DSHW4: 0016306 林天心

做法:

定義 node 除了基本的的 data、lchild、rchild 又多出一個平衡因子 bf , 左高 bf=1 等高 bf=0 右高 bf=-1

寫一個類別為 bool 的 InsertAVL 函式 1.若樹中不存在要插入的數字,則插入成功回傳 true 反之回傳 false 2.失去平衡,做平衡旋轉處理插入分 6 種狀況

- 1. 插入在左子樹 原本左比右高 須做左平衡處理
- 2. 插入在左子樹 原本左右等高 現在因左子樹增高而樹增高
- 3. 插入在左子樹 原本右比左高 現在等高
- 4. 插入到右子樹 原本左比右高 現在等高
- 5. 插入到右子樹 原本左右子樹等高 現在因右子樹增高而樹增高
- 6. 插入到右子樹 原本右子樹比左子樹高 須作右平衡處裡 狀況 1、6 須做旋轉處理,分別寫 leftbalance 和 rightbalance 函式檢查左(右) 子樹平衡度,並做相對應的處理

leftbalance中: T 指向bf值>=|2|的node; L 指向 T 的lchild; Lr 指向 L 的rchild 裡面又分成 2 種狀況:

- 1. 新節點插在 T 的左孩子的左子樹上 作右旋轉,調整 bf 值 (*T)->bf=L->bf=0
- 2. 新節點插在 T 的左孩子的右子樹上 作雙旋轉,調整 bf 值:

```
switch(Lr->bf)
{     case LH:
         (*T)->bf=-1;     L->bf=0;
     case EH:
         (*T)->bf=L->bf=0;
     case RH:
         (*T)->bf=0;L->bf=1;     }
Lr->bf=EH;
```

右旋轉和左旋轉也分別寫兩個函式 r_rotate 和 I_rotate, AVLtree 就完成了工作站執行結果

Input: 9,10,15,3,6,21,8,19,24,3,1,0,48,92,7

```
14:17 tienhsin@bsd1 [~/dshw4] > gcc avltree.c

14:17 tienhsin@bsd1 [~/dshw4] > ./a.out test_hw4

10,6,2,1,0,3,8,7,9,24,19,15,21,48,92,

LL : 2

LR : 1

RL : 1

RR : 3

14:17 tienhsin@bsd1 [~/dshw4] >

14:19 tienhsin@bsd1 [~/dshw4] >
```