CS 225

Data Structures

February 14 – BST Balance

G Carl Evans

Exam 1 Issues

Several of the free response questions were poorly written which impacted the answers given.

- Grades and feedback assume that the questions were understood
- You can see feedback in office hours now / later will be available
- Grades will be changed to be full points for everyone

BST Analysis – Running Time

Operation	BST Worst Case
find	
insert	
delete	
traverse	

BST Analysis

Therefore, for all BST:

Lower bound: h >= O(lg(n))

U**pper bound: h <= O(n)**

BST Analysis

The height of a BST depends on the order in which the data is inserted into it.

ex: 1 3 2 4 5 7 6 vs. 4 2 3 6 7 1 5

Q: How many different ways are there to insert keys into a BST?

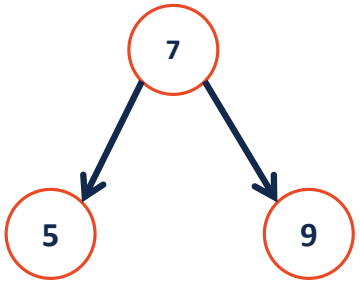
Q: What is the average height of all the arrangements?

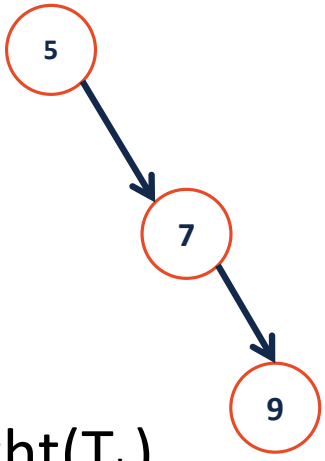
Dictionary Analysis – Running Time

Operation	BST Average case*	BST Worst case	Sorted array	Sorted List
find				
insert				
delete				
traverse				

Height-Balanced Tree

What tree makes you happier?





Height balance: $b = height(T_R) - height(T_L)$

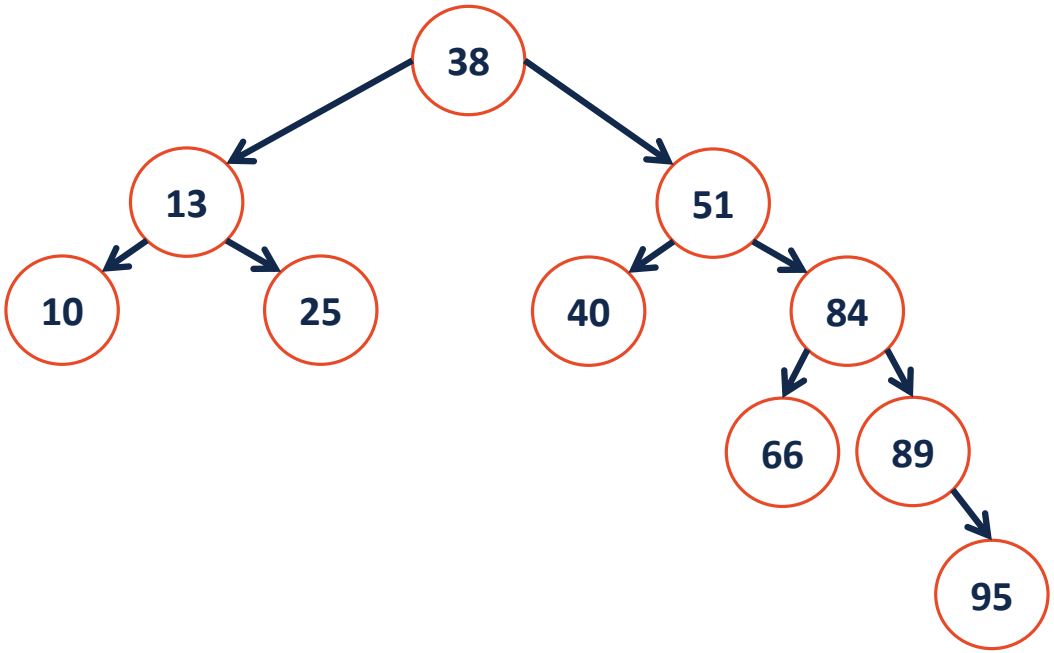
A tree is height balanced if:

