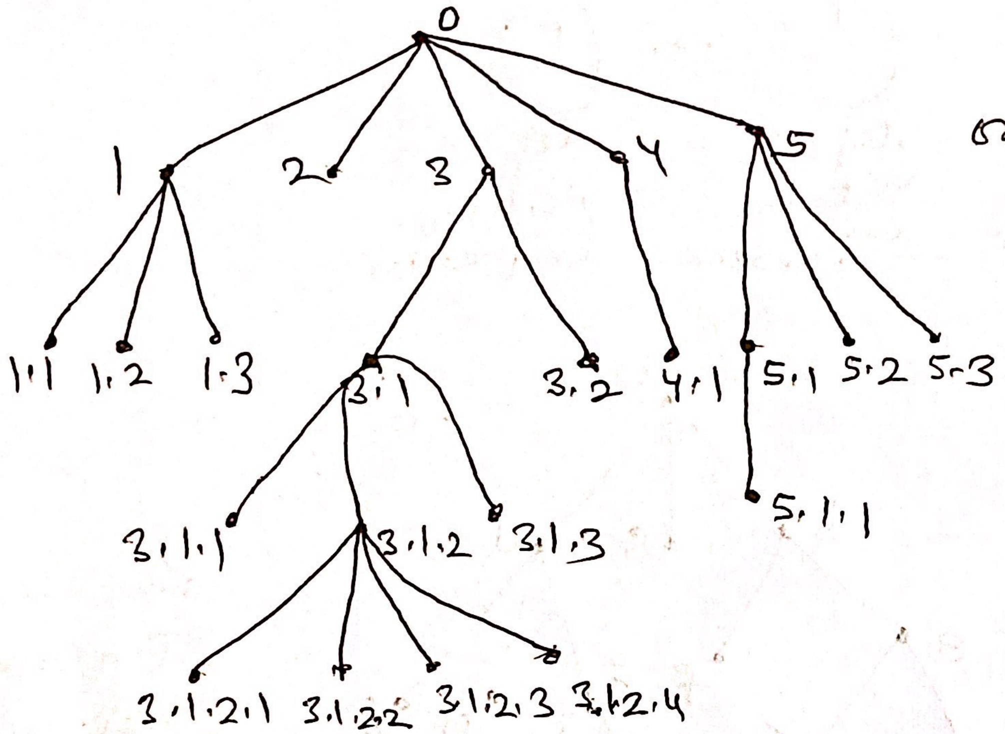
TREE TRAVERSAL

- Ordered rooted trees are often used to store info.
→ We need procedures for visiting each vertex of an ordered rooted tree to access data.

Universal Address systems



ordered rooted tree

$0 < 1 < 1.1 < 1.2 < 1.3 < 2 < 3 < 3.1 < 3.1.1 < 3.1.2$
$< 3.1.2.1 < 3.1.2.2 < 3.1.2.3 < 3.12.2.4$
$< 3.1.3 < 3.2 < 4 < 4.1 < 5 < 5.1 < 5.1.1 < 5.2$
< 5.3

