

# 编译原理 实验指导

## 目 录

预备实验 1 文法的读入和输出.....	4
实验一 词法分析程序的设计.....	6
预备实验 2 LL (1) 文法构造.....	8
实验二 递归下降分析法设计与实现.....	11
实验三 算符优先分析法设计与实现.....	13
预备实验 3 逆波兰式的翻译和计算.....	16
实验四 中间代码生成器的设计.....	18



# 预备实验 1 文法的读入和输出

## 一、实验目的

熟悉文法的结构，了解文法在计算机内的表示方法。

## 二、实验内容

- 1、设计一个表示文法的数据结构；
- 2、从文本文件中读入文法，利用定义的数据结构存放文法，并输出；
- 3、本实验结果还将用于实验 3。

## 三、实验要求

- 1、了解文法定义的 4 个部分：

$G(V_n, V_t, S, P)$

$V_n$  文法的非终结符号集合，在实验中用大写的英文字母表示；

$V_t$  文法的终结符号集合，在实验中用小写的英文字母表示；

$S$  开始符号，在实验中是  $V_n$  集合中的一个元素；

$P$  产生式，分左部和右部，左部为非终结符号中的一个，右部为终结符号或非终结符号组成的字符串，如  $S \rightarrow ab|c$

- 2、根据文法各个部分的性质，设计一个合理的数据结构用来表示文法，
  - 1) 若使用 C 语言编写，则文法可以设计成结构体形式，结构体中应包含上述的 4 部分，
  - 2) 若使用 C++ 语言编写，则文法可以设计成文法类形式，类中至少含有 4 个数据成员，分别表示上述 4 个部分文法数据结构的具体设计由学生根据自己想法完成，并使用 C 或 C++ 语言实现设计的数据结构。
- 3、利用完成的数据结构完成以下功能：
  - 1) 从文本文件中读入文法（文法事先应写入文本文件）；
  - 2) 根据文法产生式的结构，分析出文法的 4 个部分，分别写入定义好的文法数据结构中的相应部分；
  - 3) 整理文法的结构；
  - 4) 在计算机屏幕或者文本框中输出文法，文法输出按照一个非终结符号一行，开始符号引出的产生式写在第一行，同一个非终结符号的候选式用“|”分隔的方式输出。

## 四、实验环境

PC 微机

DOS 操作系统或 Windows 操作系统

Turbo C 程序集成环境或 Visual C++ 程序集成环境

## 五、实验步骤

- 1、根据文法定义，设计出文法数据结构
- 2、用学生选择的语言，实现文法的数据结构
- 3、编写调试文法读入和输出程序，
- 4、测试程序运行效果：从文本文件中读入一个文法，在屏幕上输出，检查输出结果。

## 六、测试数据

输入数据：

编辑一个文本文件 g.txt，在文件中输入如下内容：

S->Qc;

S->c;

Q->Rb;

Q->b;

R->Sa;

R->a;

正确结果：

上述文法整理后的输出形式：

S->Qc|c;

Q->Rb|b;

R->Sa|a;

## 七、实验报告要求

实验报告应包括以下几个部分：

- 1、文法数据结构的设计和实现；
- 2、文法的读入算法
- 3、文法的输出方法
- 4、程序的测试结果和问题
- 5、实验总结

## 八、思考题

- 1、如何让设计的文法结构满足各种文法的要求？
- 2、如何设计文法才能跟简单地表示文法，同时又降低程序编写难度？

# 实验一 词法分析程序的设计

## 一、实验目的

掌握计算机语言的词法分析程序的开发方法。加深对词法分析器的工作过程的理解；加强对词法分析方法的掌握；能够采用一种编程语言实现简单的词法分析程序；能够使用自己编写的分析程序对简单的程序段进行词法分析。

## 二、实验内容

编制一个能够分析三种整数、标识符、主要运算符和主要关键字的词法分析程序。自定义一种程序设计语言，或者选择已有的一种高级语言，编制它的词法分析程序。从输入的源程序中，识别出各个具有独立意义的单词，即关键字、标识符、常数、运算符、界符。并依次输出各个单词的内部编码及单词符号自身值。（遇到错误时可显示“Error”，然后跳过错误部分继续显示）

