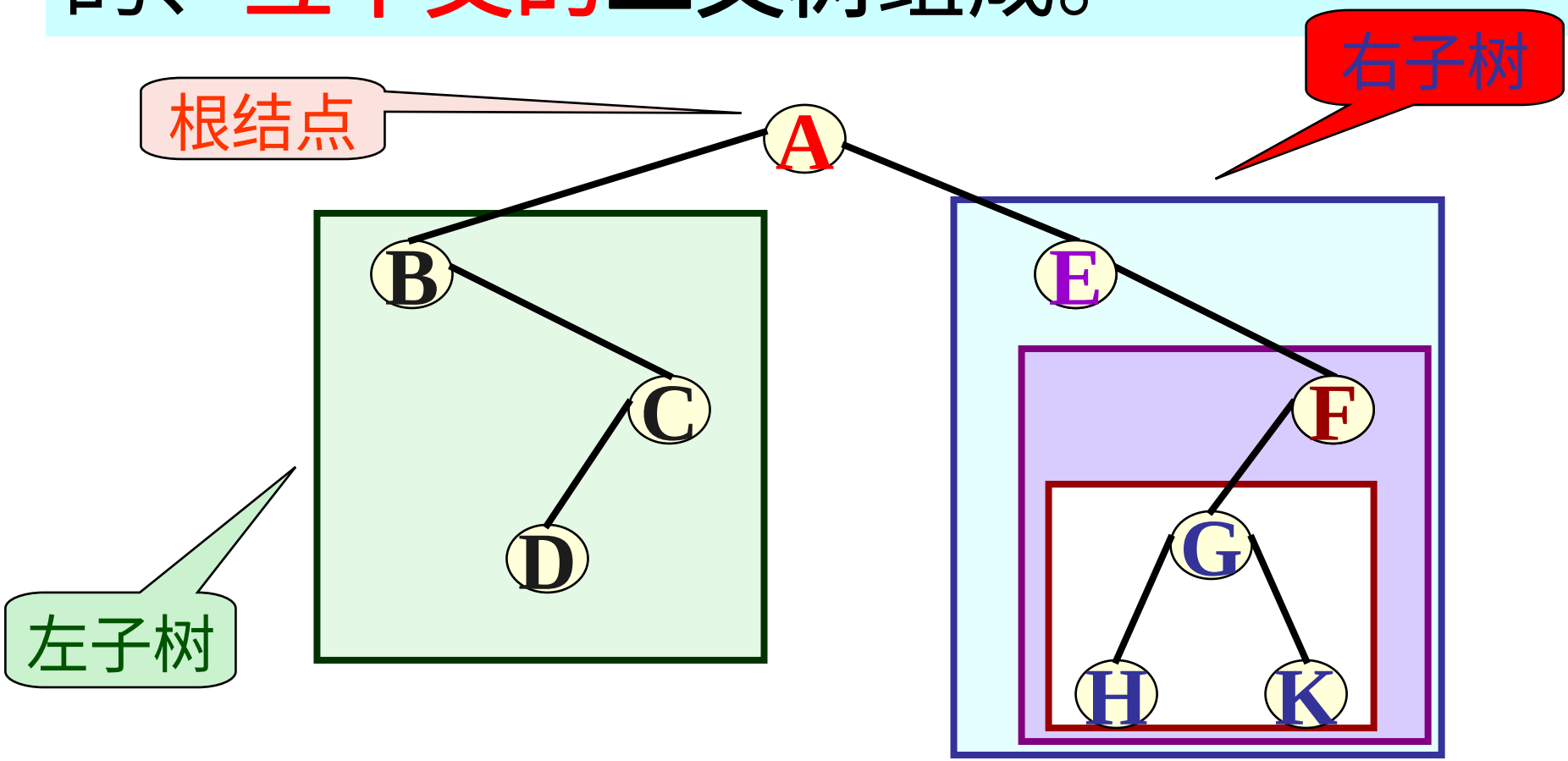
第六章 二叉树

二叉树或为空树；或是由一个根结点加上两棵分别称为左子树和右子树的、互不交的二叉树组成。



二叉树的五种基本形态：

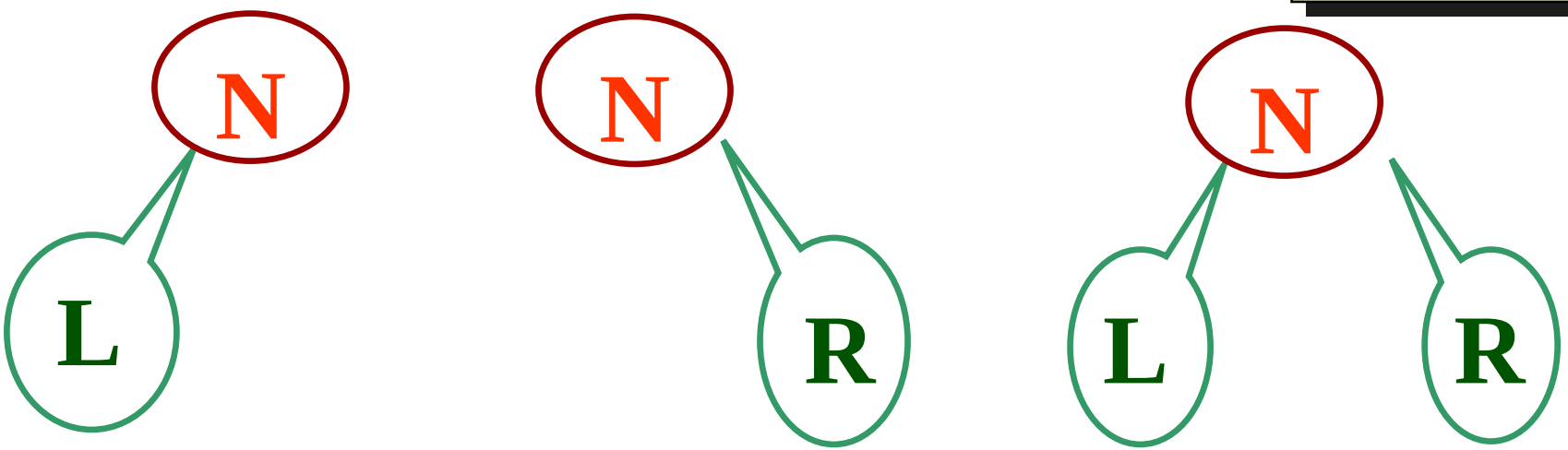
空树

只含根结点

左右子树均不为空树

右子树为空树

左子树为空树



二叉树的特性

- **性质 1 ：**

在二叉树的第 i 层上至多有 2^{i-1} 个结点。 (i≥1)