Ordering of the labelings

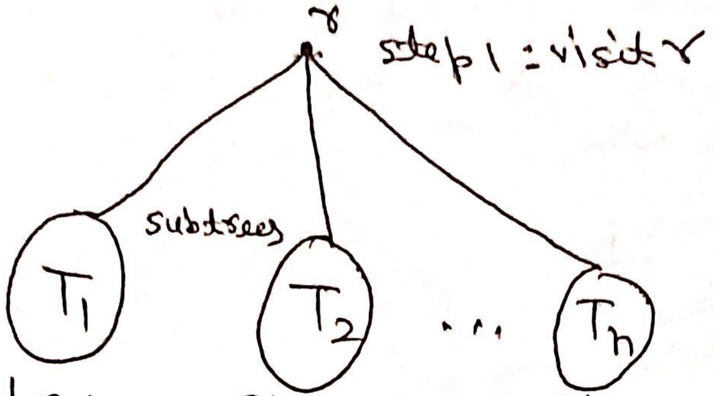
TRAVERSAL ALGORITHMS

preorder traversal

~~inodor~~ inorder traversal

postorder traversal

PREORDER TRAVERSAL



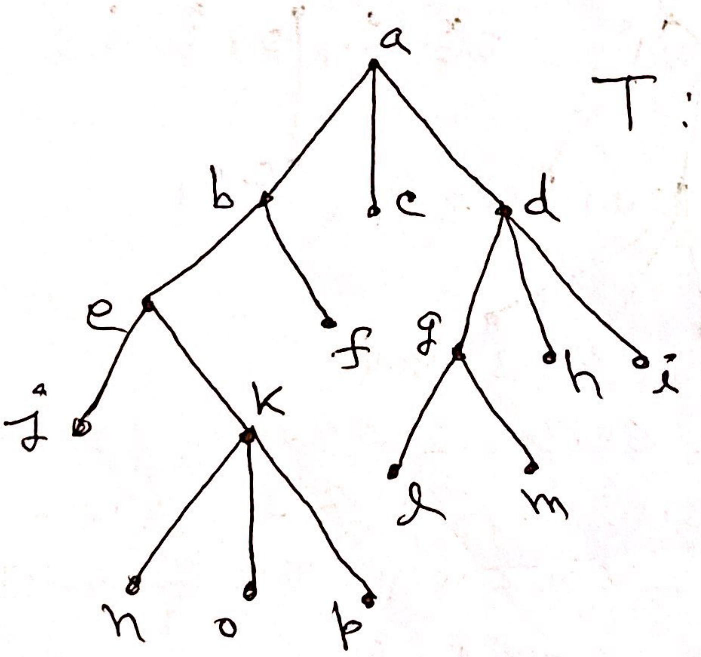
step 1 : visit r

Step 2: visit T_1 in preorder

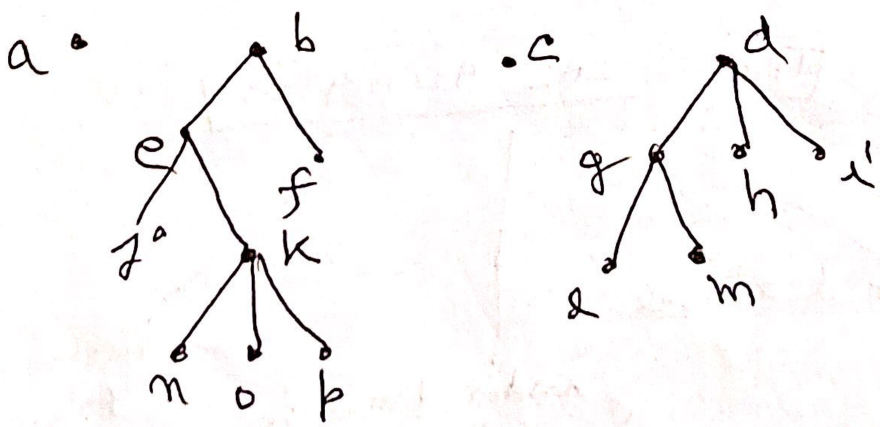
Step 3: visit T_2 in preorder

