

实验报告名称: 词法分析器

班级：15信息安全

姓名：端启航

学号：1552212

完成日期：2017.10.22

目录

[1. 实验目的及要求 2](#_Toc496547801)

[2. 实验步骤 2](#_Toc496547802)

[2.1. 符号分类 2](#_Toc496547803)

[2.2. 标号 3](#_Toc496547804)

[2.3. 读取文件 4](#_Toc496547805)

[2.4. 预处理 4](#_Toc496547806)

[2.5. 读取字符 4](#_Toc496547807)

[2.6. 特殊字符处理 5](#_Toc496547808)

[2.6.1. 注释 5](#_Toc496547809)

[2.6.2. 运算符 6](#_Toc496547810)

[2.6.3. 字符串常量 6](#_Toc496547811)

[2.6.4. 字符常量 7](#_Toc496547812)

[2.6.5. 界符 7](#_Toc496547813)

[2.7. 单词处理 7](#_Toc496547814)

[2.7.1. 保留字 7](#_Toc496547815)

[2.7.2. 整数常量 8](#_Toc496547816)

[2.7.3. 浮点数常量 8](#_Toc496547817)

[2.7.4. 标识符 8](#_Toc496547818)

[2.8. 错误提示 9](#_Toc496547819)

[2.9. 其他 9](#_Toc496547820)

[2.9.1. 保存文件 9](#_Toc496547821)

[2.9.2. 输出屏幕 10](#_Toc496547822)

[2.9.3. 构建常量表和标识符表 10](#_Toc496547823)

[2.9.4. 自定义数据结构 11](#_Toc496547824)

[2.9.5. 处理命令行选项 11](#_Toc496547825)

[2.9.6. 执行 12](#_Toc496547826)

[3. 程序编译 12](#_Toc496547827)

[4. 程序使用说明 13](#_Toc496547828)

[5. 程序使用演示 14](#_Toc496547829)

[6. 错误测试 17](#_Toc496547830)

1. **实验目的及要求**
2. 给出某高级程序设计语言的单词子集及机内表示，试编写一个词法分析器，输入为源程序字符串，输出为单词的机内表示序列。
3. 在完成以上基本要求的情况下，对程序功能扩充：

（1）增加单词（如保留字、运算符、分隔符等）的数量；

（2）将整常数扩充为实常数；

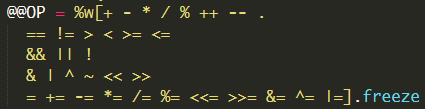
（3）增加出错处理功能；

（4）增加预处理程序，每次调用时都将下一个完整的语句读入扫描缓冲区，去掉注解行，并能够对源程序列表打印。

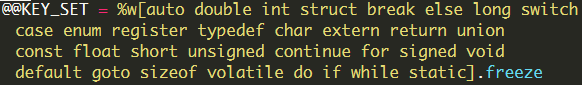
1. **实验步骤**
   1. **符号分类**

所有的符号分为关键字，运算符，界符和标识符

该词法分析器支持大部的运算符，总共34种，分别是



支持c语言所有的关键字，总共32个



以及10种界符



* 1. **标号**

**关键字标号**

|  |  |  |  |
| --- | --- | --- | --- |
| 0 | auto | 16 | const |
| 1 | double | 17 | float |
| 2 | int | 18 | short |
| 3 | struct | 19 | unsigned |
| 4 | break | 20 | continue |
| 5 | else | 21 | for |
| 6 | long | 22 | signed |
| 7 | switch | 23 | void |
| 8 | case | 24 | default |
| 9 | enum | 25 | goto |
| 10 | register | 26 | sizeof |
| 11 | typedef | 27 | volatile |
| 12 | char | 28 | do |
| 13 | extern | 29 | if |
| 14 | return | 30 | while |
| 15 | union | 31 | static |

**界符标号**

|  |  |
| --- | --- |
| 100 | , |
| 101 | ; |
| 102 | { |
| 103 | } |
| 104 | [ |
| 105 | ] |
| 106 | ( |
| 107 | ) |
| 108 | < |
| 109 | > |