用归纳法证明：

归纳基： $i = 1$ 层时，只有一个根结点，

$$2^{i-1} = 2^0 = 1 ;$$

归纳假设：假设对所有的 j，$1 \le j < i$，命题成立;

归纳证明：二叉树上每个结点至多有两棵子树，

则第 i 层的结点数 $= 2^{i-2} \times 2 = 2^{i-1}$ 。

- 性质 2 ：

深度为 k 的二叉树上至多含 2^k-1 个结点（k≥1）

证明：

基于上一条性质，深度为 k 的二叉树上的结点数至多为

$$2^0+2^1+\cdots\cdots+2^{k-1}=2^k-1$$

性质 3 ：

对任何一棵二叉树，若它含有 n_0 个叶子结点、n_2 个度为 2 的结点，则必存在关系式：$n_0 = n_2+1$

证明：

设 二叉树上结点总数 $n = n_0 + n_1 + n_2$

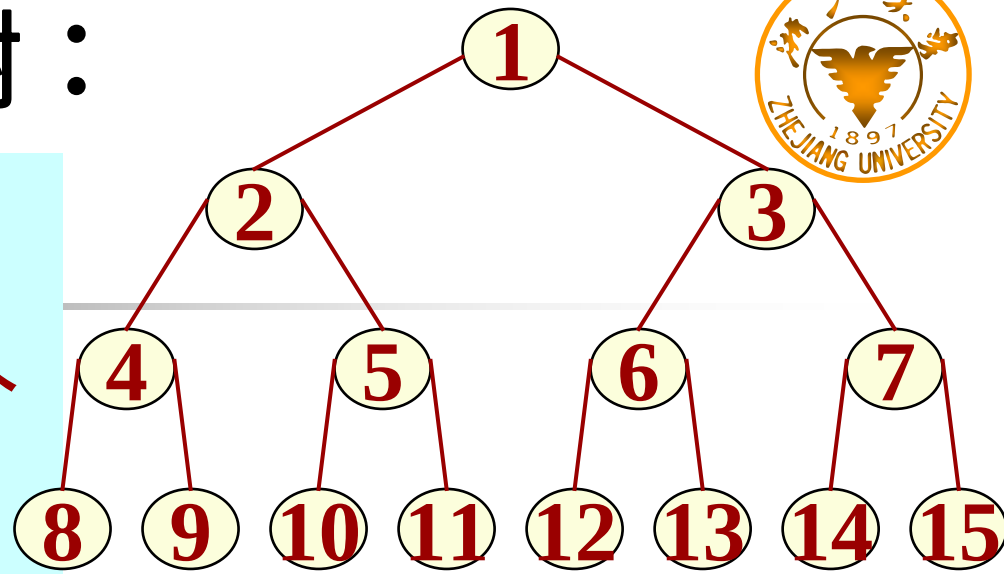
又 二叉树上分支总数 $b = n_1 + 2n_2$

而 $b = n-1 = n_0 + n_1 + n_2 - 1$

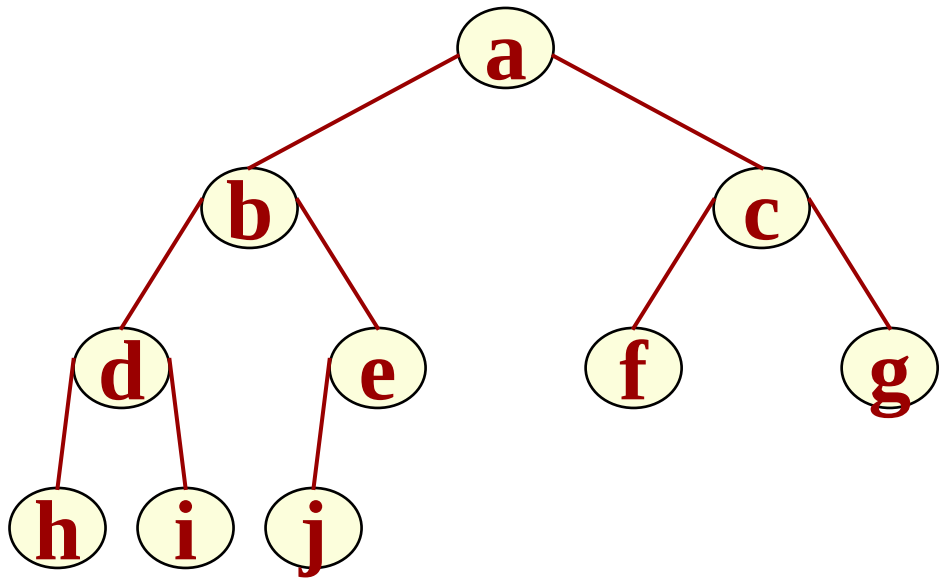
由此， $n_0 = n_2 + 1$

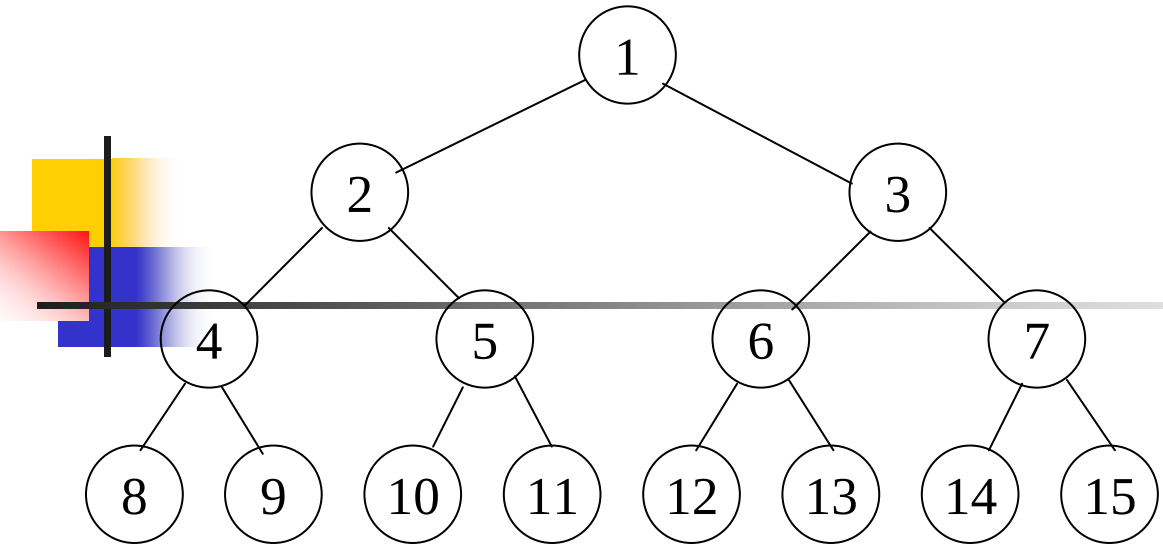
两类特殊的二叉树：

