### 第二章 线性表

1. C ； 2. D ； 3. D
2. （1）p != NULL

（2）p = p->next

5. 输出单链表（或者将链表的所以值打印出来）

### 第三章 栈和队列

1. B ，D ；2 A ； 3 B； 4 A； 5 stack； 6 30 10 62 30；

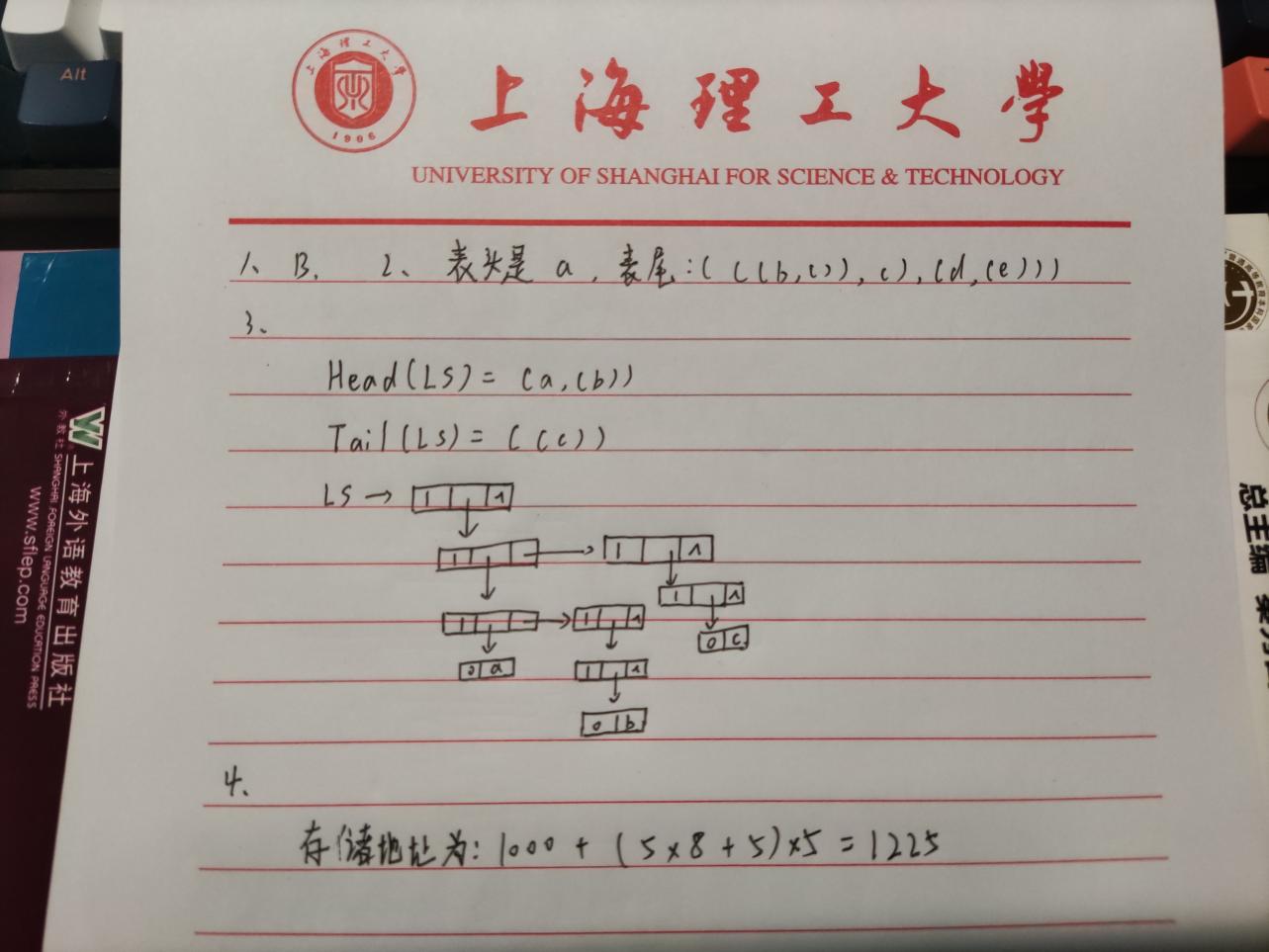
7.