## 三、实验要求

1、根据以下的正规式，编制正规文法，画出状态图：

标识符	$\langle \text{字母} \rangle (\langle \text{字母} \rangle   \langle \text{数字字符} \rangle)^*$
十进制整数	$0   ( (1 2 3 4 5 6 7 8 9) (0 1 2 3 4 5 6 7 8 9)^* )$
八进制整数	$0 (1 2 3 4 5 6 7) (0 1 2 3 4 5 6 7)^*$
十六进制整数	$0x (0 1 2 3 4 5 6 7 8 9 a b c d e f) (0 1 2 3 4 5 6 7 8 9 a b c d e f)^*$
运算符和界符	$+ \ - \ * \ / \ > \ < \ = \ ( \ ) \ ;$
关键字	<code>if then else while do main</code>

2、根据状态图，设计词法分析函数 `int scan()`，完成以下功能：

- 1）从文本文件中读入测试源代码，根据状态转换图，分析出一个单词，
- 2）以二元式形式输出单词(单词种类，值)

关键字：`if`、`int`、`for`、`while`、`do`、`return`、`break`、`continue`；单词种别码为 1。

标识符；单词种别码为 2。

常数为无符号整形数；单词种别码为 3。

运算符包括：`+`、`-`、`*`、`/`、`=`、`<`、`>`、`==`、`<=`、`!=`；单词种别码为 4。

分隔符包括：`,`、`;`、`{`、`}`、`(`、`)`；单词种别码为 5。3、编写测试程序，反复调用函数 `scan()`，输出单词种别和属性。

## 四、实验环境

PC 微机

DOS 操作系统或 Windows 操作系统

codeblocks 程序集成环境或 Visual C++ 程序集成环境

## 五、实验步骤

- 1、根据正规式，画出状态转换图；
- 2、根据状态图，设计词法分析算法；

- 3、采用 C 或 C++ 语言，设计函数 `scan()`，实现该算法；
- 4、编制测试程序（主函数 `main`）；
- 5、调试程序：读入文本文件，检查输出结果。

## 六、测试数据

输入数据举例：

编辑一个文本文件 `program.txt`，在文件中输入如下内容：

```
main()
{
int  a,b;
a = 10;
b = a + 20;
}
```

正确结果：

```
(2," main")
(5," ("
(5," )"
(5," { "
```

```
(1," int")
(2," a")
(5," ,"
(2," b")
(5," ;")
(2," a")
(4," ="
(3," 10")
(5," ;")
(2," b")
(4," ="
(2," a")
(4," +"
(3," 20")
(5," ;")
(5," } "
```

## 七、实验报告要求

实验报告应包括以下几个部分：

- 1、词法的正规式描述；
- 2、变换后的 状态图；

- 3、词法分析程序的数据结构与算法。
- 4、实验记录（实验时间、实验中遇到的问题及解决方法、总结体会）
- 5、报告主要包括：实验目的、要求、内容、步骤；软件设计要有总体设计、模块划分、流程图、关键算法、重要数据结构；测试结果及分析等。

## 八、思考题

- 1、词法分析能否采用空格来区分单词？
- 2、程序设计中哪些环节影响词法分析的效率？如何提高效率？

## 预备实验 2 LL(1) 文法构造

### 一、实验目的

熟悉 LL(1) 文法的分析条件，了解 LL(1) 文法的构造方法。

### 二、实验内容

- 1、编制一个能够将一个非 LL(1) 文法转换为 LL(1) 文法；
- 2、消除左递归；
- 3、消除回溯。

### 三、实验要求

- 1、将一个可转换非 LL(1) 文法转换为 LL(1) 文法，要经过两个阶段，1) 消除文法左递归，2) 提取左因子，消除回溯。
- 2、提取文法左因子算法：

1) 对文法 G 的所有非终结符进行排序

2) 按上述顺序对每一个非终结符  $P_i$  依次执行：

for(  $j=1$ ;  $j < i-1$ ;  $j++$ )

将  $P_j$  代入  $P_i$  的产生式（若可代入的话）；

消除关于  $P_i$  的直接左递归：

$P_i \rightarrow P_i \alpha \mid \beta$ ，其中  $\beta$  不以  $P_i$  开头，则修改产生式为：

$P_i \rightarrow \beta P_i'$

$P_i' \rightarrow \alpha P_i' \mid \varepsilon$

3) 化简上述所得文法。

- 3、提取左因子的算法：

$A \rightarrow \delta \beta_1 \mid \delta \beta_2 \mid \dots \mid \delta \beta_n \mid \gamma_1 \mid \gamma_2 \mid \dots \mid \gamma_m$