**运算符标号**

|  |  |  |  |
| --- | --- | --- | --- |
| 200 | + | 217 | & |
| 201 | - | 218 | | |
| 202 | \* | 219 | ^ |
| 203 | / | 220 | ~ |
| 204 | % | 221 | << |
| 205 | ++ | 222 | >> |
| 206 | -- | 223 | = |
| 207 | . | 224 | += |
| 208 | == | 225 | -= |
| 209 | != | 226 | \*= |
| 210 | > | 227 | /= |
| 211 | < | 228 | %= |
| 212 | >= | 229 | <<= |
| 213 | <= | 230 | >>= |
| 214 | && | 231 | &= |
| 215 | || | 232 | ^= |
| 216 | ! | 233 | |= |

**其他**

#301 整数常量

#302 浮点数常量

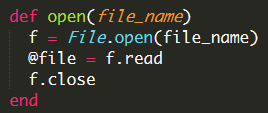
#303 字符串常量

#304 字符常量

#400 标识符

* 1. **读取文件**

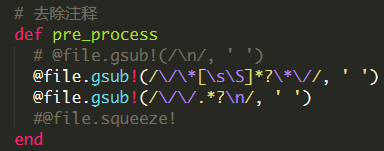
打开文件并全部读入，放入变量@file中



* 1. **预处理**

在预处理中，去除所有的注释。采用的方法是使用正则表达式/\/\\*[\s\S]\*?\\*\//，选中所有的以/\*开头和\*/结束的字符串，并且包括换行符号，然后去除就行。

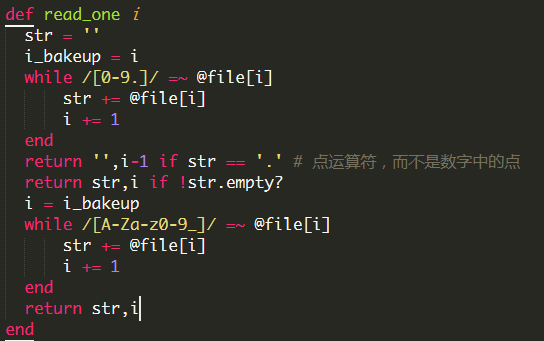
在去除注释后可以使代码更加简洁，但是，去除注释后，词法分析器报错的行号将会按照去除注释后的代码来计算。如果不去除注释的话，词法分析器也能正常工作。



