# 12

struct tree

{

    int data;

    int depth;

    struct tree \*l,\*r;

};

int maxdep(tree \*root)

{

    int dep=0;

    tree \*q[MAX];

    root->depth=1;

    for (head=1,tail=1;head<=tail;head++)

    {

        if (q[head]->depth>dep) dep=q[head]->depth;

        if (q[head]->l!=NULL) q[++tail]=q[head]->l,q[head]->l->depth=q[head]->depth+1;

        if (q[head]->r!=NULL) q[++tail]=q[head]->r,q[head]->r->depth=q[head]->depth+1;

    }

    return dep;

}

# 13

1. 当叶子节点都在第h层时，内节点最多，此时第h-1层的节点最多为2^(h-2)，依此类推，故从第1层到第h-1层一共有2^(h-1)-1个节点，即内节点最多为2^(h-1)-1个节点。
2. 反证法：若叶子节点大于2^(h-1)，因为叶子节点都在最底层时，内节点最多，总节点最多，此时第h-1层节点大于2^(h-2)，以此类推，前h-1层的节点总数，即内节点大于2^(h-1)-1个，与Q1矛盾。所以，叶子结点的个数小于等于2^(h-1)。

# 14

int path(int x)

{

    if (x<=0) return 0;

    return max(path(left[x])+lw[x],path(right[x])+rw[x]);

}

# 16

void preprocessing(int x)

{

    if (!x) return;

    int y=l[x];

    while (y) y=r[y];

    next[y]=x;

    preprocessing(l[x]);

    preprocessing(r[x]);

}

void trave(int root)

{

    int x=root;

    while (x) x=l[x];

    while (x)

    {

        printf("%d ",x);

        x=next[x];

    }

}

# 34

void print\_tree(int x)

{

    if (x<=0) return;

    printf("(%d",key[x]);

    if (left[x]||right[x]) printf(":");

    print\_tree(left[x]);

    if (left[x]&&right[x]) printf(",");

    print\_tree(right[x]);

    printf(")");

}

# 36