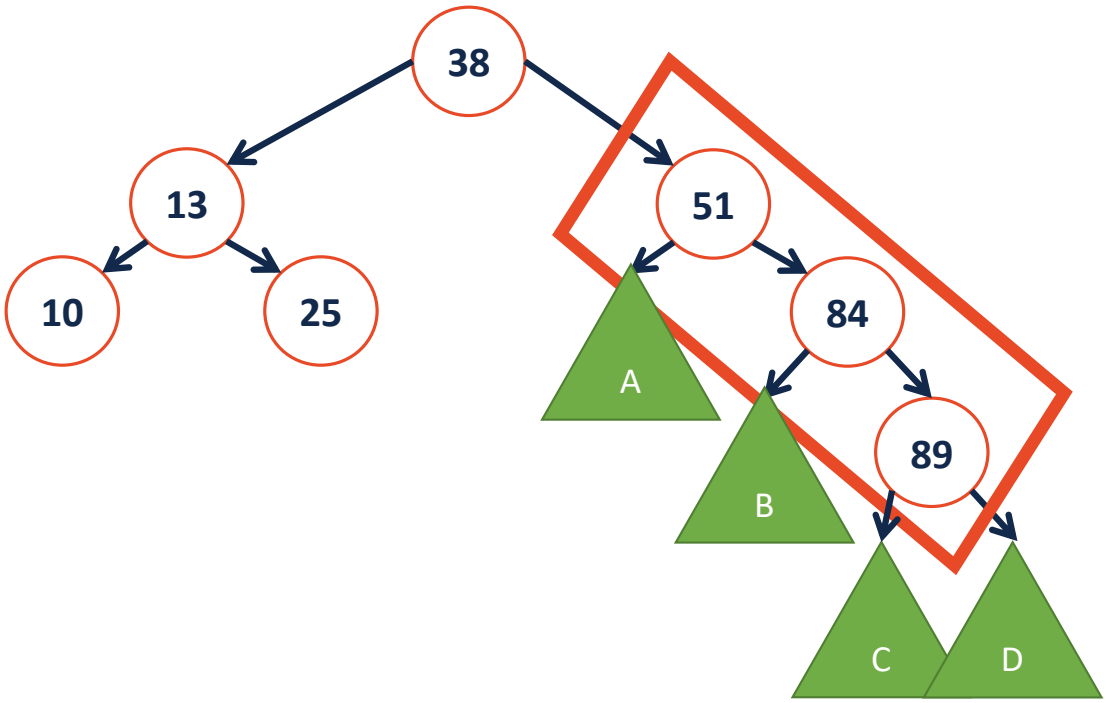
BST Rotation

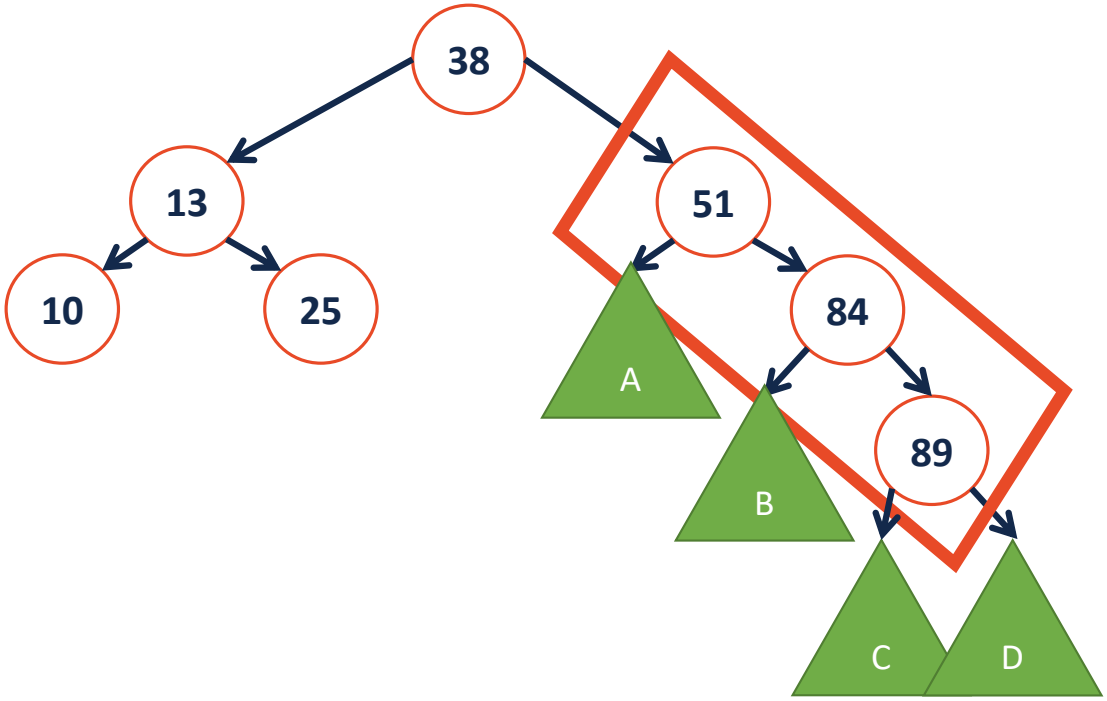
We will perform a rotation that maintains two properties:

