```
/**
 * Return the height of node t, or -1, if null.
 */
private int height( AvlNode<AnyType> t )
{
    return t == null ? -1 : t.height;
}
```

图 4-38　计算 AVL 节点的高度的方法

```
/**
 * Internal method to insert into a subtree.
 * @param x the item to insert.
 * @param t the node that roots the subtree.
 * @return the new root of the subtree.
 */
private AvlNode<AnyType> insert( AnyType x, AvlNode<AnyType> t )
{
    if( t == null )
        return new AvlNode<AnyType>( x, null, null );

    int compareResult = compare( x, t.element );

    if( compareResult < 0 )
    {
        t.left = insert( x, t.left );
        if( height( t.left ) - height( t.right ) == 2 )
            if( compare( x, t.left.element ) < 0 )
                t = rotateWithLeftChild( t );
            else
                t = doubleWithLeftChild( t );
    }
    else if( compareResult > 0 )
    {
        t.right = insert( x, t.right );
        if( height( t.right ) - height( t.left ) == 2 )
            if( compare( x, t.right.element ) > 0 )
                t = rotateWithRightChild( t );
            else
                t = doubleWithRightChild( t );
    }
    else
        ;  // Duplicate; do nothing
    t.height = Math.max( height( t.left ), height( t.right )) + 1;
    return t;
}
```

图 4-39　向 AVL 树的插入例程

对于图 4-40 中的那些树，方法 rotateWithLeftChild 把左边的树变成右边的树，并返回对新根的引用。方法 routateWithRightChild 是对称的。程序在图 4-41 中表出。