Data Structures
Final Exam., Jan. 3

1. Union/Find Operations: Suppose that the sets are implemented by using forest.
   (a) If Union is implemented by the weighting rule, show that the height of any tree in the forest is $O(\log n)$.
   (b) Show that this bound can be achieved.
2. Hashing:
   (a) Describe the hashing functions, division and folding.
   (b) Name 2 overflow handling techniques and describe the techniques.
3. Some questions regarding sorting algorithms.
   (a) What is the sorting algorithm best for the case that the input sequence is almost sorted? Why?
   (b) You are going to sort n records. Suppose that the size of the records are large. What is the data structure we should use? And Why?
   (c) What is the bad input for a standard implementation of quick sort? Is there bad input for the randomized quick sort? What is the worst case time complexity for the randomized quick sort?
4. Heap Sort:
   (a) Show that building a heap takes $O(n)$ time.
   (b) Describe the heap sort algorithm.
   (c) Is it "in place"? How about standard implementation of merge sort and quick sort? Are they in place? Why?
   (d) Is it "stable"? How about standard implementation of insertion sort and quick sort? Are they stable? Why?
5. LSD radix sort:
   (a) Describe the algorithm and show the correctness of the algorithm.
   (b) It can be done in linear time. Why does it beat the lower bound $\Omega(n \log n)$ obtained under the linear decision tree model?
6. Minimum Leftist Tree: Describe the delete minimum algorithm and delete the minimum from the minimum leftist minimum tree shown in Figure 1.
7. Given the Symmetric Min-Max Heap (SMMH) shown in Figure 2, delete minimum from the SMMH then insert the 2 into the resulted SMMH.
8. Given the AVL-tree shown in Figure 3, insert Hune then insert Oct into the AVL-tree.
9. Balance search Tree:
   (a) How can we search for the kth largest in a balance tree in $O(\log n)$ time.
   (b) Suppose we add two "fingers" into the tree, one finger points to the smallest and the other one points to the largest. And suppose that each node has pointers to its children as well as a pointer to its parent. Describe an algorithm that we can search for a node, x, in $O(\log d,)$ time where d is the "distance" between x to the smallest or the largest node.
10. Given a binomial heap shown in Figure 5, describe the algorithm to delete a given node and delete the node x from the heap.

11. Biconnected component: Given a graph shown in Figure 6,

    (a) draw the adj. list presention of the graph.

    (b) Draw the Depth First Search Tree obtained by applying the DFS according to the adj. list you gave.

    (c) Mark the *dfn* and *low* in the tree.

    (d) Which vertices are the articulation poitns?

12. Given a weighted graph $G = (V, E)$. We would to compute the MST of G.

    (a) State the Kruskal's algorithm.

    (b) What is the best algorithm that we can find the least weight edge? What is the data structure we need? And what is the time complexity?

    (c) What is the best algorithm that we can detect a cycle if it is formed? What is the data structure we need? And what is the time complexity?