

# **Tree traversals**

(Weiss 4.6)

# Tree traversal

*Traversing* a tree means visiting all its nodes in some order

This can be implemented by recursion. For example, printing the value of all nodes in the tree:

```
class Node<E> {
    E value;
    Node<E> left, right;
}

void printAll(Node<E> node) {
    if (node == null) return; // base case
    System.out.println(node.value);
    printAll(node.left);
    printAll(node.right);
}
```

This kind of traversal is called a *preorder traversal* because we visit the node before we recurse into its children.

# Inorder traversal

In an *inorder traversal* we first visit the node's left child, then the node itself, then the node's right child.

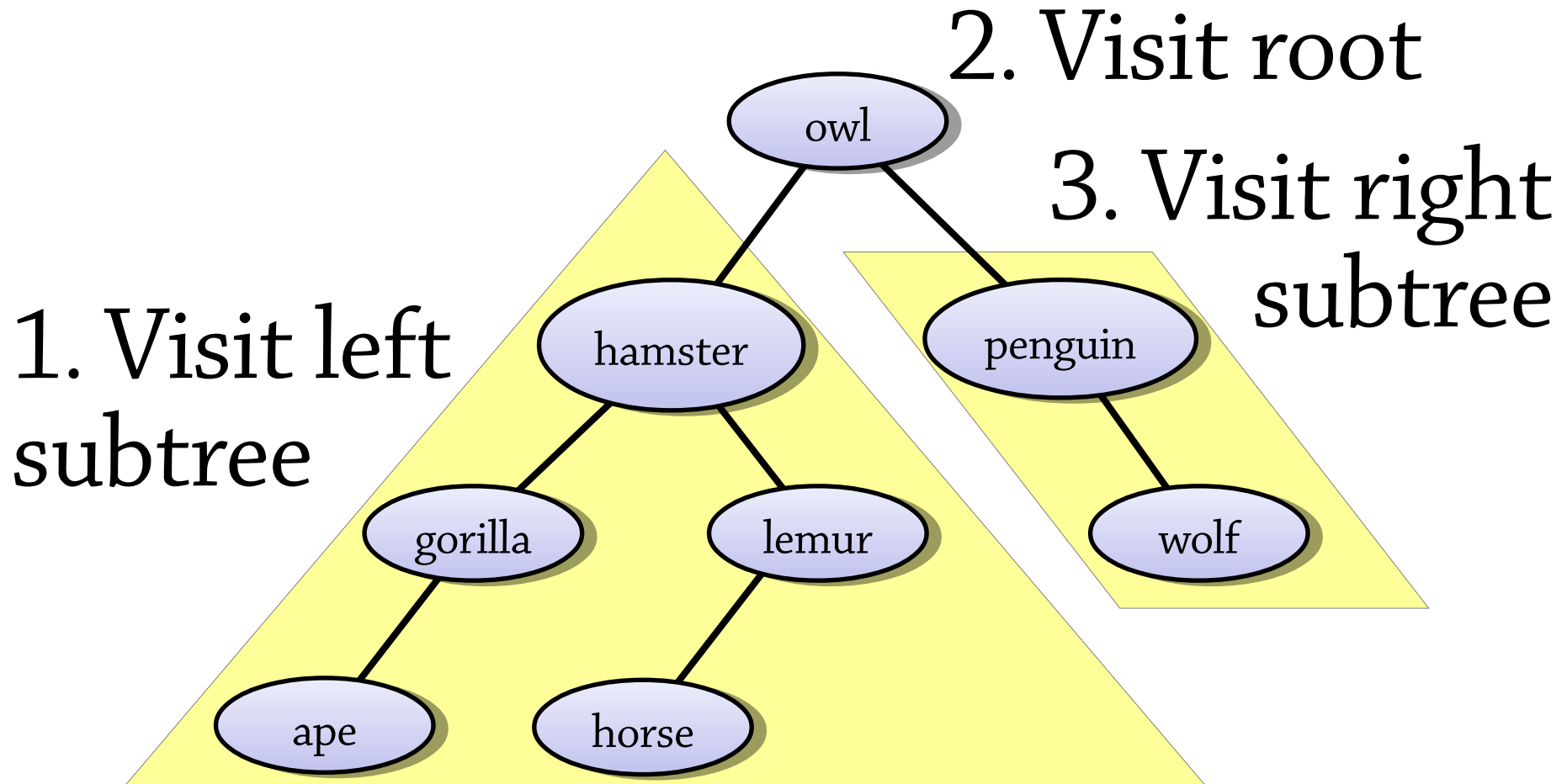
Example: printing out the contents of a binary search tree in sorted order!

```
class Node<E> {
    E value;
    Node<E> left, right;
}

void printSorted(Node<E> node) {
    if (node == null) return;
    printSorted(node.left);
    System.out.println(node.value);
    printSorted(node.right);
}
```

# Sorting a binary search tree

If we do an inorder traversal of a BST, we visit its elements in sorted order!



# Postorder traversal

In a *postorder traversal* we visit each node's children before visiting the node itself.

Example: calculating the height of a binary tree:

```
int height(Node<E> node) {  
    if (node == null) return 0;  
    else {  
        int leftHeight = height(node.left);  
        int rightHeight = height(node.right);  
        return 1 + Math.max(leftHeight, rightHeight);  
    }  
}
```

# Summary – traversals

Tree traversals capture different patterns of recursion for visiting all nodes in a tree

- Preorder – visit root, then recursively traverse children
- Postorder – recursively traverse children, then visit root
- Inorder – recursively traverse one child, then visit root, then recursively traverse other child (makes sense for e.g. 2-3 trees too)

Nothing sophisticated needed – a simple recursive function does the job

Less common: level-order traversal – visit nodes in order of level in the tree