**《编译系统设计实践》**

实验项目三：语法制导翻译与生成中间代码

学号：

姓名：

年级：

学院： 数计学院

专业： 计算机

本组其它成员：学号 姓名

学号 姓名

实验时间：2016－2017学年第一学期

任课教师：

**一、实验目的**

通过语法制导或翻译模式生成中间代码。

**二、实验内容**

在自底向上语法分析基础上设计语义规则（语法制导翻译），将源程序翻译为四元式输出，若有错误将错误信息输出。

**三、设计思路**

1. 分析过程

主函数，读取文件，存入字符串数组，调用语义分析，判断关键字，调用相应的语义规则（这里只有if和while和赋值语句），赋值语句调用表达式处理，if语句调用条件表达式处理，while也是调用表达式处理，然后是一个递归过程，不断的递归调用，按序输出三地址语句。在本例程序中选用expr及num作为运算数。

1. 主要函数

string link() //字符串和数字的连接

string element() //获取表达式中的元素对象

string expression() //处理表达式

string expression\_1() //处理表达式

string biaodashi() //处理表达式，转为三地址输出

string biaodashi\_1() //递归---处理表达式，转为三地址输出

string getOperator() //判断并获取运算符

void condition(int L1,int L2) //输出if语句的条件的三地址代码

void yuyifenxi\_list() //生成并输出条件返回地址

void yuyifenxi\_list\_1() //递归---生成并输出条件返回地址

void yuyifenxi(int next,int &flag) //判断关键字，调用相应的产生式分析

void readfile() //文件读入

**四、测试报告**

1. 第一组测试：

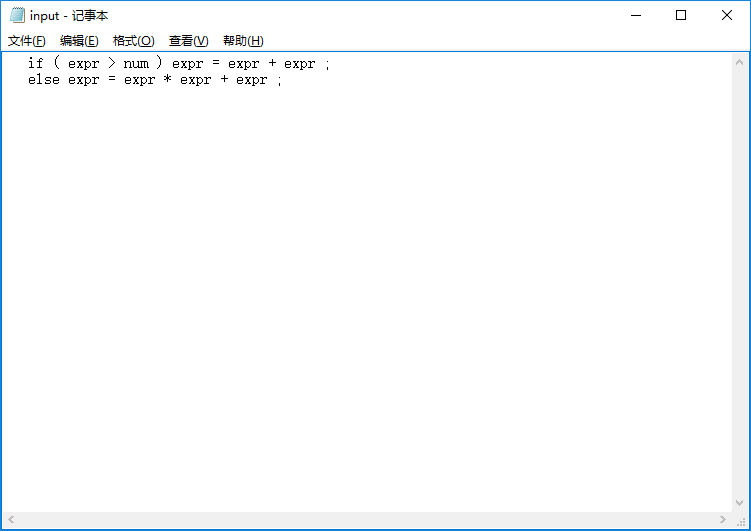


图1-1 输入待翻译代码

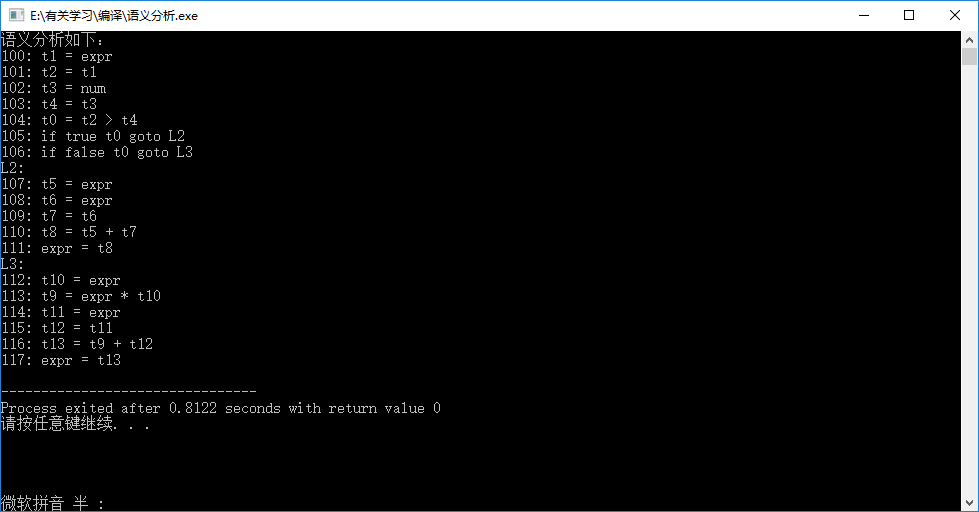


图1-2 中间代码生成

1. 第二组测试：



图2-1 输入待翻译代码

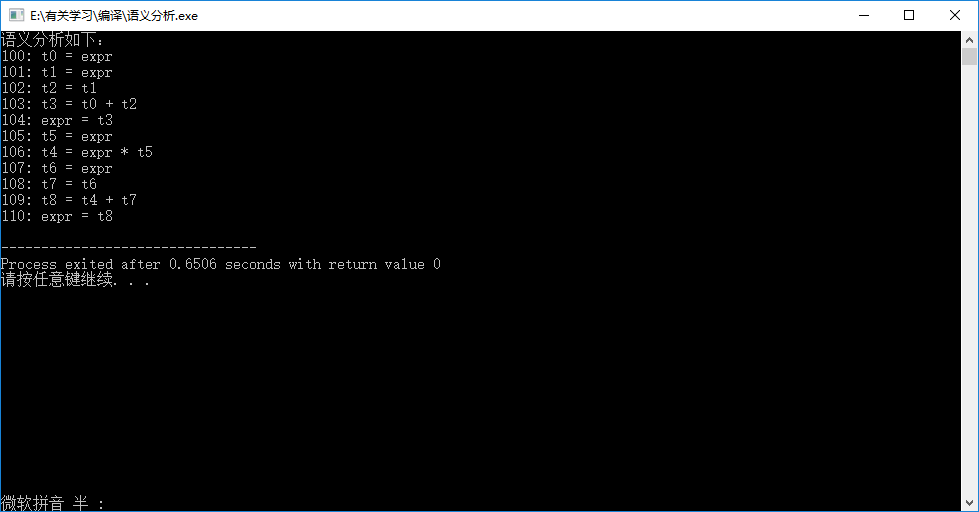


图2-2 中间代码生成

3.第三组测试：错误待翻译代码

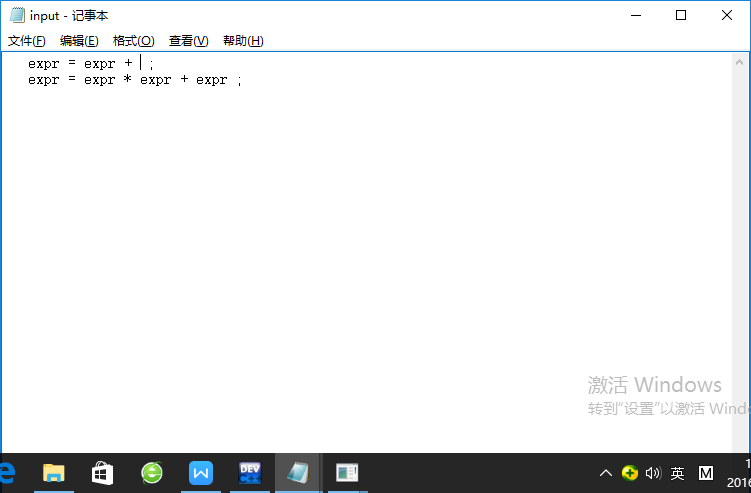


图3-1 输入待翻译代码