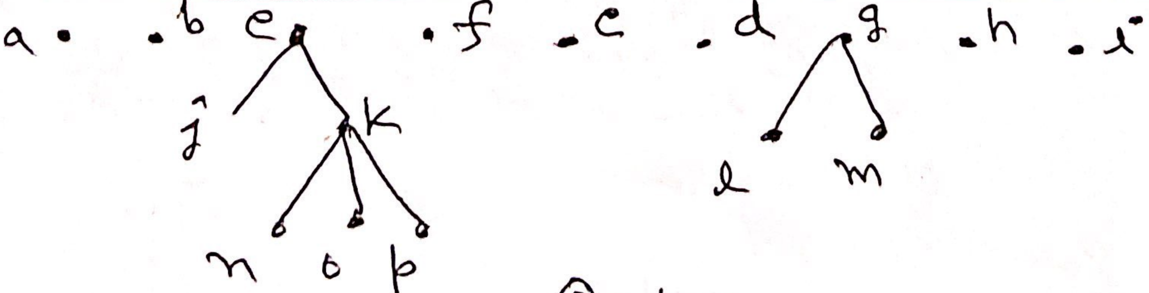
Step n+1: visit T_n in preorder

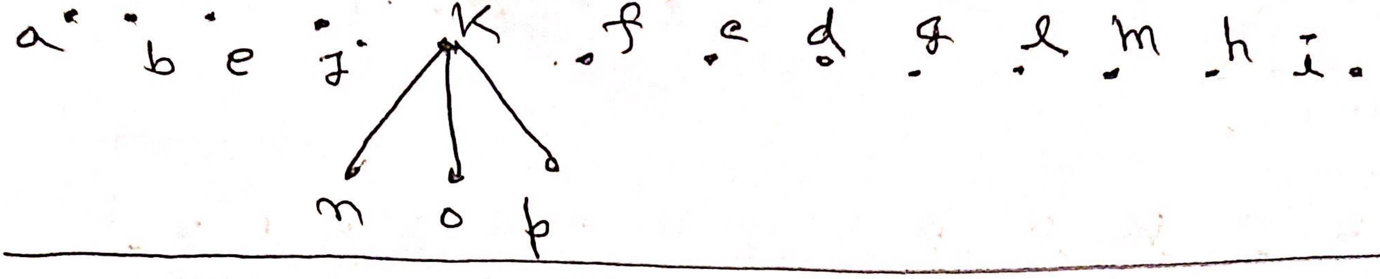
Example



Perorder Traversal: visit root visit subtrees left to right

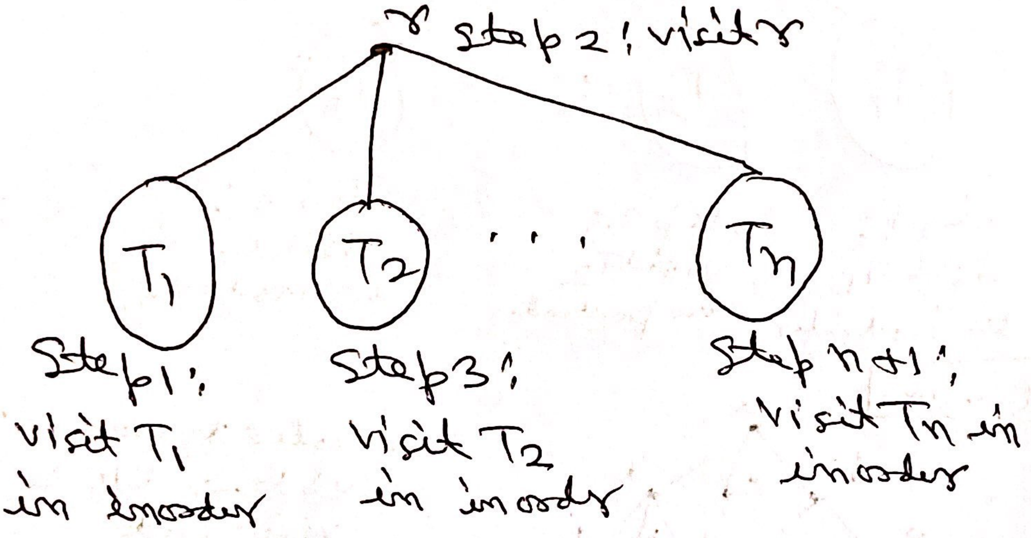




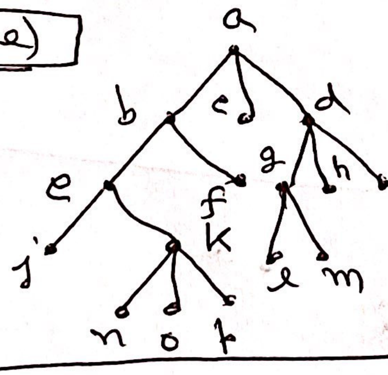


a, b e j k n o p f c d g l m h i

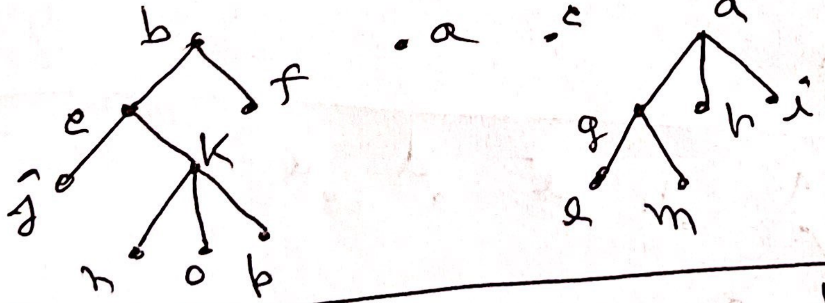