* 1. **读取字符**

从字符流中一个个读取字符，将其取到字符串中组成一个单词，或者读取一个特殊字符



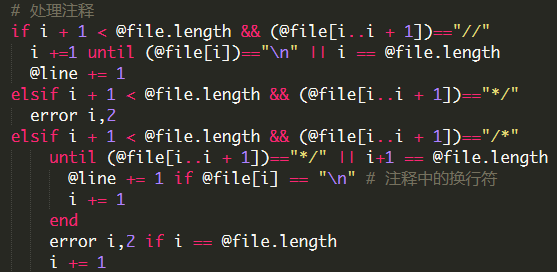


* 1. **特殊字符处理**

当读到了特殊字符时，不会存入字符串中，而是进行单独的处理

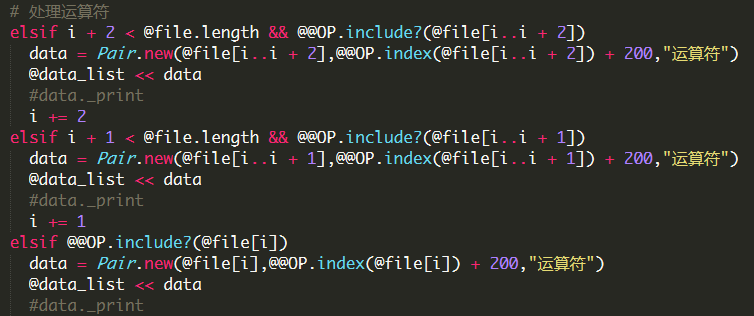
* + 1. **注释**





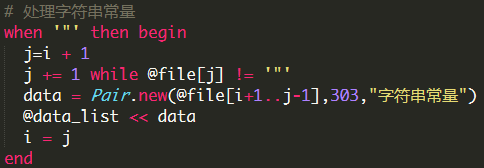
* + 1. **运算符**





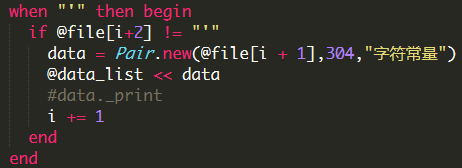
* + 1. **字符串常量**





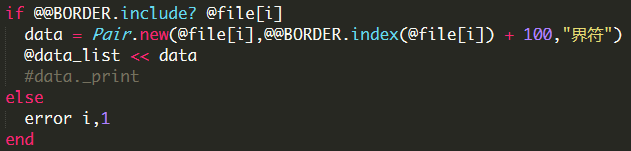
* + 1. **字符常量**





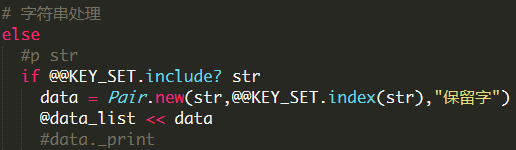
* + 1. **界符**





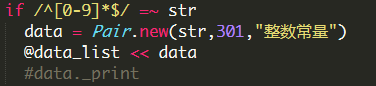
* 1. **单词处理**
     1. **保留字**

如果该单词在保留字列表，则判断为保留字



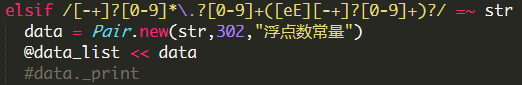
* + 1. **整数常量**





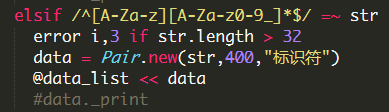
* + 1. **浮点数常量**





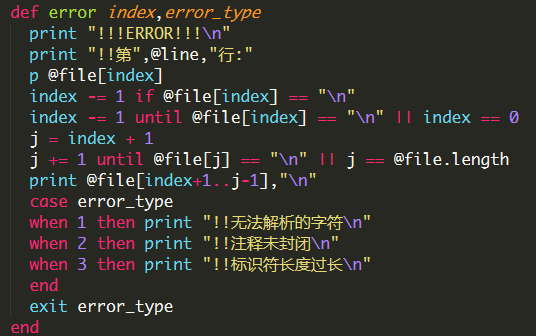
* + 1. **标识符**



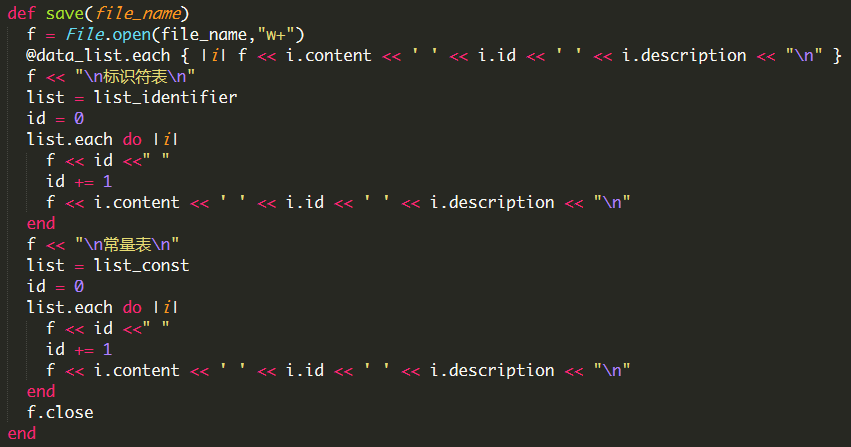


* 1. **错误提示**

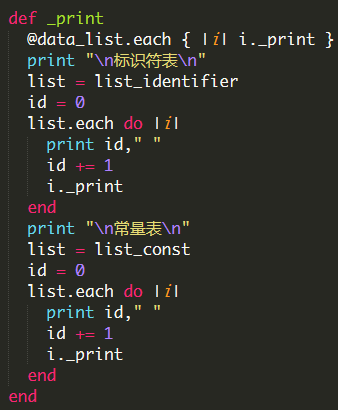
当解释出现错误时，会出现错误提示并且停止解析，错误提示将显示当前处理的字符和当前所在行号和当前所在行的代码，和错误信息



