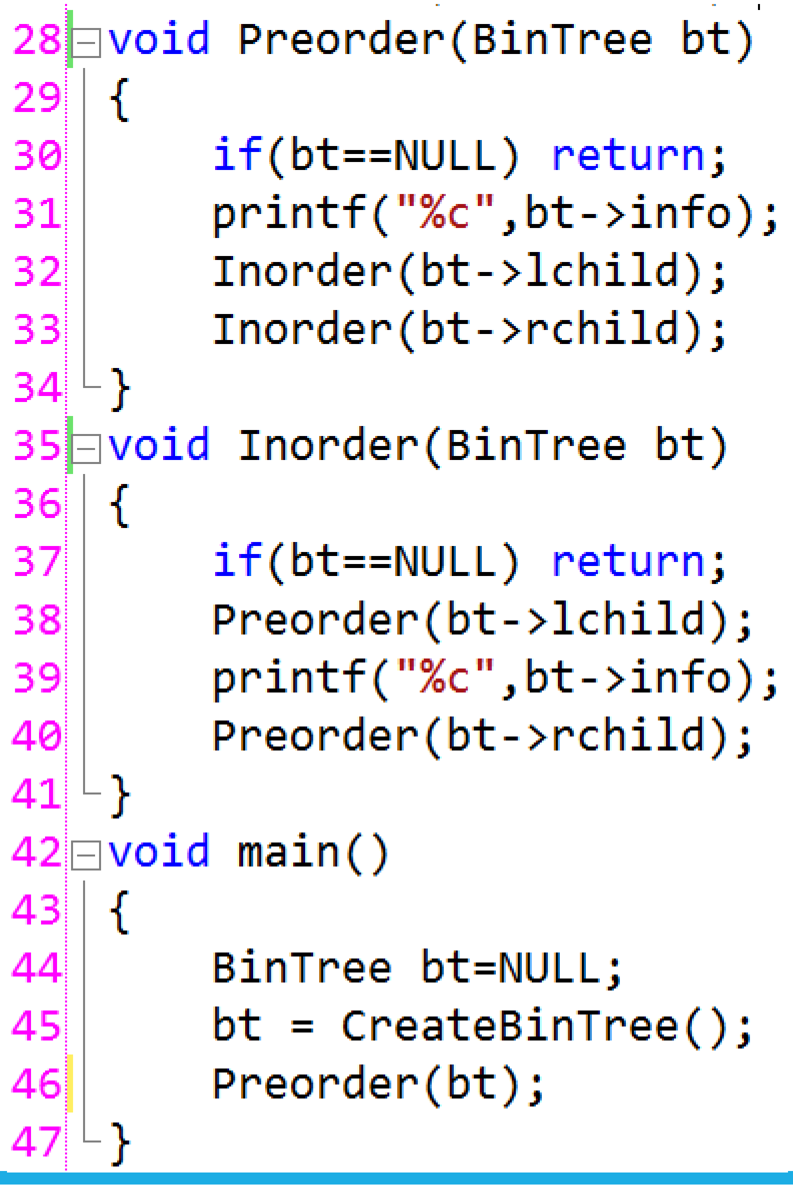
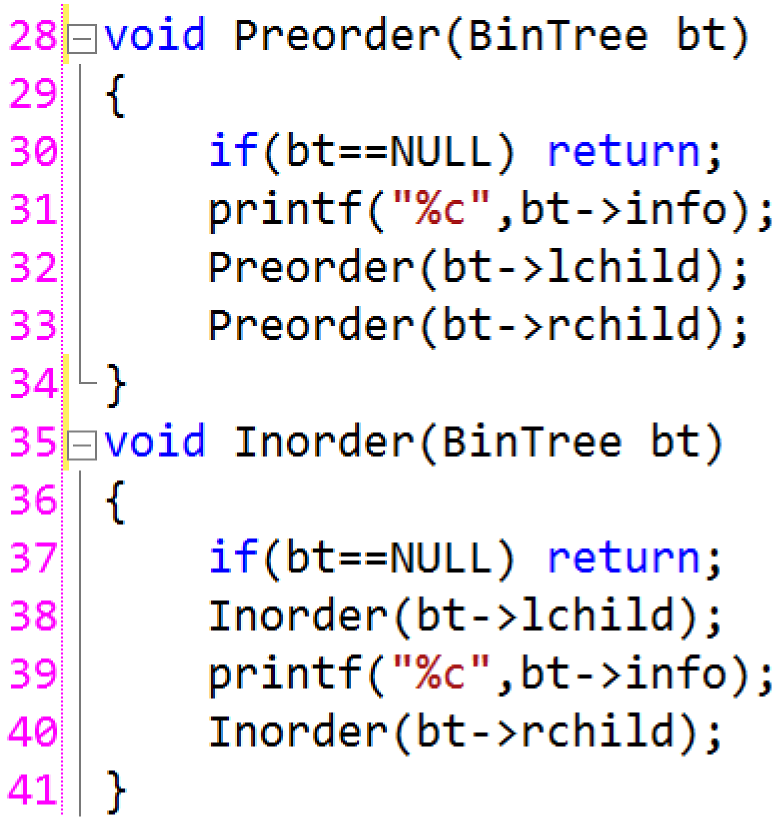
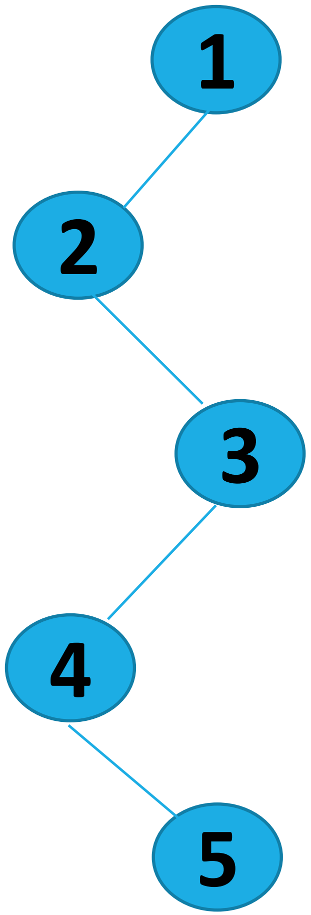
交叉遍历调用

