Homework 12.14 12.16

PB19010450 和泳毅 36号

9.38 试写一算法,将两棵二叉排序树合并为一棵二叉排序树

```
void Insert(BiTree &T,KeyType key){
    if(!T)){ //如果为空树，直接操作
        T = (BinTree)malloc(sizeof(BinNode));
        T->data = key;
        T->lchild = T->rchild = NULL;
        return;
    }
    if(key == T->data)  return;
    if(key > T->data)   Insert(T->rchild,key);
    else    Insert(T->lchild,key);
}

void MergeBST(BiTree T1,BiTree T2){
    if(T2){
        InsertBST(T1,T2->lchild);
        Insert(T1,T2->data); //递归
        InsertBST(T1,T2->rchild);
    }
}
```

9.40 在平衡二叉排序树的每个结点中增设一个lsize域,其值为它的左子树中的结点数加1。试写一时间复杂度为O(logn)的算法,确定树中第k小的结点的位置。

```
BSNode Locate(BSNode T,int k){
    //二叉排序树中第k小的结点，即为二叉排序树中序序列中顺序号为k的结点;
    //将k与根结点的顺序号进行比较。
    p = T;
    while(p){
        j = i + p->lsize;
        if(j == k) return p;//找到
        if(j > k) p = p->lchild;
        else{
            i += p->lsize;
            p = p->rchild
        }//右孩子结点号为根结点号与右孩子结点lsize域值之和
    }
    return NULL:
}
```

查找不超过树的高度，$T(n) = O(logn)$。

9.41 为B+树设计结点类型并写出算法,随机(而不是顺序)地查找给定的关键字K,求得它所在的叶子结点指针和它在该结点中的位置(提示:B+树中有两种类型的指针，结点结构也不尽相同,考虑利用记录的变体。)

```
Result SearchBTree(BTree T, KeyType K){
    Result R = {NULL, 0, 0};
    p = T.root;//root指向B+树的根
    found = FALSE;
    i = 0;                          //p->key[i]<=K<p->key[i+1]
    while(p && !found){
        i = Search(p,K);            //在p->key[1...keynum]中查找
        if(p->leaf == 1){
            if(i > 0 && p->key[i - 1] == K){
                found = TRUE;
                i--;
            }
            else    break;
        }
        else{
            if(i > 0)   i--;
            p = p->ptr[i];
        }
    }
    R.i = i;
    R.pt = p;
    if(found)   R.tag = 1;//1成功，0失败
    else    R.tag = 0;
    return R;
}
int Search(BTNode &p, KeyType K){
    for(i = 0; i < p->keynum && p->key[i] <= K; i++);
    return i;
}
```