Problem Number(s)	Possible Points	Earned Points	Course Outcome(s)
1	10		
2	10		
3	10		
4	12		
5	8		
6	30		
7	15		
8	5		
	TOTAL POINTS 100		

Exam 2
100 points

1) (10 points) A priority Queue has the following two basic operations:
Insert a new item
Remove the item with the largest key. (This also includes FINDING the max.)

We considered several underlying implementations of the priority queue each with widely varying performance characteristics. Complete the following table using O(1), O(lg N), O(N), O(N lg N), or $O(N^2)$ to denote the complexity of each operation using the given implementation strategy on the left. Please assume the array will not need a resize operation during an insert or remove for this exercise and that list implies a singly list with head and tail pointers.

	Insert	Remove Maximum
Unordered Array		
Unordered List		
Ordered Array		
Ordered List		
Heap		

2) (10 points)
(A.) Describe an algorithm for removing the maximum from a priority queue implemented as an unordered list.

(B.) Describe an algorithm for inserting a new key value into a priority queue implemented as an ordered array.

3) (A.) (6 points) Draw a box around and label the resulting max-heaps after inserting the following numbers in the order in which they appear one by one. You should have one heap diagram for each number.
3, 7, 6, 11, 5, 13, 0

(B.)(4 points) Show the resulting max-heap after a remove max operation.

4) For heap sort our first task is to make the given array satisfy the max-heap property but…
(A.) (3 points) What is the max-heap property?

(B.) (9 points) In class we turned normal arrays into heaps. We called this operation heapify. Given the following array of numbers; show the final heap and the final corresponding array resulting from our heapify operation. Show all work.
3, 7, 6, 11, 5, 13, 0

5) (8 points) Full vs Complete binary trees
(A.) Give an example of a binary tree that is full but not complete.

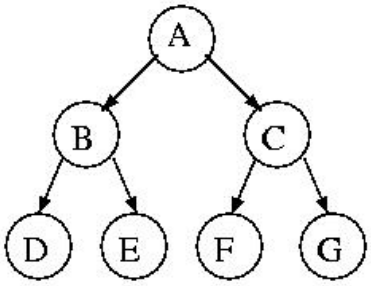
(B.) Give an example of a binary tree that is complete but not full.

(C.) Give an example of a binary tree that is neither full nor complete.

(D.) Give an example of a binary tree that is both full and complete.

6) Tree Traversals (30points)

I. Consider the following binary tree (not satisfying the BST property) and a visit operation that prints the contents of a node.



a) Give the output for a preorder traversal calling visit.

b) Give the output for an inorder traversal calling visit.

c) Give the output for a postorder traversal calling visit.

II. Given the following Binary tree structure, write recursive functions to traversal through the tree, printing out the key held by each node.

```
typedef struct Node treeNode;

struct Node{
     int key;
     treeNode* lchild;
     treeNode* rchild;

};
```

treeNode* root;

NB: assume there is a function insert that has already inserted data items to the tree.

a) Pre-order Traversal

b) In-order Traversal

c) Post-order traversal

7). AVL Balancing (15 points)
Assuming the following values arrive in the order they appear and are inserted into an empty AVL tree, show the resulting tree.

14, 17, 11, 7, 53, 4, 13

a) Insert 12
b) Insert 8
c) Remove 53

8). (5 points) Given the following preorder and in-order traversals for an unknown binary tree, determine the exact tree that would generate these traversals and then draw that tree. Once you have generated the tree be sure to check your work.

Pre-order: {7, 10, 4, 3, 1, 2, 8, 11}
In-order: {4, 10, 3, 1, 7, 11, 8, 2}