CS 610-001, Fall 2009, FINAL EXAM Exam #: _____ Name:_____
Prof. David Nassimi (Time: 2 Hours) First Last

1. (a) Insert the following sequence of elements in an AVL tree, starting with an empty tree. Show the rebalancing as needed after each insert operation.

$$50, 20, 70, 60, 80, 65, 62, 68, 61$$

	GRADE	
1		/20
2		/20
3		/20
4		/20
5		/20
SUM		/100

(b) An AVL tree has the following **preorder** listing. Show the corresponding AVL tree from this list. Briefly explain how this is done.

$$50, 20, 10, 30, 25, 35, 70, 60, 55, 65, 80$$

2. (a) Obtain a Huffman-Code for the following alphabet with the given probabilities. Briefly explain how the code is obtained. Show the corresponding code tree.

Symbol	Prob	Huff Code
A	0.05	
B	0.05	
C	0.07	
D	0.08	
E	0.1	
F	0.15	
G	0.2	
H	0.3	

(b) What is the time complexity of this algorithm for finding the optimal code for an alphabet with n symbols? Give a brief explanation.

(c) Which of the following design strategies does the algorithm use?

- Divide-and-Conquer
- Greedy
- Dynamic Programming
- None of the above

3. This problem is concerned with finding the worst-case number of key comparisons (exact number, not order) for sorting $n = 4$ real numbers.

   (a) Derive a lower bound on the worst-case number of key comparisons needed by any comparison-based sorting algorithm for $n = 4$.

   (b) Derive the worst-case number of key-comparisons used by insertion-sort algorithm for $n = 4$.

   (c) Derive the worst-case number of key-comparisons used by mergesort algorithm for $n = 4$.

   (d) Is either one of the above two algorithms optimal in terms of the worst-case number of key-comparisons? Explain.

4. Given a directed graph (not weighted) represented by its adjacency matrix A, where $A[i,j] = 1$ if (i,j) is an edge, or if $i = j$, and $A[i,j] = 0$ otherwise.

   We want to compute the **transitive-closure** of the graph, defined by matrix T, where $T[i,j] = 1$ if there is a **path** from i to j, or if $i = j$, and $T[i,j] = 0$ otherwise.

   (a) Briefly explain how to adapt one of the more general algorithms we discussed to apply to this special case. What is the time complexity of the algorithm?

   (b) Give the pseudocode for computing T from A.

   (c) Suppose we want to compute the transitive closure of an **undirected graph**. Is there a more efficient algorithm for this case? Briefly explain what algorithm could be used and its time complexity.

5. Consider Prim's MST algorithm for finding a minimum-cost spanning tree of an undirected weighted graph. Assume the graph is connected, is sparse, and is represented by its adjacency-lists.

   (a) What is the time complexity of Prim's algorithm for this graph representation? Briefly explain what data structure is used and how each iteration is implemented.

   (b) Now suppose we have an undirected weighted graph, where all edge weights are **integers** in the range 1 to k, for some small constant k. (For example, $k = 10$.) Is there a more efficient implementation of Prim's algorithm for this special case? What data structure is used? What is the time complexity for this case?

(c) Illustrate the latter algorithm for the following 5-vertex graph (represented by its adjacency lists). First, draw the graph. Then, show the result of each iteration. Assume vertex 1 is the source.

i	$(j, C_{i,j})$
1	$(2,6), (3,7)$
2	$(1,6), (4,2), (5,3)$
3	$(1,7), (4,5)$
4	$(2,2), (3,5), (5,4)$
5	$(2,3), (4,4)$