# 实验 3：递归下降语法分析实验

### 一、实验目的

通过设计、开发一个高级语言的递归下降语法分析程序，实现

对词法分析程序所提供的单词序列进行语法检查和结构分析，加

深对相关课堂教学内容的理解，提高语法分析方法的实践能力。

二、实验要求

(1)理解语法分析在编译程序中的作用，以及它与词法分析程序的

关系；

(2)掌握递归下降语法分析方法的主要原理；

(3)理解递归下降分析法对文法的要求；

(4)熟练掌握Select集合的求解方法；

(5)熟练掌握文法变换方法（消除左递归和提取公因子）。

### 三、实验原理

递归下降分析法是语法分析中最易懂的一种方法，基本原理

是：对每个非终结符号（分别代表一个语法单位）按其产生式结

构构造相应语法分析子程序，以完成该非终结符号所对应的语法

单位的分析和识别任务。其中终结符号产生匹配命令，而非终结

符号则产生过程调用命令。因为文法可以递归，相应子程序也是

递归的，所以称这种方法为递归子程序下降法或递归下降法。其

中子程序的结构与产生式结构几乎是一致的。

假设一个文法中的非终结符号 A的全部产生式为

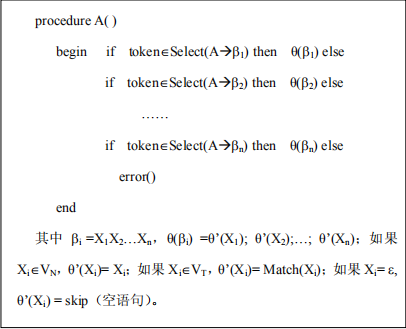
AÆα1|α2|……|αn ，则必须满足以下条件才能保证可以唯一的选

择合适的产生式，才能采用递归下降分析法：

Select(AÆαi)∩Select(AÆαj)=Φ,其中i≠j

假设文法中有如下的产生式AÆβ1|β2|…|βn，则应按如下方法

编写语法分析子程序。



### 四、实验步骤

(1)根据S语言BNF形式的语法规则（见附件A），写出S语言的上下

文无关文法；

(2)求每个产生式的Select集：

procedure A( )

begin if token∈Select(AÆβ1) then θ(β1) else

if token∈Select(AÆβ2) then θ(β2) else

……

if token∈Select(AÆβn) then θ(βn) else

error()

end

其中 βi =X1X2…Xn，θ(βi) =θ’(X1); θ’(X2);…; θ’(Xn)；如果

Xi∈VN，θ’(Xi)= Xi；如果 Xi∈VT，θ’(Xi)= Match(Xi)；如果 Xi= ε,

θ’(Xi) = skip（空语句）。

《编译原理》实验指导书

10

Select(AÆβ)= First(β)，当 ε∉First(β)

= (First(β)-{ε})∪Follow(A)，当ε∈First(β)

(3)判断是否满足递归下降法分析条件，若不满足用消除左递归和

提取公因子等文法等价变换操作对文法进行变换，使其满足递

归下降法的要求；

(4)构造递归下降语法分析程序，对文法中的每个非终结符号按其

产生式结构产生相应的语法分析子程序,完成相应的识别任

务。其中终结符号产生匹配命令，非终结符号则产生调用命令。

实际的语法分析工作从调用主程序（开始符号S对应的程序）

开始，根据产生式递归调用各个分析子程序；

1. 撰写实验报告。

### 实现代码

#include <stdio.h>

#include <string.h>

char word[1000];

char scan[100];

int line=1;

int t=0,i=0,sum=0;

char c;

int syn=0;

void scanner();

void program();

void constExp(int shendu);

void constDef(int shendu);

void constRec(int shendu);

void uInt(int shendu);

void ide(int shendu);

void varExp(int shendu);

void ideRec(int shendu);

void sen(int shendu);

void assSen(int shendu);

void expression(int shendu);

void itemRec(int shendu);

void item(int shendu);

void factorRec(int shendu);

void factor(int shendu);

void addOpt(int shendu);

void mulOpt(int shendu);

void conSen(int shendu);

void conOpt(int shendu);

void condition(int shendu);

void relOpt(int shendu);

void cirSen(int shendu);

void comSen(int shendu);

void senRec(int shendu);

void scanner();