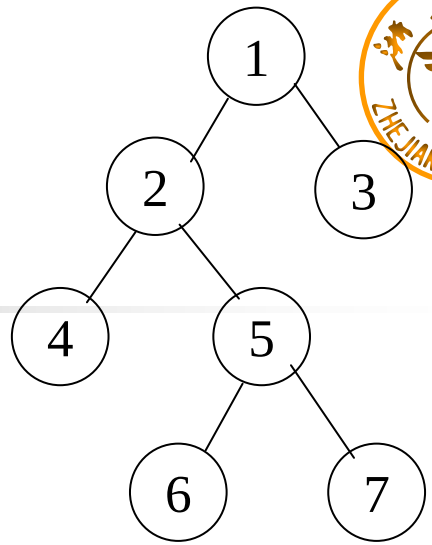
满二叉树：指的是深度为k且含有2^k-1个结点的二叉树。

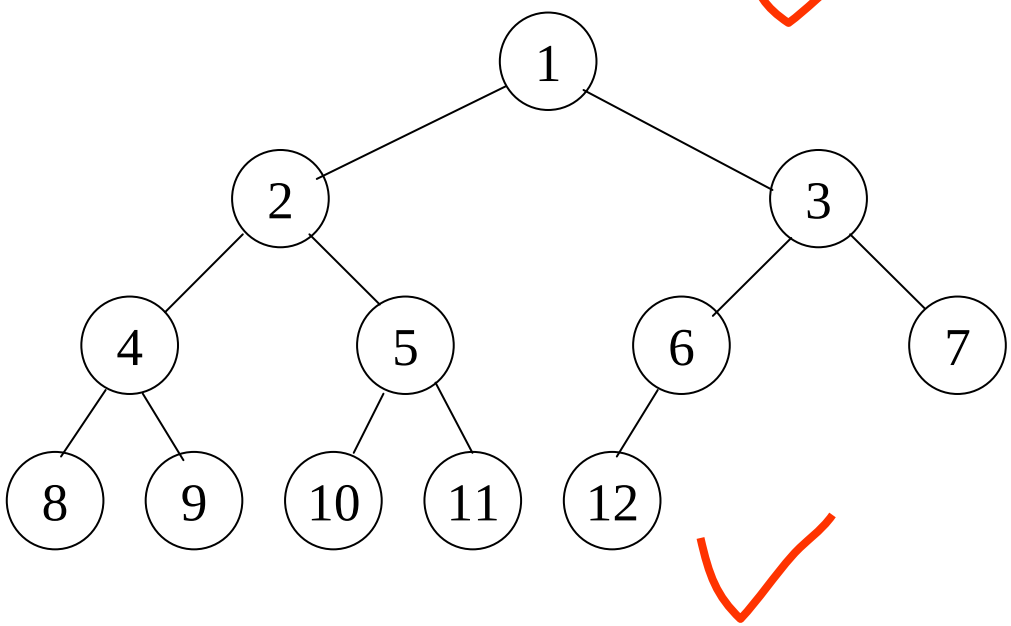


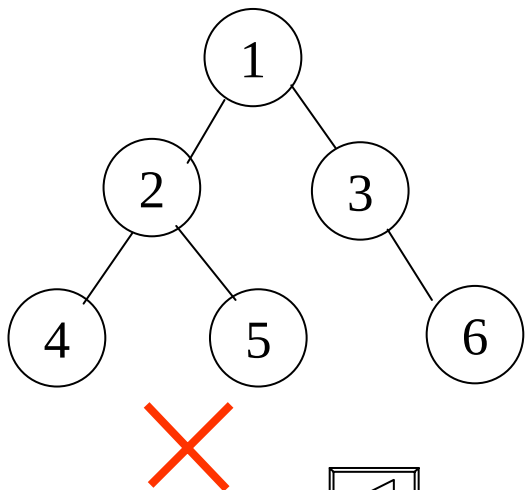
完全二叉树：树中所含的 n 个结点和满二叉树中编号为 1 至 n 的结点一一对应。




1
2
3
4
5
6
7
8
9
10
11
12
13
14
15


1
2
3
4
5
6
7


1
2
3
4
5
6
7
8
9
10
11
12


1
2
3
4
5
6

- **性质 4 ：**

具有 n 个结点的完全二叉树的**深度**为 $[\log_2 n]+1$

证明：

设 完全二叉树的深度为 k

则根据第二条性质得 $2^{k-1} \leq n < 2^k$

即 $k-1 \leq \log_2 n < k$

因为 k 只能是整数，因此， $k = \lfloor \log_2 n \rfloor + 1$

性质 5 ：

若对含 n 个结点的完全二叉树从上到下且从左至右进行 1 至 n 的编号，则对完全二叉树中任意一个编号为 i 的结点：

(1) 若 $i=1$，则该结点是二叉树的根，无双亲, 否则，编号为 $\lfloor i/2 \rfloor$ 的结点为其**双亲**结点;