INORDER TRAVERSAL

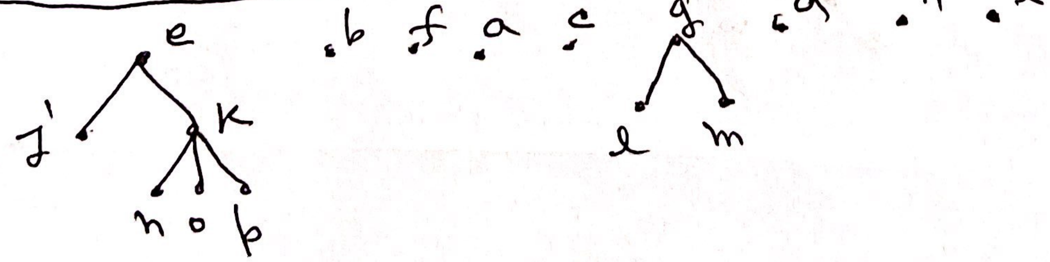

r step 2: visit r
T1
T2
...
Tn
Step 1: visit T1 in inorder
Step 3: visit T2 in inorder
Step n+1: visit Tn in inorder

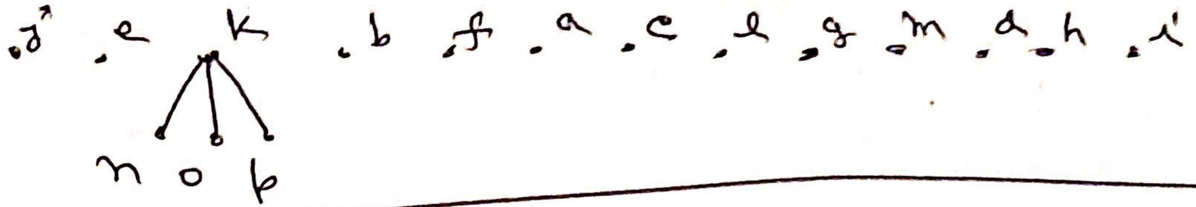
Example (same)



Inorder traversal:
visit leftmost
subtree, visit root,
visit other subtrees
left to right