/\*

1: Const

2: Var

3: if

4: then

5: else

6: while

7: do

8: begin

9: end

10: Indetifier

11: Number

12: >=

13: >

14: <=

15: <>

16: <

17: ==

18: =

19: ,

20: ;

21: (

22: )

23: +

24: -

25: \*

26: /

\*/

void star(int shendu){ //用于输出所在层次\*\*\*\*

for(int i=0;i<shendu;i++)

printf(" ");

}

void program() // <程序>→[<常量说明>][<变量说明>]<语句>

{

int shendu=1;//常量说明

if(syn==1)

{

star(shendu);

printf("constExp\n");

constExp(shendu);

}

else if(syn==2)//变量说明

{

star(shendu);

printf("VarExp\n");

varExp(shendu);

}

else if(syn==3 || syn==6 || syn==8)//语句

{

star(shendu);

printf("Sen\n");

sen(shendu);

}

}

//<常量说明>→Const <常量定义>{，<常量定义>}；

void constExp(int shendu)

{

if(syn==1)

{

shendu++;

star(shendu);

printf("Const\n");

scanner( );

constDef(shendu); //常量定义

if(syn==20)

{

star(shendu);

printf(";\n");

scanner();

program();

}

else

printf("; Error!\n");

}

else

printf("const Error!\n");

}

//<变量说明>->Var <标识符>{, <标识符>}

void varExp(int shendu)

{

if(syn==2)

{

shendu++;

star(shendu);

printf("Var\n");

scanner();

ide(shendu);

ideRec(shendu);

if(syn==20)

{

star(shendu);

printf(";\n");

scanner();

program();

}

else

printf("; Error!\n");

}

else

printf("var Error!\n");

}

// <常量定义>→<标识符>＝<无符号整数>

void constDef(int shendu)

{

star(shendu);

printf("ConstDef\n");

shendu++;

ide(shendu);

if(syn==18)

{

star(shendu);

printf("=\n");

scanner( );

uInt(shendu);

constRec(shendu-1);

}

else

printf("constDef Error!\n");

}

////可以多个定义常量 如 const a=1,b=2;

void constRec(int shendu)

{

if(syn==19)

{

star(shendu);

printf(",\n");

scanner();

constDef(shendu);

}

}

//<无符号整数>-><数字>{<数字>}

void uInt(int shendu)

{

if(syn==11)

{

star(shendu);

printf("Number: %s\n",scan);

scanner();

}

else

printf("Number Error!");

}

//<标识符>-><字母>{<字母>，<数字>}

void ide(int shendu)

{

if(syn==10)

{

star(shendu);

printf("Identifier: %s\n",scan);

scanner();

//printf("%d",syn);

//ideRec();

}

else

printf("Identifier Error!");

}

//可以多个说明变量 如 var a,b;

void ideRec(int shendu)

{

if(syn==19)

{

star(shendu);

printf(",\n");

scanner();

ide(shendu);

ideRec(shendu);

}

}

//<语句>→<赋值语句>|<条件语句>|<当循环语句>|<复合语句>|ε

void sen(int shendu)

{

shendu++;

if(syn==10)

{

star(shendu);

printf("assSen\n");

assSen(shendu); //赋值语句

}

else if(syn==8)

{

star(shendu);

printf("comSen\n");//复合语句

comSen(shendu);

}

else if(syn==3)

{

star(shendu);

printf("conSen\n");//条件语句

conSen(shendu);

}

else if(syn==6)

{

star(shendu);

printf("cirSen\n");//循环语句

cirSen(shendu);

}

}

//<赋值语句>→<标识符>＝<表达式>;

void assSen(int shendu)

{

shendu++;

ide(shendu); // 标识符

if(syn==18)