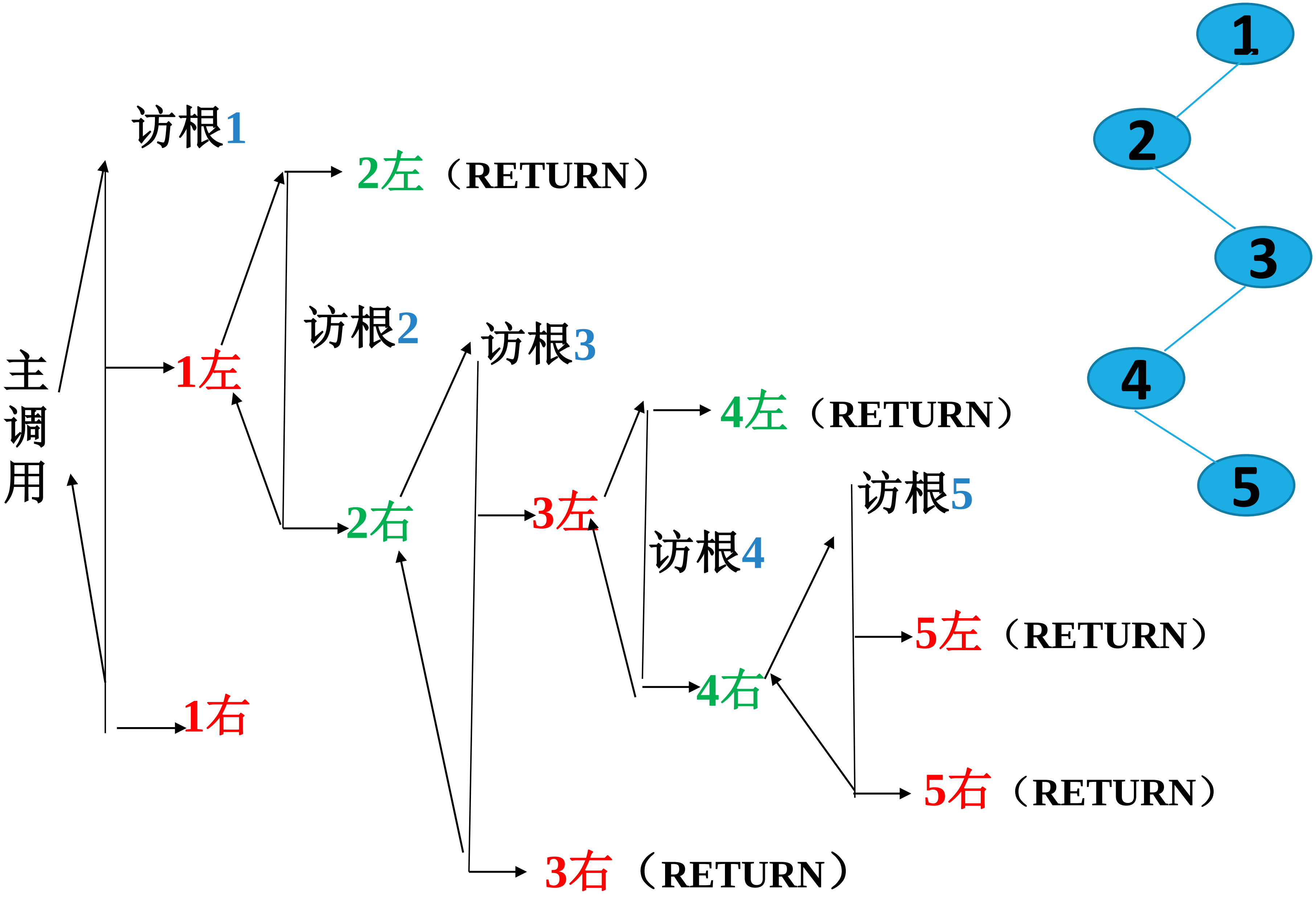


```c
void Preorder(BinTree bt)
{
	if(bt==NULL) return;
	printf("%c",bt->info);
	Inorder(bt->lchild);
	Inorder(bt->rchild);
}
void Inorder(BinTree bt)
{
	if(bt==NULL) return;
	Preorder(bt->lchild);
	printf("%c",bt->info);
	Preorder(bt->rchild);
}
void main()
{
	BinTree bt=NULL;
	bt = CreateBinTree();
	Preorder(bt);
}
```



```c
void Preorder(BinTree bt)
{
	if(bt==NULL) return;
	printf("%c",bt->info);
	Preorder(bt->lchild);
	Preorder(bt->rchild);
}
void Inorder(BinTree bt)
{
	if(bt==NULL) return;
	Inorder(bt->lchild);
	printf("%c",bt->info);
	Inorder(bt->rchild);
}
```





