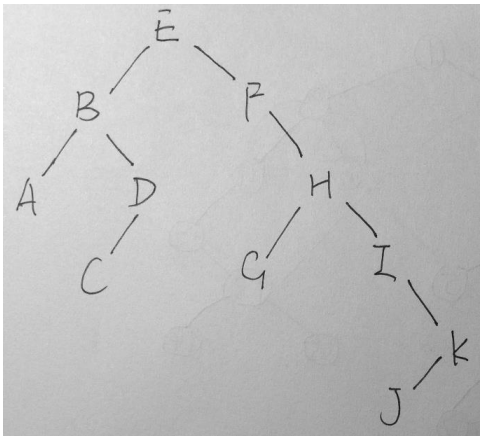
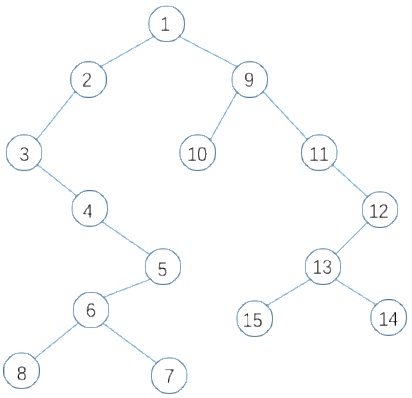
DS Homework 9

注：请使用 A4 纸作答，写上姓名学号，并于下一次上课时提交。

1、假设一棵二叉树的先序序列为 EBADCFHGIKJ 和中序序列为 ABCDEFGHIJK，请绘制出该树。



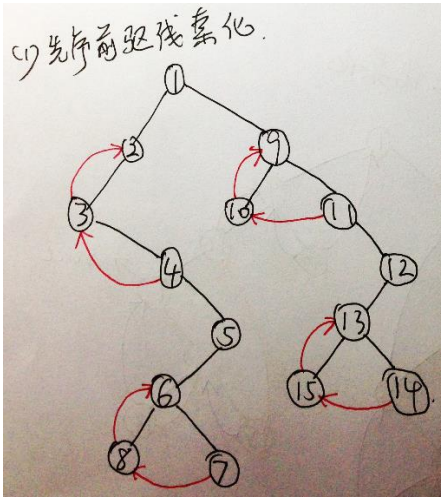
2、已知有二叉树：

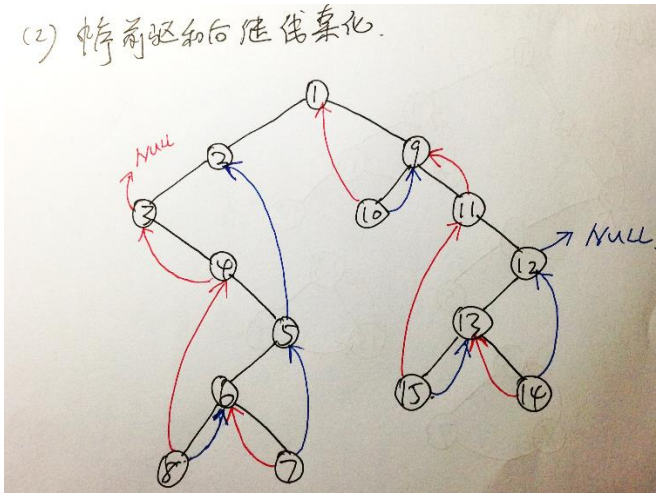


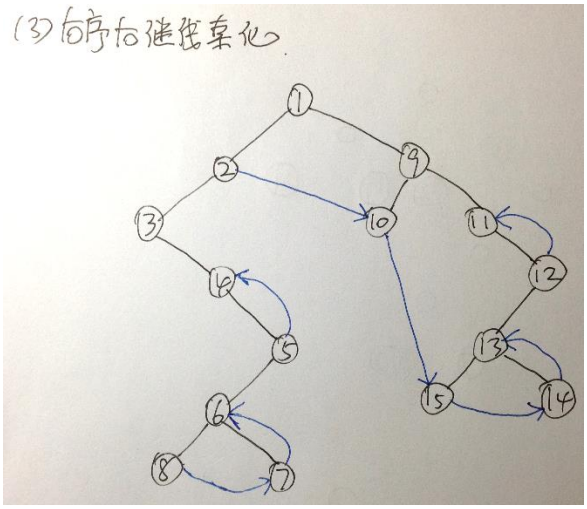
(1) 绘出先序前驱线索化示意图

(2) 绘出中序前驱和后继线索化示意图

(3) 绘出后序后继线索化示意图







3、在二叉树中查找值为 x 的结点，试设计输出值为 x 的结点的所有祖先的算法，假设值为 x 的结点不多于 1 个。

解答：采用后序遍历的非递归算法解决这一问题。注意到，当访问完左右子树后，保存在栈中的结点由栈顶及其祖先构成。因退栈时需区分其左右子树是否已遍历，因此在结点入栈的同时附带一个标志：0 表示是左子树，1 表示是右子树，用栈 stack 保存结点指针及其标志。具体实现算法如下：

```
void ancestor(BiTree  T, int x)
{
struct node
    {
       BiTree  p;
       int tag;      //取 0 或 1
    }s[100];
int top=0,i;
```

```
while(1)
{
      while(T)  //将 t 所指结点的所有左结点入栈
     {   top++;
     s[top].p=t;   s[top].tag=0;
     t=T->lchild;
}
```

```
if (top==0) return;   //栈空
      if (s[top].tag==0  &&  s[top].p->rchild)  //从左子树返回，进入右子树
      {    s[top].tag=1;
t=s[top].p->rchild;
}
else  //左右子树访问完，访问根结点
{   if (s[top].p->data==x)
{
     for(i=1;i<top;i++)
       printf("%d",s[i].p->data);
     return;
}
top--;  //出栈
}
}
}
```