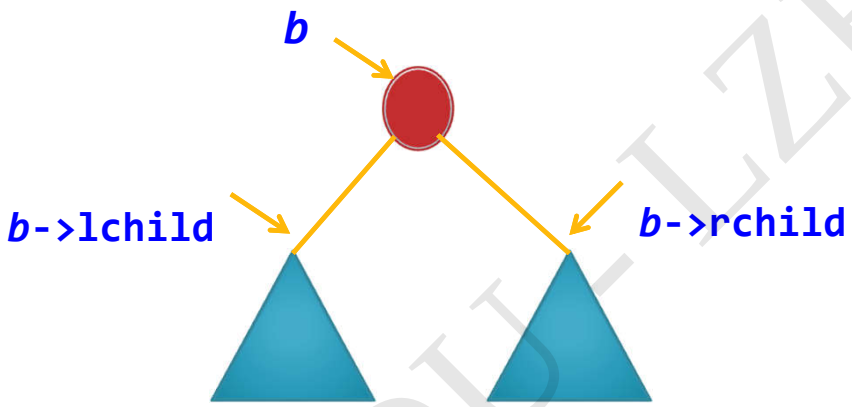
6.3 二叉树的遍历

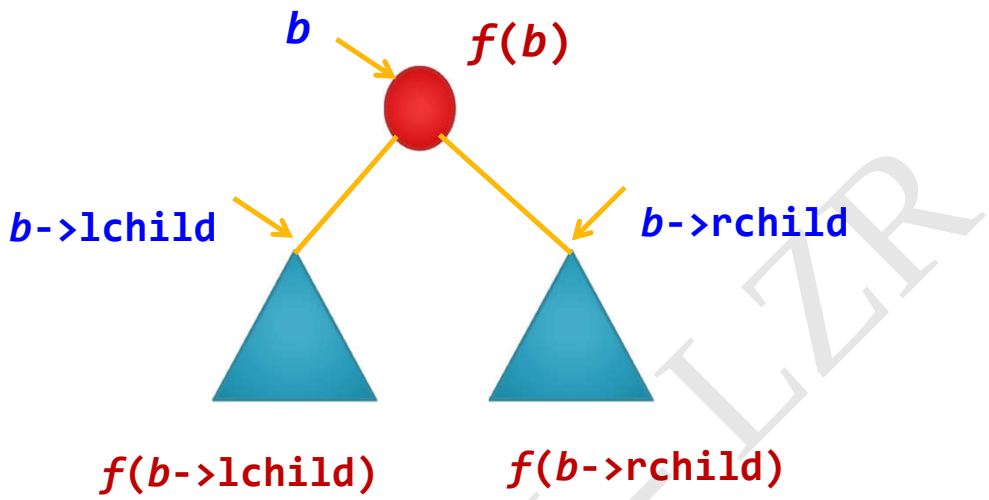
二叉树的递归算法设计思想

- 递归算法设计便是从递归数据结构的基本递归运算入手的。
- 对于二叉树，以二叉链表为存储结构，其基本递归运算就是求一个结点p的左子树（p->lchild）和右子树（p->rchild）。
- p->lchild和p->rchild一定是一棵二叉树。

一般地，二叉树（Binary Tree）的递归结构如下：



- b
- b->lchild
- b->rchild



- 对于二叉树b，设f(b)是求解的“大问题”。
- f(b->lchild)和f(b->rchild)为“小问题”。
- 假设f(b->lchild)和f(b->rchild)是可求的，在此基础上得出f(b)和f(b->lchild)、f(b->rchild)之间的关系，从而得到递归体。
- 再考虑b=NULL或只有一个结点的特殊情况，从而得到递归出口。

【示例-1】 假设二叉树中所有结点值为整数，采用二叉链表存储结构，求该二叉树*b*中所有结点值之和。

分析:

- 设*f*(*b*)为二叉树*b*中所有结点值之和，则*f*(*b*->lchild)和*f*(*b*->rchild)分别求根结点*b*的左、右子树的所有结点值之和。
- 显然有:

  f(*b*) = *b*->data + *f*(*b*->lchild) + *f*(*b*->rchild)

- 当*b*=NULL时*f*(*b*)=0。

递归模型

$f(b)=0$ 当b=NULL

$f(b)=b\text{->data}+f(b\text{->lchild})+f(b\text{->rchild})$ 其他情况

```
int Sum(BiTNode *b)
{
   if (b==NULL)
      return 0;
   else
     return(b->data+Sum(b->lchild)+Sum(b->rchild));
}
```

【示例-2】试设计销毁二叉树b的算法。

销毁二叉树bt的递归模型$f(b)$如下：

$f(b)$ ≡ 不做任何事情　　当b=NULL
$f(b)$ ≡ f(b->lchild);f(b->rchild);free(b);　　其他情况

```
void DestroyBTree(BiTNode *&b)
{   if (b!=NULL)
    {   DestroyBTree(b->lchild);
        DestroyBTree(b->rchild);
        free(b);
    }
    else
        return;
}
```

END