(2) 若 $2i>n$，则该结点无左孩子，

否则，编号为 $2i$ 的结点为其**左孩子**结点；

(3) 若 $2i+1>n$，则该结点无右孩子结点，

否则，编号为 $2i+1$ 的结点为其**右孩子**结点。

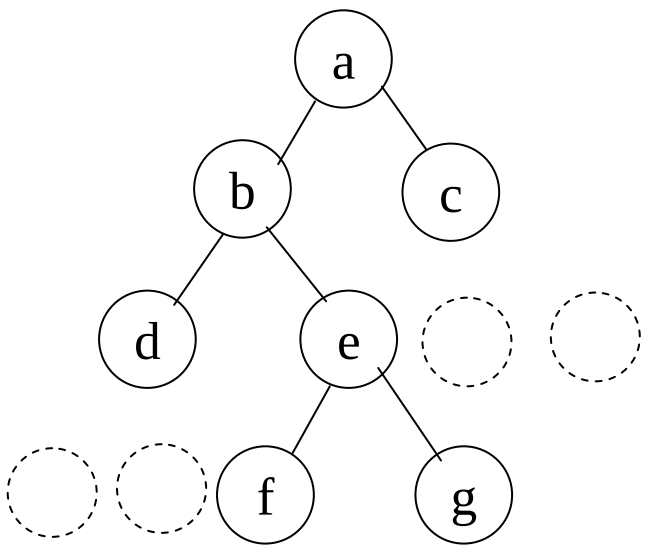
树的存储结构

1. 二叉树的顺序存储表示
2. 二叉树的链式存储表示

二叉树的顺序存储表示

1. 顺序存储结构

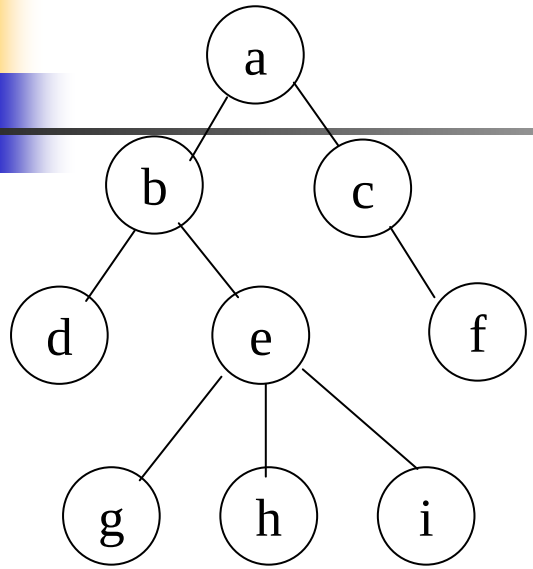
- 实现：按满二叉树的结点层次编号，依次存放二叉树中的数据元素
- 特点：
  - 结点间关系蕴含在其存储位置中
  - 浪费空间，适于存满二叉树和完全二叉树



1	2	3	4	5	6	7	8	9	10	11
a	b	c	d	e	0	0	0	0	f	g

2. 双亲表示法

- 实现：定义结构数组存放树的结点，每个结点含两个域：
  - 数据域：存放结点本身信息
  - 双亲域：指示本结点的双亲结点在数组中位置
- 特点：找双亲容易，找孩子难



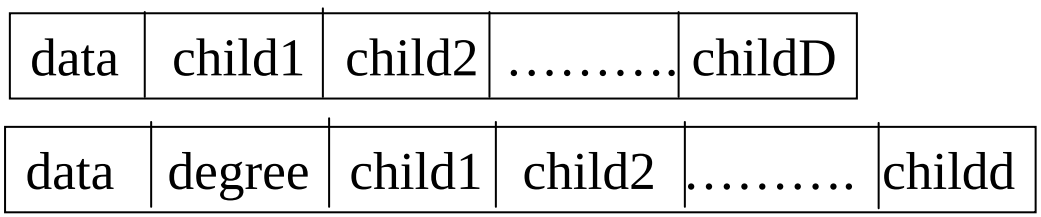
0号单元不用或存结点个数

	data	parent
0	0	9
1	a	0
2	b	1
3	c	1
4	d	2
5	e	2
6	f	3
7	g	5
8	h	5
9	i	5

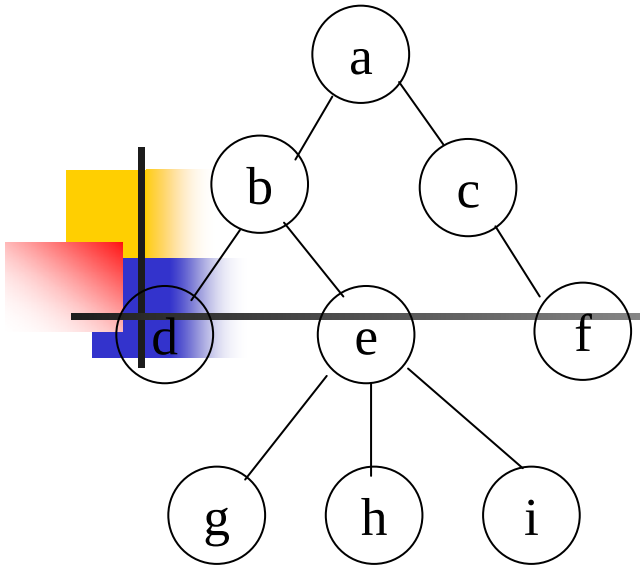
如何找孩子结点

3. 孩子表示法

- 多重链表：每个结点有多个指针域，分别指向其子树的根
  - 结点同构：结点的指针个数相等，为树的度D
  - 结点不同构：结点指针个数不等，为该结点的度d



- 孩子链表：每个结点的孩子结点用单链表存储，再用含n个元素的结构数组指向每个孩子链表

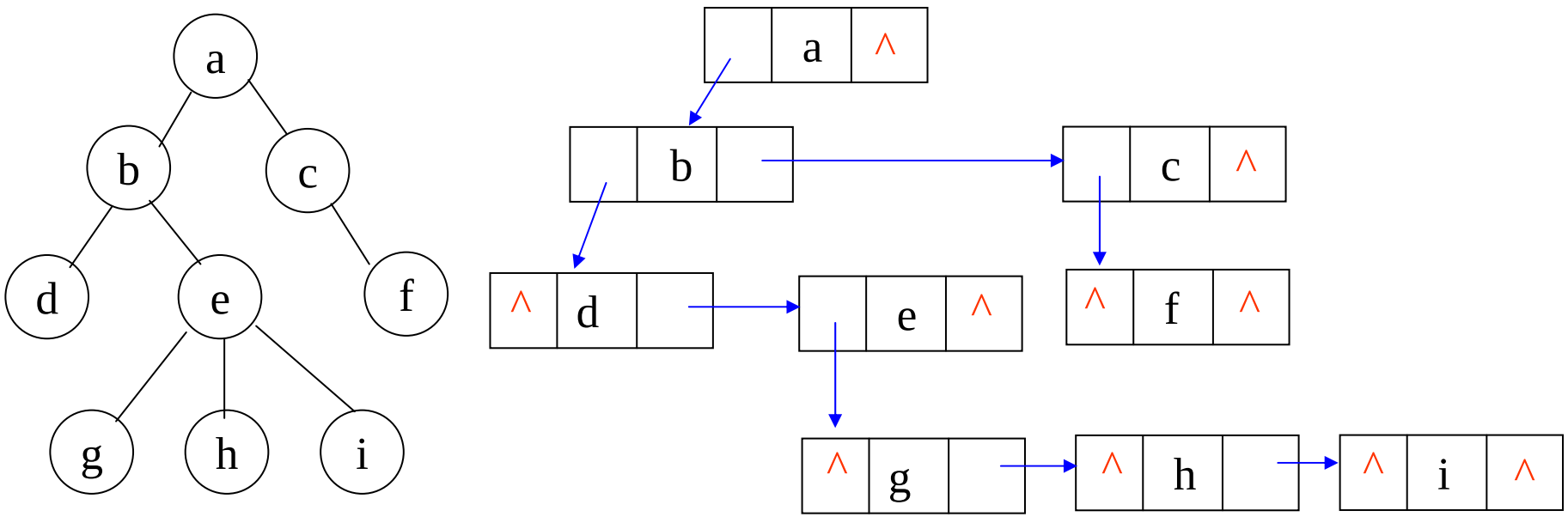


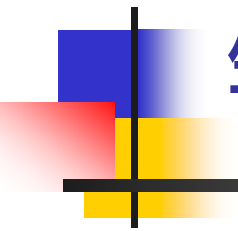
	data	fc
0		
1	a	
2	b	
3	c	
4	d	∧
5	e	
6	f	∧
7	g	∧
8	h	∧
9	i	∧