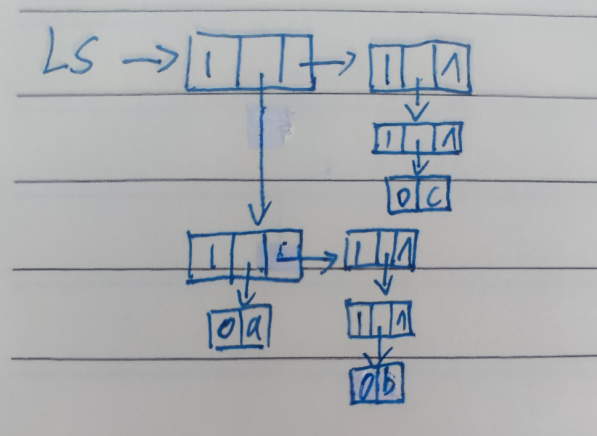
* 1  
  22  
  333  
  4444  
  55555

### 第五章 数组和广义表

1. B ；2. 表头是a, 表尾： (((b,( )),c),(d,(e)))

3. ((a,(b)),(c))

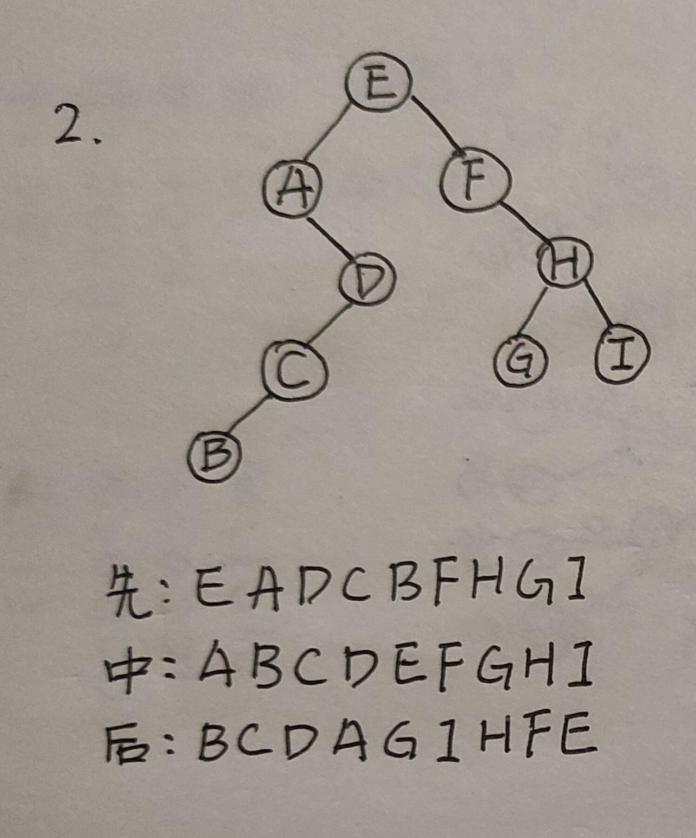




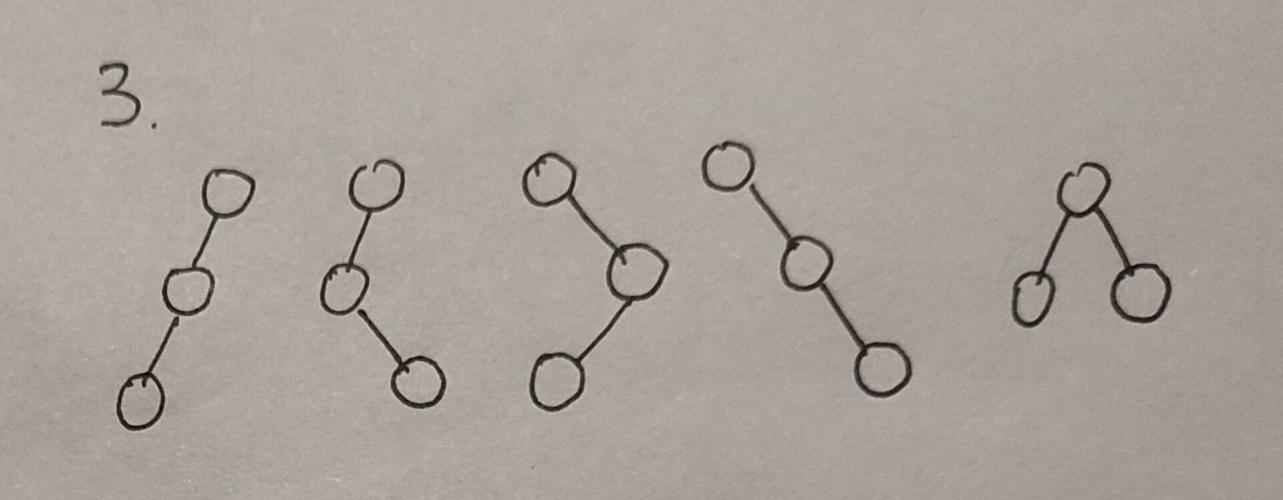
1. 1000+(5\*7+5)\*5=1200