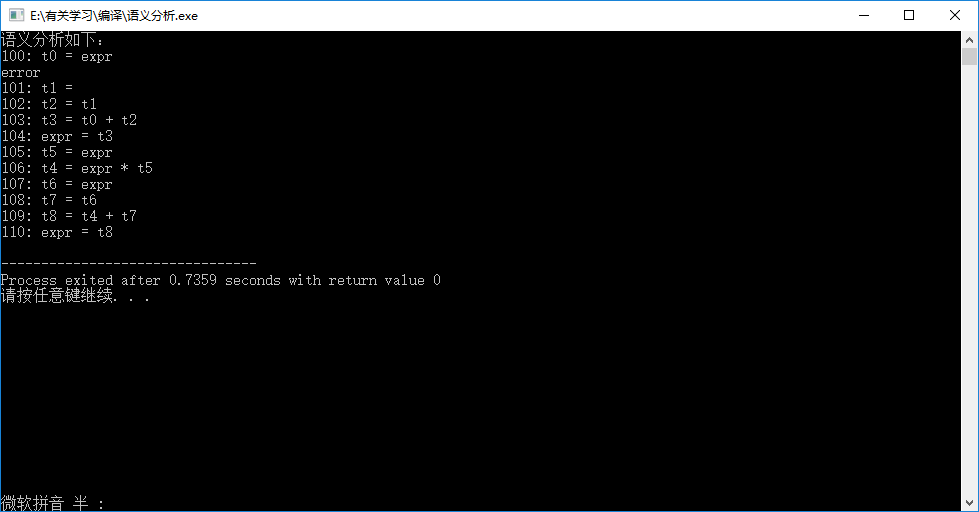


图3-2 中间代码生成

**五、实验总结**

实验三的重点在于判断关键字，调用相应的产生式分析及处理表达式，转为三地址输出部分，也是很费时间的难点部分，但通过查阅书本及网上资料，还是将其以多个处理函数的递归调用实现了，虽然最后实现结果对错误的分析还不够精确有些差强人意，但毕竟还算有些收获了。另外，通过三次实验下来，对于一个简易编译器的实现已经有了一个整体的构架了，相信在通过自己以后的深入学习，一定能写出属于自己的编译器。

**六、附录代码**

#include <iostream>

#include <algorithm>

#include<conio.h>

using namespace std;

int address=100; //每条分析语句的地址

int LID=0; //表示过程执行到相应位置的地址符号

int tID=0; //用于替换表达式的标识符

int ip=0;

string shuru[666]; //存放从文件读入的字符串

int maxsize; //设置存放数组的长度

string biaodashi();

/\*\*\*\*\*字符串和数字的连接\*\*\*\*\*/

string link(string a,int b)

{

string t="";

do

{

t+=b%10+'0';

b/=10;

}

while(b);

reverse(t.begin(),t.end());

return a+t;

}

/\*\*\*\*\*获取表达式中的元素对象\*\*\*\*\*/

string element()

{

if(shuru[ip]=="expr"||shuru[ip]=="num")

{

ip++;

return shuru[ip-1];

}

else if(shuru[ip]=="(")

{

ip++;

string result=biaodashi();

if(shuru[ip]==")")ip++;

else puts("Lack)");

return result;

}

else puts("error");

return "";

}

/\*\*\*\*\*处理表达式\*\*\*\*\*/

string expression\_1(string &op)

{

if(shuru[ip]=="\*"||shuru[ip]=="/")

{

op=shuru[ip];

ip++;

string arg1=element();

string op\_1="",result=link("t",tID++);

