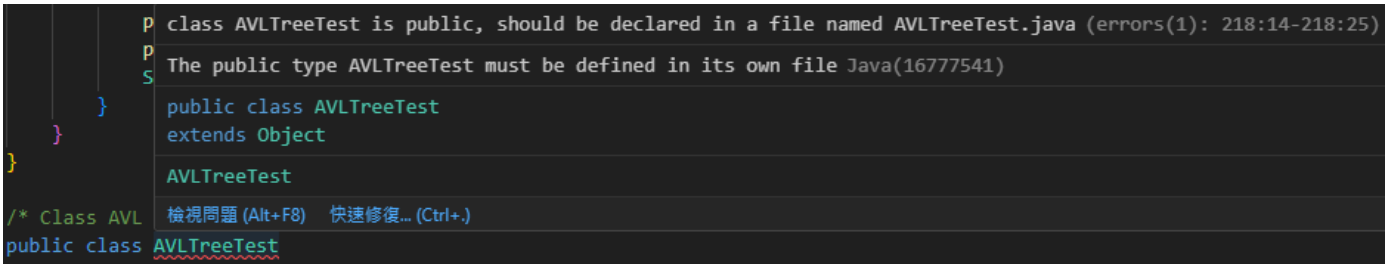
軟體工程實務 HW1

學號：113522118
姓名：韓志鴻

1. **Bug**：Java 檔名和 public class 不同。
   Solution：因為 Java 檔名和 public class 的名稱相同才可以順利執行，所以要先把原始檔名 AVLtree-incorrect.java 改成 AVLTreeTest.java 才可以執行。

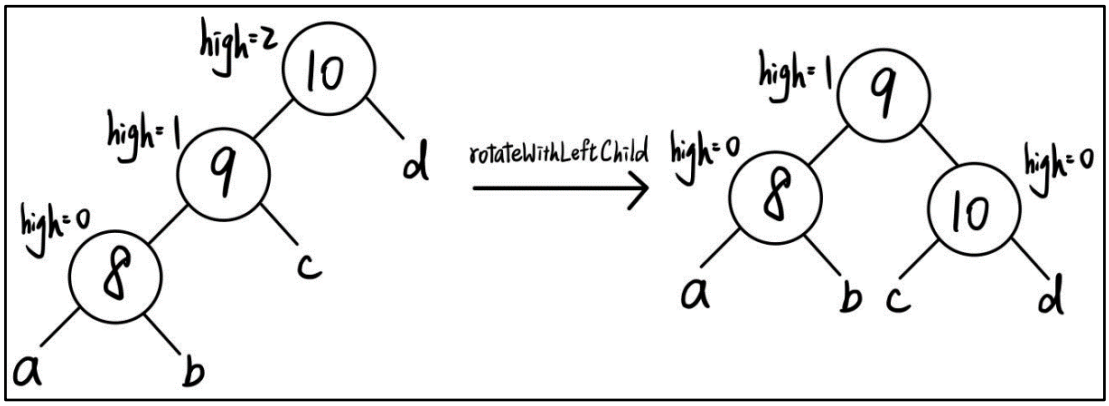


▲ 圖 1：在 public class 顯示的錯誤訊息

2. **Bug**：rotateWithLeftChild()**參數傳遞錯誤**，以及**旋轉寫法錯誤**。
   a、 Solution：如下圖 2 和圖 3，當 t＝10 時，若按原先寫法 rotateWithLeftChild(t.left)，會把 9 傳入 rotateWithLeftChild 函式內部進行向右旋轉。然而在函式內部無法從 9 取得父子點 10 的位置，因此無法把 10 改成 9 的右子點。故必須改成 rotateWithLeftChild(t)，把旋轉目標當作參數傳入，後續才能順利執行。

```
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 ); // 原 rotateWithLeftChild( t.left )
        else
            t = doubleWithLeftChild( t );
}
```

