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:

```
AVL.h (snippet)
23  class TreeNode {
24    public:
25        T key;
26        unsigned height;
27        TreeNode *left;
28        TreeNode *right;
...

AVL.cpp
151  template <typename K, typename V>
152  void AVL<K, D>::_insert(const K & key, const V & data, TreeNode * & cur) {
153      if (cur == NULL) { cur = new TreeNode(key, data); }
154      else if (key < cur->key) { _insert(key, data, cur->left); }
155      else if (key > cur->key) { _insert(key, data, cur->right); }
156      _ensureBalance(cur);
157  }
158
159  template <typename K, typename V>
160  void AVL<K, D>::_ensureBalance(TreeNode * & cur) {
161      // Calculate the balance factor:
162      int balance = height(cur->right) - height(cur->left);
163      // Check if the node is current not in balance:
164      if (balance == -2) {
165          int l_balance = height(cur->left->right) - height(cur->left->left);
166          if (l_balance == -1) { _________________; } // Left-left case
167          else { _________________; } // Left-right case
168      } else if (balance == 2) {
169          int r_balance = height(cur->right->right) - height(cur->right->left);
170          if (r_balance == 1) { _________________; } // Right-right case
171          else { _________________; } // Right-left case
172      } _updateHeight(cur);
173  }
```