**深 圳 大 学 实 验 报 告**

**课程名称：­ 编译原理**

**实验项目名称： 高级语言及其文法**

**学院： 计算机与软件学院**

**专业： 计算机科学与技术**

**指导教师： 张昊迪**

**报告人： 郑雨婷 学号： 2021150122 班级： 高性能**

**实验时间： 2024年 4月 12 日 -- 2024年 4 月 19 日**

**实验报告提交时间： 2024年 4月 20 日**

**教务处制**

1. 实验内容

* TINY 语言的词法由 TINY Syntax.ppt 描述；
* TINY 语言的词法分析器由 TINY Scanner.rar 的 C 语言代码实现；
* TINY+语言的词法由 TINY+ Syntax.doc 描述。

**任务：**理解 TINY 语言的词法及词法分析器的实现，并基于该词法分析器， 实现拓展语言 TINY+的词法分析器。

**要求：**

（1） TINY+词法分析器以 TINY+源代码为输入，输出为识别出的 token 序 列；

（2） 词法分析器以最长匹配为原则，例如‘:=’应识别为赋值符号而非单 独的‘：’及‘=’；

（3） Token 以（种别码，属性值）表示，包含以下类型的种别码：

a) KEY 为关键字；

b) SYM 为系统特殊字符；

c) ID 为变量；

d) NUM 为数值常量；

e) STR 为字符串常量。

（4） 识别词法错误。词法分析器可以给出词法错误的行号并打印出对应的 出错消息，主要包含以下类型的词法错误：

a) 非法字符。即不属于 TINY+字母表的字符，比如$就是一个非法字符；

b) 字符串匹配错误，比如右部引号丢失，如‘scanner

c) 注释的右部括号丢失或匹配错误，如 {this is an example

二、实验要求

* 完成 TINY+词法分析程序的编写及测试（使用提供的测试代码或自己编写测 试文件）；
* 使用实验所提供的模板撰写实验报告，要求内容详实，有具体的设计描述、 关键的代码片段、及实验结果屏幕截图；
* 在截止日期前将代码、实验报告、测试文件（如有）等所有实验相关文件压 缩到一个压缩包姓名\_学号\_实验二.zip 上传至 Blackboard。

三、实验步骤

1. 分析TINY语言的词法

TINY语言有关键字：if、then、else、end、repeat、until、read、write…..特殊字符：+ - \* 、 = < { } ； := 等，其他的可以认作是数字或标识符。TINY语言的每条语句用分号（；）分割，变量不会单独声明，在复制到时候声明，使用:=赋值，花括号{}中的认为是注释read、write是输入输出语句。以下是一个简单的示例：

想要分析出所有的token，可以一个字符一个字符地读取，然后进行状态转移。

开始还未读取任何字符的时候是START状态，若读到空格或tab等直接跳过，也就是还是START状态；读入数字认为是数字状态，一直读到一个非数字时，设置状态为DONE，表示这一个token读取结束；读取字母后进入字母状态，一直读到一个非字母才结束；读取:后紧接着读取=，结束；读取{之后，进入注释状态，之后读取的任何字符都是注释中的成员，直到读取到}；

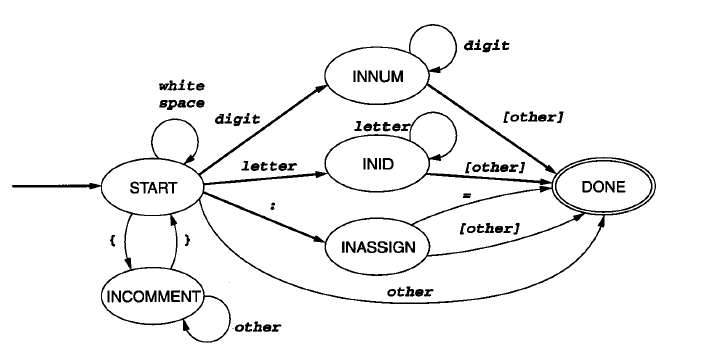


图1 TINY语法Token的状态转移图

2．分析TINY词法分析器程序

MAIN.C是程序的主程序，在 main 函数中，我们可以看到如下图2所示的代码。main函数中有 argc 和 argv 这两个参数，argc 是参数的个数，argv 是参数的内容。首先检验参数个数是否合法。之后根据argv[1]中的参数打开测试文件，所以就需要我们在运行这段程序的时候把文件名作为参数传入。之后设置了fprintf的输出为标准输出，打印到屏幕上。之后就在 while 循环中调用 getToken 函数了，直到返回的结果为ENDFILE。