如何找双亲结点

4. 孩子兄弟表示法（二叉树表示法）

- 实现：用二叉链表作树的存储结构，链表中每个结点的两个指针域分别指向其第一个孩子结点和下一个兄弟结点
- 特点
  - 操作容易
  - 破坏了树的层次



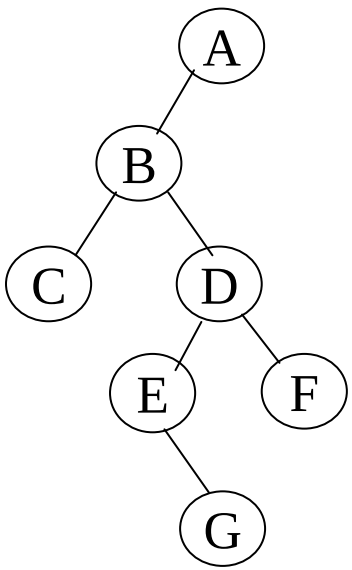


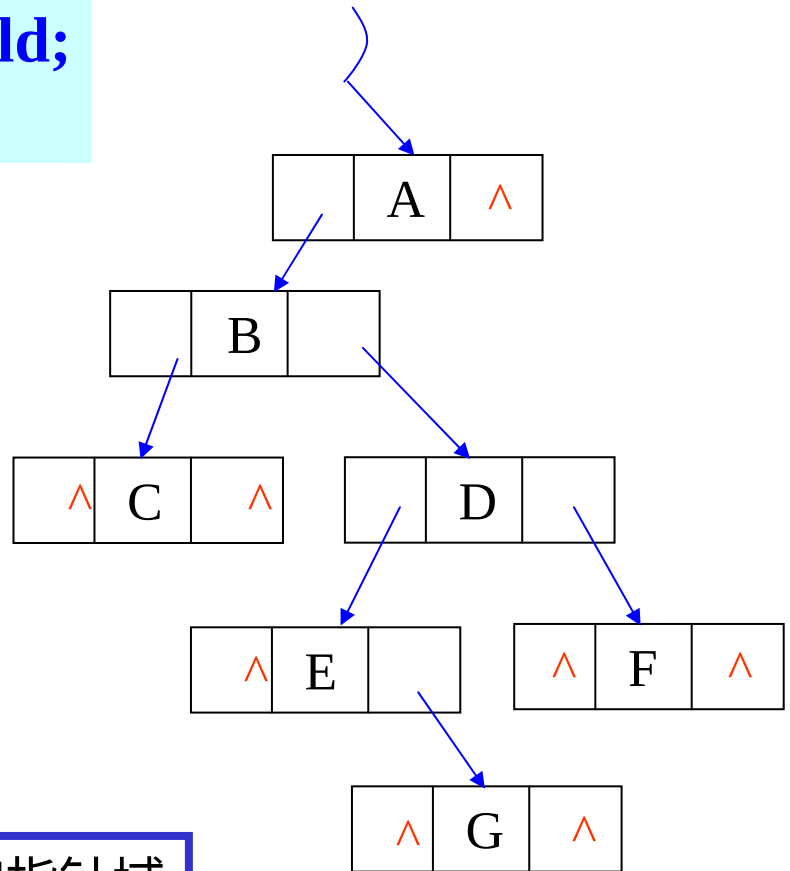
链式存储结构

1. 二叉链表

```
typedef  int datatype;
typedef    struct  node
{    datatype    data;
      struct  node  *lchild,  *rchild;
}bitree;
```

