# 一、

## 2、

#include <stdio.h>

#include <stdlib.h>

#define m 1024

typedef struct bn

{

int d;

struct bn \*lc, \*rc;

}BN;

typedef BN \*QE;

typedef struct qn

{

QE data[m];

int f, r;

}QN;

void Init(QN \*q)

{

q->f=0;

q->r=0;

}

int Empty(QN \*q)

{

if(q->r==q->f)

return 1;

return 0;

}

QE Head(QN \*q)

{

if(q->f==q->r)

return NULL;

else

return (q->data[(q->f+1)%m]);

}

void EQ(QN \*q, QE x)

{

if((q->r+1)%m==q->f)

printf("Overflow\n");

else

{

q->r=(q->r+1)%m;

q->data[q->r]=x;

}

}

QE DQ(QN \*q)

{

if(q->f==q->r)

return NULL;

else

{

q->f=(q->f+1)%m;

return (q->data[q->f]);

}

}

BN \*create()

{

int x;

BN \*bt;

scanf("%d",&x);

if(x==0)return(NULL);

else

{

bt=(BN\*)malloc(sizeof(BN));

bt->d=x;

bt->lc=create();

bt->rc=create();

return (bt);

}

}

void preorder(BN \*t)

{

if(t)

{

printf("%d",t->d);

preorder(t->lc);

preorder(t->rc);

}

}

void inorder(BN \*t)

{

if(t)

{

inorder(t->lc);

printf("%d",t->d);

inorder(t->rc);

}

}

void postorder(BN \*t)

{

if(t)

{

postorder(t->lc);

postorder(t->rc);

printf("%d",t->d);

}

}

void LT(BN \*t)

{

QN \* q;

BN \* p;

q=(QN\*)malloc(sizeof(QN));;

Init(q);

p=t;

if(p==NULL)

printf("树为空！\n") ;

EQ(q,p) ;

while(!Empty(q))

{

p = NULL ;

p=DQ(q);

if(p->lc!=NULL)

EQ(q,p->lc) ;

if(p->rc!=NULL)

EQ(q,p->rc) ;

printf("%d", p->d) ;

}

}

int GD(BN \*t)

{

int a,b;

if(!t)

return 0;

else

{

a=GD(t->lc);

b=GD(t->rc);

return (a>b?a:b)+1;

}

}

void GSD(BN \*t, int x)

{

if(t->d==x)

{

printf("%d\n",GD(t));

}

else

{

if(t->lc)

GSD(t->lc,x);

if(t->rc)

GSD(t->rc,x);

}

}

int CN(BN \*t)

{

int count=0;

if(t)

{

count++;

return count+CN(t->lc)+CN(t->rc);

}

else

return count;

}

int CL(BN \*t)

{

int i;

if(t==NULL)

i=0;

else if((t->lc==NULL)&&(t->rc==NULL))

i=1;

else

i=CL(t->lc)+CL(t->rc);

return (i);

}

int main()

{

BN \*bn;

bn=create();

printf("%d\n",GD(bn));

GSD(bn,2);

printf("%d\n",CL(bn));

printf("%d\n",CN(bn));

preorder(bn);

printf("\n");

inorder(bn);

printf("\n");

postorder(bn);

printf("\n");

LT(bn);

printf("\n");

return 0;

}

# 二、

## 1、

#include <stdio.h>

#include <stdlib.h>

#define m 1024

typedef struct bn

{

int d;

struct bn \*lc, \*rc, \*p;

}BN;

BN \*create()

{

int x;

BN \*bt;

scanf("%d",&x);

if(x==0)return(NULL);

else

{

bt=(BN\*)malloc(sizeof(BN));

bt->d=x;

bt->lc=create();

if(bt->lc)

bt->lc->p=bt;

bt->rc=create();

if(bt->rc)

bt->rc->p=bt;

return (bt);

}

}

void preorder(BN \*t)

{

if(t)

{

printf("%d",t->d);

preorder(t->lc);

preorder(t->rc);

}

}

int GD(BN \*t)

{

int num=0;

if(t)

{

if(t->lc)

num++;

num+=GD(t->lc)+GD(t->rc);

}

return num;

}

void GSD(BN \*t, int x)

{

if(t->d==x)

{

printf("%d\n",GD(t)+1);

}

else

{

if(t->lc)

GSD(t->lc,x);

if(t->rc)

GSD(t->rc,x);

}

}

int CL(BN \*t)

{

int i;

if(t==NULL)

i=0;

else if((t->lc==NULL)&&(t->rc==NULL))

i=1;

else

i=CL(t->lc)+CL(t->rc);

return (i);

}

int HZ(BN \*t)

{

int num=0;

struct bn \*tt;

if(t)

{

if(t->lc)

{

tt=t->lc;

while(tt)

{

tt=tt->rc;

num++;

}

}

}

return num;

}

void HZS(BN \*t, int x)

{

if(t->d==x)

{

printf("%d\n",HZ(t));

}

else

{

if(t->lc)

HZS(t->lc,x);

if(t->rc)

HZS(t->rc,x);

}

}

void XD(BN \*t, int x)

{

struct bn \*tt;

if(t->d==x)

{

tt=t;

while(1)

{

if(tt->p->lc==tt)

{

while(tt)

{

printf("%d\t",tt->d);

tt=tt->rc;

}

break;

}

else

{

tt=tt->p;

}

}

}

else

{

if(t->lc)

XD(t->lc,x);

if(t->rc)

XD(t->rc,x);

}

}

void SS(BN \*t, int x)

{

struct bn \*tt;

if(t->d==x)

{

tt=t;

while(1)

{

if(tt->p->lc==tt)

{

tt=tt->p;

while(1)

{

if(tt->p->lc==tt)

{

while(tt)

{

printf("%d\t",tt->d);

tt=tt->rc;

}

break;

}

else

{

tt=tt->p;

}

}

break;

}

else

{

tt=tt->p;

}

}

}

else

{

if(t->lc)

{

SS(t->lc,x);

}

if(t->rc)

{

SS(t->rc,x);

}

}

}

//1 2 0 3 4 0 0 5 0 0 0

int main()

{

BN \*bn;

bn=create();

preorder(bn);

printf("\n");

printf("%d\n",GD(bn)+1);

GSD(bn,2);

HZS(bn,3);

XD(bn,5);

printf("\n");

printf("%d\n",CL(bn));

SS(bn,4);;

printf("\n");

return 0;

}