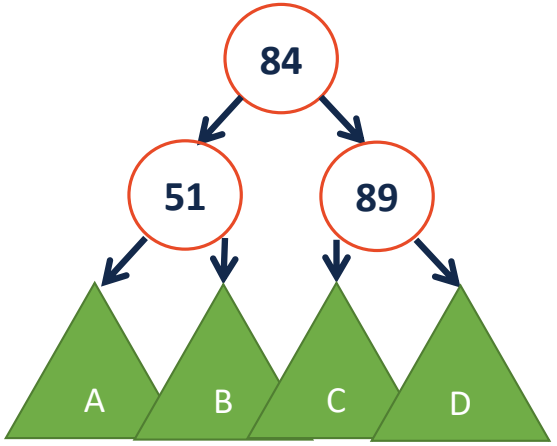
1.

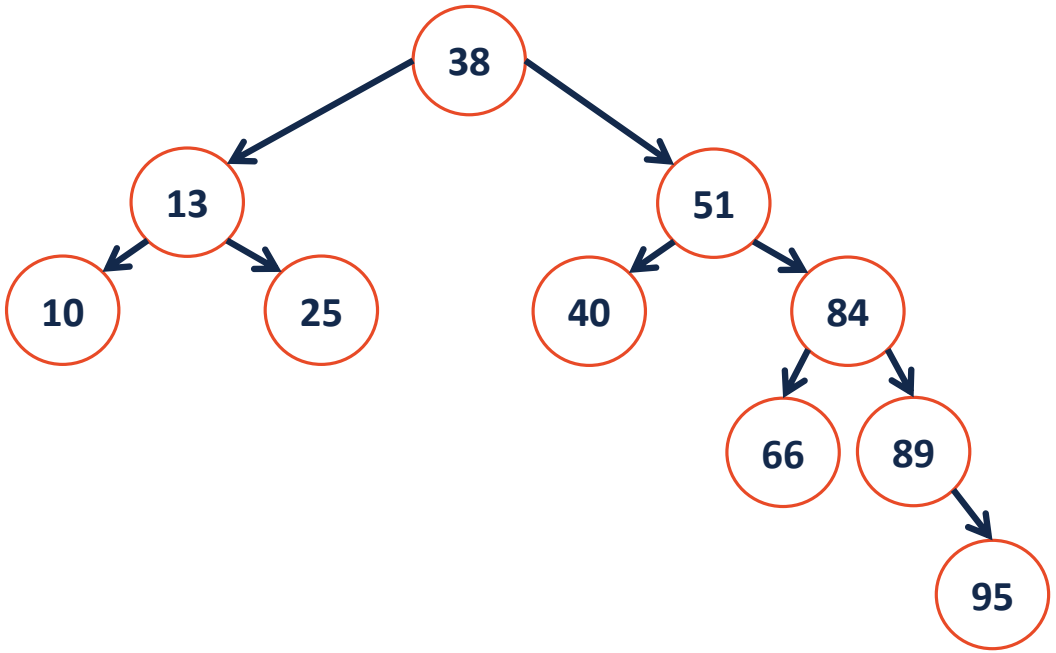
2.

