# 二叉树



前序遍历

根->左->右

1 2 4 5 7 8 3 6

func preOderTraverse(root \*TreeNode) {

if nil != root {

fmt.Println(root.Val, “ ”)

preOderTraverse(root.Left)

preOderTraverse(root.Right)

}

}

中序遍历

左->根->右

4 2 7 5 8 1 3 6

func midOderTraverse(root \*TreeNode) {

if nil != root {

preOderTraverse(root.Left)

fmt.Println(root.Val, “ ”)

preOderTraverse(root.Right)

}

}

后序遍历

左->右->根

4 7 8 5 2 6 3 1

func postOderTraverse(root \*TreeNode) {

if nil != root {

postOderTraverse(root.Left)

postOderTraverse(root.Right)

fmt.Println(root.Val, “ ”)

}

}

二叉树深度

func treeDepth(root \*TreeNode) int {

if nil == root {

return 0

}

return math.Max(treeDepth(root.Left), treeDepth(root.Right)) + 1

}

二叉树第K层的节点个数

# 二叉查找树

对于树中任意一个节点X，其左子树中任意节点的值都小于X的值，其右子树中任意节点的值都大于X。

# AVL树

# 完全二叉树

# 红黑树