# 词法分析程序设计

## 程序功能描述

输入C语言源程序文件，可以对其进行词法分析，词法分析结果为二元式序列组成的中间文件，二元式序列包括单词类别编号，若为标识符则进一步输出内容。输出结果保存在中间文件中。

## 数据结构描述

程序设立了一个保留字数组用以存放C语言的保留字，数组元素类型为C++的string类型。

## 程序结构描述

## 3.1设计方法

设计的文法如下所示，

<标识符>→字母︱ <标识符>字母︱ <标识符>数字

<无符号整数>→数字︱ <无符号整数>数字

<单字符分界符> →+ ︱－ ︱\* ︱；︱, ︱(︱) ︱{︱}

<双字符分界符>→<大于>=︱<小于>=︱<小于>>︱<感叹号>=︱<等于>=︱<斜竖>\*

<小于>→< <等于>→= <大于>→> <斜竖> →/

<感叹号>→!

<注释1>→<斜竖><斜竖>

<注释2>→<注释前半部><注释后半部>

<注释前半部>→<斜竖>\*

<注释后半部>→\*<斜竖>

<自增运算符>→++ <自减运算符>→--

<左移运算符>→<< <右移运算符>→>>

<加等于>→+<等于> <减等于>→-<等于>

<乘等于>→\*<等于> <除等于>→/<等于>

<与运算>→& <或运算>→|

<逻辑与>→&& <逻辑或>→||

<双引号>→” <单引号>→’

<结束符号>→#

保留字包括：void、int、float、double、if、else、for、do、while

此外，该语言设计成大小写不敏感；用户定义的标识符最长不超过32个字符；字母为 a-z A-Z，数字为 0-9； “/\*……\*/”和“//”(一行内)为程序的注释部分。

状态转换图请见附录。

## 3.2程序模块

程序分为三个模块: main.cpp、lex.cpp和lex.h。main.cpp负责读取源程序文件，将文件内容传送给lex.cpp的scan扫描函数进行处理，输出至中间文件；lex.cpp负责词法分析，包括扫描源文件、判断单词、输出结果；lex.h中定义了一系列宏，这些宏的含义是C语言的保留字、标识符和各种其他符号的助记符，用于词法结果的输出。

这里做一个说明，本词法分析程序并不能识别头文件。

## 3.3函数说明

lex.cpp包括isAlpha、isNumber、lookup、out和scan这五个函数。isAlpha函数输入一个字符判断是否为英文字母；isNumber函数输入一个字符判断是否为数字0-9；lookup函数输入一个字符串token，在保留字数组中查找token是否存在，若存在则返回保留字的宏，否则返回0；out函数传入单词类别编号、单词和中间文件，将二元组写入中间文件；scan函数传入一个源程序文件和中间文件，从源程序文件中成功识别一个单词后将其二元组表示输出到中间文件中。

在主函数中，当文件未结束时，每识别出一个单词，就将word清空，进行下一轮循环。

在lex.cpp中设置全局变量endFile、word、row和col。endFile是文件结束的标志；word用来存放一个单词；row和col用于记录扫描过程中当前所在的行与列。

重点说明scan函数，此函数按照标识符、无符号整数、其余符号三个顺序来进行单词识别。在标识符识别中，当字符为字母或数字或下划线且单词长度不超过32时，就将字符加到wod后面，若最终长度大于32则报错，否则在保留字数组reserved\_words中查找word，若查到则将类别号置为相应的保留字宏，否则置为ID即标识符的类别，作为输入到out函数中进行输出。

判断无符号整数中，若字符为0-9之间的数字则一直加到word后面，直到出现非数字的字符，此时调用out函数输出。

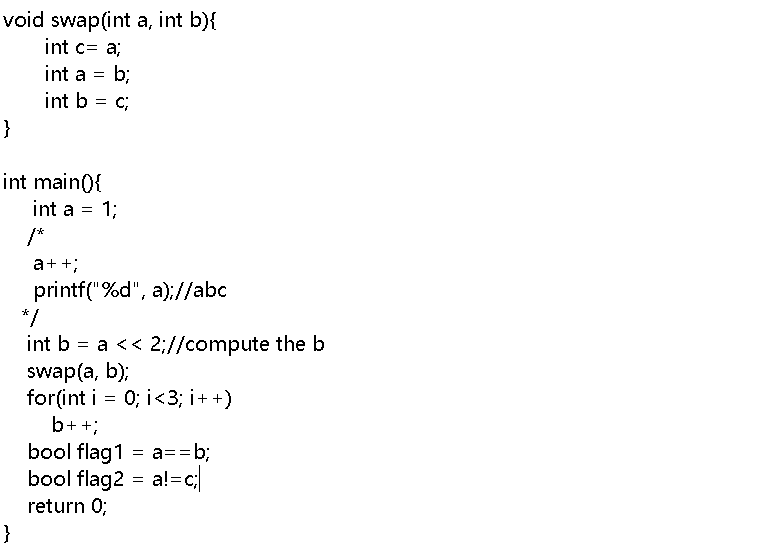
剩余的一类符号识别起来相对简单，用switch-case结构依次判断属于哪种符号即可，这里识别起来较为复杂的是//、/\*\*/这2种注释。首先说明//的识别方法，若当前扫描字符为/，则再向后面扫描一个字符，若仍为/表示出现了//注释，此时继续向后扫描直到遇到回车，调用out函数输出，在回车之前的字符属于注释内容，不用关心。/\*\*/的识别过程与//类似，仍然是先扫描到/符号，此时向后扫描字符若为\*，则进入循环扫描过程，期间扫描到\*时，再向后扫描一个字符若为/，表示注释闭合，调用out函数输出。

