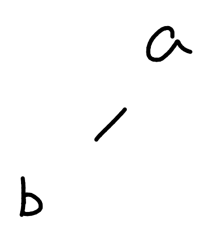
Step 1: look at preorder, it starts with a $\Rightarrow$ root is a

a

$\Downarrow$

$\underbrace{d\ b\ h\ e\ i}_{\text{left subtree}}\ \boxed{a}\ \underbrace{f\ c\ g}_{\text{right subtree}}$

Step 2: root of left subtree is b



$\underset{\text{left}}{d}\ \boxed{b}\ \underbrace{h\ e\ i}_{\text{right}}$

Preorder

a b d e h i c f g

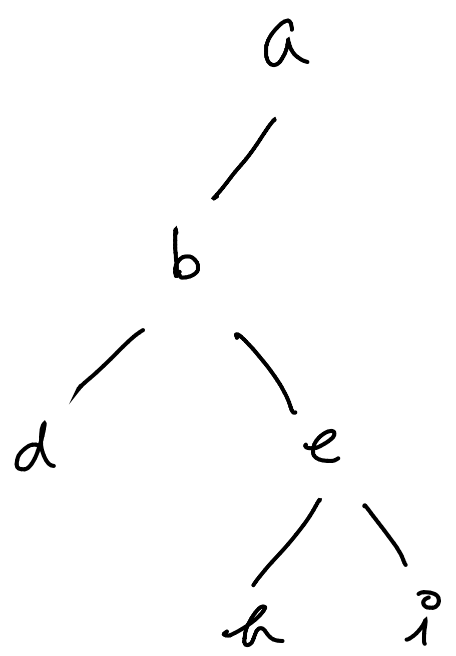
Inorder

$\underbrace{d\ b\ h\ e\ i}$ $\boxed{a}$ f c g

Step 3 : look at pre order e occurs before h and i

⇒ e is the root of the right sub-tree of b

h $\boxed{e}$ i



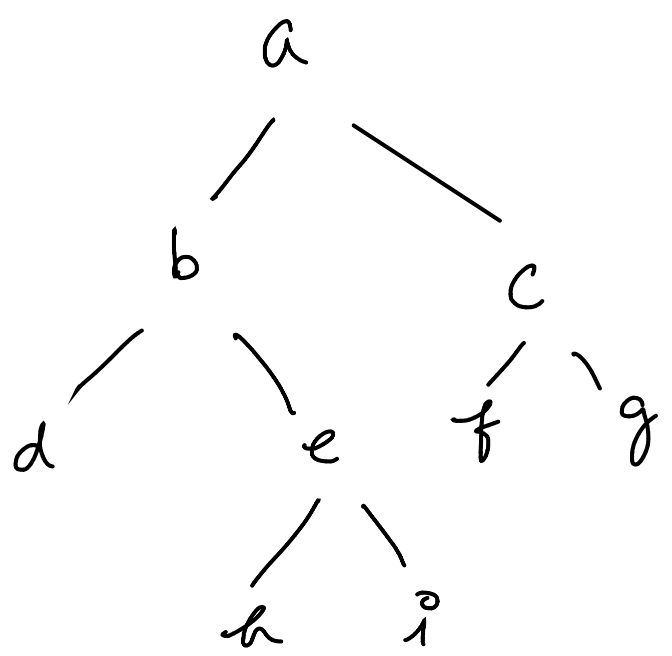
Preorder

a b d e h i c f g

Inorder

$\underbrace{d\ b\ h\ e\ i}\ \boxed{a}\ \underbrace{f\ c\ g}$

Step 4: look at preorder, c occurs before f and g



$f\ \boxed{c}\ g$

Preorder and Post order : Can you reconstruct ? (Not always) only when tree is proper

Preorder

a b e i c

Post order

i e b c a

step 1: look at either preorder or post order ⇒ a is root

a

step 2: b e i c can belong to L alone, ~~R alone~~, ~~both~~

(look at preorder) b e i c cannot be R alone

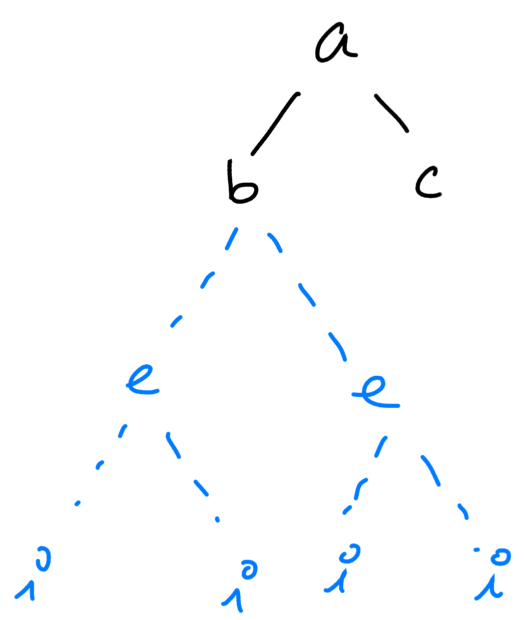
(look at pre + post order) if b e i c belongs to R alone ⇒ from preorder b should be a's right child

⇒ b should be second from last in post order ✗

b e i c belongs L alone ⇒ from post order c should be a's left child

⇒ c should be second from beginning in preorder ✗

$\Downarrow$



Exercise :

Preorder

a b d e h i c f g

Post order

d h i e b f g c a