string arg2=expression\_1(op\_1);

if(op\_1=="")op\_1="=";

if(arg2=="") cout<<address++<<":"<<" "<<result<<" = "<<arg1<<endl;

else cout<<address++<<":"<<" "<<result<<" = "<<arg1<<" "<<op\_1<<" "<<arg2<<endl;

return result;

}

return "";

}

/\*\*\*\*\*处理表达式\*\*\*\*\*/

string expression()

{

string op="",result=link("t",tID++);

string arg1=element();

string arg2=expression\_1(op);

if(op=="")

{

op="=";

}

if(arg2=="")

{

cout<<address++<<":"<<" "<<result<<" = "<<arg1<<endl;

}

else

{

cout<<address++<<":"<<" "<<result<<" = "<<arg1<<" "<<op<<" "<<arg2<<endl;

}

return result;

}

/\*\*\*\*\*递归---处理表达式，转为三地址输出\*\*\*\*\*/

string biaodashi\_1(string &op)

{

string result="";

if(shuru[ip]=="+"||shuru[ip]=="-")

{

op=shuru[ip];

ip++;

string arg1=expression();

string op\_1="";

string arg2=biaodashi\_1(op\_1);

result=link("t",tID++);

if(op\_1=="")

{

op\_1="=";

}

if(arg2=="")

{

cout<<address++<<":"<<" "<<result<<" = "<<arg1<<endl;

}

else

{

cout<<address++<<":"<<" "<<result<<" = "<<arg1<<" "<<op\_1<<" "<<arg2<<endl;

}

}

return result;

}

/\*\*\*\*\*处理表达式，转为三地址输出\*\*\*\*\*/

string biaodashi()

