

三. 若二叉树中有 n_1 个度为 1 的结点, n_2 个度为 2 的结点. 问此树最高可以达到多高? 若 $n_1 = 3, n_2 = 4$, 画出一棵这样的树. (10 分)

① $n_1 + n_2 + 1$

② $n_0 + n_1 + n_2 = n_1 + 2n_2 + 1$
 $n_0 = n_2 + 1$

四. 已知一棵二叉树的中序遍历序列为 C D A B E F H G I J, 后序遍历序列为 A B C D E F H G I J. 请画出这棵二叉树, 并将其转化为对应的森林. (10 分)

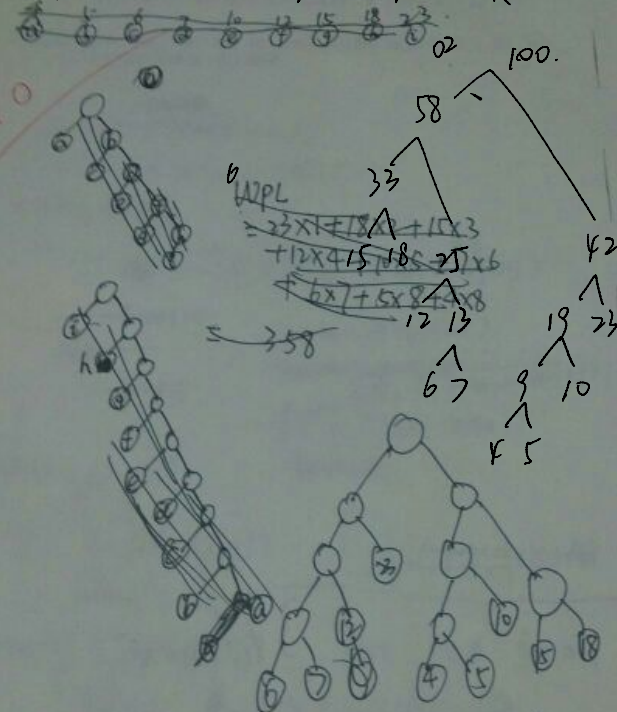
l r ldr ldr ldr ldr

二又树:

森林:

五. 对下面给出的数据序列, 构造一棵哈夫曼树, 并求出其带权路径长

成. ~~45 6 7 10 12 15 18 23~~ (10分) ~~9 12 15 23 33 42~~



$$WPL = 4 \times 6 + 7 \times 4 + 12 \times 3 + 23 \times 1 + 10 \times 2 + 4 \times 4 + 5 \times 4 + 15 \times 3 + 18 \times 3 = 299$$

六. 已知二叉链表表示的二叉树中有值为 e 、 e_1 、 e_2 三个结点，下面算法是判断 e 是否为 e_1 和 e_2 的共同祖先，请在空格处填上相应的语句或表达式。(10分)

```
typedef struct node{
    datatype data;
    node *lc, *rc;
} node, *bitptr;
```



```
int forefather(bitptr t, datatype e, datatype e1, datatype e2)
{
    f = p1 = p2 = NULL;
    search(t, f, e);

    S1;

    S2;

    if S3 return TRUE;
    return FALSE;
}
```

```
void search(bitptr t, bitptr &s, datatype e) //dlr.
```

```
{
    if ( S4 && !s )
    {
        if ( t->data == e ) s = t;
        S5;
        search(t->lc, s, e);
    }
}
```

Handwritten notes:
 - Above S4: t
 - Next to $t \rightarrow data == e$: $t \rightarrow data == e$
 - Next to S5: t

S1: $search(f, p_1, e)$

S2: $search(f, p_2, e)$

S3: $(p_1 \&\& p_2)$

S4: t

S5: $search(t \rightarrow rc, s, e);$

七. 已知矩阵 $A_{m \times n}$ 中, 从上到下、从左到右元素值从小到大, 有元素 x 一定存在于 A 中。给出算法不超过 $m+n$ 次比较找到元素 x 的下标 (15 分)

void search(int A[][50], int m, int n, int x, int &i, int &j)
// x 的下标由引用类型参数 i 和 j 返回

```

180
for (int i=0, j=n-1; i<m; i++)
{
    if (A[i][j] == x) return;
    else if (A[i][j] > x) j--;
    else if (A[i][j] < x) j++;
}

while (j < n)
{
    if (x < A[k][j])
    {
        break;
    }
    else
    {
        k++;
    }
}

while (j < n)
{
    if (x == A[i][j])
    {
        break;
    }
    else if (x > A[i][j])
    {
        i++;
    }
    else if (x < A[i][j])
    {
        j--;
    }
}

key

```

for (int i=0, j=n-1; i<m; i++)
{ if (A[i][j] == x) return;
else if (A[i][j] > x) j--;
else if (A[i][j] < x) j++;
}

while (j < n)
{ if (x < A[k][j])
break;
else
{ k++;
}
}

while (j < n)
{ if (x == A[i][j])
break;
else if (x > A[i][j])
{ i++;
}
else if (x < A[i][j])
{ j--;
}
}

key

八. 试编写算法, 对一棵以孩子—兄弟链表表示的树统计叶子节点的个

数。(15分)

```
typedef struct node{
    datatype data;
    node *child, *sibling;
} node, *bitptr;
```

int countleaf (bitptr t) //t为根节点, 函数返回值为叶子结点个数

```
{ int i int i = 0
  if (!t)
    return 0;
```

```
  else {
```

```
    if (t->child == NULL)
```

```
        i++;
```

```
    else t->sibling
```

```
        { countleaf(t->child);
```

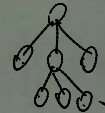
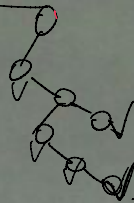
```
        i += countleaf(t->sibling);
```

```
    }
```

```
}
```

```
return i;
```

```
}
```



不是二叉树

是树