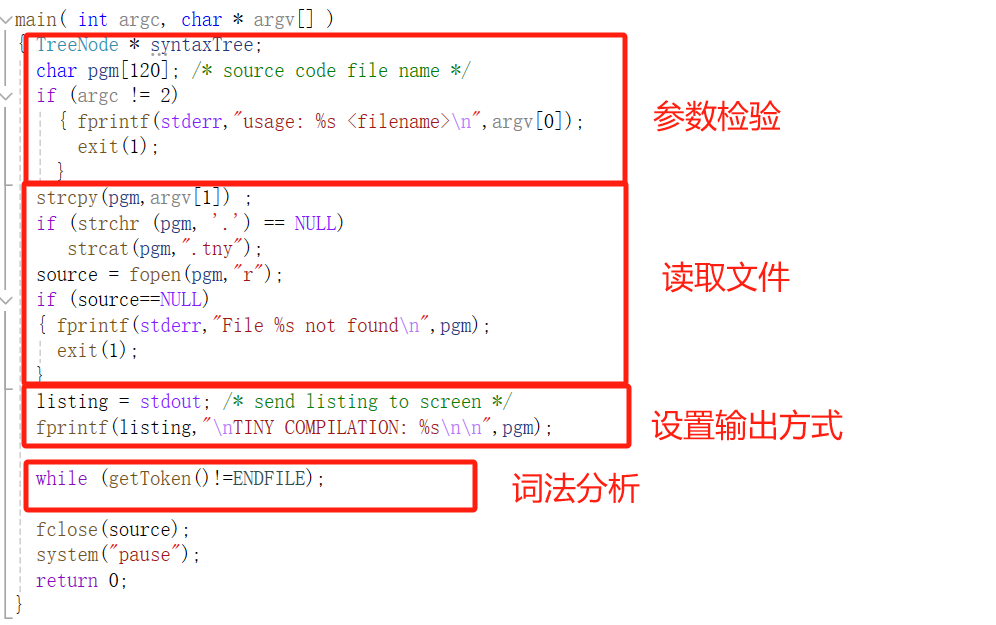


图2 main函数代码

接下来分析getToken函数。函数的返回值是一个枚举类型，代表识别到的这个词法单元的类型。其中有关键字、数字、标识符、特殊符号等。

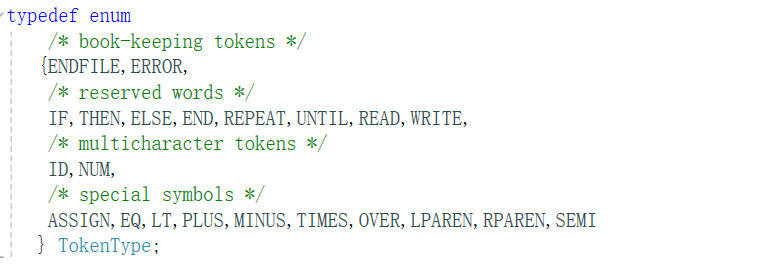


图3 TokenType定义

在getToken 函数中，使用枚举类型 state 记录每一个状态，这些状态是之前分析的DFA中的每一个状态。包括START、INASSAGIN、INCOMMENT、INNUM、INID、DONE.

