

**《编译原理》**

**实验五实验报告**

|  |  |
| --- | --- |
|  |  |
| 学 号： |  |
| 姓 名： |  |
| 学 院： |  |
| 日 期： |  |

## 一、 理论传授

编译器前端的组成，理解编译程序的整体概念，语法制导翻译的基本概念和实现方法。

## 二、 目标任务

**[实验项目]** 将词法分析程序设计原理与实现（专题 1）和基于 SLR(1)分析法的语法制导翻译及中间代码生成程序设计原理与实现（专题 5）形成一个程序，即程序的输入为符号 串源程序，输出为中间代码四元式序列。

**[设计要求]**（1）词法分析的输入为符号串，词法分析的输出为二元式形式的中间文件；

（2）语法制导翻译读取二元式形式的中间文件并生成中间代码四元式形式的中间文件；

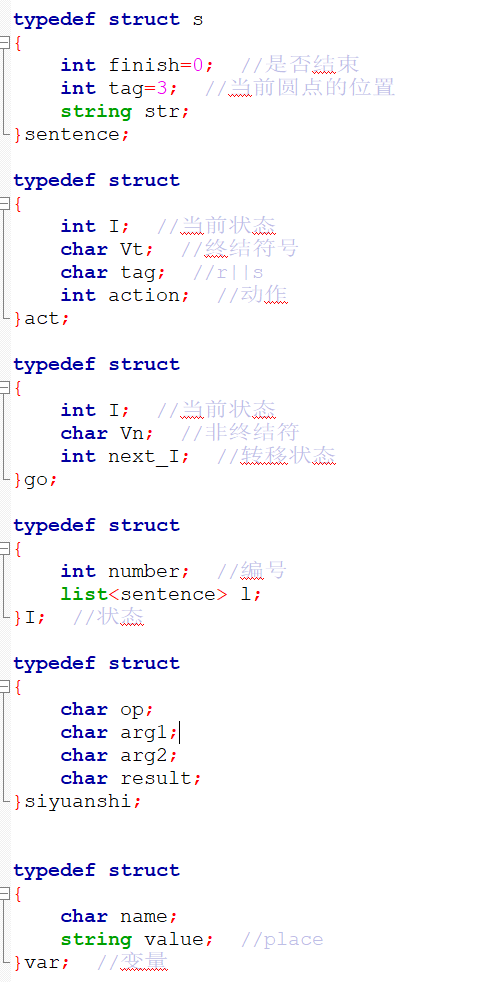
（3） 设计两个测试用例（尽可能完备），并给出程序执行结果。

三、任务分析

重点解决编译程序总体结构理解问题；词法分析的输出为语法分析的输入，语法分析和语义分析同时完成，即语法制导翻译

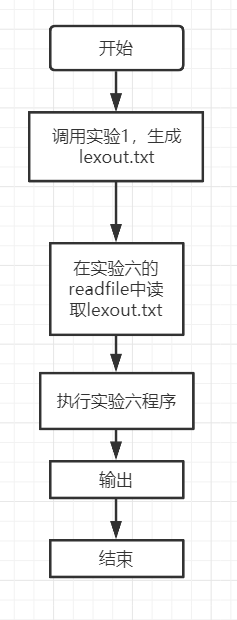
## 四、数据结构描述

|  |  |
| --- | --- |
| 变量及类型 | 用途 |
| sentence | 用于存放输入的句子，并判断是否匹配 |
| act | 用于存放状态与动作 |
| go | 用于存放当前状态，非终结符和转移状态 |
| I | 用于存放算出的闭包I |
| Siyuanshi | 用于存放和输出四元式 |
| Var | 用于存放变量 |
| list<I> DFA; | 用于存放状态集 |
| char input[100]; | 用于读取词法分析的输入 |
| list<char> Vn; | 用于存放非终结符号集 |
| list<char> Vt; | 用于存放终结符号集 |
| list<go> GOTO; | GO[i][j] |
| list<act> ACTION; | ACTION[i][j] |
| list<char>\* Follow; | 用于存放FOLLOW集 |
| list<char>\* First; | 用于存放FIRST集 |

****

### 五、程序结构描述

1、程序流程图：



本次实验结合了实验一与实验五，在控制台输入要识别的句子，调用实验一得到二元式，输出到result.txt文件中，实验五再读取result.txt，进行SLR（1）分析