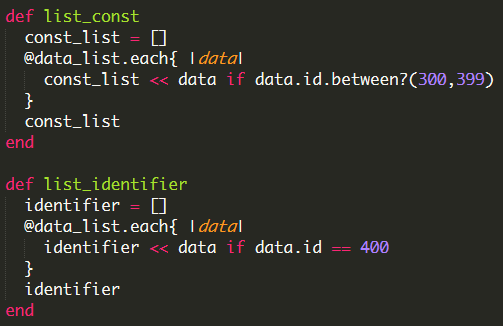
* 1. **其他**
     1. **保存文件**



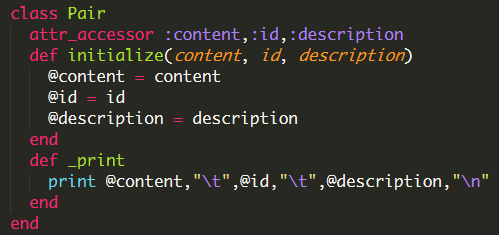
* + 1. **输出屏幕**



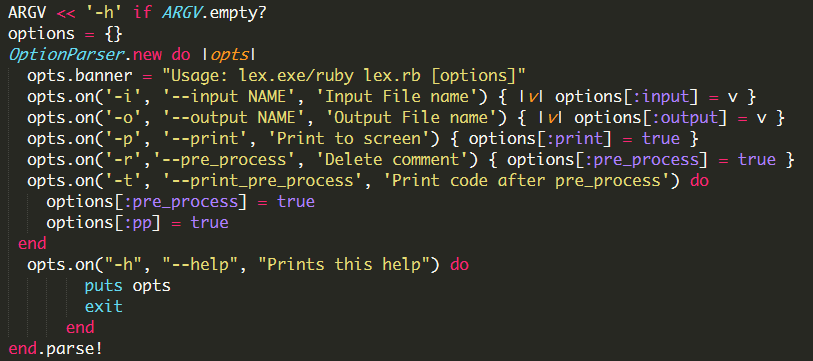
* + 1. **构建常量表和标识符表**



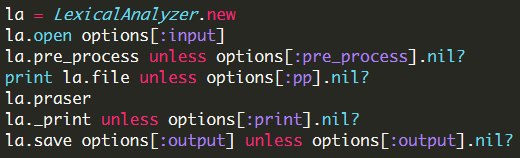
* + 1. **自定义数据结构**



* + 1. **处理命令行选项**



* + 1. **执行**



1. **程序编译**

本程序使用ruby编写，可以在装有ruby运行环境的电脑上使用ruby lex.rb运行。在没有装有ruby运行环境的电脑上可以使用编译后的lex.exe。

程序使用ocra编译，下面是程序编译过程

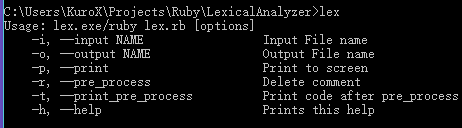


1. **程序使用说明**

本程序为命令行程序，并无图形界面。

打开命令提示符，使用cd命令，将当前路径转到程序所在目录。

使用lex或者lex.exe就可以运行程序（会显示使用帮助）



程序选项说明

-i 参数指定输入的文件

-o指定输入的文件

-p 将结果输出到屏幕上

-r 进行预处理

-t 将预处理结果输出到屏幕上

