Single/Double Rotation

\[ p < T_e < c \]

\[ T_1 < p < T_2 < c < T_3 \]

\[ \text{height at } p = 1 + \max(\text{height at } p \to \text{left, height at } p \to \text{right}) \]

\[ c \text{ is still unbalanced!} \]

\[ \text{node added} \]

\[ \text{lowest unbalanced node} \]

\[ \text{something else!} \]

\[ \text{insider: } T_1, p, T_2, g, T_2, c, T_3 \]

\[ T_1, p, T_2, g, T_2, c, T_3 \]
Inorder Traversal

Inorder traversal processes keys in sorted order

```
traverse(n)
{
    if (n != NULL) {
        printf(“%s\n”, n->key);  // preorder
        traverse(n->left)
        printf(“%s\n”, n->key);  // inorder
        > traverse(n->right)
        printf(“%s\n”, n->key);  // postorder
    }
}
```
donut hamster rule kelp pickle lemon midterm no packle numb oat

AVL Example
Removing from a BST

[Diagram of a binary search tree with nodes labeled ABE, LHR, NRT, MCI, CGK, PER, STL, SAF, SAW, SEA, YVR, TPA, YYZ, STS, TUL, TOL.]

Fill hole with something less than everything on right or greater than everything on left.
20 Questions

- Do you peel it?
  - Yes
    - Is it buzzy?
      - Yes: Kiwi
        - Does it grow in bunches?
          - Yes: Banana
          - No: Orange
      - No
        - Does it grow in bunches?
          - Yes: Grape
          - No: Apple
  - No
    - Does it smell bad?
Add branch

1. Make new node for existing class (Wrong guess)
2. Copy pointer to old class to new node
3. Make new node for new class (Correct answer)
4. Make copy of new class for new node
5. Replace text in existing node with copy of new text (New?)
6. Point pointer `new-ptr` points to node for new class
7. Point pointer `old-ptr` points to new node for old class