{

string arg1="",op="";

if(shuru[ip]=="+"||shuru[ip]=="-"){arg1=shuru[ip];ip++;}

arg1+=expression();

string arg2=biaodashi\_1(op);

string result=link("t",tID++);

if(op=="")

{

op="=";

}

if(arg2=="")

{

cout<<address++<<":"<<" "<<result<<" = "<<arg1<<endl;

}

else

{

cout<<address++<<":"<<" "<<result<<" = "<<arg1<<" "<<op<<" "<<arg2<<endl;

}

return result;

}

/\*\*\*\*\*判断并获取运算符\*\*\*\*\*/

string getOperator()

{

if(shuru[ip]=="="||shuru[ip]=="<>"||shuru[ip]=="<"||shuru[ip]==">"||

shuru[ip]=="<="||shuru[ip]==">=")

{

ip++;

return shuru[ip-1];

}

else

{

puts("error");

}

return "";

}

/\*\*\*\*\*输出if语句的条件的三地址代码\*\*\*\*\*/

void condition(int L1,int L2) //L1,L2分别为if条件为true和false时候的跳转地址

{

string result=link("t",tID++);

string arg1=biaodashi(); //获得表达式的运算符的左边内容

string op=getOperator(); //获得表达式的运算符

string arg2=biaodashi(); //获得表达式的运算符的右边内容

if(arg2=="")

{

cout<<" "<<result<<" = "<<arg1<<endl;

}

else

{

cout<<address++<<":"<<" "<<result<<" = "<<arg1<<" "<<op<<" "<<arg2<<endl;

}

cout<<address++<<":"<<" if true "<<result<<" goto "<<"L"<<L1<<endl;

cout<<address++<<":"<<" if false "<<result<<" goto "<<"L"<<L2<<endl;

}

/\*\*\*\*\*判断关键字，调用相应的产生式分析\*\*\*\*\*/

void yuyifenxi(int next,int &flag)

{

if(shuru[ip]=="expr")

{

ip++;

if(shuru[ip]=="=") //赋值语句 转化为四元式

{

ip++;

string arg1=biaodashi();

string arg2="";

if(arg2 == "") cout<<address++<<":"<<" expr = "<<arg1<<endl;

}

else puts("error");

}

else if(shuru[ip]=="if") //if的语义子程序

{

ip++;

int L1=LID++;

int L2=LID++;

if(shuru[ip]=="(")

{

ip++;

condition(L1,L2);

}

else

{

puts("Lack(");return;

}

if(shuru[ip]==")") ip++;

else {

puts("Lack)");return;

}

printf("L%d:\n",L1);

yuyifenxi(next,flag);

ip++;

if(shuru[ip]=="else")

{

printf("L%d:\n",L2);

ip++;

yuyifenxi(next,flag);

}

}

else if(shuru[ip]=="while") //while的语义子程序

{

ip++;

int L1=LID++;

int L2=LID++;

if(shuru[ip]=="(")

{

ip++;

printf("L%d:\n",L1);

condition(L2,next);

}

else

{

puts("Lack(");return;

}

if(shuru[ip]==")") ip++;

else {

puts("Lack)");return;

}

printf("L%d:\n",L2);

yuyifenxi(next,flag);

printf("goto L%d\n",L1);

flag=1;

}

}

/\*\*\*\*\*递归---生成并输出条件返回地址\*\*\*\*\*/

void yuyifenxi\_list\_1()

{

if(shuru[ip]==";")

{

ip++;

int next=LID++;

int flag=0;

yuyifenxi(next,flag);

if(flag)printf("L%d:\n",next);

yuyifenxi\_list\_1();

}

}

/\*\*\*\*\*生成并输出条件返回地址\*\*\*\*\*/

void yuyifenxi\_list()

{

int next=LID++;

int flag=0;

yuyifenxi(next,flag);

if(flag)printf("L%d:\n",next);

yuyifenxi\_list\_1();

}

void Modular()

{

int next=LID++;

int flag=0;

yuyifenxi\_list();

if(flag)printf("L%d:\n",next);

}

/\*\*\*\*\*文件读入\*\*\*\*\*/

void readfile() //将字符串输入到shuru中

{

maxsize=0;

while(cin>>shuru[maxsize])

maxsize++;

}

/\*\*\*\*\*主函数\*\*\*\*\*/

int main()

{

freopen("input.txt","r",stdin);

cout<<"语义分析如下："<<endl;

readfile();

Modular();

return 0;

}