

# 树形结构Part I

同学们要自己动手，锻炼动手能力！

（下节课我们会实现生成随机树的代码，同学们不要错过）

## 二叉树的遍历



```
#include <stdio.h>
#include <string.h>

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

void p2(node *p) {
    if (p == NULL) return ;
    printf("%d ", p->data);
    p2(p->lchild);
    p2(p->rchild);
}

void preorder(node *p) {
    printf("%d ", p->data);
    if (p->lchild != NULL) preorder(p->lchild);
    if (p->rchild != NULL) preorder(p->rchild);
}

void inorder(node *p) {
    if (p->lchild != NULL) inorder(p->lchild);
```

```
    printf("%d ", p->data);  
    if (p->rchild != NULL) inorder(p->rchild);  
}
```

```
void postorder(node *p) {  
    if (p->lchild != NULL) postorder(p->lchild);  
    if (p->rchild != NULL) postorder(p->rchild);  
    printf("%d ", p->data);  
}
```

```
void level(node *p) {  
    node *que[15];  
    int front = 0, rear = 1;  
    que[0] = p;  
    while (front != rear) {  
        node *temp = que[front];  
        front++;  
        printf("%d ", temp->data);  
        if (temp->lchild != NULL) que[rear++] = temp->lchild;  
        if (temp->rchild != NULL) que[rear++] = temp->rchild;  
    }  
    printf("\n");  
}
```

```
int main() {  
    node tree[15];  
    memset(tree, 0, sizeof(tree));  
    tree[1].data = 5, tree[1].lchild = &tree[2], tree[1].rchild = &tree[3];  
    tree[2].data = 1, tree[2].rchild = &tree[5];  
    tree[5].data = 6, tree[5].lchild = &tree[10];  
    tree[10].data = 4;  
    tree[3].data = 2, tree[3].lchild = &tree[6], tree[3].rchild = &tree[7];  
    tree[6].data = 7, tree[6].rchild = &tree[13];  
}
```

```

tree[13].data = 0;
tree[7].data = 9;
preorder(&tree[1]);
printf("\n");
inorder(&tree[1]);
printf("\n");
postorder(&tree[1]);
printf("\n");
level(&tree[1]);
return 0;
}

```

代码中的树形结构如下：

```

29 printf("%d ", p->data);
30 if (p->rchild != NULL) inorder(p->rchild);
31 }
32
33 void postorder(node *p) {
34     if (p->lchild != NULL) postorder(p->lchild);
35     if (p->rchild != NULL) postorder(p->rchild);
36     printf("%d ", p->data);
37 }
38
39 void level(node *p) {
40     node *que[15];
41     int front = 0, rear = 1;
42     que[0] = p;
43     while (front != rear) {
44         node *temp = que[front];
45         front++;
46         printf("%d ", temp->data);
47         if (temp->lchild != NULL) que[rear++] = temp->lchild;
48         if (temp->rchild != NULL) que[rear++] = temp->rchild;
49     }
50     printf("\n");
51 }
52
53 int main() {
54     node tree[15];
55     memset(tree, 0, sizeof(tree));
56     tree[1].data = 5, tree[1].lchild = &tree[2], tree[1].rchild = &tree[3];
57     tree[2].data = 1, tree[2].rchild = &tree[5];
58     tree[5].data = 6, tree[5].lchild = &tree[10];
59     tree[10].data = 4;

```