(其中, 每个  $\gamma$  不以  $\delta$  开头)

那么, 可以把这些产生式改写成

$A \rightarrow \delta A' \mid \gamma_1 \mid \gamma_2 \dots \mid \gamma_m$

$A' \rightarrow \beta_1 \mid \beta_2 \mid \dots \mid \beta_n$

- 4、利用上述算法，实现构造一个 LL(1) 文法：

- 1) 从文本文件 g.txt 中读入文法，利用实验 1 的结果，存入实验 1 设计的数据结构；



- 2) 设计函数 `remove_left_recursion()` 和 `remove_left_gene()` 实现消除左递归和提取左因子算法，分别对文法进行操作，消除文法中的左递归和提出左因子；
- 3) 整理得到的新文法；
- 4) 在一个新的文本文件 `newg.txt` 输出文法，文法输出按照一个非终结符号一行，开始符号引出的产生式写在第一行，同一个非终结符号的候选式用“|”分隔的方式输出。

## 四、实验环境

PC 微机

DOS 操作系统或 Windows 操作系统

codeblocks 程序集成环境或 Visual C++ 程序集成环境

## 五、实验步骤

- 1、学习 LL(1) 文法的分析条件；
- 2、学习构造 LL(1) 文法的算法；
- 3、结合实验 1 给出的数据结构，编程实现构造 LL(1) 文法的算法；
- 4、结合实验 1 编程和调试实现对一个具体文法运用上述算法，构造它的 LL(1) 文法形式；
- 5、把实验结果写入一个新建立的文本文件。

## 六、测试数据

输入数据：

编辑一个文本文件 `g.txt`，在文件中输入如下内容：

```
S->Qc|c|cab;  
Q->Rb|b;  
R->Sa|a;
```

正确结果：

本实验的输出结果是不唯一的，根据消除左递归是选择非终结符号的顺序不同，或选择新的非终结符号的不同，可能会得到不同的结果，下面只是可能的一个结果：

```
S->Qc|cT;  
T->@|ab; //由于无法输出  $\epsilon$ ，用@代替  
Q->Rb|b;  
R->bcaU|caU|cabaU|aU;  
U->bcaU|@;
```

## 七、实验报告要求

实验报告应包括以下几个部分：

- 1、满足 LL(1) 文法的分析条件；
- 2、构造 LL(1) 文法的算法；
- 3、消除左递归文法和提取左因子算法实现方法；

- 4、整个测试程序的流程；
- 5、程序的测试结果和问题；
- 6、实验总结。

## 八、思考题

- 1、是不是所有的文法都可以通过上述程序构造 LL（1）文法？
- 2、LL（1）文法在整个语法分析中的作用？
- 3、预备实验 1 中设计的文法数据结构对本实验的影响？
- 4、如何更好地组合预备实验 1 和实验一，使之具有更高的效率？

## 实验二 递归下降分析法设计与实现

### 一、实验目的

通过设计、编制、调试一个典型的语法分析程序，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，进一步掌握常用的语法分析中递归下降分析方法。

### 二、实验内容

设计一个文法的递归下降分析程序，判断特定表达式的正确性。

### 三、实验要求

- 1、给出文法如下：

$G[E]$

$E \rightarrow T|E+T;$

$T \rightarrow F|T * F;$

$F \rightarrow i(E);$

利用预备实验 2 的方法将上述文法转化为 LL(1) 文法；

- 2、根据递归下降分析方法为转化后的文法设计递归下降分析程序，利用 C 语言或 C++ 语言实现；
- 3、利用递归下降分析程序完成下列功能：
  - 1) 手工将测试的表达式写入文本文件，每个表达式写一行，用“;”表示结束；
  - 2) 读入文本文件中的表达式；
  - 3) 调用实验一中的词法分析程序搜索单词；
  - 4) 把单词送入递归下降分析程序，判断表达式是否正确（是否是给出文法的语言），若错误，应给出错误信息；
  - 5) 完成上述功能，有余力的同学可以对正确的表达式计算出结果。

### 四、实验环境

PC 微机

DOS 操作系统或 Windows 操作系统

codeblocks 程序集成环境或 Visual C++ 程序集成环境

### 五、实验步骤

- 1、分析文法，将给出的文法转化为 LL(1) 文法；
- 2、学习递归下降分析程序的结构，设计合理的递归下降分析程序；
- 3、编写测试程序，包括表达式的读入和结果的输出；
- 4、测试程序运行效果，测试数据可以参考下列给出的数据。