{

star(shendu);

printf("=\n");

scanner();

expression(shendu); //表达式

if(syn==20)

{

star(shendu);

printf(";\n");

scanner();

sen(shendu-2);

}

else

printf(";Error!\n");

}

else

printf("assSen Error!");

}

//<表达式>→[＋|－]<项>{<加法运算符><项>}

void expression(int shendu)

{

addOpt(shendu); //+-符号

item(shendu); // 项

}

// <项>→<因子>{<乘法运算符><因子>}

void item(int shendu)

{

factor(shendu); // 因子

factorRec(shendu+1); //乘法项

}

void itemRec(int shendu) //??????????

{

addOpt(shendu);

item(shendu);

}

//<项>→<因子>{<乘法运算符><因子>}

void factor(int shendu)

{

if(syn==10)

{

ide(shendu); //标识符

factorRec(shendu);

}

else if(syn==11)

{

uInt(shendu); //整数

factorRec(shendu+1);

}

else if(syn==21) //表达式

{

star(shendu);

printf("(\n");

scanner();

expression(shendu);

if(syn==22)

{

star(shendu);

printf(")\n");

scanner();

factorRec(shendu+1);

}

else

printf("expression Error!");

}

}

//乘法项

void factorRec(int shendu)

{

//star(shendu-1);

//printf("factorRec\n");

addOpt(shendu);

mulOpt(shendu);

factor(shendu);

}

//+-运算符

void addOpt(int shendu)

{

if(syn==23)

{

star(shendu);

printf("+\n");

scanner();

}

else if(syn==24)

{

star(shendu);

printf("-\n");

scanner();

}

}

//\* / 运算符

void mulOpt(int shendu)

{

if(syn==25)

{

star(shendu);

printf("\*\n");

scanner();

}

else if(syn==26)

{

star(shendu);

printf("/\n");

scanner();

}

}

//<条件语句>→if <条件> then <语句>| if <条件> then <语句> else <语句>

void conSen(int shendu)

{

shendu++;

if(syn==3)

{

star(shendu);

printf("if\n");

scanner();

condition(shendu);

if(syn==4)

{

star(shendu);

printf("then\n");

scanner();

sen(shendu-1);

conOpt(shendu);

}

else

printf("then Error!");

}

else

printf("if Error!");

}

void conOpt(int shendu)

{

if(syn==5)

{

star(shendu);

printf("else\n");

scanner(); //少输了else

sen(shendu-1);

}

}

void condition(int shendu) //<条件>→<表达式><关系运算符><表达式>

{

expression(shendu);

relOpt(shendu);

expression(shendu);

}

//关系运算符

void relOpt(int shendu)

{

if(syn==12)

{

star(shendu);

printf(">=\n");

scanner();

}

else if(syn==14)

{

star(shendu);

printf("<=\n");

scanner();

}

else if(syn==15)

{

star(shendu);

printf("<>\n");

scanner();

}

else if(syn==18)

{

star(shendu);

printf("==\n");

scanner();

}

else if(syn==13)

{

star(shendu);

printf(">\n");

scanner();

}

else if(syn==16)

{

star(shendu);

printf("<\n");

scanner();

}

else

printf("relOpt Error!");

}

//<当循环语句>→while <条件> do <语句>

void cirSen(int shendu)

{

shendu++;

if(syn==6)

{

star(shendu);

printf("while\n");

scanner();

condition(shendu);

if(syn==7)

{

star(shendu);

printf("do\n");

scanner();

sen(shendu-1);

}

else

printf("do Error!\n");

}

else

printf("while Error!\n");

}

//<复合语句>→begin <语句>{；<语句>} end

void comSen(int shendu)

{

shendu++;

if(syn==8)

{

star(shendu);

printf("begin\n");

scanner();

sen(shendu-1);

if(syn==9)

{

star(shendu);

printf("end\n");

scanner();

sen(shendu-2);

senRec(shendu-2); //多个语句

}

else

printf("end Error!\n");

}

else

printf("begin Error!\n");

