HW1 111522118

軟體工程實務 作業一報告

錯誤發現過程，一開始先嘗試幾組簡單測資，一開始一切正常，但發現如果有需要 rotate 調整 AVL 樹高度時，程式就會 crash，不論是 LL RR LR RL 旋轉都會 crash，於是開始去尋找 rotate 的 code 是否錯誤。

```
/* Function to insert data recursively */
private AVLNode insert(int x, AVLNode t)
{
    if (t == null)
        t = new AVLNode(x);
    else if (x < t.data)
    {
        t.left = insert( x, t.left );
        if( height( t.left ) - height( t.right ) == 2 )
            if( x < t.left.data )
                t = rotateWithLeftChild( t );
            else
                t = doubleWithLeftChild( t );
    }
    else if( x > t.data )
    {
        t.right = insert( x, t.right );
        if( height( t.right ) - height( t.left ) == 2 )
            if( x > t.right.data)
                t = rotateWithRightChild( t );
            else
                t = doubleWithRightChild( t );
    }
    else
      ;  // Duplicate; do nothing
    t.height = max( height( t.left ), height( t.right ) ) + 1;
    return t;
}
/* Rotate binary tree node with left child */
```

一開始先是用邏輯推導發現，粉紫底線處原本分別是放入 t.left、t.right，只放入 t 的左或右子樹，實際上要達成 AVL 平衡應該是 t 左右子樹都要參與這個過程，於是先將兩邊都改為 t。

```
/* Rotate binary tree node with left child */
private AVLNode rotateWithLeftChild(AVLNode k2)
{
    AVLNode k1 = k2.left;
    k2.left=k1.right;
    k1.right=k2;

    /*k2.right = k1.left;
    k1.left = k2;*/

    k2.height = max( height( k2.left ), height( k2.right ) ) + 1;
    k1.height = max( height( k1.left ), k2.height ) + 1;
    return k1;
}

/* Rotate binary tree node with right child */
private AVLNode rotateWithRightChild(AVLNode k1)
{
    AVLNode k2 = k1.right;
    k1.right=k2.left;
    k2.left=k1;

    /*k1.left = k2.right;
    k2.right = k1;*/

    k1.height = max( height( k1.left ), height( k1.right ) ) + 1;
    k2.height = max( height( k2.right ), k1.height ) + 1;
    return k2;
}
/**
```

黃色螢光筆處是我後來改的 code，上半部為例，如果是左子樹不平衡，先用 k1 指向 k2 左子樹，然後將 k1 的右子樹發給 k2 的左子樹，在將 k2 接到 k1 的右子樹，完成 AVL 的平衡。

藍筆底線部分是原本的 code，原本上下兩部分的 code 作用剛好相反，rotateWithLeftChild 的 code 是要平衡右子樹，反之一樣，所以上下的 code 都要改。