主调用

访根1

2左（RETURN）

1左

访根2

访根3

2右

4左（RETURN）

3左

访根5

访根4

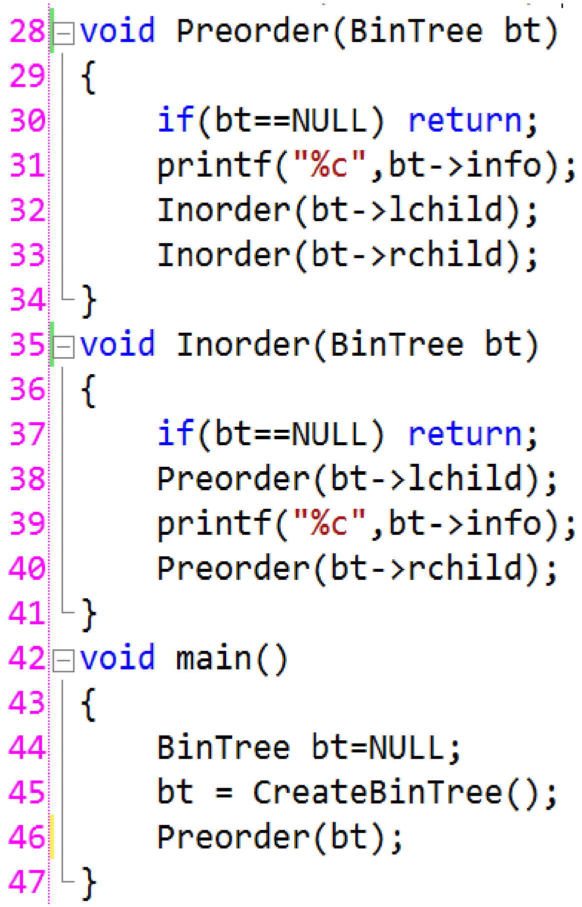
5左（RETURN）

4右

5右（RETURN）

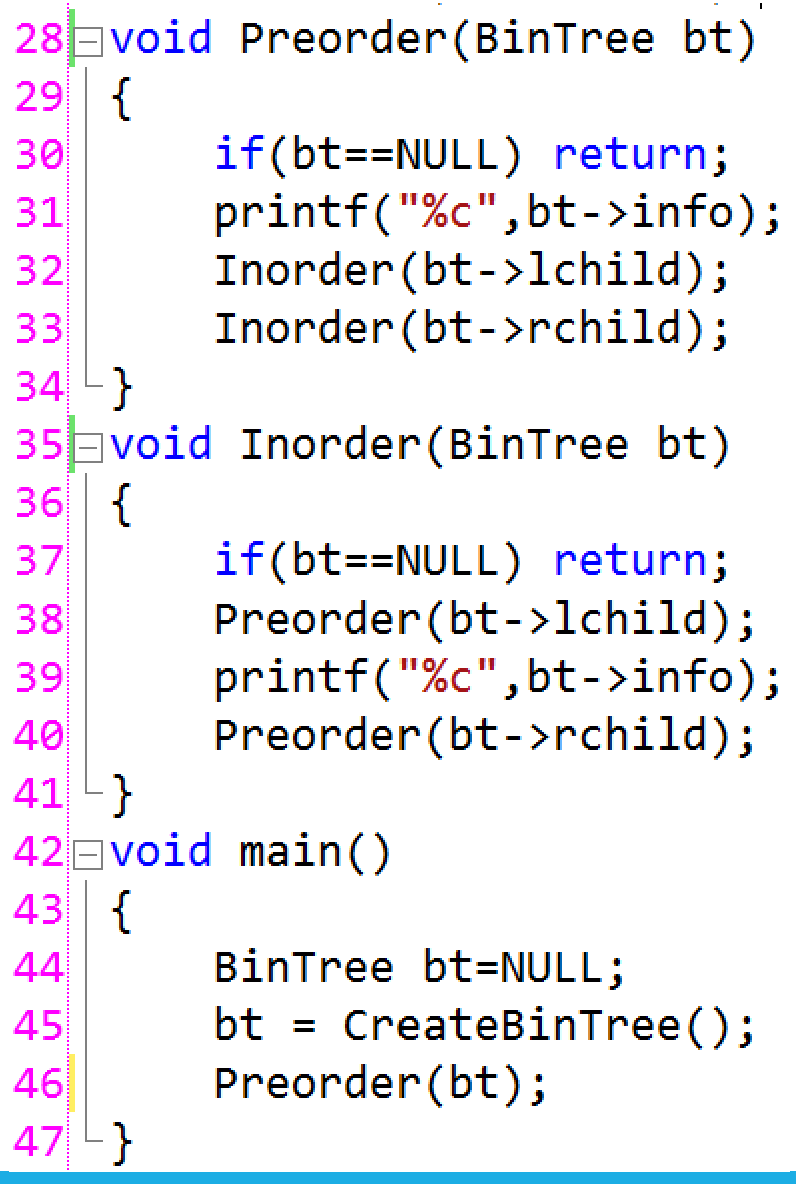
1右

3右（RETURN）

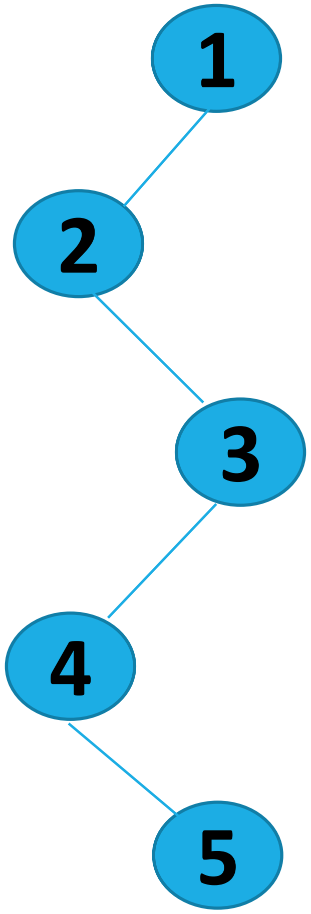


```
void Preorder(BinTree bt)
{
    if(bt==NULL) return;
    printf("%c",bt->info);
    Inorder(bt->lchild);
    Inorder(bt->rchild);
}
void Inorder(BinTree bt)
{
    if(bt==NULL) return;
    Preorder(bt->lchild);
    printf("%c",bt->info);
    Preorder(bt->rchild);
}
void main()
{
    BinTree bt=NULL;
    bt = CreateBinTree();
    Preorder(bt);
}
```