}

void senRec(int shendu)

{

sen(shendu);

}

void scanner()

{

for(i=0;i<100;i++)

scan[i]=NULL;

i=0;

c=word[t++];

while(c==' ' || c=='\n')

{

c=word[t];

t++;

}

if(c>='a' && c<='z' || c>='A' && c<='Z')

{

while(c>='a' && c<='z' || c>='A' && c<='Z' || c>='0' && c<='9')

{

scan[i++]=c;

c=word[t++];

}

scan[i++]='\0';

if(!strcmp(scan,"Const"))

syn=1;

else if(!strcmp(scan,"Var"))

syn=2;

else if(!strcmp(scan,"if"))

syn=3;

else if(!strcmp(scan,"then"))

syn=4;

else if(!strcmp(scan,"else"))

syn=5;

else if(!strcmp(scan,"while"))

syn=6;

else if(!strcmp(scan,"do"))

syn=7;

else if(!strcmp(scan,"begin"))

syn=8;

else if(!strcmp(scan,"end"))

syn=9;

else

syn=10;

t--;

i=0;

}

else if(c>='0' && c<='9')

{

while(c>='0' && c<='9')

{

scan[i++]=c;

c=word[t++];

}

syn=11;

t--;

i=0;

}

//else if(c=='\n')

//line++;

else

{

switch(c)

{

case '>':

i=0;

scan[i++]=c;

c=word[t++];

t--;

if(c=='=')

{

syn=12;

scan[i++]=c;

scan[i]='\0';

i=0;

//printf("%d: %s\n",line,scan);

break;

}

else

{

syn=13;

i=0;

//printf("%d: %s\n",line,scan);

break;

}

case '<':

i=0;

scan[i++]=c;

c=word[t++];

t--;

if(c=='=')

{

syn=14;

scan[i++]=c;

scan[i]='\0';

i=0;

//printf("%d: %s\n",line,scan);

break;

}

else if(c=='>')

{

syn=15;

scan[i++]=c;

scan[i]='\0';

i=0;

//printf("%d: %s\n",line,scan);

break;

}

else

{

syn=16;

i=0;

//printf("%d: %s\n",line,scan);

break;

}

case '=':

i=0;

scan[i++]=c;

c=word[t++];

t--;

if(c=='=')

{

syn=17;

scan[i++]=c;

scan[i]='\0';

i=0;

//printf("%d: %s\n",line,scan);

break;

}

else

{

syn=18;

i=0;

//printf("%d: %s\n",line,scan);

break;

}

case ',':syn=19;scan[0]=c;break;

case ';':syn=20;scan[0]=c;break;

case '(':syn=21;scan[0]=c;break;

case ')':syn=22;scan[0]=c;break;

case '+':syn=23;scan[0]=c;break;

case '-':syn=24;scan[0]=c;break;

case '\*':syn=25;scan[0]=c;break;

case '/':syn=26;scan[0]=c;break;

}

}

}

int main()

{

char ch;

FILE \*fp;

fp=fopen("./examplse2.txt","r");

int p=0;

while((ch=fgetc(fp))!=EOF)

word[p++]=ch;

printf("program\n");

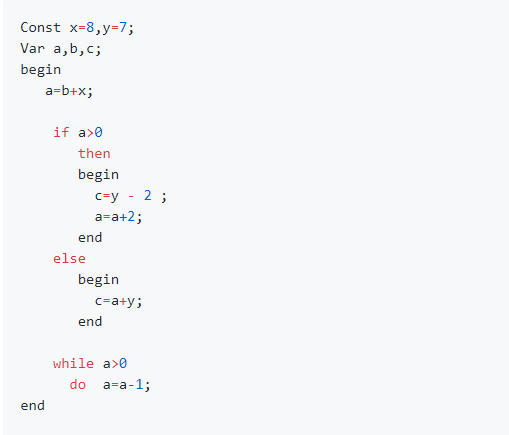
scanner();

program();

fclose(fp);

}

### 测试代码



### 结果展示