-h 将帮助输出到屏幕上

**示例**

lex.exe -i test.c -p

对文件test.c 进行词法分析并输出到屏幕上

lex.exe -i test.c -o result.txt

对文件test.c 进行词法分析并输出到result.txt

lex.exe -i test.c -o result.txt -p

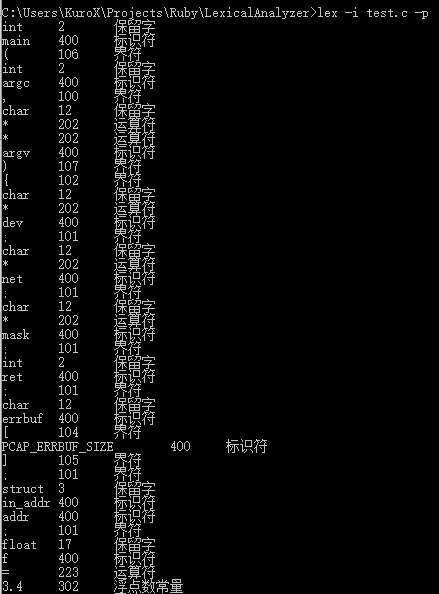
对文件test.c 进行词法分析并输出到result.txt和屏幕上

lex.exe -i test.c -t

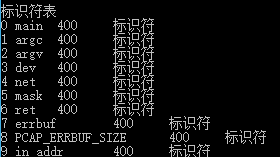
对文件test.c 进行预处理并输出到屏幕上

1. **程序使用演示**

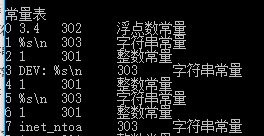
lex -i test.c -p



。。。



。。。

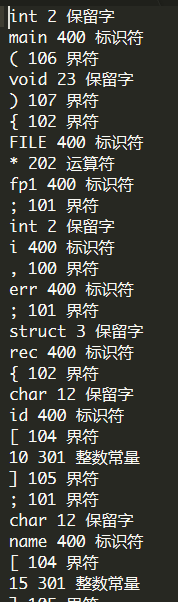
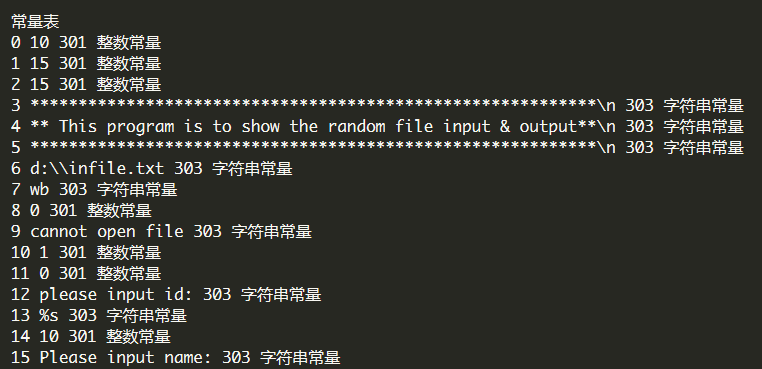


lex -i test.c -o result.txt

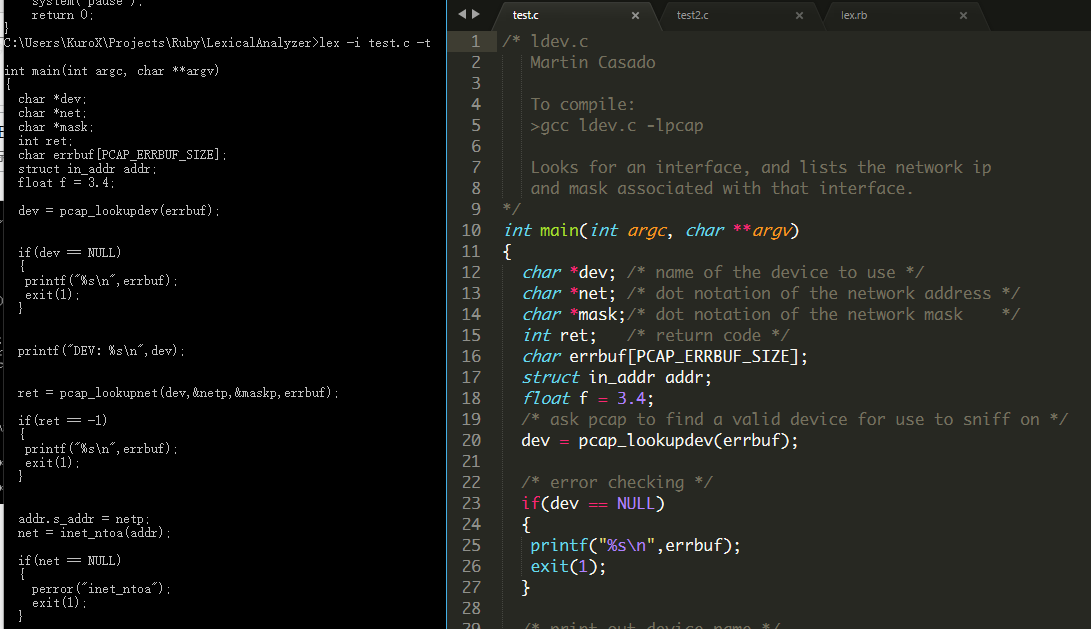


lex -i test2.c -o result2.txt



lex -i test2.c -t



1. **错误测试**