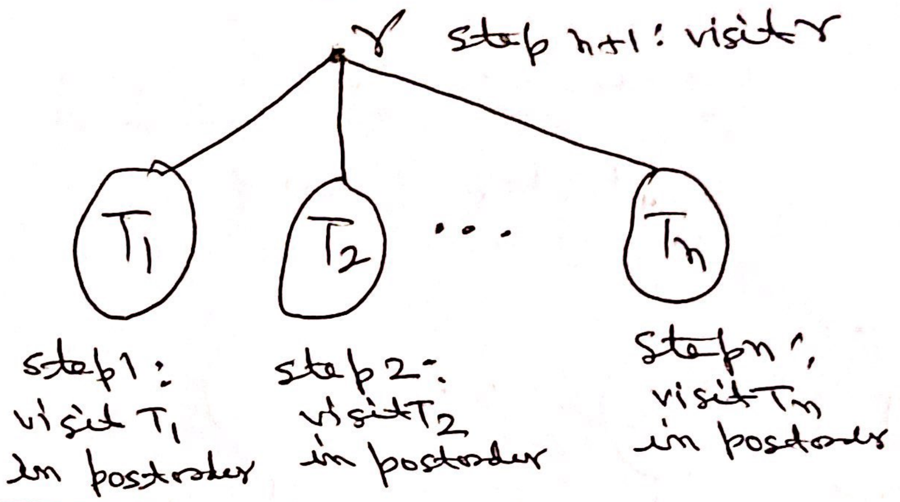




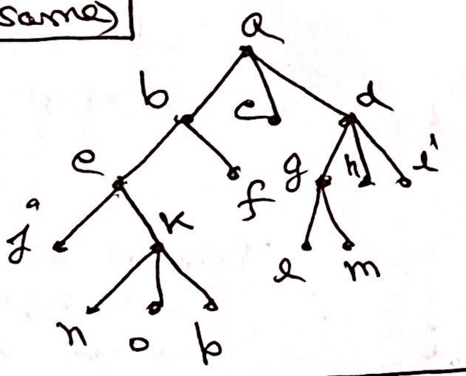


j e n k o p b f a c l g m d h i

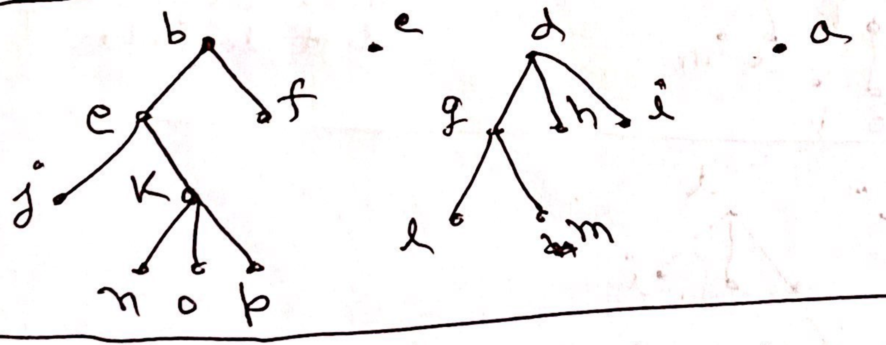
POSTORDER TRAVERSAL

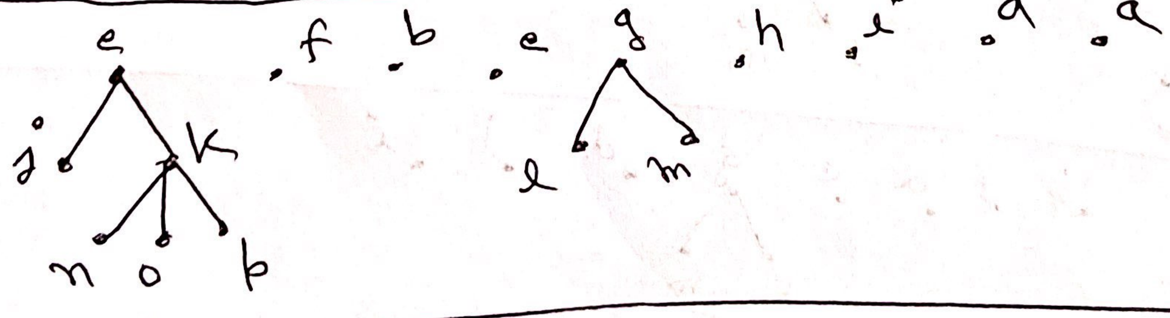

r
step n+1: visit r
T1
T2
...
Tn
step 1: visit T1 in postorder
step 2: visit T2 in postorder
step n: visit Tn in postorder

