```
//c语言实现二叉树层次遍历（借助队列实现）

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

//二叉树类型定义
typedef struct TreeNode
{
    char data;
    struct btnode *lchild,*rchild;
}TreeNode,*Tree;


//链队列类型定义
typedef struct QueueNode
{
    TreeNode *data;
    struct LinkQueueNode *next;
}QueueNode;


typedef struct Queue
{
    QueueNode *front,*rear;
}Queue;


//初始化队列
void InitQueue(Queue *LQ)
{
    QueueNode *p;
    p=(QueueNode*)malloc(sizeof(QueueNode));
    LQ->front = p;
    LQ->rear = p;
    LQ->front->next = NULL;
}


//判断队列是否为空
int EmptyQueue(Queue *LQ)
{
    if(LQ->front == LQ->rear)
        return 1;
    else
        return 0;
}


//入队列
void EnQueue(Queue *LQ,Tree x)
{
    QueueNode *p;
    p = (QueueNode*)malloc(sizeof(QueueNode));
    p->data = x;
    p->next = NULL;
    LQ->rear->next = p;
    LQ->rear = p;
}
```

```
//出队列
int OutQueue(Queue *LQ)
{
    QueueNode *s;
    if ( EmptyQueue(LQ))
    {
        exit(0);
        return 0;
    }
    else
    {
        s = LQ->front->next;
        LQ->front->next = s->next;
        if(s->next == NULL)
            LQ->rear = LQ->front;
        free(s);
        return 1;
    }
}


//取队列首元素
Tree GetHead(Queue *LQ)
{
    QueueNode *p;
    TreeNode *q;
    if(EmptyQueue(LQ))
        return q;
    else
    {
        p = LQ->front->next;
        return p->data;
    }
}


//建二叉树
Tree Initiate()
{
    char ch;
    Tree t;
    ch = getchar();
    if(ch == '#')
        t = NULL;
    else
    {
        t = (Tree)malloc(sizeof(TreeNode));
        t->data = ch;
        t->lchild = Initiate();
        t->rchild = Initiate();
    }
    return t;
}


//访问节点
void Visit(Tree p)
{
printf("%c",p->data); //输出是char
}
```

```
//树的高度
int height(Tree t)
{
    int ld,rd;
    if(t == NULL)
        return 0;
    else
    {
        ld = height(t->lchild);
        rd = height(t->rchild);
        return 1 + (ld>rd?ld:rd);
    }
}

//层次遍历
void LevelOrder(Tree bt)
{
    Queue Q;
    Tree p;
    InitQueue(&Q);
    if(bt != NULL)
    {
        EnQueue(&Q,bt);
        while(!EmptyQueue(&Q))
        {
            p = GetHead(&Q);
            OutQueue(&Q);
            Visit(p);
            if(p->lchild != NULL)
                EnQueue(&Q,p->lchild);
            if(p->rchild != NULL)
                EnQueue(&Q,p->rchild);
        }
    }
}


int main()
{
    Tree T;
    printf("\n按先序序列输入结点序列，'#'代表空:");
    T=Initiate();


    printf("\n二叉树的高度为:%d\n",height(T));


    printf("\n层次遍历序列为:");
    LevelOrder(T);


    printf("\n\n");
    return 0;
}
```