Binary Tree Traversal Methods

• In a traversal of a binary tree, each element of the binary tree is visited exactly once.
• During the visit of an element, all action (make a clone, display, evaluate the operator, etc.) with respect to this element is taken.

Preorder Traversal

```
public static void preOrder(BinaryTreeNode t)
{
    if (t != null)
    {
        visit(t);
        preOrder(t.leftChild);
        preOrder(t.rightChild);
    }
}
```

Preorder Example (visit = print)

```
a b c
da e f
g h i
```

Gives prefix form of expression!
Inorder Traversal

```java
public static void inOrder(BinaryTreeNode t) {
    if (t != null) {
        inOrder(t.leftChild);
        visit(t);
        inOrder(t.rightChild);
    }
}
```

Inorder Example (visit = print)

```
a
b

```

Inorder Example (visit = print)

```
a
b
c
d
e
f
g
h
i
j

```

Inorder By Projection (Squishing)

```
g
d
h
b
e
i
a
f
j

```

Inorder Of Expression Tree

```
a + b + c - d / e + f

```

Postorder Traversal

```java
public static void postOrder(BinaryTreeNode t) {
    if (t != null) {
        postOrder(t.leftChild);
        postOrder(t.rightChild);
        visit(t);
    }
}
```
Postorder Example (visit = print)

```
 b c a
```

Postorder Of Expression Tree

```
a b + c d - * e f + /
```

Gives postfix form of expression!

Level Order

Let t be the tree root.
while (t != null)
{
    visit t and put its children on a FIFO queue;
    remove a node from the FIFO queue and call it t;
    // remove returns null when queue is empty
}

Level-Order Example (visit = print)

```
a b c d e f g h i j
```
Binary Tree Construction

- Suppose that the elements in a binary tree are distinct.
- Can you construct the binary tree from which a given traversal sequence came?
- When a traversal sequence has more than one element, the binary tree is not uniquely defined.
- Therefore, the tree from which the sequence was obtained cannot be reconstructed uniquely.

Some Examples

<table>
<thead>
<tr>
<th>Traversal Sequence</th>
<th>Example Tree</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>preorder</code> = <code>ab</code></td>
<td><img src="tree1.png" alt="Tree Example" /></td>
</tr>
<tr>
<td><code>inorder</code> = <code>ab</code></td>
<td><img src="tree2.png" alt="Tree Example" /></td>
</tr>
<tr>
<td><code>postorder</code> = <code>ab</code></td>
<td><img src="tree3.png" alt="Tree Example" /></td>
</tr>
<tr>
<td><code>level order</code> = <code>ab</code></td>
<td><img src="tree4.png" alt="Tree Example" /></td>
</tr>
</tbody>
</table>

Binary Tree Construction

- Can you construct the binary tree, given two traversal sequences?
- Depends on which two sequences are given.

Preorder And Postorder

- Preorder and postorder do not uniquely define a binary tree.
- Nor do preorder and level order (same example).
- Nor do postorder and level order (same example).

Inorder And Preorder

- `inorder` = `gdhbeiafjc`
- `preorder` = `abcdgheicfj`
- Scan the preorder left to right using the inorder to separate left and right subtrees.
- `a` is the root of the tree; `gdhbei` are in the left subtree; `fjc` are in the right subtree.

Inorder And Preorder

- `preorder` = `abcdgheicfj`
- `b` is the next root; `gdh` are in the left subtree; `ei` are in the right subtree.
**Inorder And Preorder**

- **preorder** = a b d g h e i c f j
- d is the next root; g is in the left subtree; h is in the right subtree.

**Inorder And Postorder**

- Scan postorder from right to left using inorder to separate left and right subtrees.
- **inorder** = g d h b e i a f j c
- **postorder** = g h d i e b j f c a
- Tree root is a; gdhbei are in left subtree; fjc are in right subtree.

**Inorder And Level Order**

- Scan level order from left to right using inorder to separate left and right subtrees.
- **inorder** = g d h b e i a f j c
- **level order** = a b c d e f g h i j
- Tree root is a; gdhbei are in left subtree; fjc are in right subtree.