Instructions: Submit it through OWL online or in class.

1. (25 marks) Consider the algorithm AddSomething(A, n):

   Algorithm AddSomething(A, n)
   Input: Array A and integer n. A has size at least n
   if n < 1 return 0
   else
   return temp

(a) (10 marks) Write down recurrence equations that describe the running time of algorithm AddSomething(A, n).
(b) (15 marks) Solve these recurrence equations and give the asymptotic (big-O) complexity of the algorithm.

2. (10 marks) Prove that in a binary tree, the minimum number of internal nodes is h, and the maximum number of internal nodes is 2^h - 1, where h is the height of the tree. That is prove h ≤ i ≤ 2^h - 1, where i is the number of internal nodes.

3. (10 marks) Draw a binary tree where each node stores a character key, and preorder traversal visits nodes in the order LXZFGED, and postorder traversal visits nodes in the order ZFXEDGL.

4. (10 marks) Suppose that a heap stores 210 elements. What is its height?

5. (20 marks) Suppose that a heap has hight 10 and stores only unique keys. At which levels of the heap can the 4th largest key be stored?
6. (25 marks)

(a) (15 marks) Suppose $T$ is a binary tree that at each node stores an integer key. Let us call a node left-unbalanced if the sum of keys in its left subtree is less than the sum of keys in its right subtree. For example, in the tree above, the root node is left-unbalanced, since the sum of keys in its left subtree is 4 and in the right subtree is 5. Also the left child of the root node is left-unbalanced. Write an algorithm that takes as an input the root of the tree and returns the number of left-unbalanced nodes. For the tree above, your algorithm should return 2.

(b) (10 marks) Analyze the running time of your algorithm in part (a) as a function of the number of tree nodes $n$.

7. (Just for Fun) Prove that if every internal node in a tree has exactly 3 children, then $e = 2i + 1$, where $e$ is the number of external nodes and $i$ is the number of internal nodes.