COSC 310 Final

**Trees**

Binary search trees

Dictionary

Hashtable

Balancing of trees

Heaps

Ways to keep binary search tree to keep O(logn)

Tree best case is O(logn) worst case is O(n)

Trees are graphs where there are no cycles and at most one parent to any given vertex

Binary tree representation of a general tree

**Sort algorithms**

Merge sort O(nlogn) – sort into two sorted lists then merge the two lists

\*Heap sort O(nlogn) – add the elements to a heap. Fill in inserts level by level from left to right. If there is a violation just swap the two numbers. When removing, always take off from the top.

Quick sort (O(nlogn) unless it is already sorted then O(n2))

Selection sort O(n2)

Insertion sort O(n2) – deck of cards

Bubble sort O(n2)

Be able to say what methods are in a particular interface not how to implement them

A Set ignores duplicates. A dictionary does not allow duplicate keys.

Directed or undirected graph

Connected graph – any vertex can get to any vertex

Undirected graphs can be fully connected unless part is not connected

Depth first search will search for unknown areas until it cannot search anymore

Binary search tree: insert 10, 15, 20, 19, 1

10

1 15

5 20

8 19

Insert into AVL tree – find point of violation then rotate. Know left left heavy – right right heavy – right left heavy…

2-3 B tree – 2 keys 3 children, middle number moves up because the other 2 keys are sorted

Red black trees – node is either red or black, the root is black, and the number of black nodes is the same, never have…

Go to visulation sites:

\*\*Red Black tree

Insert and remove from heap

B tree

2 3 tree

AVL

\*\*In-order traversal  
Hashing

Handle collision