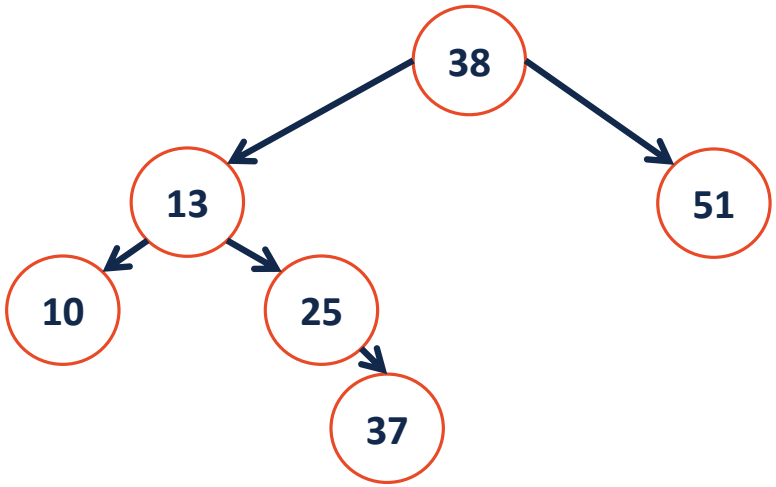

38
13
51
10
25
40
84
66
89
95

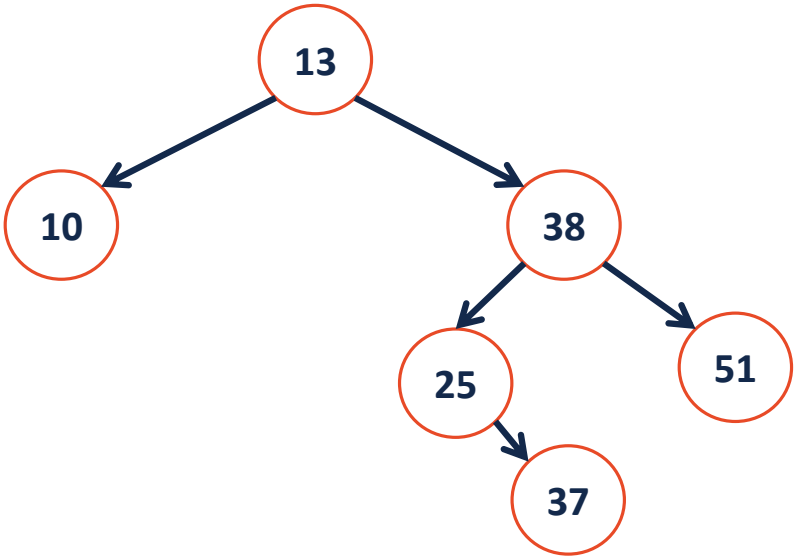

38
13
51
10
25
84
A
89
B
C
D

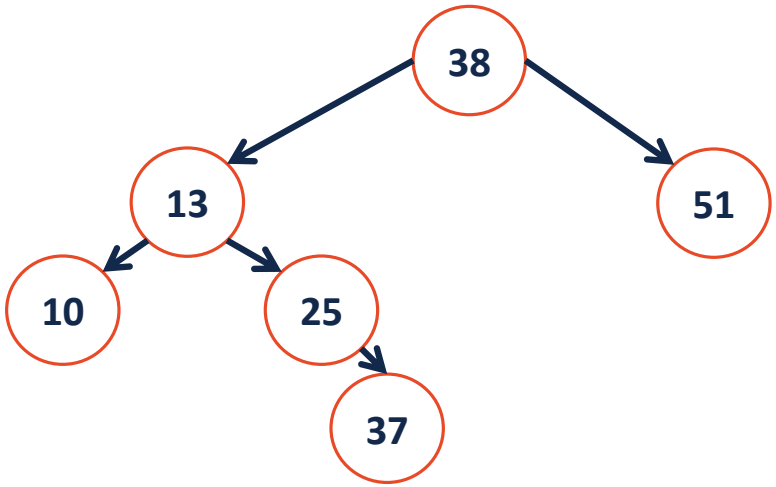

38
13
51
10
25
84
A
89
B
C
D


84
51
89
A
B
C
D


38
13
51
10
25
40
84
66
89
95


38
13
51
10
25
37


13
10
38
25
51
37


38
13
51
10
25
37

BST Rotation Summary

- Four kinds of rotations (L, R, LR, RL)
- All rotations are local (subtrees are not impacted)
- All rotations are constant time: O(1)
- BST property maintained