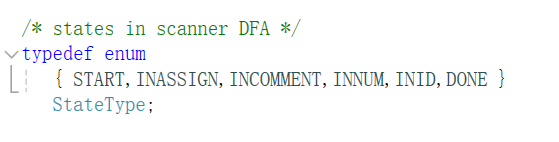


图4 StateType定义

在getToken 函数中,调用设置初始状态为START，然后读取文件中的字符，进行状态转移。例如读入一个数字会进入 INNUM 状态，读入一个英文字母会进入INID状态，读入冒号：会进入INASSIGN状态，读入空格或回车继续保持START状态，读入左大括号{会进入 INCOMMENT 状态，其他情况看可能是读入特殊字符，也可能是读入无效的错误字符，也可能是结束标志，总之下一个状态都将设置为DONE。

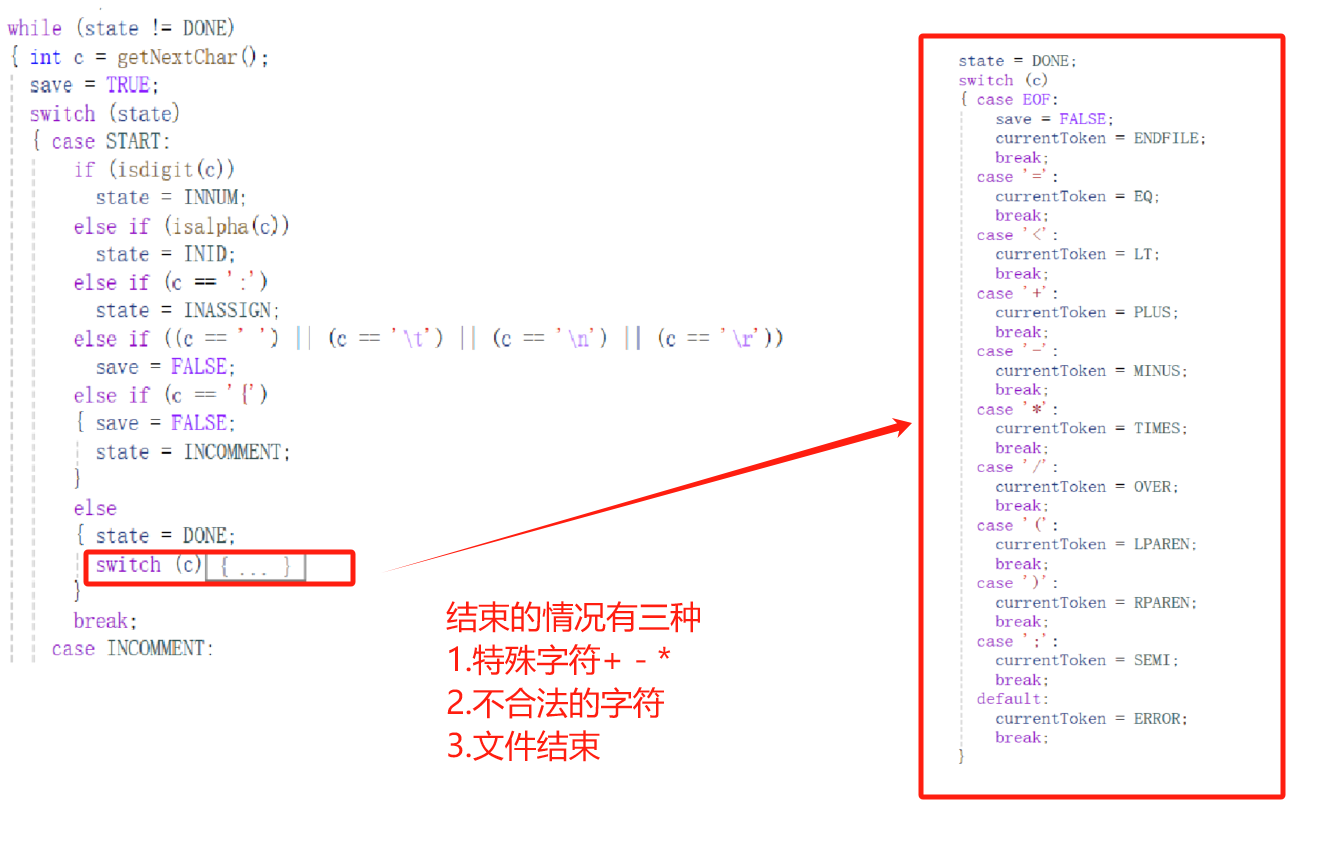


图5 START状态转移

其他状态INASSAGIN、INCOMMENT、INNUM、INID与START状态x相似，根据状态转移图进行状态转移，其中需要注意的是，当把状态设置为DONE后，也需要根据情况设置currentToken的值，这样在一次getToken函数执行完毕之后，可以返回给main函数以判断是否可以停止循环。