## 六、测试数据

输入数据举例：

编辑一个文本文件 `expression.txt`，在文件中输入如下内容：

```
10;
1+2;
(1+2)*3+(5+6*7);
((1+2)*3+4;
1+2+3+(*4+5);
(a+b)*(c+d);
((ab3+de4)**5)+1;
```

正确结果：

- (1) 10;  
输出：正确
- (2) 1+2;  
输出：正确
- (3) (1+2)\*3+(5+6\*7);  
输出：正确
- (4) ((1+2)\*3+4  
输出：错误
- (5) 1+2+3+(\*4+5)  
输出：错误
- (6) (a+b)\*(c+d)  
输出：正确
- (7) ((ab3+de4)\*\*5)+1  
输出：错误

## 七、实验报告要求

实验报告应包括以下几个部分：

1. 给定文法的 LL (1) 形式；
2. 递归下降分析程序的算法和结构；
3. 程序运行流程；
4. 程序的测试结果和问题；
5. 实验记录（实验时间、实验中遇到的问题及解决方法、总结体会）
6. 报告主要包括：实验目的、要求、内容、步骤；软件设计要有总体设计、模块划分、流程图、关键算法、重要数据结构；测试结果及分析等。

## 八、思考题

- 1、为什么文法必须先转化为 LL (1) 文法再来做递归下降分析？
- 2、如果用预测分析法来改写本程序，如何实现？

## 实验三 算符优先分析法设计与实现

### 一、实验目的

通过设计、编制、调试一个典型的语法分析程序，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，进一步掌握常用的语法分析中算符优先分析方法。

### 二、实验内容

设计一个文法的算符优先分析程序，判断特定表达式的正确性。

### 三、实验要求

1、给出文法如下：

```
G[E]
E->T|E+T;
T->F|T*F;
F->i(E);
```

可以构造算符优先表如下：

	+	*	(	)	i
+	$\triangleright$	$\triangleleft$	$\triangleleft$	$\triangleright$	$\triangleleft$
*	$\triangleright$	$\triangleright$	$\triangleleft$	$\triangleright$	$\triangleleft$
(	$\triangleleft$	$\triangleleft$	$\triangleleft$	$\equiv$	$\triangleleft$
)	$\triangleright$	$\triangleright$		$\triangleright$	
i	$\triangleright$	$\triangleright$		$\triangleright$	

2、计算机中表示上述优先关系，优先关系的机内存放方式有两种 1) 直接存放，2) 为优先关系建立优先函数，这里由学生自己选择一种方式；

3、给出算符优先分析算法如下：

```
k:=1;    S[k]:='#';
REPEAT
  把下一个输入符号读进 a 中;
  IF S[k] ∈ VT THEN j:=k ELSE j:=k-1;
  WHILE S[j]  $\triangleright$  a DO
  BEGIN
    REPEAT
      Q:=S[j];
      IF S[j-1] ∈ VT THEN j:=j-1 ELSE j:=j-2
    UNTIL S[j]  $\triangleleft$  Q
```

```

    把 S[j+1]...S[k]归约为某个 N;
    k:=j+1;
    S[k]:=N;
END OF WHILE;
IF S[j] <= a OR S[j] = a THEN
BEGIN
    k:=k+1;S[k]:=a
END
ELSE ERROR
UNTIL a='#'

```

- 4、根据给出算法，利用适当的数据结构实现算符优先分析程序；
- 5、利用算符优先分析程序完成下列功能：
  - 1) 手工将测试的表达式写入文本文件，每个表达式写一行，用“;”表示结束；
  - 2) 读入文本文件中的表达式；
  - 3) 调用实验一中的词法分析程序搜索单词；
  - 4) 把单词送入算法优先分析程序，判断表达式是否正确（是否是给出文法的语言），若错误，应给出错误信息；
  - 5) 完成上述功能，有余力的同学可以对正确的表达式计算出结果。

## 四、实验环境

PC 微机

DOS 操作系统或 Windows 操作系统

codeblocks 程序集成环境或 Visual C++ 程序集成环境

## 五、实验步骤

- 1、分析文法中终结符号的优先关系；
- 2、存放优先关系或构造优先函数；
- 3、利用算符优先分析的算法编写分析程序；
- 4、写测试程序，包括表达式的读入和结果的输出；
- 5、程序运行效果，测试数据可以参考下列给出的数据。

