/\*\*

\* BiThrTree.c

\* 线索二叉树的代码实现

\*

\*

\*/

#include <stdio.h>

#include <stdlib.h>

typedef enum {Link,Thread} PointerTag;

typedef char ElemType;

typedef struct BiThrNode

{

ElemType data;

struct BiThrNode \*lchild,\*rchild;

PointerTag ltag;

PointerTag rtag;

}BiThrNode,\*BiThrTree;

BiThrTree pre;

CreateBiThrTree(BiThrTree \*T)

{

char c;

scanf("%c",&c);

if(' ' == c)

{

(\*T) = NULL;

}

else

{

(\*T) = (BiThrNode \*)malloc(sizeof(BiThrTree));

(\*T)->data = c;

(\*T)->ltag = Link;

(\*T)->ltag = Link;

CreateBiThrTree(&(\*T)->lchild);

CreateBiThrTree(&(\*T)->rchild);

}

}

InThreading(BiThrTree T)

{

if(T)

{

InThreading(T->lchild);

if(!T->lchild)

{

T->ltag = Thread;

T->lchild = pre;

}

if(!pre->rchild)

{

pre->rtag = Thread;

pre->rchild = T;

}

pre = T;

InThreading(T->rchild);

}

}

InOrderThreading(BiThrTree \*p,BiThrTree T)

{

\*p =(BiThrTree)malloc(sizeof(BiThrNode));

(\*p)->ltag = Link;

(\*p)->rtag = Thread;

(\*p)->rchild = \*p;

if(!T)

{

(\*p)->lchild=\*p;

}

else

{

(\*p)->lchild=T;

pre = \*p;

InThreading(T);

pre->rchild = \*p;

pre->rtag = Thread;

(\*p)->rchild = pre;

}

}

void visit(char c)

{

printf("%c",c);

}

void InOrderTraverse(BiThrTree T)

{

BiThrTree p;

p = T->lchild;

while(p!=T)

{

while(p->ltag==Link)

{

p = p->lchild;

}

visit(p->data);

while(p->rtag == Thread && p->rchild!=T)

{

p = p->rchild;

visit(p->data);

}

p = p->rchild;

}

}

int main()

{

BiThrTree P,T = NULL;

CreateBiThrTree(&T);

InOrderThreading(&P,T);

InOrderTraverse(P);

printf("\n");

return 0;

}



