CS280 – Heaps, Red-Black Trees
March 30, 2016

http://azrael.digipen.edu/~mmead/www/Courses/CS280/Graphs-1.html

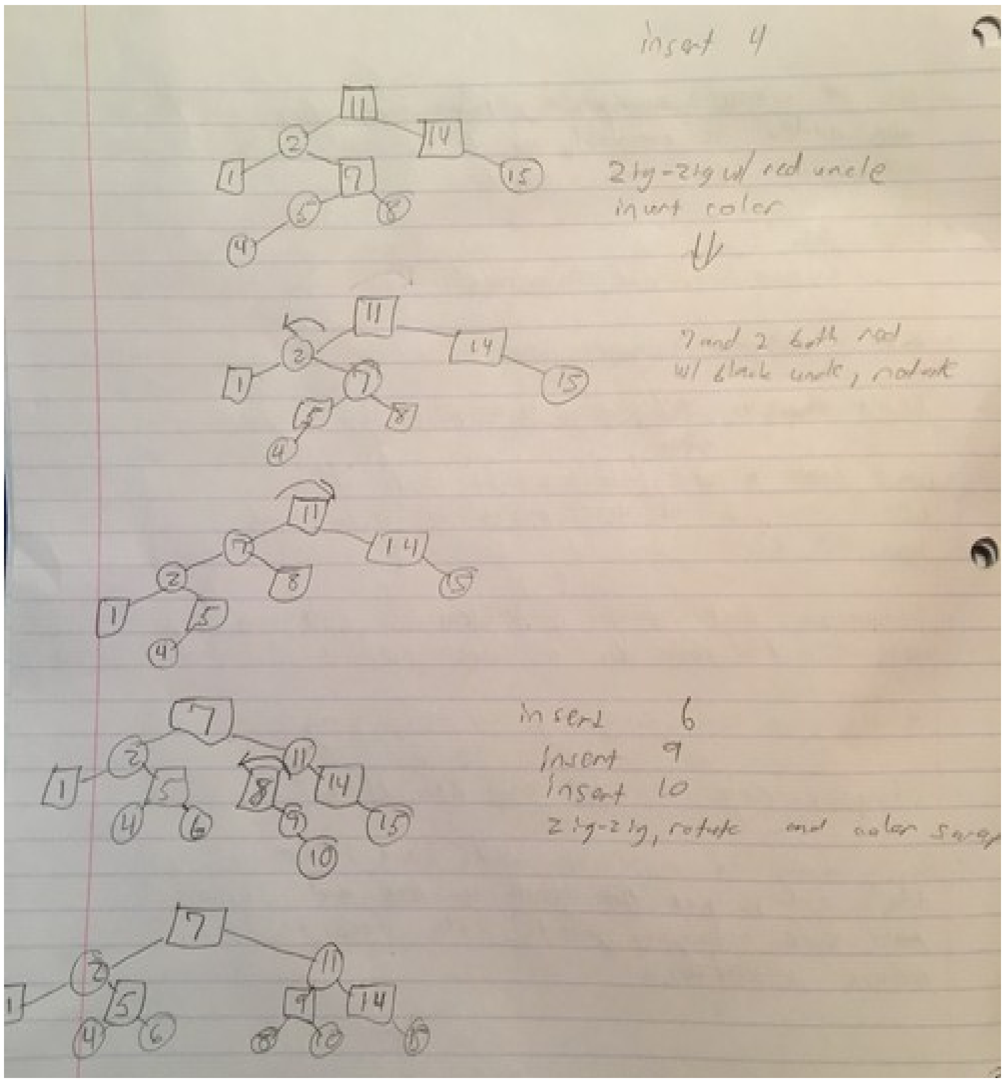
Homework

- How we create the adjacency list and store the nodes is completely up to us
    - Overloaded less than operator will make things easier
- Use the STL!
    - Don't have to, but it's recommended
- Use std::priority_queu

Red-Black Trees

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

- Can't have two red nodes in a row
- Don't have to worry about height since it is based on 2-3-4 trees, which are a balanced type of tree
- Situation that causes a problem is when we have two red nodes in a row
- We need to ask zig-zig or zig-zag
- Changing color is the only new thing
- Really easy if uncle is red, just in very the colors. But then walk up the tree to make sure changing parent's color doesn't cause violation



Inserting into a red-black tree

Red-Black Trees

- Root is always black to keep the algorithm simple
- Red-black tree is not guaranteed to be balanced
- Black node represents 2-node, red node can be 3 or 4 node
- GNU libAVL
- If you are getting sorted data, use AVL tree. If not, red-black can be better because the weird balance situation shouldn't occur.
    - Overall, red-black tree and AVL tree are about equal for random data

- Skip lists let you do a binary search of a list
- Multiple keys per node in a 2-3-4 tree
    - Basis of a B-Tree
    - STL sets are implemented using a red-black tree

Heaps

http://azrael.digipen.edu/~mmead/www/Courses/CS280/Heaps-1.html

- A BST where each child's value <= parent
    - Can contain duplicates
- Root is always the largest value
- If we remove the root, we have to 're-heapify' the tree
    - $\log_2(N)$
- How do you find last element in a tree>
    - Left children are even, right children are odd
- Can use binary to traverse to element
- Complete binary trees can be implemented using arrays
- When looking at priority queu in visual studio don't think it is sorted, a tree is being implemented using an array! Visual studio isn't broken
- Heap is visualized as tree, but almost always implemented as an array
- Memory heap is unrelated to the data structure!