CS280 – More Trees
February, 2016

http://azrael.digipen.edu/~mmead/www/Courses/CS280/Trees-2.html

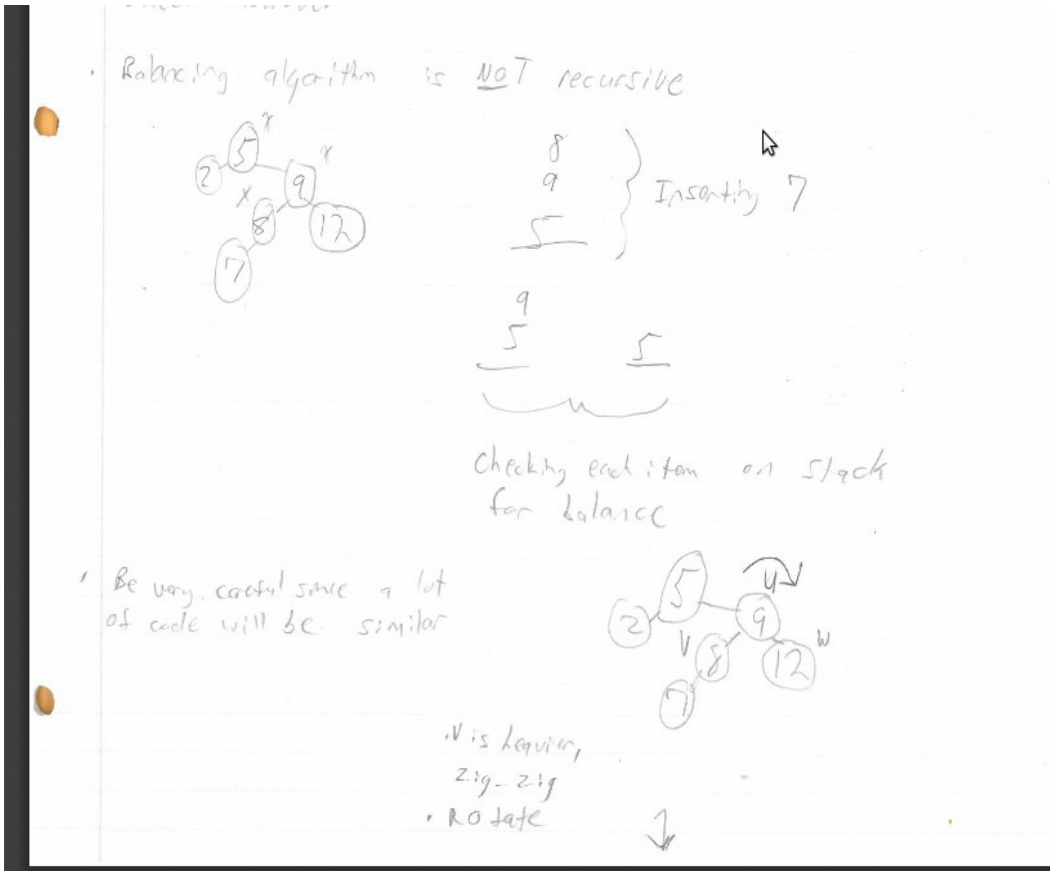
- Stuff from these notes not on the midterm
- **Midterm Wednesday**

Trees

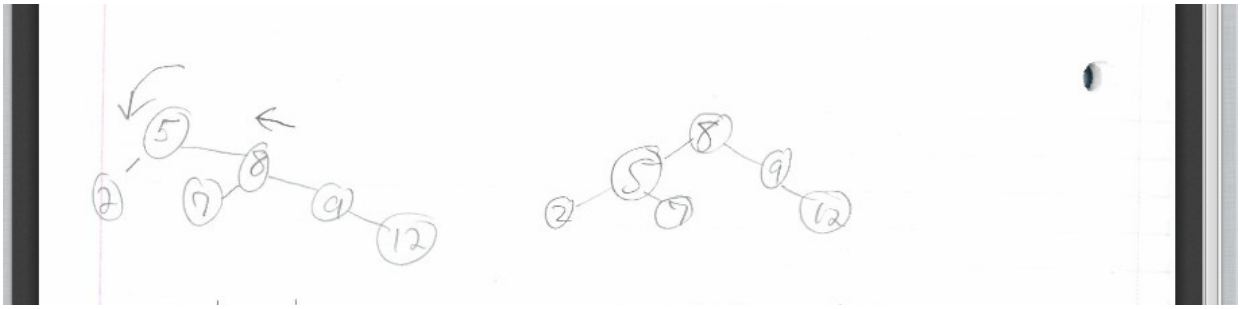
- Main tree operation is rotating nodes

AVL Trees

- Always insert at the bottom of a tree
    - If this makes the tree unbalanced, find the offending node and do rotations to balance it out
- Delete must go all the way to the root of the tree to check balance
- Balancing algorithm is NOT recursive
- Be careful with balance algorithm since code will be similar



Inserting and balancing using a stack



- Do not have insert and delete balancing the tree
  - Use a helper function
- We will lose points if we have two different balance functions for insert and delete
- Just use a flag, since they are almost identical
- Store a "balance factor" for each node
  - Balanced = 0
  - Left heavy = -1

    - Right heavy = 1
    - This lets you do constant time balance checking
    - Actually put "+1" to make it easier to read
- Keeping track of balance facor is non-trivial, so we will use more inefficient height algorithm
- We are using the memory manager to allocate nodes
- Do NOT make copies of the object allocator
- Use DummyOA since it is what the driver will use
- Not templated at first so we can get the algortihm working easier
- "Placement new" takes address so it just does construction step, skilling the allocation step
- Main point of this assignment is the balancing algorithm
- Read through assignment and make sure we can draw the diagrams TODAY

2-3 Search Trees

- Only way to grow the tree is to split the root
- Trees grow wider than they do tall