lchild	data	rchild



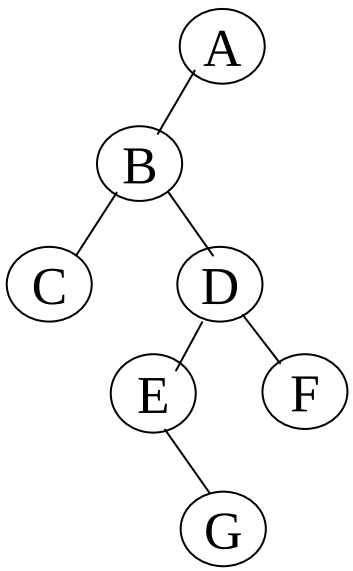


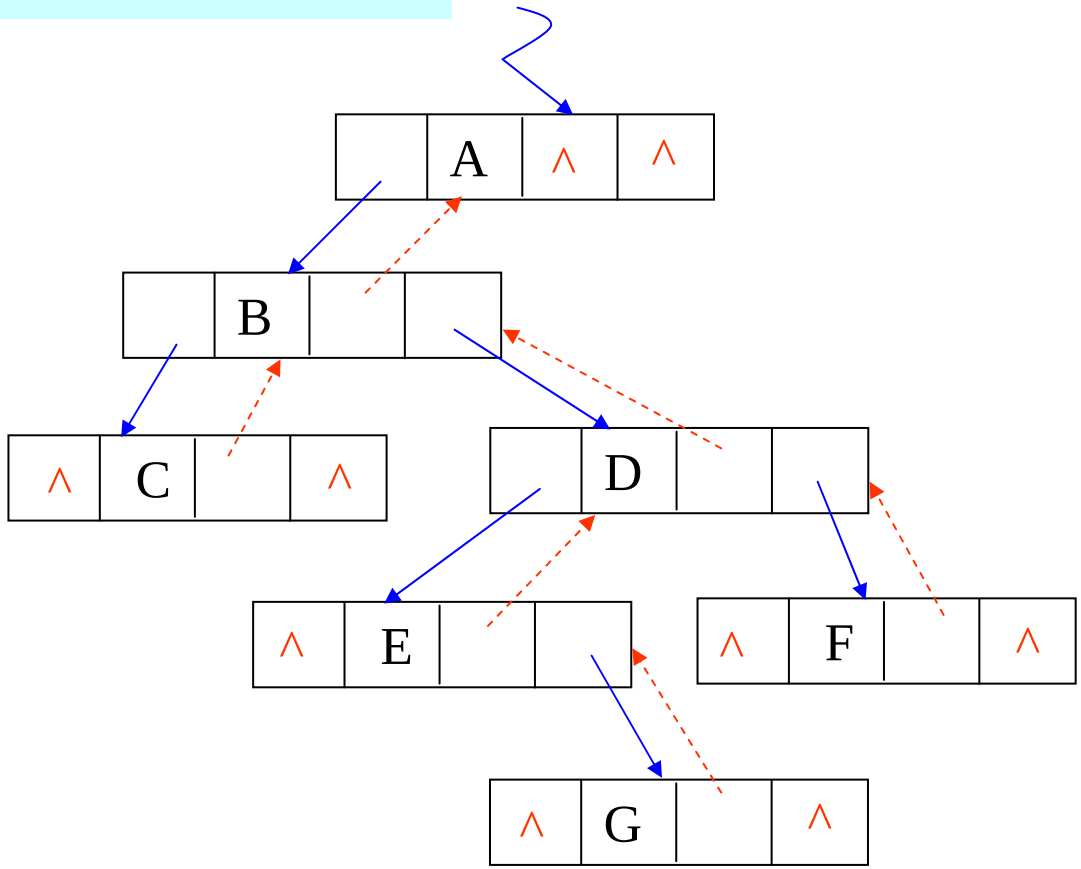
在n个结点的二叉链表中，有n+1个空指针域

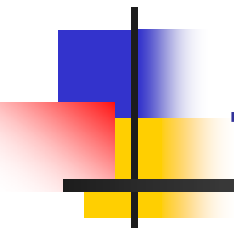
2. 三叉链表

```
typedef  struct  node
{  datatype  data;
     struct  node  *lchild,  *rchild,  *parent;
}JD;
```

lchild	data	parent	rchild







二叉树的建立

❖ 二叉树的建立

二叉树的建立是指在内存中建立二叉树存储结构。

二叉树的顺序存储结构的建立比较简单，只需将二叉树各个结点的信息（值）按原有的逻辑关系送入相应的向量单元中即可。

二叉树链式存储结构的建立算法有多种。下面介绍按完全二叉树的层次顺序，依次输入结点信息建立二叉链表的算法。对于一般的二叉树，必须添加若干个虚结点使其成为完全二叉树。

算法的基本思想

依次输入结点信息，若输入的结点不是虚结点，则建立一个新结点。若新结点是第一个结点，则令其为根结点，否则将新结点作为孩子链接到它的双亲结点上。如此反复进行，直到输入结束标志“#”为止。为了使新结点能够与双亲结点正确相连，并考虑到这种方法中先建立的结点其孩子结点也一定先建立的特点，可以设置一个指针类型的数组构成的队列来保存已输入结点的地址，并使队尾（rear）指向当前输入的结点，队头 （FRONT）指向这个结点的双亲结点。

```
class bitree
{   int   data;
    struct  node  *lchild,  *rchild;
};
```

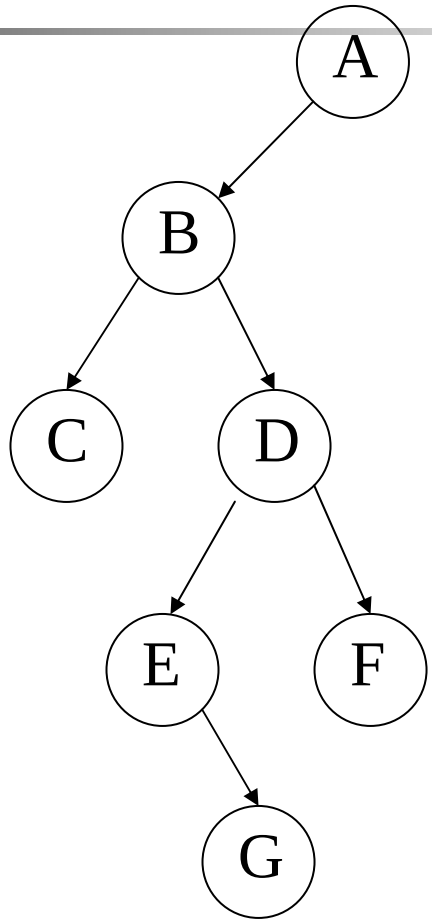
具体算法如下：

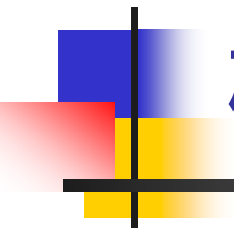
```
bitree *Q[maxsize];
bitree *CREATREE( )
{ char ch;
  int front, rear;     /* 队头和队尾指针 */
  bitree *root, *s;
  root=NULL;   /* 置空二叉树 */
  front =1;  rear = 0;  /* 设置队列指针变量初值*/
  while(ch =getchar()!='#')     /*输入一个字符，
                  当不是结束符时执行以下操作*/
```

```
{s = NULL;
 if (ch!='@')     /* @表示虚结点 */
 {                /* 不是虚结点时建立新结点 */
   s = new bitree;
   s->data=ch;
   s->lchild=NULL;
   s->rchild=NULL;
 }
 rear++ ;         /* 队尾指针增1，指向新结点地址应存
                     放的单元*/
 Q[rear] = s ;    /* 将新结点地址或虚结点指针NULL
                     入队 */
```