## 六、测试数据

输入数据：

编辑一个文本文件 expression.txt，在文件中输入如下内容：

```

10;
1+2;
(1+2)*3+(5+6*7);
((1+2)*3+4;
1+2+3+(*4+5);
(a+b)*(c+d);
((ab3+de4)**5)+1;

```

正确结果:

(1) 10;

输出: 正确

(2) 1+2;

输出: 正确

(3)  $(1+2)*3+(5+6*7)$ ;

输出: 正确

(4)  $((1+2)*3+4$

输出: 错误

(5)  $1+2+3+(*4+5)$

输出: 错误

(6)  $(a+b)*(c+d)$

输出: 正确

(7)  $((ab3+de4)**5)+1$

输出: 错误

## 七、实验报告要求

实验报告应包括以下几个部分:

1. 给定文法优先关系和存放方式;
2. 算符优先分析程序的算法和结构;
3. 程序运行流程;
4. 程序的测试结果和问题;
5. 实验记录 (实验时间、实验中遇到的问题及解决方法、总结体会)
6. 报告主要包括: 实验目的、要求、内容、步骤; 软件设计要有总体设计、模块划分、流程图、关键算法、重要数据结构; 测试结果及分析等。

## 八、思考题

- 1、算符优先关系表示的是一种什么样的关系, 它在自下而上的分析中起到了什么作用?
- 2、比较本实验和实验二, 实现同样的功能, 两种方法有什么不同?

## 预备实验 3 逆波兰式的翻译和计算

### 一、实验目的

通过实验加深对语法指导翻译原理的理解，掌握算符优先分析的方法，将语法分析所识别的表达式变换成中间代码的翻译方法。

### 二、实验内容

设计一个表示能把普通表达式（中缀式）翻译成后缀式，并计算出结果的程序。

### 三、实验要求

1、给出文法如下：

```
G[E]
E->T|E+T;
T->F|T*F;
F->i(E);
```

对应的转化为逆波兰式的语义动作如下：

```
E-> E(1) op E(2)      { E.CODE := E(1).CODE || E(2).CODE || op }
E-> (E(1))             { E.CODE := E(1).CODE }
E-> id                  { E.CODE := id }
```

2、利用实验 5 中的算符优先分析算法，结合上面给出的语义动作实现逆波兰式的构造；

3、利用栈，计算生成的逆波兰式，步骤如下：

- 1）中缀表达式，从文本文件读入，每一行存放一个表达式，为了降低难度，表达式采用常数表达式；
- 2）利用结合语法制导翻译的算符优先分析，构造逆波兰式；
- 3）利用栈计算出后缀式的结果，并输出；

### 四、实验环境

PC 微机

DOS 操作系统或 Windows 操作系统

codeblocks 程序集成环境或 Visual C++ 程序集成环境

### 五、实验步骤

- 1、了解语法制导翻译的方法，学习后缀式构造的语义动作；
- 2、结合实验三的算符优先程序，设计程序构造后缀式；
- 3、利用栈，编程实现后缀式的计算；
- 4、测试程序运行效果：从文本文件中读表达式，在屏幕上输出，检查输出结果。

### 六、测试数据

输入数据：

编辑一个文本文件 expression.txt，在文件中输入如下内容：



1+2;  
(1+2)\*3;  
(10+20)\*30+(50+60\*70);

正确结果:

(1) 1+2;

输出: 1,2,+      3

(2) (1+2)\*3;

输出: 1,2,+,3,\*      9

(3) (10+20)\*30+(50+60\*70)

输出: 10,20,+,30,\*50,60,70,\*,+,+      5150

## 七、实验报告要求

实验报告应包括以下几个部分:

- 1、构造逆波兰式的语义动作;
- 2、结合算符优先分析构造逆波兰式的算法和过程;
- 3、语法制导翻译的运行方法;
- 4、程序的测试结果和问题;
- 5、实验总结。

## 八、思考题

- 1、语法制导翻译的工作方式?
- 2、为什么编译程序要设计生成中间代码方式?

