**题目描述: 给定一颗二叉搜索树，请找出其中的第k大的结点。例如， 5 / \ 3 7 /\ /\ 2 4 6 8 中，按结点数值大小顺序第三个结点的值为4.**

**代码(mine):**

class Solution {

public:

    void DLR(TreeNode \*head){

        if (head != NULL){

            DLR(head->left);

            num.push\_back(head);

            DLR(head->right);

        }

    }

    TreeNode\* KthNode(TreeNode\* pRoot, unsigned int k){

        DLR(pRoot);

        if (num.size() >= k&&k>0)return num[k - 1];

        else return NULL;

    }

    vector<TreeNode \*>num;

};

**示例：**

class Solution {

public:

    int count=0;

    TreeNode\* KthNode(TreeNode\* pRoot, unsigned int k){

       if(pRoot){

           TreeNode\* ret=KthNode(pRoot->left,k);

           if(ret) return ret; //如果是空的话就不返回

           if(++count==k) return pRoot;

           ret=KthNode(pRoot->right,k);

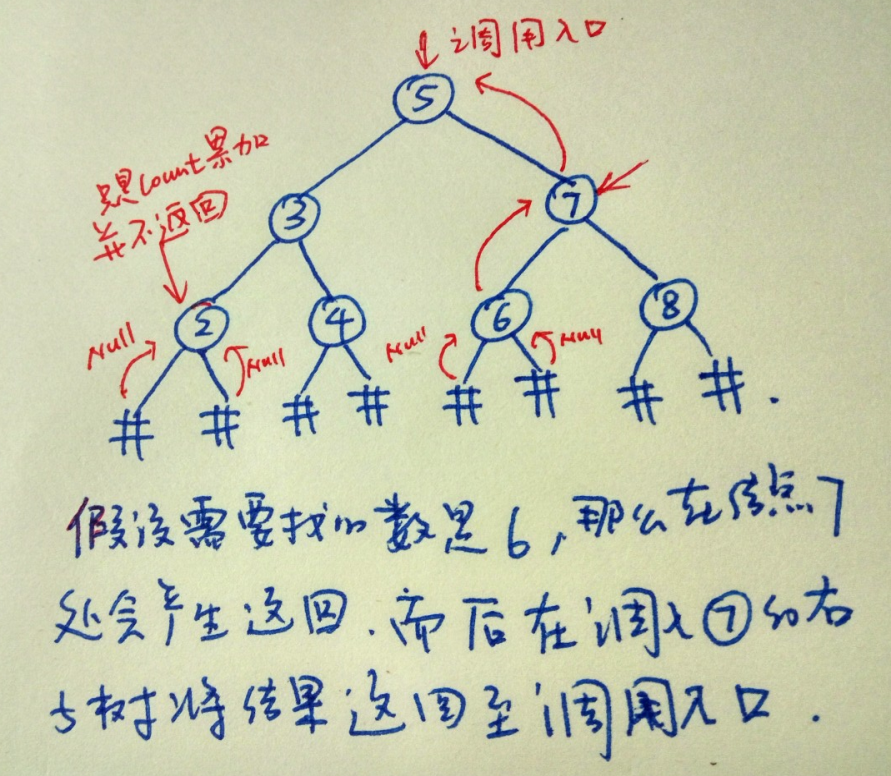
           if(ret) return ret;

       }

        return nullptr;

    }

};



**分析: 示例代码先访问根结点的左子树,从左子树中寻找是否满足条件的结点,若左子树没有满足条件的结点,然后再从右子树中寻找。在没有找到满足条件的结点的时候函数一直返回NULL,一旦找到满足条件的结点就会将结果一直返回直至函数调用总入口。**