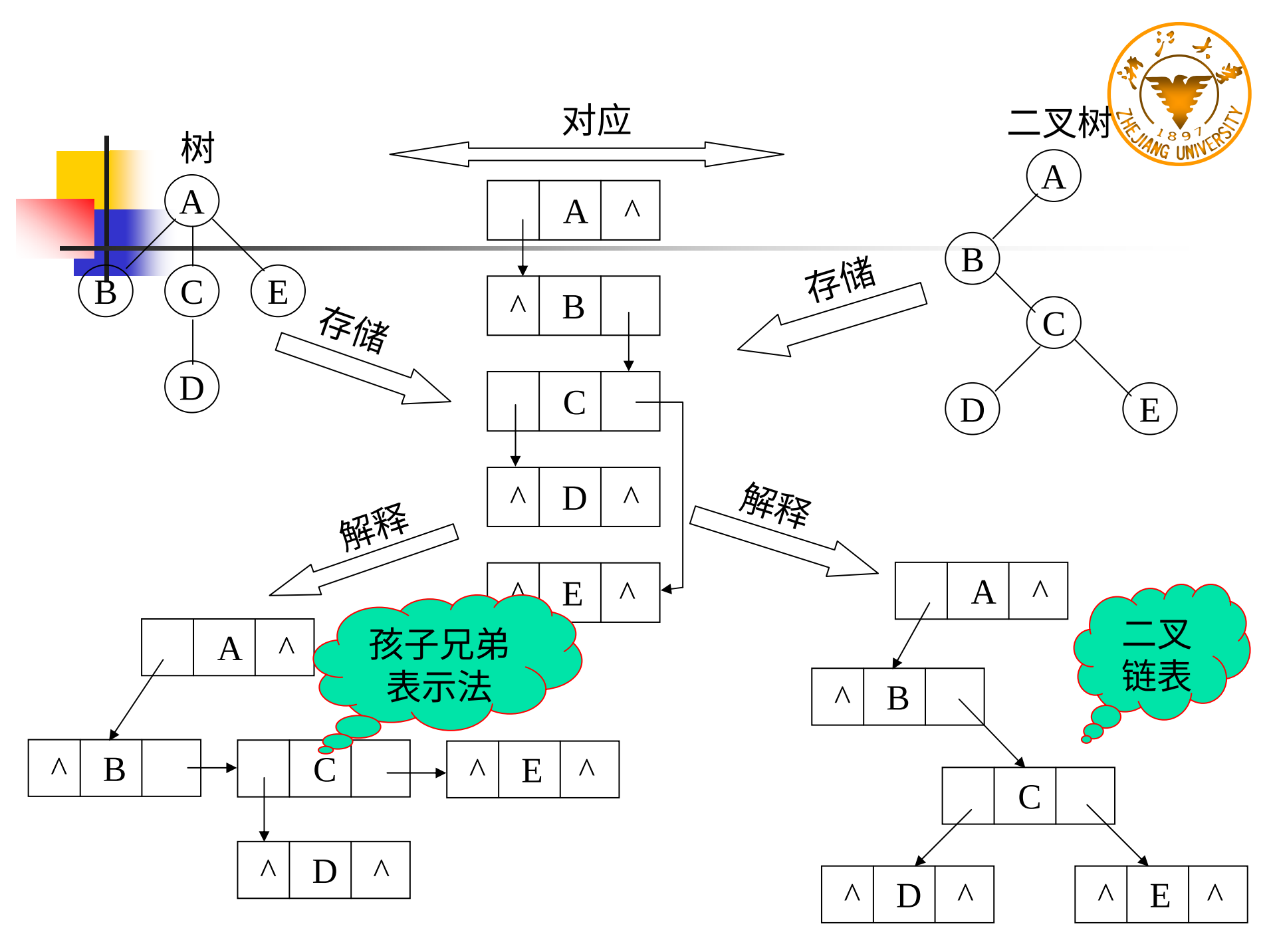
```
    if(rear = = 1) root = s ;  /* 输入的第一个结点为根结点 */
    else
    { if (s && Q[front]) /* 孩子和双亲结点都不是虚结点时 */
          if (rear % 2 = = 0)  Q[front]->lchild = s ;
          else Q[front]->rchild = s ;
      if (rear % 2 = = 1) front + + ;
    }
}
  return root;
}
```

