```
(e).  k=0;
      for (i=1;i<N;i++)
         for (j=1;j<i*i;j++)
            if (j%i==0)
               for (z=0;z<j;z++)
                  k++
```

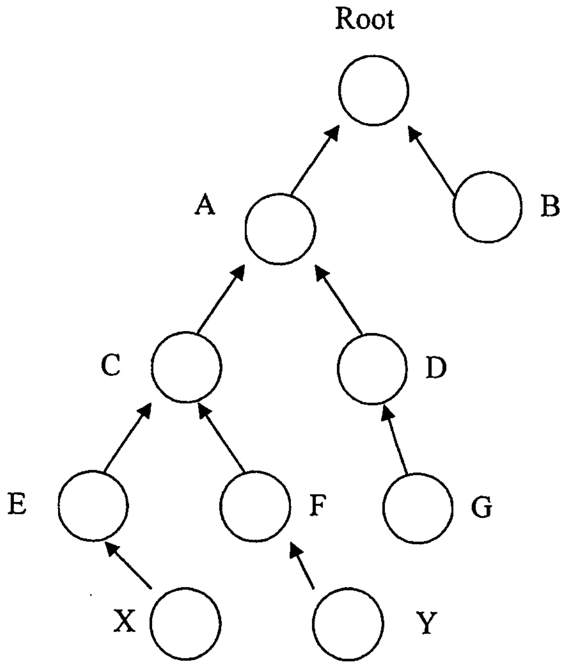
4. Answer the following two subproblems:

   (a) (5%) Show the results of inserting 2, 1, 4, 5, 9, 6, 7 into an initially empty (1) binary search tree, (2) AVL tree.

   (b) (10%) Give a precise expression for the minimum number of nodes in an AVL tree with its height H. Derive a recursive function of N(H), where N(H) denotes the minimum number of nodes for an AVL tree of an height H. What is the minimum number of nodes in an AVL tree of height 10?

5. (10%) Consider the following procedure **Mystery**, whose input is a node x in a rooted tree. Here each node in the tree has a pointer pointing to its parent except the root which points to itself. Consider the following input what will be the output and the rooted tree after executing **Mystery(Y)**. Draw the tree.

```
Procedure Mystery(x){
   if (x is not equal to Parent[x])
      then Parent[x] ← Mystery(Parent[x]);
   return(Parent[x]);   }
```



6. (15%) Given an N × N positive matrix R (i.e., each entry R[I,J] is positive) design an efficient algorithm to determine whether or not there exists a sequence of

＊＊作答前請先核對試題、答案卷(試卷)與准考證之所組別與考科是否相符！

distinct indices:I_1, I_2, ...I_k, where 1<=k<=N, such that $R[I_1, I_2] \times R[I_2, I_3] \times .... \times R[I_{k-1}, I_k] \times R[I_k, I_1] > 1$. State your algorithm precisely and analyze the running time of your algorithm.

7. (7%) Briefly describe a Huffman coding algorithm and illustrate it with the compression of the string ABADCDABEBAAABCABEBCBADAAFBAACAA.

8. (18%) Design a data structure, called `hashtable`, which requires the following operations

   (1) `find(k)`: returns the element with key `k`, where `k` is an integer.
   Let `k` be less than a sufficiently large prime number p for simplicity.

   (2) `Insert(e,k)`: insert the element `e` with key `k`.

   Please design the data structure, `hashtable`, such that

   (a) the averaged time complexity for `find(k)` is O(1) for *all* keys (even if the keys are well selected),

   (b) the amortized time complexity for `Insert(e,k)` is O(1), by assuming that the time complexity for each `find(k)` is O(1) and that each memory allocation (e.g., malloc) function call takes O(1), and

   (c) the space complexity is always O(n) if there are n elements in `hashtable`.

   Please also explain why (a), (b), and (c) are satisfied in your data structure. (Hint: consider the design of `java.util.Hashtable` in Java's utility library.)

國立交通大學 96 學年度碩士班考試入學試題

科目：資料結構與演算法(1001)　　考試日期：96 年 3 月 17 日 第 1 節

系所班別：資訊學院聯招　組別：資訊聯招　第 1 頁, 共 3 頁

＊＊作答前請先核對試題、答案卷(試卷)與准考證之所組別與考科是否相符！

1. Answer the following three subproblems:

   (a) (5%) Consider using the following pseudocode of quicksort to sort the following number sequence. Please show numbers which have been selected as pivots in increasing order.

   26 55 40 45 60 10 20 50 25 30 15 24 5 22

```
function Qsort(class List q)
  class List less, pivotList, greater
  if length(q) ≤ 1
    return q
  select the first element of q as the pivot
  for each x in q except the pivot element
    if x < pivot then add x into less
    if x ≥ pivot then add x into greater
  add pivot to pivotList
  return concatenate(Qsort(less), pivotList, Qsort(greater))
```

   (b) (5%) Please derive the best-case time complexity of quicksort.

   (c) (5%) Please derive the worst-case time complexity of quicksort.

2. (10%) Consider a tree **T** whose postorder and inorder sequences are as follows.

   Postorder: A, C, E, D, B, H, I, G, F.

   Inorder: A, B, C, D, E, F, G, H, I.

   Please show the preorder sequence of **T**.

3. (10%) Please derive the corresponding time complexity (Big-Oh) for each of the following six program segments.

```
(a).k=0;
    for (i=0;i<N;i++)
        k++;
(b). k=0;
     for (i=0;i<N;i++)
         for (j=0;j<N;j++)
             k++;
(c). k=0;
     for (i=0;i<N;i++)
         for (j=0;j<i;j++)
             k++;
(d). k=0;
     for (i=0;i<N;i++)
         for (j=0;j<i*i;j++)
             for (z=0;z<j;z++)
                 k++;
```