▲ 圖 2：程式碼修改



▲ 圖 3：rotateWithLeftChild 執行示意圖

   b、 Solution：如上圖 3，若要從左邊的樹旋轉成右邊的樹，則需要以 9 為軸，將 c 節點變成 10 的左節點，接著把 10 當作 9 的右子點。同理對應下圖 4 的 code，其中 k1＜k2(因為 k1 是 k2 的左子點)，以 k1 為軸，要先把 k1 的右子點變成 k2 的左子點，接著把 k2 當成 k1 的右子點，才能正確完成 AVL 樹的平衡。故原 code 中的 left 和 right 部分全部都寫相反的寫法是錯誤的。

```
/* Rotate binary tree node with left child */
private AVLNode rotateWithLeftChild(AVLNode k2) // 以左子節點為軸向右旋轉
{
    AVLNode k1 = k2.left;
    k2.left = k1.right; // 原 k2.right = k1.left
    k1.right = k2; // 原 k1.left = k2
    k2.height = max( height( k2.left ), height( k2.right ) ) + 1;
    k1.height = max( height( k1.left ), k2.height ) + 1;
    return k1;
}
```

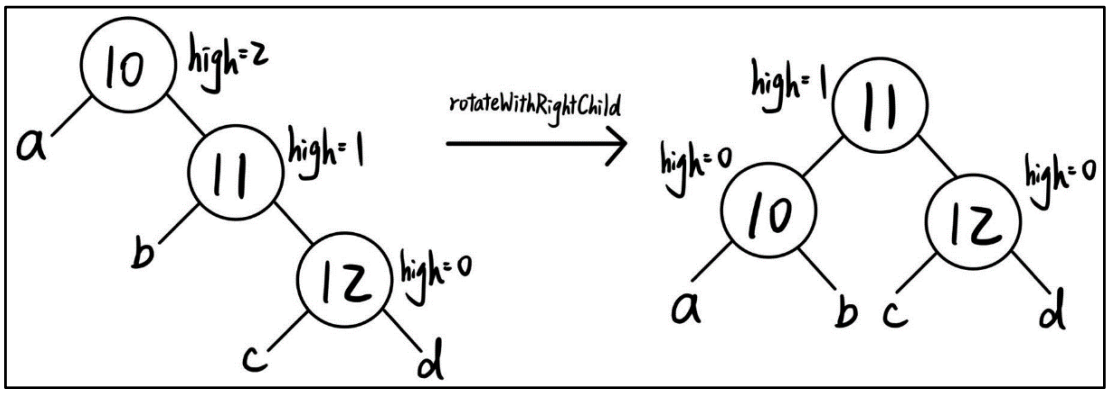
▲ 圖 4：rotateWithLeftChild 內部旋轉程式碼修改

3. **Bug**：rotateWithRightChild ()**參數傳遞錯誤**，以及**旋轉寫法錯誤**。

a、 Solution：如下圖 5 和圖 6，與先前的 rotateWithLeftChild 同理。當 t = 10 時，若按原先寫法 rotateWithRightChild (t.right)，會把 11 傳入 rotateWithRightChild 函式內部進行向左旋轉。然而在函式內部無法從 11 取得父子點 10 的位置，因此無法把 10 改成 11 的左子點。故必須改成 rotateWithRightChild (t)，把旋轉目標當作參數傳入，後續才能順利執行。

```
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 ); // 原 rotateWithRightChild( t.right )
        else
            t = doubleWithRightChild( t );
}
```

▲ 圖 5：程式碼修改



▲ 圖 6：rotateWithRightChild 執行示意圖

b、 Solution：如上圖 6，若要從左邊的樹旋轉成右邊的樹，則需要以 11 為軸，將 b 節點變成 10 的右節點，接著把 10 當作 11 的左子點。同理對應下圖 7 的 code，其中 k1＜k2(因為 k2 是 k1 的右子點)，以 k2 為軸，要先把 k2 的左子點變成 k1 的右子點，接著把 k1 當成 k2 的左子點，才能正確完成 AVL 樹的平衡。故原 code 中的 left 和 right 部分全部都寫相反的寫法是錯誤的。

```
/* Rotate binary tree node with right child */
private AVLNode rotateWithRightChild(AVLNode k1) // 以右子節點為軸向左旋轉
{
    AVLNode k2 = k1.right;
    k1.right = k2.left; // 原 k1.left = k2.right
    k2.left = k1; // 原 k2.right = k1
    k1.height = max( height( k1.left ), height( k1.right ) ) + 1;
    k2.height = max( height( k2.right ), k1.height ) + 1;
    return k2;
}
```

▲ 圖 7：rotateWithRightChild 內部旋轉程式碼修改