上机练习：以该二叉树为例，建立二叉链表。



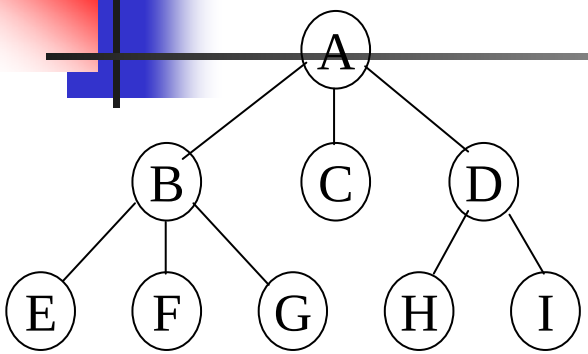


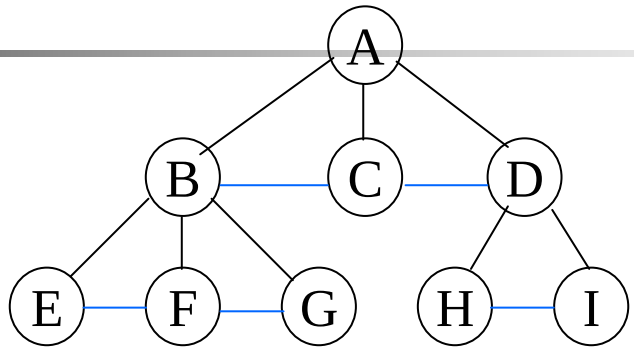
树与二叉树转换

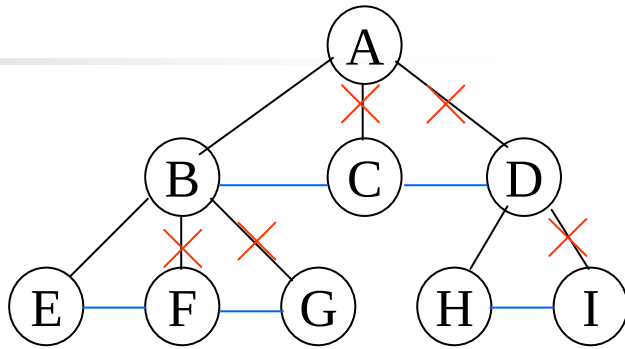

树
对应
二叉树
A
B
C
D
E
存储
存储
解释
解释
孩子兄弟
表示法
二叉
链表
^

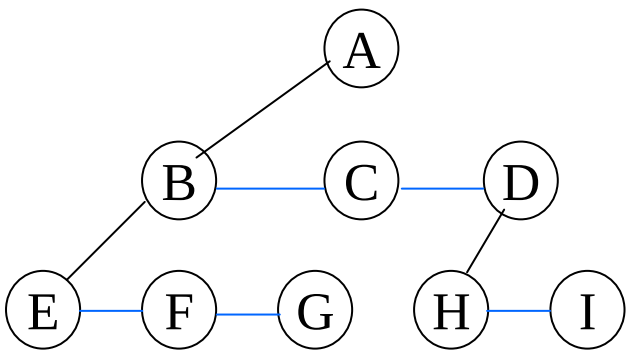
将树转换成二叉树

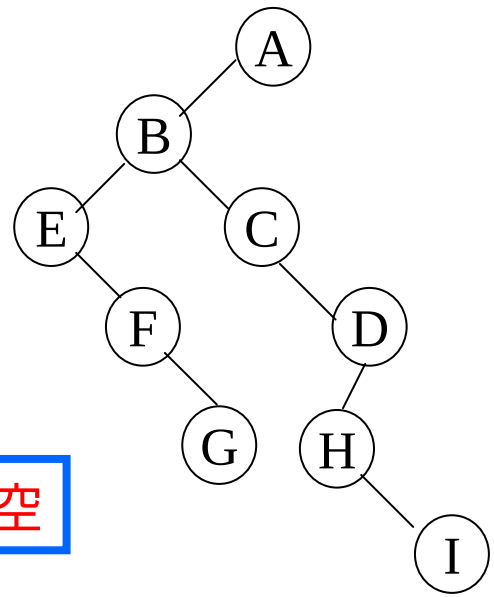
- 加线：在兄弟之间加一连线
- 抹线：对每个结点，除了其左孩子外，去除其与其余孩子之间的关系
- 旋转：以各子树的根结点为轴心，将水平结点顺时针转45°







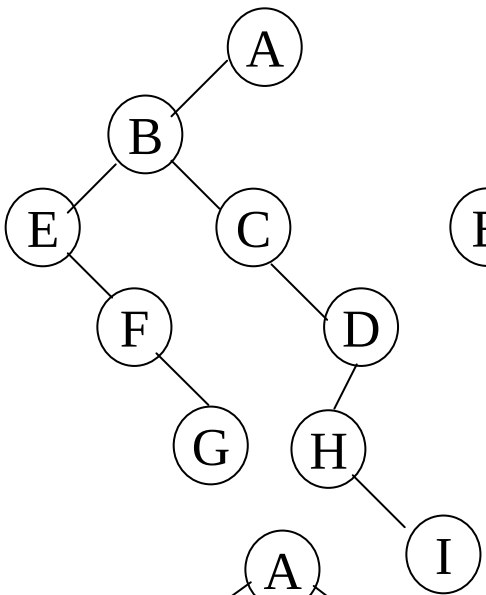


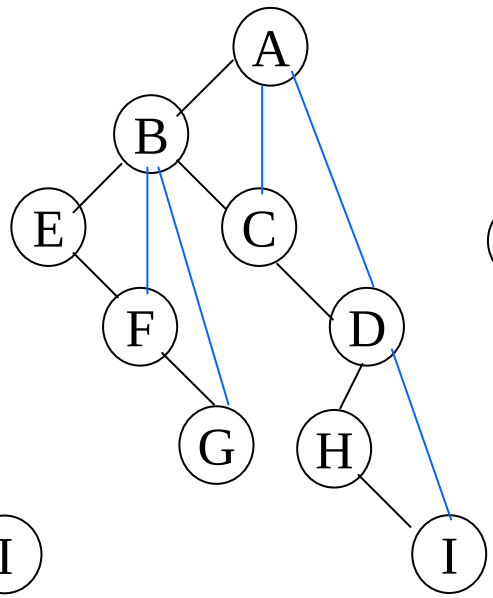


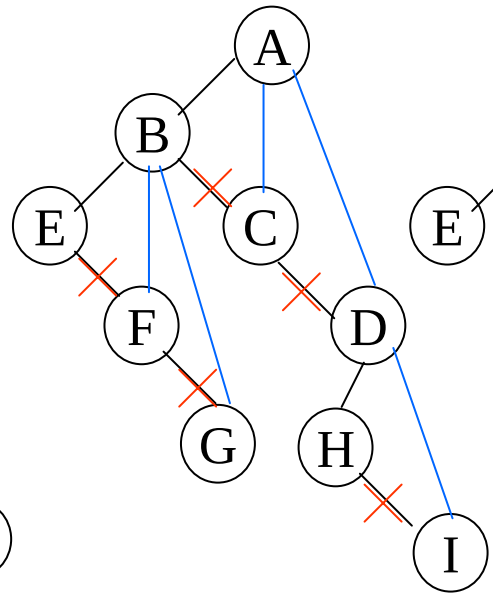
树转换成的二叉树其右子树一定为空

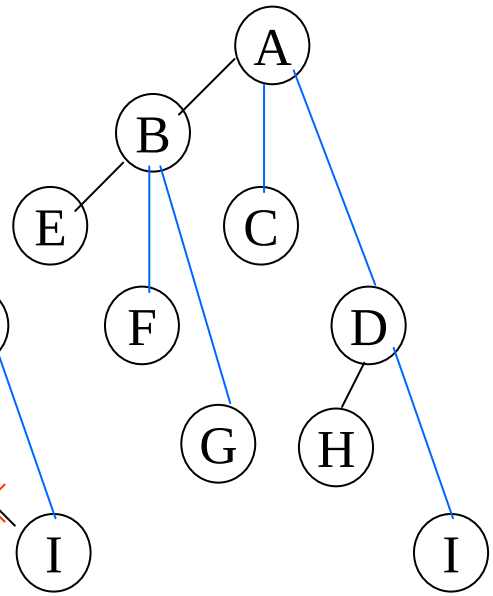
将二叉树转换成树

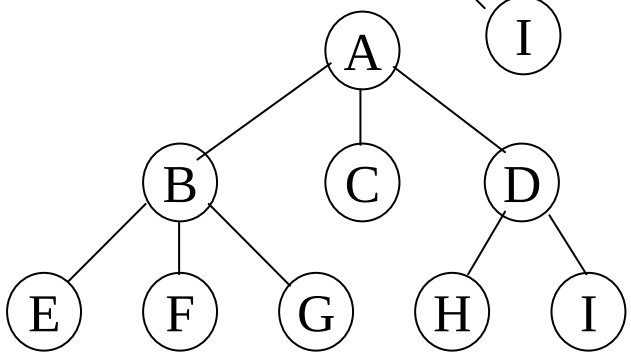
- 加线：若p结点是双亲结点的左孩子，则将p的右孩子，右孩子的右孩子，……沿分支找到的所有右孩子，都与p的双亲用线连起来
- 抹线：抹掉原二叉树中双亲与右孩子之间的连线
- 调整：将结点按层次排列，形成树结构

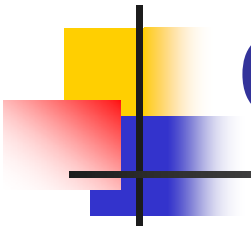

A
B
E
C
F
D
G
H
I


A
B
E
C
F
D
G
H
I


A
B
E
C
F
D
G
H
I


A
B
E
C
F
D
G
H
I


A
B
C
D
E
F
G
H
I



Q&A