GOAL:

We call these trees:

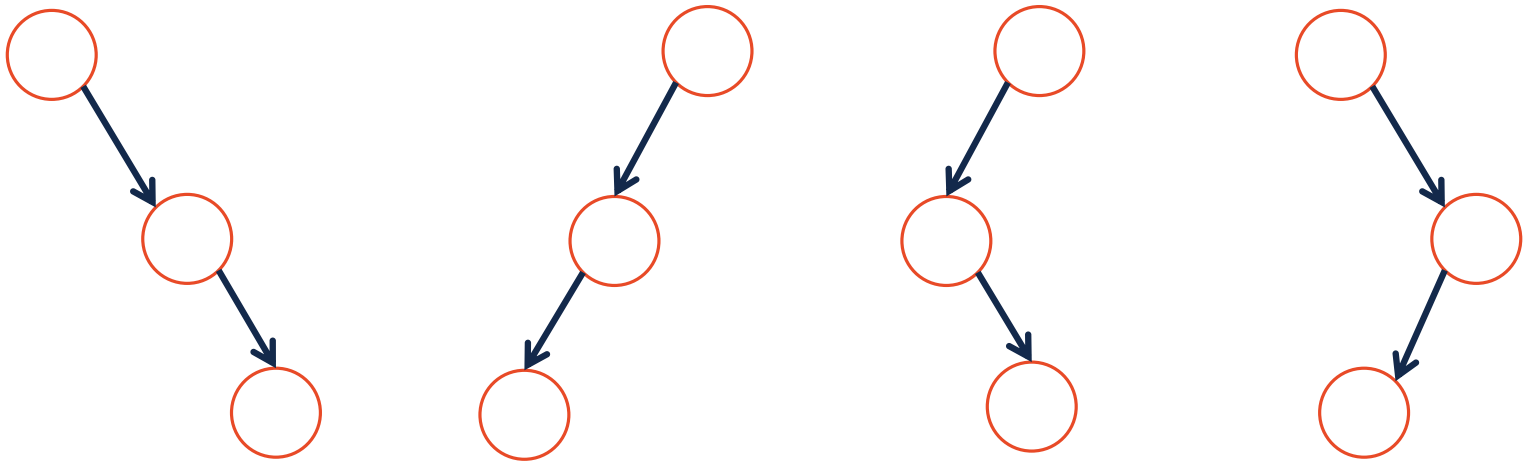
AVL Trees

Three issues for consideration:

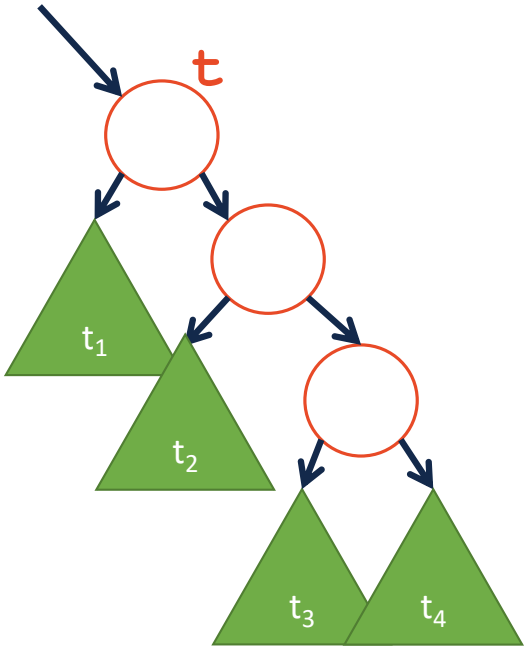
- Rotations
- Maintaining Height
- Detecting Imbalance

AVL Tree Rotations

Four templates for rotations:



