Example 2: A Complex Rotation

BST Rotation Summary:
1. Four kinds of rotations (L, R, LR, and RL)
2. All rotations are local
3. All rotations run in constant time, O(1)
4. BST property is maintained!

Overall Goal:
...and we call these trees:
...additional property:

**AVL Theorem #1:** If an insertion occurred in subtrees $t_3$ or $t_4$ and a subtree was detected at $t$, then a __________ rotation about $t$ restores the balance of the tree.

**AVL Theorem #2:** If an insertion occurred in subtrees $t_2$ or $t_3$ and a subtree was detected at $t$, then a __________ rotation about $t$ restores the balance of the tree.
AVL Insertion

Pseudocode:

```cpp
class TreeNode {
  public:
    T key;
    unsigned height;
    TreeNode *left;
    TreeNode *right;
}
```

AVL Insertion

```cpp
template <typename K, typename V>
void AVL<K, V>::_insert(const K & key, const V & data, TreeNode *& cur) {
  if (cur == NULL) { cur = new TreeNode(key, data); }
  else if (key < cur->key) { _insert(key, data, cur->left); }
  else if (key > cur->key) { _insert(key, data, cur->right); }
  _ensureBalance(cur);
}

void AVL<K, V>::_ensureBalance(TreeNode *& cur) {
  int balance = height(cur->right) - height(cur->left);
  if (balance == -2) {
    int l_balance = height(cur->left->right) - height(cur->left->left);
    if (l_balance == -1) { ____________________________;
    } else { ____________________________;
    }
  } else if (balance == 2) {
    int r_balance = height(cur->right->right) - height(cur->right->left);
    if (r_balance == 1) { ____________________________;
    } else { ____________________________;
    }
  }
  _updateHeight(cur);
}
```

AVL Removal

CS 225 – Things To Be Doing:

1. mp_traversal EC deadline today
2. Daily POTDs