**数据结构**

# 二叉树

## 二叉树的遍历方式

### 先序遍历

#### 递归遍历

#### 非递归遍历

# 平衡二叉树

## 为什么引入二叉平衡树

将{61,88,58,47,35,73,51,99,37,93}构建如图1的二叉排序树。但如果数组元素的次序是从小到大有序，如{35,37,47,51,58,62,73,88,93,99}，则二叉排序树就成了极端的右斜树，注意它依然是一颗二叉排序树，如图2,。此时，同样是查找结点99，左图只需要比较三次，而右图就需要11次比较才可以得到结果，二者差异很大。

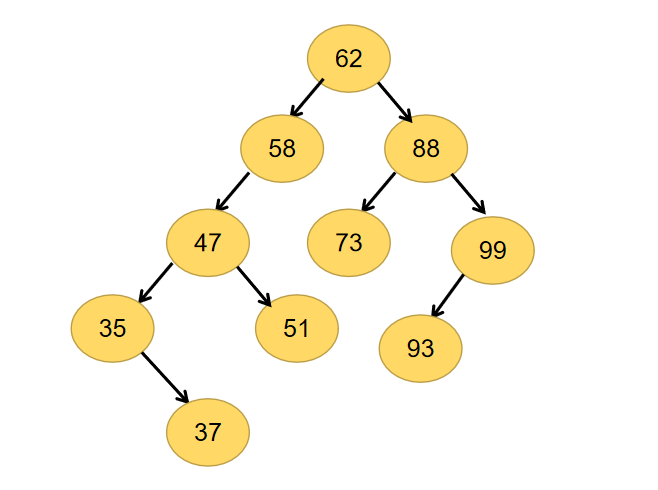


图1

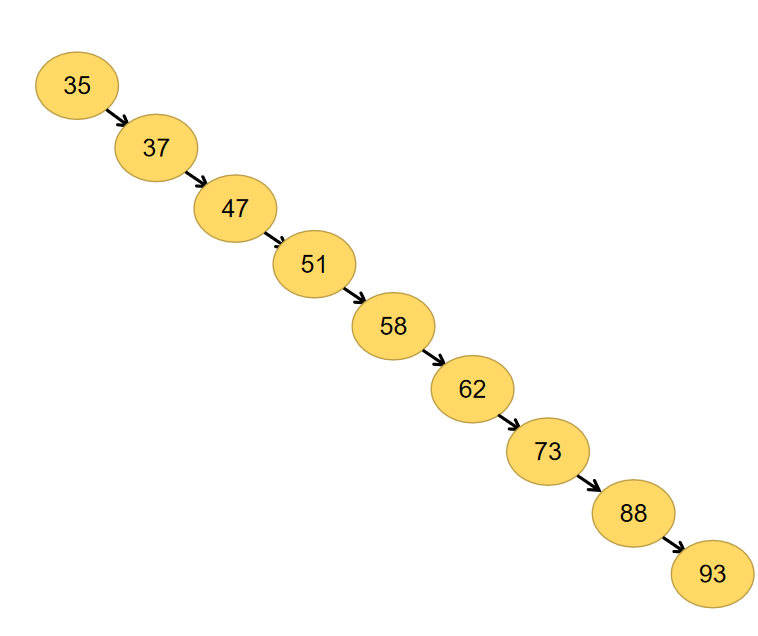


图2

我们希望二叉树是比较平衡的，即其深度与完全二叉树相同，均为，

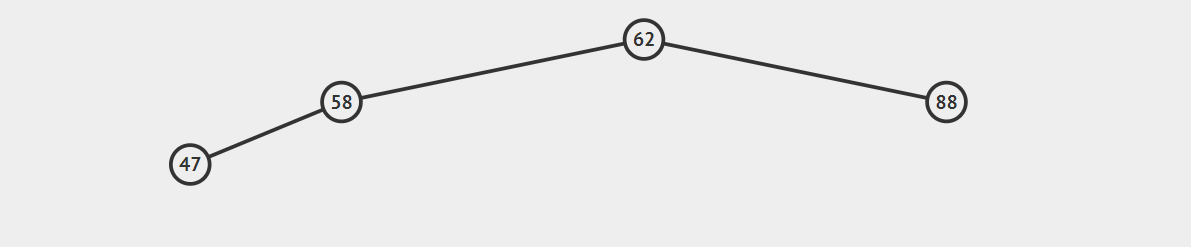
那么查找的时间复杂度也就O()，近似于折半查找。

不平衡的最坏情况就是像图2的斜树，查找的时间复杂度为O(n)，这等同于顺序查找。

因此，如果我们希望对一个集合按二叉排序树查找，最好是把它建成一棵平衡的二叉排序树。这样我们就引申出另一个问题，如何让二叉排序树平衡的问题。

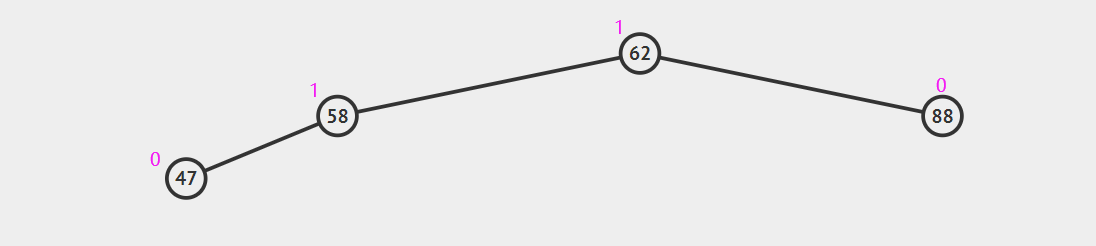
## 平衡二叉树的定义

平衡二叉树是一种二叉排序树，其中每一个节点的左子树和右子树的高度差至多等于1。



## 平衡因子

将二叉树上节点的左子树的高度减去右子树深度的值称之为平衡因子BF(Balance Factor),那么平衡二叉树上所有节点的平衡因子只可能是0，-1，1。只要二叉树上有一个节点的平衡因子的绝对值大于1，则该二叉树就是不平衡的。



## 最小不平衡子树

距离插入节点最近的，且平衡因子的绝对值大于1的节点为根的子树，我们称为最小不平衡子树。

## 旋转

## 插入