

实验三 二叉查找树、红黑树的基本操作实现

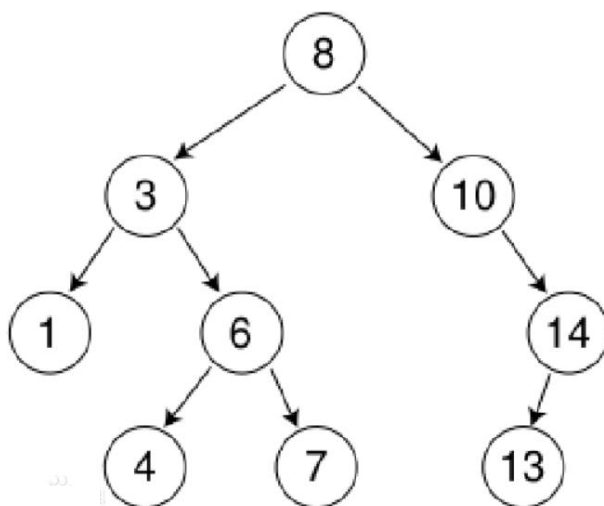
一、实验原理

1. 二叉查找树 (Binary Search Tree)

二叉查找树又称二叉搜索树，二叉排序树，特点如下：

1. 左子树上所有结点值均小于根结点
2. 右子树上所有结点值均大于根结点
3. 结点的左右子树本身又是一颗二叉查找树
4. 二叉查找树中序遍历得到结果是递增排序的结点序列。

典型操作：其插入、删除操作请见教材 P151-P155



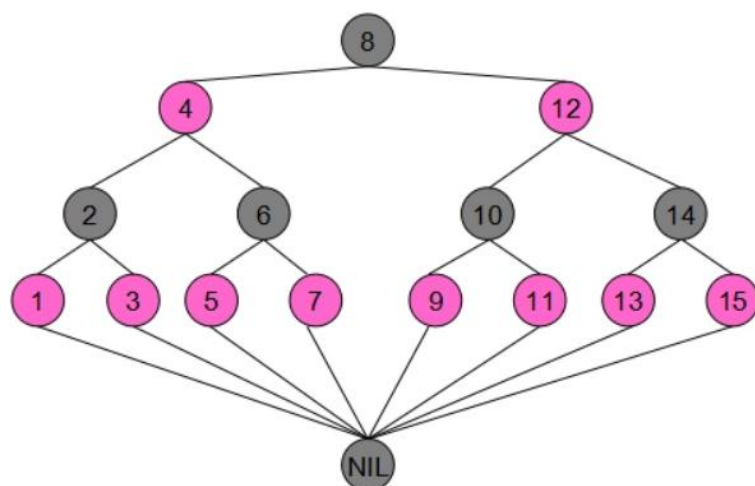
2. 红黑树

红黑树中每个结点包含五个域：color, key, left, right 和 p。如果某结点没有子结点或父结点，则该域指向 NIL。

一棵二叉树如果满足下面的红黑性质，则为一棵红黑树：

- 1) 每个结点或是红的，或是黑的。
- 2) 根结点是黑的。
- 3) 每个叶结点 (NIL) 是黑的。
- 4) 如果一个结点是红的，则它的两个儿子都是黑的。
- 5) 对每个结点，从该结点到其子孙结点的所有路径上包含相同数目的黑结

点。



从某个结点 x 出发 (不包括该结点) 到达一个叶结点的任意一条路径上, 黑色结点的个数称为该结点 x 的黑高度, 用 $bh(x)$ 表示。

引理: 一颗有 n 个内结点的红黑树的高度至多为 $2\lg(n+1)$ 。

典型操作: 其旋转、插入、删除、扩张操作请见教材 P165-P186

3. 二者性能比较

操作名(h树高)	二叉查找数	红黑树
查找	$O(h)$	$O(\lg n)$
查最大/小元素	$O(h)$	$O(\lg n)$
前驱/后继	$O(h)$	$O(\lg n)$
插入	$O(h)$	$O(\lg n)$
删除	$O(h)$	$O(\lg n)$
旋转	无	$O(1)$
高度	下取整 $(\lg n)+1 \leq h \leq n$	$\leq 2\lg(n+1)$

二、实验要求

1. 实现下列关于二叉查找树、红黑树的判断、构建、删除等操作。并写出这些操作的流程图或伪代码。
2. 请说明二叉查找树和红黑树的区别以及时间、空间性能。

Pro1: Given a binary tree, determine if it is a valid binary search tree (BST).

Assume a BST is defined as follows:

- ① The left subtree of a node contains only nodes with keys less than the node's key.
- ② The right subtree of a node contains only nodes with keys greater than the node's key.
- ③ Both the left and right subtrees must also be binary search trees.

e.g.

```
1  输入:
2      5
3     /\
4    1  4
5     /\
6    3  6
7  输出: false
8  解释: 输入为: [5,1,4,null,null,3,6]。
```

Pro2: Construct a red black tree by {1, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15}, show the result. Then delete the points in order {14,9,5}, show the result too.

e.g.

```
依次插入值为41,38,31,12,19,8的节点后, 中序遍历红黑树各节点:
red      8
black    12
red      19
black    31
black    38
black    41
```

注意: 当节课实验结束前找老师检查, 不需提交实验报告