2、与分专题关系；

如图，由于本次实验是两个实验的调用，因此主要需要处理好接口之间的关系。词法分析的输出为语法分析的输入，语法分析和语义分析同时完成，即语法制导翻译。

实验五的内容是基于 SLR(1)分析法的语法制导翻译及中间代码生成程序设计原理与实验，而实验一的内容是词法分析程序设计原理与实现并生成二元式。因此需要把实验一生成二元式的输出结果输出到文件，再作为实验五的文件输入即可。

具体接口处代码为：

fp1 = fopen("result.txt", "w");

fp = fopen("result.txt", "r");

fp2 = fopen("siyuanshi.txt", "w");

bool r = AnalyzeProcess(fp);

if (r)

printf("输入串分析成功\n");

else

printf("输入串不匹配\n");

这里要注意的点为实验一完成后内容是写入result.txt中，即覆盖关系。所以在实验五代码使用的时候需要重新打开result.txt

4.3程序设计方法、函数定义及函数之间的调用关系等；

1. void InitKeyword()

功能：初始化关键字表

实现：将关键字表定义为vector结构，逐个插入。

1. void InitOperator()

功能：初始化运算符表

实现：将运算符表定义为vector结构，逐个插入。

1. void InitBoundary()

功能：初始化界符表

实现：将界符表定义为vector结构，逐个插入。

1. void InitAll()

功能：调用InitKeyword();InitOperator();InitBoundary()总体初始化，包括初始化关键字表、运算符表、界符表、总表，初始化全局变量

实现：调用逐个初始化，将关键字表、运算符表、界符表所汇总的总表定义为map结构、各个子表起始位置不同，可实现一个符号对应一个种类码。

1. bool IsLetter()

功能：判断character是否为字母

实现：与‘A’‘Z’‘a’‘z’作比较

1. bool IsNum()

功能：判断character是否为数字

实现：与‘0’‘9’做比较

1. void retract()

功能：字符回退

实现：index--，character=’’

1. char GetChar()

功能：读取一个字符到全局变量character中

实现：取instr中index位置的字符，index++

1. void removeBlank()

功能：去掉空格和tab

实现：当character为空格和tab时跳过

1. void concat()

功能：连接字符串

实现：token = token + character;

1. int matchKeyword()

功能：关键字表匹配

实现：for循环逐个判断

1. string matchSymbol()

功能：标识符表匹配，当单词不在标识符表中时，添加该单词到标识符表中

实现：通过迭代器，当ite= Symbol.find(token)，ite为Symbol.end()时，说明该单词不在标识符表中。

1. string matchNum()

功能：常熟表匹配，当常数不在常数表中时，添加该数到常数表中

实现：与（12）类似

1. Binary scanner()

功能：词法分析扫描器，将输入的程序拆分，以二元组（符号，种类码）的形式输出。

实现：从输入缓冲区读取一个字符到character中，去掉空格和tab，判断是否为单词，为单词判断是关键字还是标识符，是否为数字，若都不是则为符号，逐个符号判断，若该符号可能会后接一个符号如++，--，+=，再读入一个字符，进行判断。

1. void showtable()

功能：展示表

实现：逐个打印各个表

（16）readfile()

读取文件，将文件中的实验一的运行结果读入，整合为字符数组，准备好以待匹配。

（17）scan()

扫描文法：scan()将文法数组G[]中的string字符串进行扫描，构造终结符号集和非终结符号集；之后按照SLR(1)文法中FIRST集和FOLLOW集的构造方法进行这两个集合的构造。在构造FIRST集时，用到了find函数，用来不断地递归查找产生式首符号为非终结符号的产生式，以构造FIRST集；在构造FOLLOW集时，使用到了findVn函数，用来返回指定的非终结符对应的标号，以进行FOLLOW集的计算。同时，通过不断的循环、加入，当所有非终结符的FOLLOW集总大小不变时，跳出循环，构造完成。至此，造表前的准备工作全部完成，接下来进入造表阶段。

（18）table()

