# 数据结构

## 一．树

### 1.树的深度遍历(DFS)

树的深度遍历(DFS)

DFS一般可以用递归来实现，非递归实现一般用stack;

DFS递归实现：

这是前序遍历

public void preOrderTraverse1(TreeNode root) {

        if (root != null) {

            System.out.print(root.val+"  ");

            preOrderTraverse1(root.left);

            preOrderTraverse1(root.right);

        }

}

DFS非递归实现

**public** **void** preOrderNR() {

Stack<Node> stack = **new** Stack<Node>();

stack.push(root);

**while**(!stack.isEmpty()) {

Node pop = stack.pop();

System.***out***.println(pop);

**if**(pop.right != **null**) {

stack.push(pop.right);

}

**if**(pop.left != **null**) {

stack.push(pop.left);

}

}

}

### 2.层次遍历： (利用队列)

public void levelTraverse(TreeNode root) {

        if (root == null) {

            return;

        }

        LinkedList<TreeNode> queue = new LinkedList<>();

        queue.offer(root);

        while (!queue.isEmpty()) {

            TreeNode node = queue.poll();

            System.out.print(node.val+"  ");

            if (node.left != null) {

                queue.offer(node.left);

            }

            if (node.right != null) {

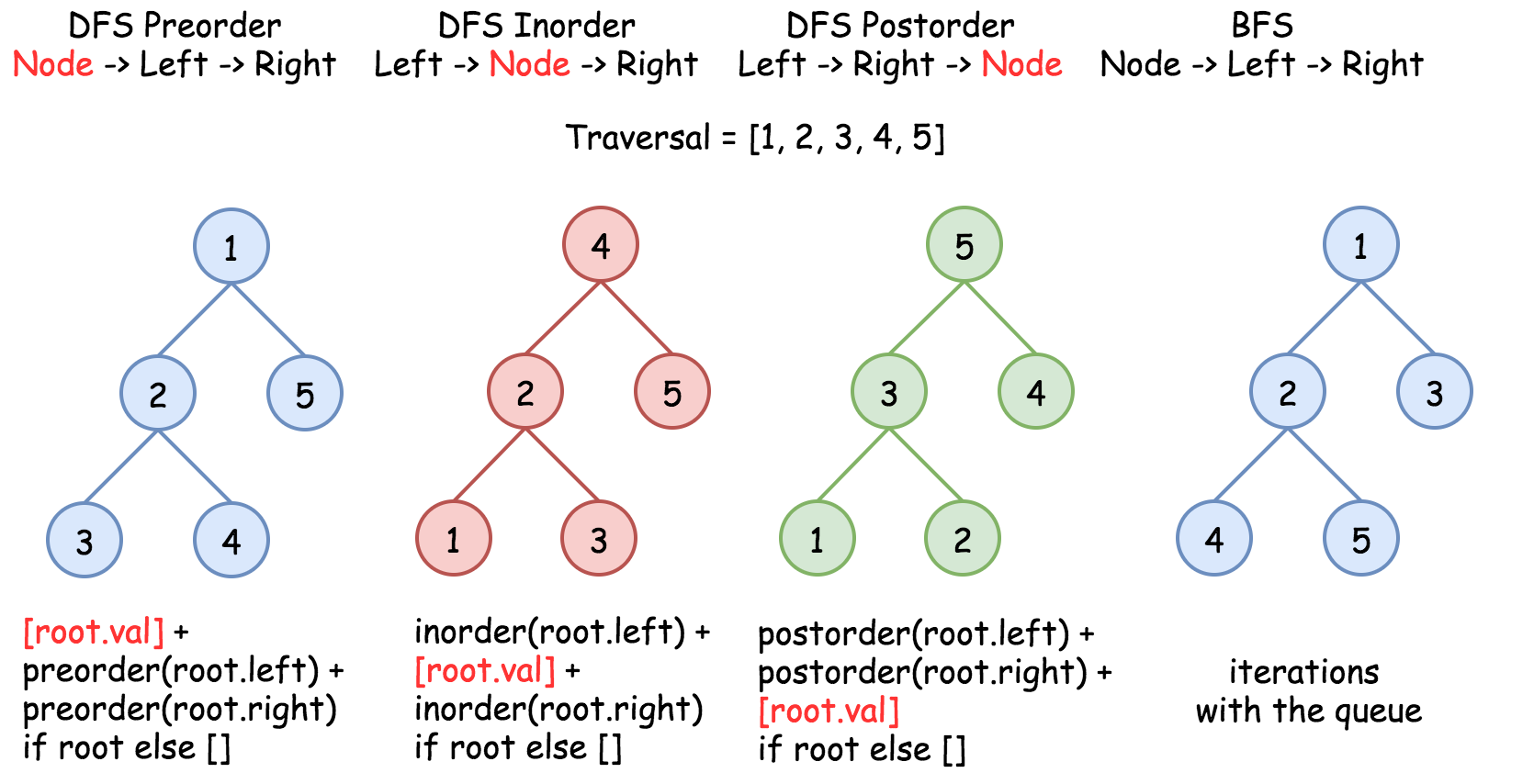
                queue.offer(node.right);

            }

        }

}

### 3下图为DFS前、中、后序遍历以及BFS



**Tips:DFS的中序遍历为升序的**