## 题目描述：

## 输入一棵二叉搜索树，将该二叉搜索树转换成一个排序的双向链表。要求不能创建任何新的结点，只能调整树中结点指针的指向。

## 解题思路：

1.将左子树构造成双链表，并返回链表头节点。

2.定位至左子树双链表最后一个节点。

3.如果左子树链表不为空的话，将当前root追加到左子树链表。

4.将右子树构造成双链表，并返回链表头节点。

5.如果右子树链表不为空的话，将该链表追加到root节点之后。

6.根据左子树链表是否为空确定返回的节点。

## 算法描述：

/\*

struct TreeNode {

int val;

struct TreeNode \*left;

struct TreeNode \*right;

TreeNode(int x) :

val(x), left(NULL), right(NULL) {

}

};\*/

class Solution {

public:

TreeNode\* Convert(TreeNode\* root)

{

if (root == NULL)

return NULL;

if (root->left == NULL && root->right ==NULL)

return root;

TreeNode\* left = Convert(root->left); //将左子树链表化，并且返回头结点。

TreeNode\* p =left;

while( p != NULL && p->right !=NULL)

{

p = p->right; //最终p指向左子树链表化以后的最右节点

}

if (left!=NULL)

{

p->right = root; //将左子树链表化以后与根节点连接

root->left = p;

}

TreeNode \*right = Convert(root->right); //右子树链表化

if (right!=NULL)

{

right->left = root; //与根节点连接

root->right = right;

}

return left!=NULL?left:root;

}

};