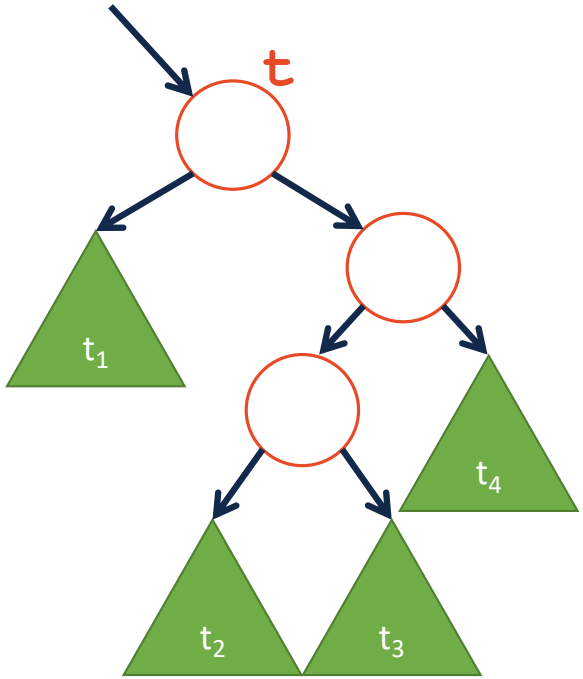
Finding the Rotation on Insert



Theorem:
If an insertion occurred in subtrees $\mathbf{t_3}$ or $\mathbf{t_4}$ and a subtree was detected at **t**, then a _____ rotation about **t** restores the balance of the tree.

We gauge this by noting the balance factor of **t->right** is _____.

Finding the Rotation on Insert



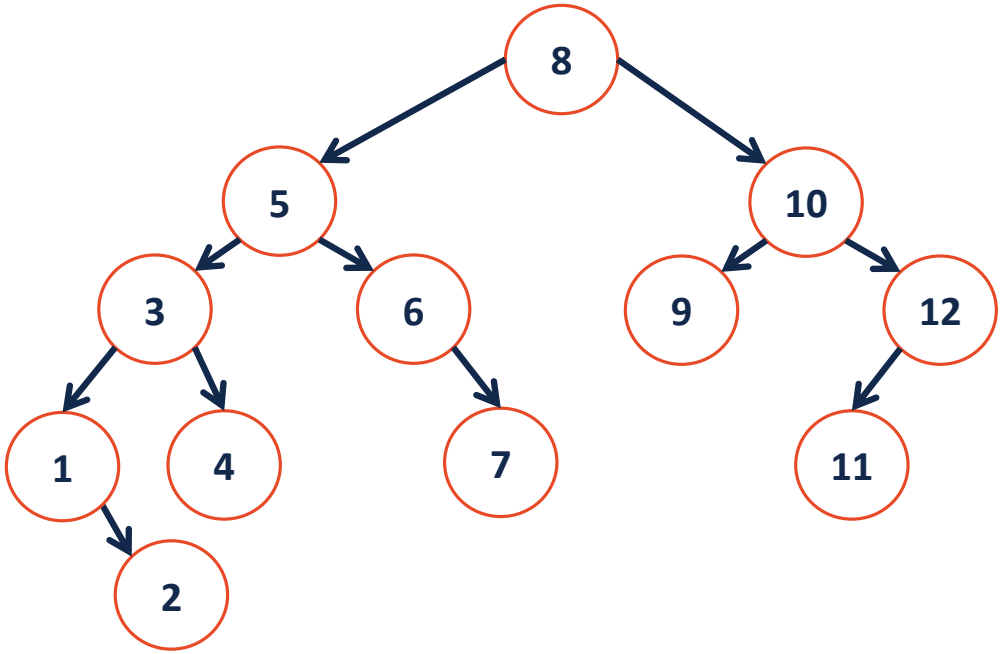
Theorem:
If an insertion occurred in subtrees $\mathbf{t_2}$ or $\mathbf{t_3}$ and a subtree was detected at **t**, then a _____ rotation about **t** restores the balance of the tree.

We gauge this by noting the balance factor of **t->right** is _____.

Insertion into an AVL Tree

`_insert(6.5)`

```
1  struct TreeNode {
2    T key;
3    unsigned height;
4    TreeNode *left;
5    TreeNode *right;
6  };
```



Insertion into an AVL Tree

_insert(6.5)

Insert (pseudo code):

1: Insert at proper place
2: Check for imbalance
3: Rotate, if necessary
4: Update height

```
1  struct TreeNode {
2    T key;
3    unsigned height;
4    TreeNode *left;
5    TreeNode *right;
6  };
```