## 实验四 中间代码生成器的设计

### 一、实验目的

掌握计算机语言的语法分析程序设计与属性文法应用的实现方法。

### 二、实验内容

编制一个能够进行语法分析并生成三地址代码的微型编译程序。

### 三、实验要求

- 1、考虑下述语法制导定义中文法，采用递归子程序法，改写文法，构造语法分析程序；
- 2、考虑下述语法制导定义中语义规则，改写语法分析程序，构造三地址代码生成程序。

产生式	语义规则
$S \rightarrow id = E ;$	$S.code = E.code \parallel gen(id.place' := E.place)$
$S \rightarrow if C then S$	$C.true = newlabel;$ $C.false = S.next;$ $S1.next = S.next;$ $S.code = C.code \parallel gen(E.true' :') \parallel S1.code$
$S \rightarrow while C do S$	$S.begin = newlabel;$ $C.true = newlabel;$ $C.false = S.next;$ $S1.next = S.begin;$ $S.code = gen(S.begin' :') \parallel C.code \parallel$ $gen(E.true' :') \parallel S1.code \parallel gen('goto' S.begin);$
$C \rightarrow E1 > E2$	$C.code = E1.code \parallel E2.code \parallel$ $gen('if' E1.place' > E2.place' goto' C.true) \parallel$ $gen('goto' C.false)$
$C \rightarrow E1 < E2$	$C.code = E1.code \parallel E2.code \parallel$ $gen('if' E1.place' < E2.place' goto' C.true) \parallel$ $gen('goto' C.false)$
$C \rightarrow E1 = E2$	$C.code = E1.code \parallel E2.code \parallel$ $gen('if' E1.place' = E2.place' goto' C.true) \parallel$ $gen('goto' C.false)$
$E \rightarrow E1 + T$	$E.place = newtemp;$ $E.code = E1.code \parallel T.code \parallel$ $gen(E.place' := E1.place' + T.place)$
$E \rightarrow E1 - T$	$E.place = newtemp;$ $E.code = E1.code \parallel T.code \parallel$ $gen(E.place' := E1.place' - T.place)$
$T \rightarrow T1 * F$	$T.place = newtemp;$ $T.code = T1.code \parallel F.code \parallel$ $gen(T.place' := T1.place' * F.place)$

T→T1 / F	T.place = newtemp; T.code = T1.code    F.code    gen(T.place := ' T1.place' /' F.place)
F→( E )	F.place = E.place; F.code = E.code
F→id	F.place = id.name; F.code = ' '
F→int8	F.place = int8.value; F.code = ' '
F→int10	F.place = int10.value; F.code = ' '
F→int16	F.place = int16.value; F.code = ' '

## 四、实验环境

PC 微机

DOS 操作系统或 Windows 操作系统

codeblocks 程序集成环境或 Visual C++ 程序集成环境

## 五、实验步骤

- 1、考虑给定的文法，消除左递归，提取左因子。
- 2、编制并化简语法图
- 3、编制递归子程序的算法
- 4、编制各个递归子程序函数
- 5、连接实验一的词法分析函数 scan( )，进行测试
- 6、设计三地址代码生成的数据结构和算法
- 7、将各个递归子程序函数改写为代码生成函数
- 8、编制测试程序（main 函数）
- 9、调试程序：输入一个语句，检查输出的三地址代码

## 六、测试数据

输入数据举例：while (a3+15)>0xa do if x2 = 07 then while y<z do y = x \* y / z;

正确结果：等效的三地址代码序列

```

L1: t1 := a3 + 15
    if t1 > 10 goto L2
    goto L0
L2:
L3:
    if x2 > 7 goto L4
    goto L1
L4: if y < z goto L5
    goto L1
L5: t2 = x * y

```

```
t3 = t2 / z
y = t3
goto L3
goto L1
L0:    // S.next
```

## 七、实验报告要求

实验报告应包括以下几个部分：

- 1、语法制导定义
- 2、改写后的产生式集合
- 3、化简后的语法图
- 4、递归子程序的算法
- 5、三地址代码生成器的数据结构
- 6、程序结构的说明
- 7、实验记录（实验时间、实验中遇到的问题及解决方法、总结体会）
- 8、报告主要包括：实验目的、要求、内容、步骤；软件设计要有总体设计、模块划分、流程图、关键算法、重要数据结构；测试结果及分析等。

## 八、思考题

- 1、生成的三地址代码可否直接输出（不采用数据结构来实现属性 code）？
- 2、如何保证四则运算的优先关系和左结合性？
- 3、如何采用代码段相对地址代替三地址代码序列中的标号？

