1. Provide solutions (using big-Oh or big-Theta) for the following recurrence relations.

   (a) \( T(n) = 9T(n/3) + n^2 \)
   (b) \( T(n) = 5T(n/3) + n \)
   (c) \( T(n) = 5T(n/3) + n^3 \)

   sol’n The time bounds are \( \Theta(n^2 \lg n) \), \( \Theta(n^{\log_3 5}) \), and \( \Theta(n^3) \), respectively.

2. Into an initially empty AVL tree, insert the following values: 22, 30, 25, 10, 8, 6, 12, 15, 24, 20, 18.

3. Insert the values above into an initially empty 2-3-4 tree.

4. Consider the following routine which takes an array of integers \( A[1], A[2], \ldots, A[n] \) and constructs a BST containing them.

   ```
   buildBST(array A)
   BST T
   for i=1 to n
     T.insert(A[i])
   return T
   ```

   (a) What is the worst case time of this routine? Explain.

   sol’n In the case of say, a sorted input, we end up with a skew tree. Total insertion time will be \( \Theta(n^2) \).

   (b) What is the average case time of this routine? Explain.

   sol’n The total insertion time in the average case will be the expected value of the internal path length. This will be \( \Theta(n \lg n) \)

   (c) What is the best case time of this routine? Explain.

   sol’n In the best case we will get a fully balanced BST of height \( \Theta(\lg n) \). The total insertion time will then be \( \Theta(n \lg n) \)

5. For this question we allow a BST to contain duplicate values: for example, there may be three nodes \( p \) for which \( p.value=5 \). If node \( p \) stores a 5, another 5 may be in the left subtree of \( p \), but not in the right subtree.

Write a short routine \( \text{mode}(\text{BST} \ T, \ \text{int} \ k) \) which determines the number of nodes in the tree \( T \) which contain the value \( k \).

sol’n
mode(BST T, int k)
    return mode1(T.root, k)

mode1(node p, int k)
    if (p=null) return 0

    if (p.value = k) return 1+mode1(p.left, k)

    if (p.value < k) return mode1(p.left, k)
    else return mode1(p.right, k)

Total: 50 points