

1.6 二叉树的递归遍历_物联网 / 嵌入式工程师 - 慕课网

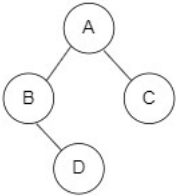
“ 慕课网慕课教程 1.6 二叉树的递归遍历涵盖海量编程基础技术教程，以图文图表的形式，把晦涩难懂的编程专业用语，以通俗易懂的方式呈现给用户。

6. 二叉树的递归遍历

前序遍历：若二叉树为空树，则空操作；否则先访问根结点 在遍历左子树 最后遍历右子树

中序遍历：若二叉树为空树，则空操作；否则先访问左子树 在遍历根结点 最后遍历右子树

后序遍历：若二叉树为空树，则空操作；否则先访问左子树 在遍历右子树 最后遍历根节点



```
void PREORDER ( bitree *r)
{
    if ( r == NULL )
        return ; //空树返回

    printf ( " %c ",r->data ); //先访问当前结点
    PREORDER( r->lchild ); //再访问该结点的左子树
    PREORDER( r->rchild ); //最后访问该结点右子树
}

void INORDER ( bitree *r)
{
    if ( r == NULL )
        return ; //空树返回

    INORDER( r->lchild ); //先该结点的左子树
    printf ( " %c ",r->data ); //在访问当前结点
    INORDER( r->rchild ); //最后访问该结点右子树
}

void POSTORDER ( bitree *r)
{
    if ( r == NULL )
        return ; //空树返回

    POSTORDER ( r->lchild ); //先该结点的左子树
    POSTORDER ( r->rchild ); //在访问该结点右子树
    printf ( " %c ",r->data ); //最后访问当前结点
}
```

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

#define N 6

typedef char data_t;
```

```

typedef struct bitree
{
    int n;
    data_t data;
    struct bitree *lchild;
    struct bitree *rchild;
}bitree_t;

bitree_t *create_binary_tree(int n)
{
    bitree_t *root = NULL;

    root = (bitree_t *)malloc(sizeof(bitree_t));
    memset(root,0,sizeof(bitree_t));

    root->n = n;
    root->lchild = root->rchild = NULL;

    printf("Input %d node data: ",n);
    scanf("%c",&root->data);
    while(getchar() != '\n');

    if(2 * n <= N)
    {
        root->lchild = create_binary_tree(2 * n);
    }

    if(2 * n + 1 <= N)
    {
        root->rchild = create_binary_tree(2 * n + 1);
    }

    return root;
}

void pre_order(bitree_t *root)
{
    if(root == NULL)
        return ;

    printf("(%d:%c) ",root->n,root->data);

    pre_order(root->lchild);

    pre_order(root->rchild);

    return ;
}

void in_order(bitree_t *root)
{
    if(root == NULL)
        return ;

    in_order(root->lchild);

    printf("(%d:%c) ",root->n,root->data);

    in_order(root->rchild);

    return ;
}

void post_order(bitree_t *root)
{
    if(root == NULL)
        return ;
    post_order(root->lchild);

    post_order(root->rchild);

    printf("(%d:%c) ",root->n,root->data);

    return ;
}

int main()
{
    bitree_t *root = NULL;

    root = create_binary_tree(1);

    pre_order(root);
    putchar('\n');
}

```

```
        return 0;
    }
```

运行结果:

```
Input 1 node data: A
Input 2 node data: B
Input 4 node data: D
Input 5 node data: E
Input 3 node data: C
Input 6 node data: F
(1:A) (2:B) (4:D) (5:E) (3:C) (6:F)
```

全文完

本文由 简悦 SimpRead 优化，用以提升阅读体验

使用了 全新的简悦词法分析引擎 beta，点击查看详细说明

