- Carefully read each question before answering it.
- Print your name on the cover of the exam booklet

1. (15 points)
   a) Give the precise definition of Big Oh (e.g., T(n) = O(f(n))).
   b) Suppose T1(n) = O(f(n)) and T2(n) = O(f(n)). Which of the following are true?
      i. T1(n) + T2(n) = O(f(n))
      ii. T1(n) – T2(n) = O(1)
      iii. T1(n) = O(T2(n))
2. (5 Points) Put the following growth functions in the ascending order of their asymptotic growth rates:

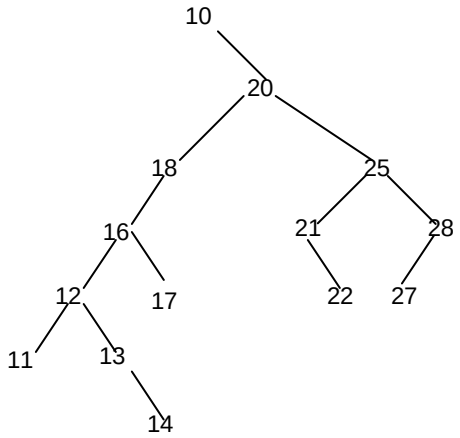
   $n,\ 1.001^n, \log n^n,\ n^2, \log^2 n, 1000000,\ n^{1/2}.$
3. (15 points) What is the asymptotic worst time performance (Big Oh) for each of the following code segments? Please note: no justification is asked and no partial credit for wrong answers.

   a)
   ```
   int sum=0;
   for (int i =1; i <= n; i++)
       for (int j = 0; j < m; j++)
         sum++;
   ```

   b)
   ```
   int sum=0;
   for (int i =1; i <= n; i *=2)
       for (int j = 0; j*j < n; j++)
         sum++;
   ```

   c)
   ```
   int sum=0;
   for (int i =1; i <= n; i *=2)
       for (int j = 0; j < i; j++)
         sum++;
   ```
4. (30 points) Lists can be used to represent sets. Let lists L1 and L2 represent two sets of n1 and n2 elements of the same type, respectively. Assume there is no duplicate within each set. You are asked to design an algorithm for set intersection L1 = L1 ∩ L2. Put it in another way, your code should keep all nodes on L1 which also appear on L2, and remove every node on L1 that does not appear on L2 (L1 will be changed after the operation).
   a) What implementation (ArrayList or LinkedList) you would choose for the lists? Justify your choice.
   b) Write pseudo code for this set intersection operation in your chosen implementation.
   c) What is the time performance of your code? Your answer should be given in Big-Oh of |L1| and |L2|.
   d) If both L1 and L2 are sorted, can you modify your code so that this operation can be done more efficiently? If your answer is yes, give the modified pseudo code and its time performance. If your answer is no, explain why.

5. (10 points) Write a recursive function in pseudo code that returns the total number of internal nodes in a binary tree.
6. (7 points) Suppose the result of the post-order traversal of a BST is 1, 3, 2, 8, 10, 5, 4. Draw this BST.
7. (8 points) Consider the following BST



   a) Draw the new tree after deleting key 20.
   b) Draw the new tree after inserting key 20 back into the BST generated from a).
8. (10 points) Prove by induction that a full binary tree with n internal nodes has n + 1 leaves (external nodes)