# Data structure
# Workshop 5

## I. Exercise

1.1 When storing a tree using a binary linked list, what does the right pointer of the root node point to?

1.2 Among the following storage forms, which one is NOT a storage form of a tree?
A. Parent representation
B. Child linked list representation
C. Child-sibling representation
D. Sequential storage representation

1.3 Suppose there are 199 nodes in a Huffman tree, then how many leaf nodes are there in this Huffman tree?

1.4 If X is a node with a left child in a binary in-order threaded tree and X is not the root, then the predecessor of X is ().
A. The parent of X
B. The leftmost node in the right subtree of X
C. The rightmost node in the left subtree of X
D. The rightmost leaf node in the left subtree of X

1.5 The purpose of introducing a binary threaded tree is ().
A. To speed up the search for the predecessor or successor of a node
B. To facilitate insertion and deletion in a binary tree
C. To find the parent conveniently
D. To make the traversal result of the binary tree unique

1.6 Let F be a forest and B be the binary tree transformed from F. If there are n non-leaf nodes in F, then the number of nodes with a null right pointer in B is ().

1.7 A Huffman tree is composed of n ($n \geq 2$) characters with distinct weights. Which of the following statements about this tree is incorrect?
A. There must be no nodes with degree 1 in the tree
B. The two nodes with the smallest weights in the tree must be sibling nodes
C. The weight of any non-leaf node in the tree must be no less than the weight of any node in the next layer
D. This tree must be a complete binary tree

1.8 Suppose the message for communication consists of only 8 letters, and the frequencies of the letters in the message are 0.07, 0.12, 0.20, 0.32, 0.16, 0.03, and 0.10 respectively.

(1) Try to design Huffman codes for these 8 letters.

(2) Try to design another coding scheme uses 3-bit binary representations (000 to 111).

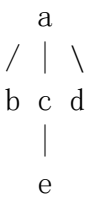
(3) Compare these the two schemes in terms of weighted path length.

## II. Experiment

**Problem Description:**

Input each edge of the tree from top to bottom, (1)establish the child-sibling linked list storage structure of the tree, (2)and find the depth of the tree.

For example, for the tree shown below, the input is **#a, ab, ac, ad, ce, ##**, where #a indicates that the root of the tree is a, and ## indicates the end of input.




**Input Format:**

A continuous character sequence in one line.

The first character is #, and the second character is the root of the tree (## indicates an empty tree);

Next, give xy in the order from top to bottom and left to right, where x is the parent node of y;

Finally, ## indicates the end of input.

The node value is a single character from a~z, without repetition.

**Output Format:**

Output the depth of the tree.