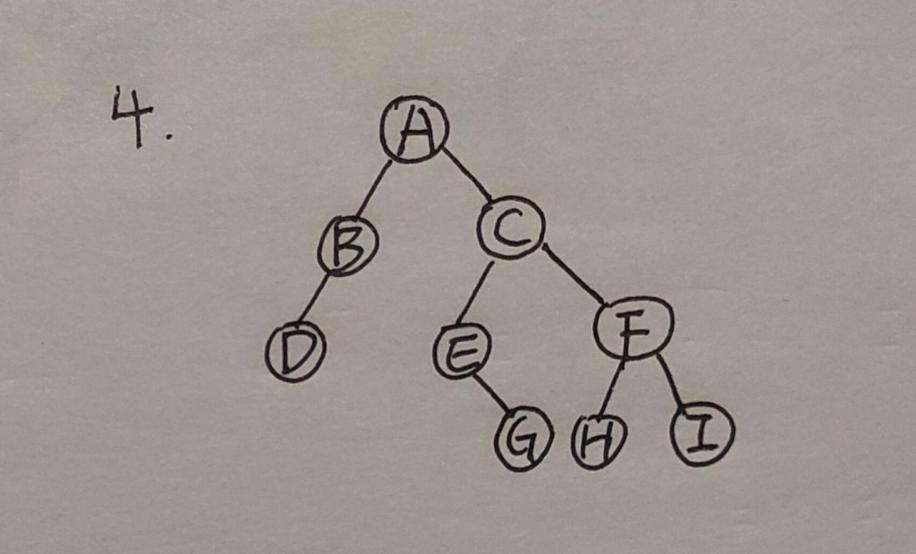
### 第六章 二叉树

1. 叶子节点数：



1. 5种

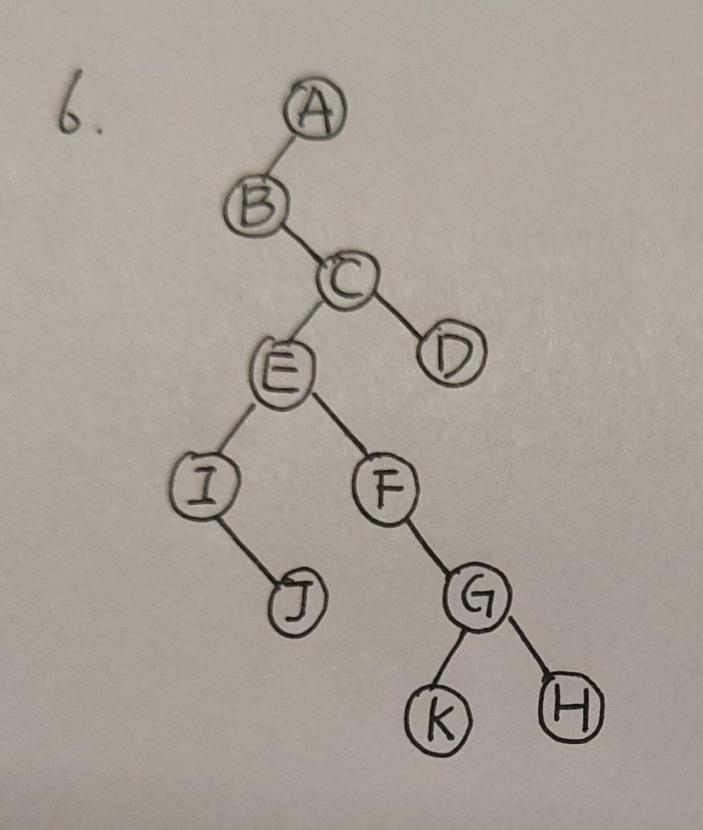
* 

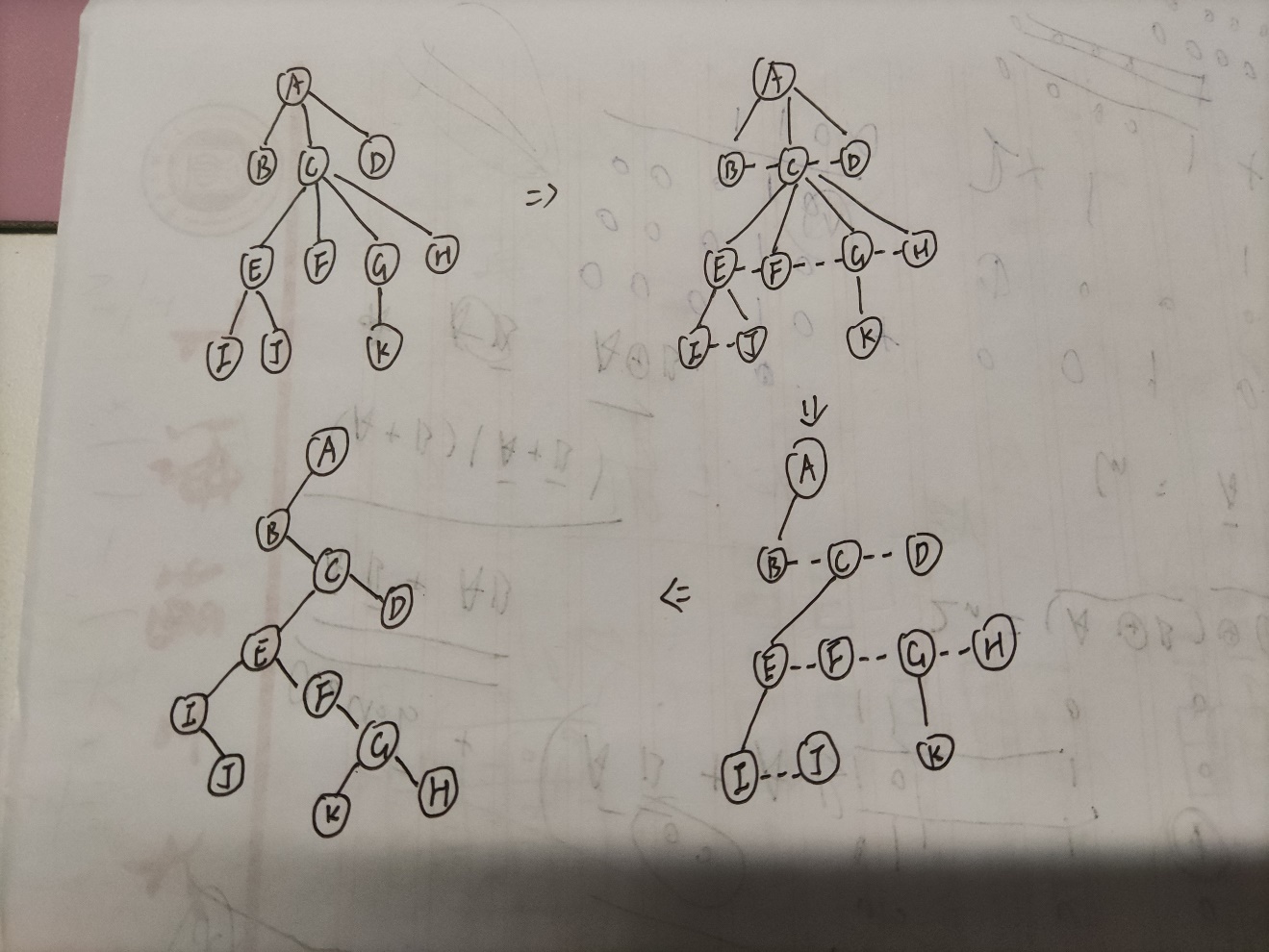
1. 
2. 原式：

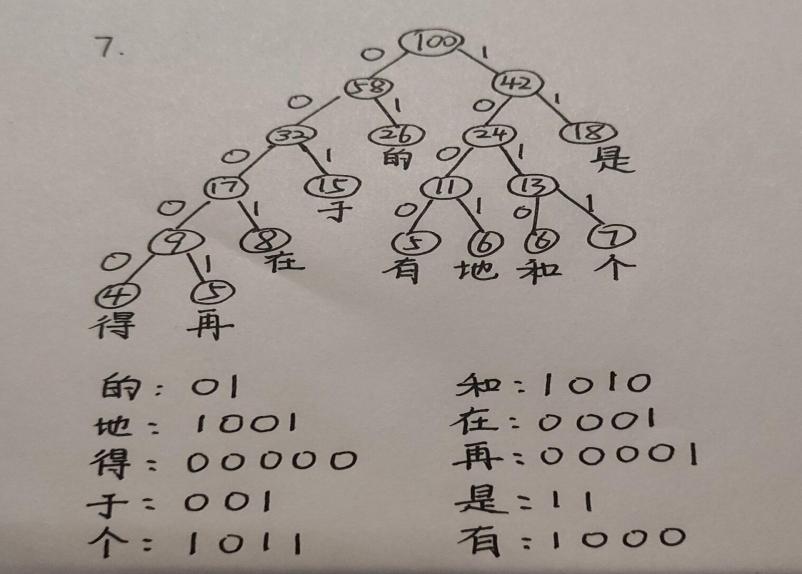
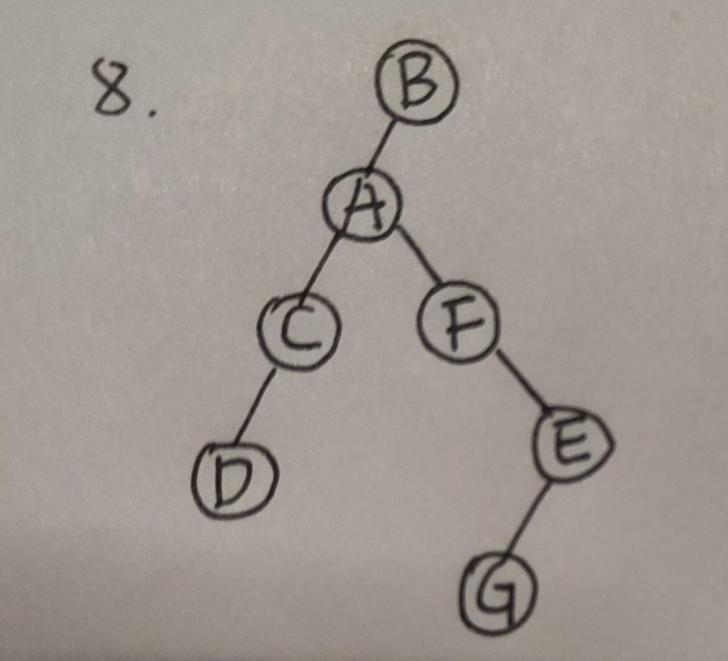
（1）波兰式：

（2）逆波兰式：

6.

1. 



1. 
2. 

* **定义：**
* **传入参数：**
* 二叉树的根指针：root
* **算法：**
  1. 判断root是否为空，不为为空，继续下面步骤，否则结束函数。
  2. 输出root节点的值。
  3. 对root的左子树，递归调用该函数
  4. 对root的右子树，递归调用该函数

//结构体  
typedef struct node  
{  
 char data;  
 struct node \*left\_child;  
 struct node \*right\_child;  
} BTnode;  
typedef BTnode \*BTptr;  
  
// 先序遍历  
void preOrder(BTptr root)  
{  
 if (root != NULL)  
 {  
 printf("%c", root->data);  
 preOrder(root->left\_child);  
 preOrder(root->right\_child);  
 }  
}