在识别过程中会出现多向前扫描一个字符的情况，以确定当前的单词。

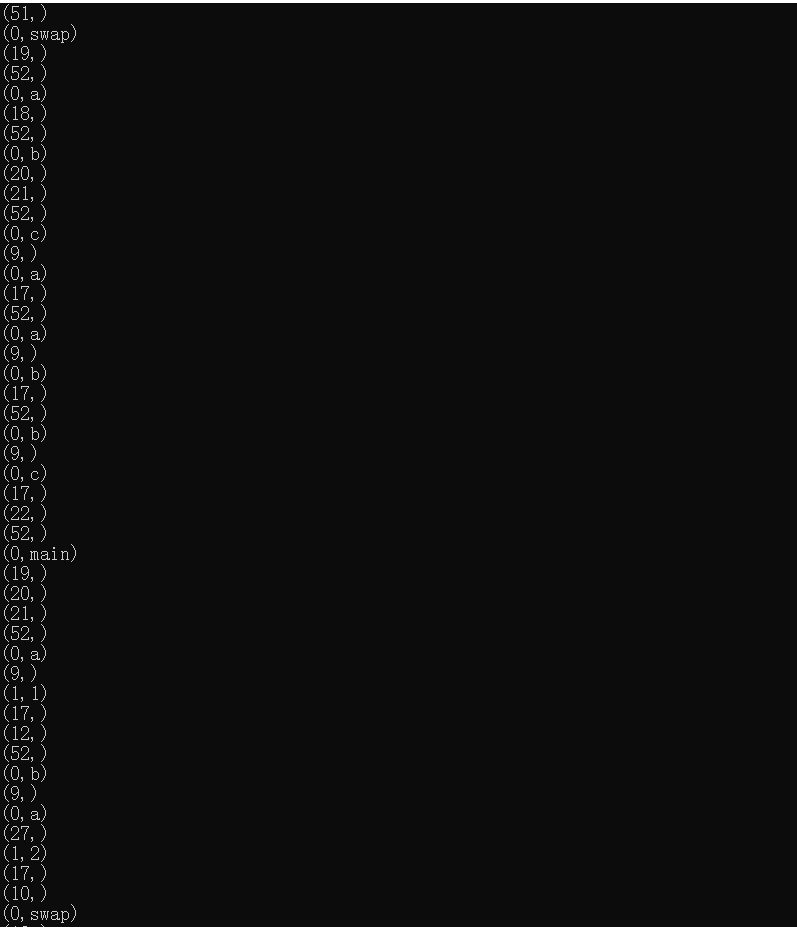
## 程序测试

设计两个例子，一个例子用于展示正确的程序输出，另一个例子展示源程序有错误时的输出。

第一个例子如下，



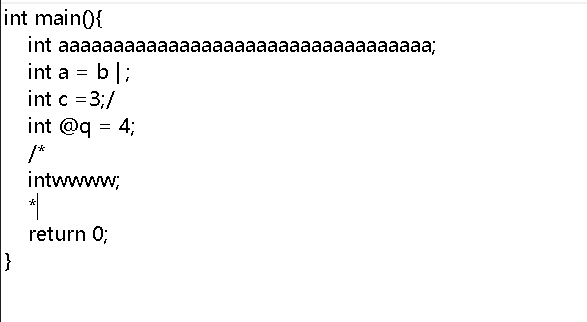
输出结果如下所示，



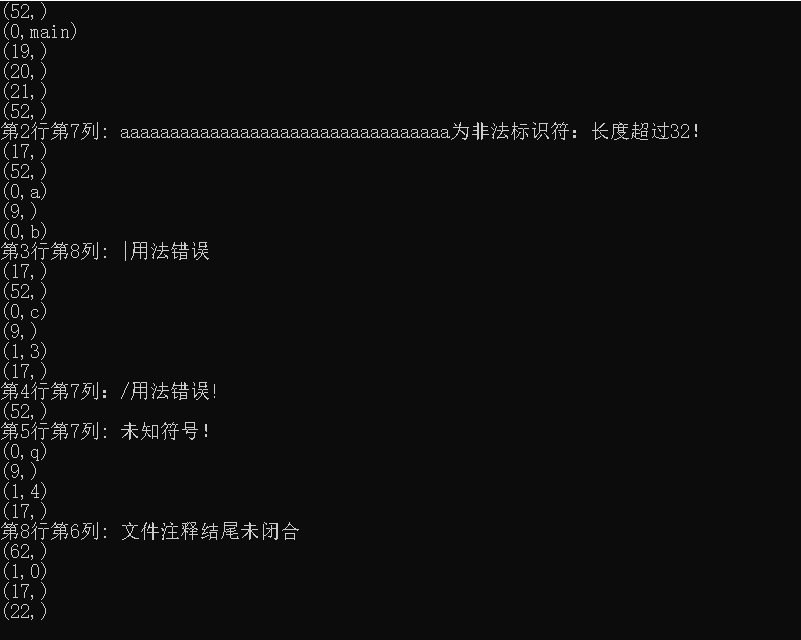


二元组的第一个数字表示单词的类别编号，也就是lex.h中的宏。比如，51代表void，0代表标识符，19代表左括号等。

第二个例子如下所示，



输出结果如下所示，



## 附录

状态转换图如下所示，