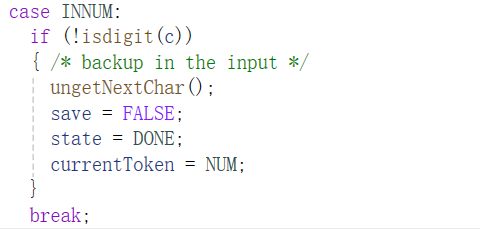
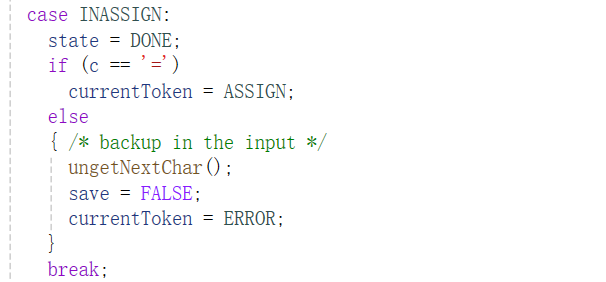
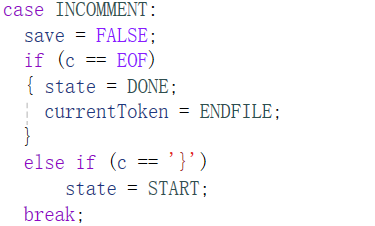
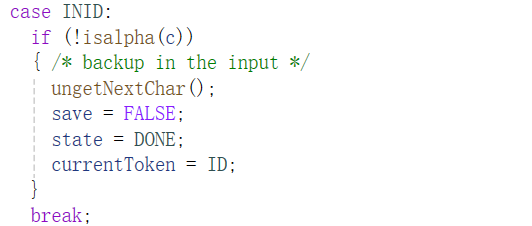
 

图6 INASSAGIN、INCOMMENT、INNUM、INID状态转移

此外，在INNUM、INID、INASSIGN中，当读取到非数字、非标识符、非等号时，说明遇到了下一个语法单元的字符，因此需要调用ungetNextChar()函数，用来撤销本次读取的字符。还注意到有一个变量save，有时会被定义为FALSE,但大多数时候定义为TRUE。这是用来标记当前字符是否需要保存到tokenString当中，tokenStringIndex 来记录需要存入的位置。

在每读取一个字符，进行状态转移之后，都判断是否保存该字符。并且在状态为DONE时，如果认为当前的Token是标识符，是我们初步判断都是字母，但是并非所有字母组成的字符串都是标识符外，还可能是保留的关键字。