生成表：table()完成整个项目集族的生成以及ACTION/GOTO的构造，以及在造表过程中对于四元式的标注，也是个人认为整个程序中最难的部分。

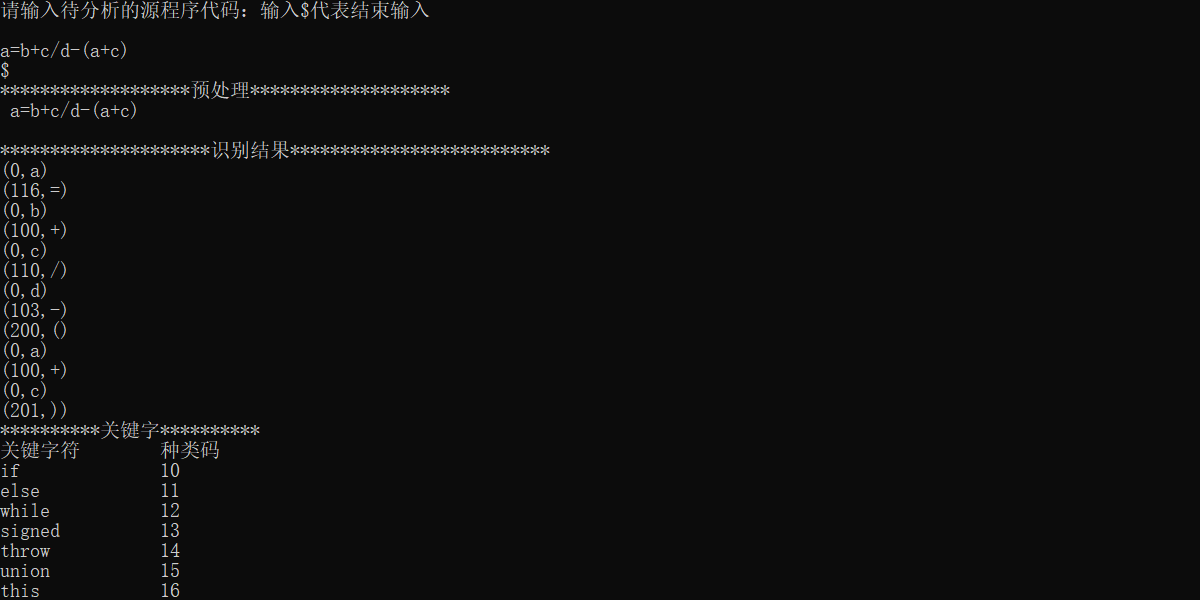
首先，进行初始设定：添加文法的起始符号对应的产生式，用以从初态进行迭代；之后将该产生式(点还处在最开始的部位tag=3, 尚未进入规约状态finish=0)加入第一个状态I0中，接下来进行一波循环操作：首先对于当前所有的项目集族进行闭包操作，该功能被封装在闭包函数closure()中（函数中对于每一个点操作的非终结符均进行闭包操作，加入该状态中，最终返回经过闭包操作的该状态）；之后对于当前状态的每一条待移进的产生式分别进行移进操作，并添加为新的状态，等待循环到该状态时的闭包操作。在移进过程中，对于不是规约的（即尚可移进的）产生式，若点操作为非终结符，置GOTO(当前状态, 非终结符) = 新状态号；若为终结符，则置ACTION(当前状态，终结符号) = s+新状态号。若存在移进时的相同符号，则加入同一状态中，不再另设新状态。之后即为对于新状态的加入判断：若新状态的闭包操作closure()已经在已形成的项目集族中出现了，则不再添加这个新状态，并且修改对应的ACTION或者GOTO（一个pop()操作+一个push()操作）；否则添加。对于需要规约的产生式，首先对其进行判断是否可以产生四元式：若可以，则进行记录，以便之后判断；否则不进行记录。至此，造表结束，接下来正式进入分析阶段。

（19）SLR()

SLR分析：SLR()完成最终整个SLR(1)文法的分析任务，也包括分析过程中完成的四元式构建。在分析过程起始时，首先将状态I0压入栈中，分析开始：若ACTION(栈顶，当前符号)不是acc（在程序中为r0），则开始循环：如果是err（在程序中使用0代替），报错；否则如果是sj，状态栈、符号栈、属性栈均压栈，读取下一个符号；如果是ri，则通过需要规约第i条产生式，对于三个栈，执行第i条产生式右部的符号数次pop()操作，同时更新属性栈，用以后续的四元式生成工作；在生成四元式时，首先判断正在规约的产生式是否可以产生四元式，同时确定四元式的op，其次判断两个参数（arg1，arg2）是否为变量or变量组成的表达式。如果是表达式，则查找之前的符号表，进行代替操作。若中间不报错而跳出循环，则成功匹配。至此，按照事先约定好的输出格式进行输出，分析任务完成。

## 五、程序测试：测试用例、测试结果

a=b+c/d-(a+c)





## 六、实验体会和总结

本次实验是实验一与实验五的结合，构建了编译器前端的组成，理解编译程序的整体概念，语法制导翻译的基本概念和实现方法。本实验结合1-5，完成了词法分析，语法分析和中间代码生成（四元式）的内容。使我更进一步了解了编译的过程和原理。