1. 详细设计

#include<iostream>

#include <malloc.h>

#include <cstdio>

using namespace std;

#define Maxsize 100

typedef char Elemtype;

typedef struct node

{

char data; //数据域

struct node\*lchild, \*rchild; //指针域

}BTNode;

void CreatrBTree(BTNode\*&b,char\*str) //创建二叉树

{

BTNode\*St[Maxsize], \*p;

int top=-1, k, j=0;

char ch;

b=NULL;

ch=str[j];

while(ch!='\0')

{ switch(ch)

{

case'(':top++;St[top]=p;k=1;break;

case')':top--;break;

case',':k=2;break;

default:

p=(BTNode\*)malloc(sizeof(BTNode));

p->data=ch; p->lchild=p->rchild=NULL;

if (b==NULL)

b=p;

else

{ switch(k)

{

case 1:St[top]->lchild=p; break;

case 2:St[top]->rchild=p; break;

}

}

}

j++; ch=str[j];

}

}

void DispBTNode(BTNode\*b) //输出二叉树

{ if(b!=NULL)

{ printf("%c",b->data);

if(b->lchild!=NULL||b->rchild!=NULL)

{ printf("(");

DispBTNode(b->lchild);

if(b->rchild!=NULL)

printf(",");

DispBTNode(b->rchild);

printf(")");

}

}

}

void PreOrder(BTNode\*b) //先序遍历

{

if(b!=NULL)

{ printf("%c", b->data);

PreOrder(b->lchild);

PreOrder(b->rchild);

}

}

int Nodes(BTNode\*b) //计算结点个数

{

if(b==NULL)

return 0;

else

return Nodes(b->lchild)+Nodes(b->rchild)+1;

}

int main() //主函数

{ char str[200];

scanf("%s",str);

BTNode\*b;

CreatrBTree(b,str);

DispBTNode(b);

cout << endl;

cout << "先序遍历的结果为:";

PreOrder(b);

cout << endl;

printf("结点数量为: %d\n", Nodes(b));

return 0;

}

四、调试分析

1. 程序使用说明
2. 本程序的运行环境为 VC6.0；
3. 进入演示程序后显示提示信息：

用括号法输入一棵二叉树。

1. 测试结果

如果输入：A(B(D,E),C(F))

运行结果：先序遍历的结果为：ABDECF

结点数量为：6