交叉遍历调用



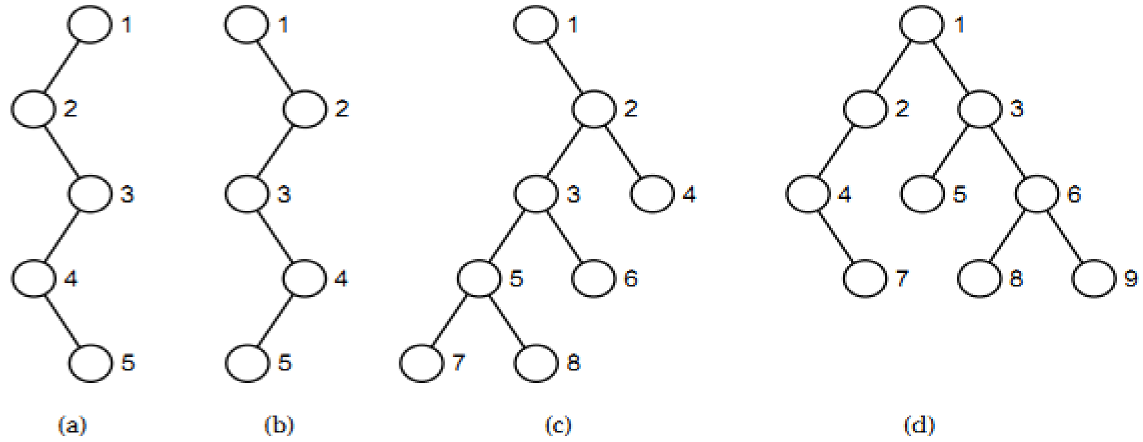
```c
void Preorder(BinTree bt)
{
        if(bt==NULL) return;
        printf("%c",bt->info);
        Inorder(bt->lchild);
        Inorder(bt->rchild);
}
void Inorder(BinTree bt)
{
        if(bt==NULL) return;
        Preorder(bt->lchild);
        printf("%c",bt->info);
        Preorder(bt->rchild);
}
void main()
{
        BinTree bt=NULL;
        bt = CreateBinTree();
        Preorder(bt);
}
```



TIPS：

- 什么时候调用进去?
- 什么时候调用返回?

动脑加动手自己分析



(a) 1, 2, 3, 4, 5　(b) 1, 3, 5, 4, 2　(c) 1, 3, 7, 5, 8, 6, 2, 4　(d) 1, 4, 7, 2, 5, 3, 6, 8, 9