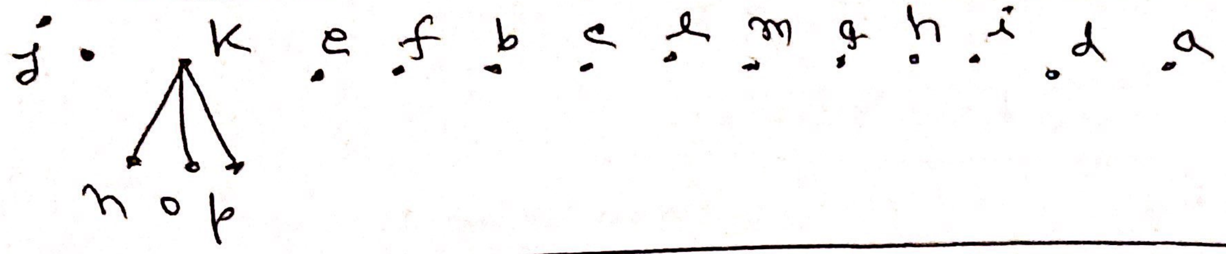
Example (same)



Postorder traversal: visit subtrees left to right; visit root

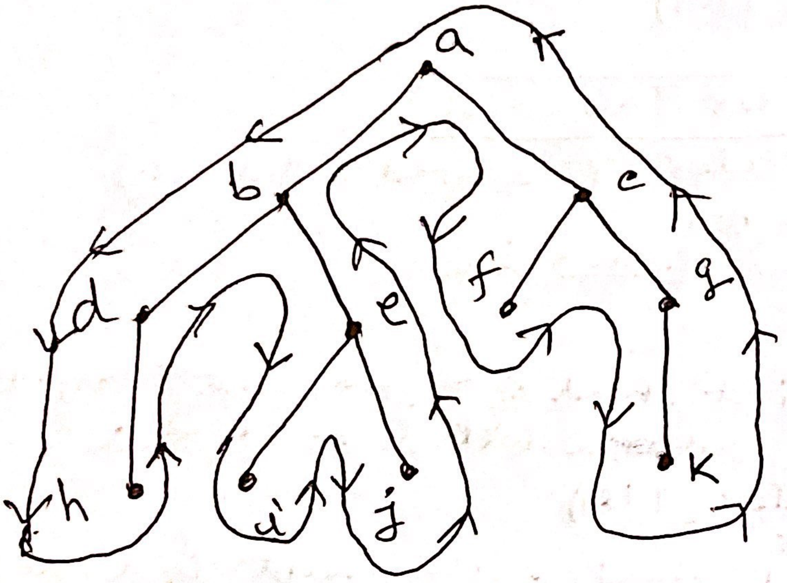






j n o p k e f b c l m g h i d a

Easy Ways



— First draw a curve (starting at root & moving around the edges)

— Preorder: We can list the vertices in preorder by listing each vertex the first time this curve passes it.

a, b, d, h, e, i, j, c, f, g, k

— Inorder by listing a leaf the first time the curve passes it and listing each internal vertex the second time the curve passes it

h, d, b, i, e, j, a, f, c, k, g

— Postorder by listing a vertex the last time it is passed on the way back up to its parent.

h, d, i, j, e, b, f, k, g, c, a

Algorithms

Algo 1: Preorder Traversal

```
procedure preorder (T: ordered rooted tree)
  r := root of T
  list r
  for each child c of r from left to right
  begin
    T(c) := subtree with c as its root
    preorder (T(c))
  end
```

ALGO 2: Inorder Traversel

```
procedure inorder (T: ordered rooted tree)
  r := root of T
  if r is a leaf then list r
  else
  begin
    l := first child of r from left to right
    T(l) := subtree with l as its root
    inorder (T(l))
    list r
    for each child c of r except for l from left to right
      T(c) := subtree with c as its root
      inorder (T(c))
  end
```

ALGO 3: Postorder Traversal

```
procedure postorder (T: ordered rooted tree)
  r := root of T
  for each child c of r from left to right begin
  T(c) := subtree with c as its root
    postorder (T(c))
  end
  list r
```