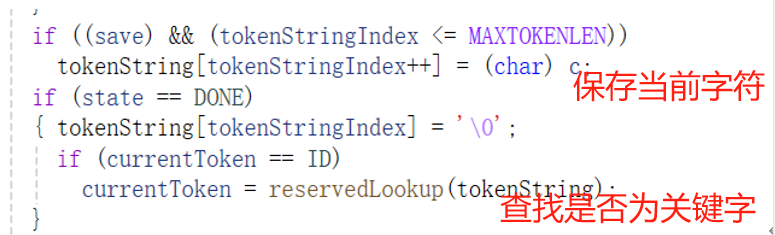


图7 状态转移后保存字符

在reservedLookup 函数中循环遍历查看是否为保留字，之后把对应保留字的枚举返回。若没有找到匹配的保留字，那就确定是一个标识符。

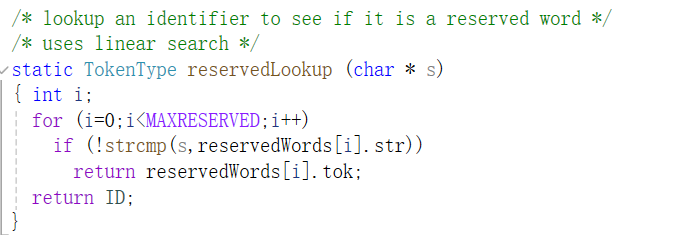
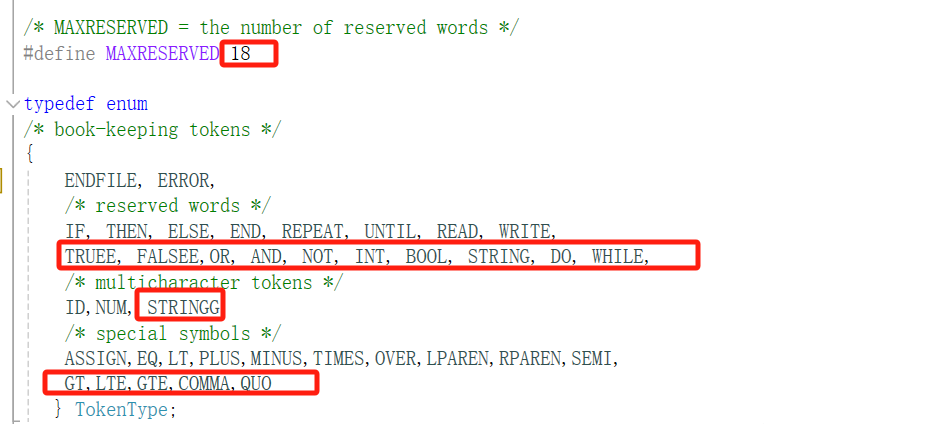


图8 遍历查找判断是否为关键字

最后，循环逐个读取字符直到状态为DONE结束后，用fprintf输出行号，用printToken输出Token的类型及值。其中printToken函数就是根据传入的参数currentToken和tokenString去输出信息到屏幕。至此，分析TINY词法分析器程序完毕。

3. 实现拓展语言 TINY+的词法分析器。

　　①TINY+相较于TINY增加了一些关键字和特殊字符，还多了一个STRING的类型，所以在TokenType中增加TRUEE, FALSEE,OR, AND, NOT, INT, BOOL, STRING, DO, WHILE,GT,LTE,GTE,COMMA,QUO等表示新增的类型。同时，寻找保留字的reservedWords数组也需要修改，MAXRESERVED也修改为１８，不然会数组越界。



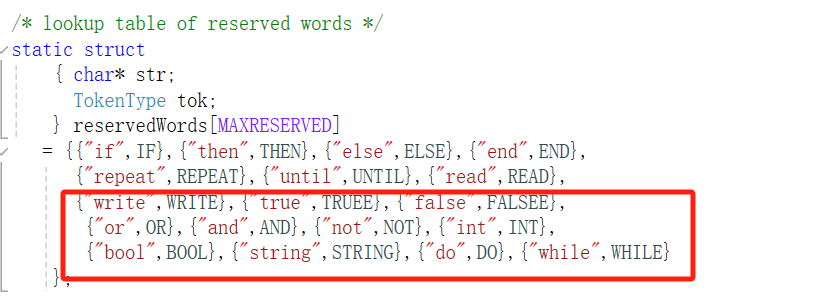


图9 增加关键字、特殊字符

②对于新增的字符，＝与，的读取和TINY语法相同，最开始只要读到就进入DONE状态，设置currentToken为相应的类型。但是>和>=需要额外考虑。读取>后先暂且认定它是大于，但是需要判断下一个是否为=，若是，改变curentToken为大于等于；若不是撤回刚才读取的字符。小于等于的处理方式相同。

若从START状态开始读取的第一个字符既不是数字又不是字母也不是特殊字符空格等，那就是非法字符。即不属于 TINY+字母表的字符，比如$就是一个非法字符；将currentToken设置为NOCHARERROR，代表当前字符非法。

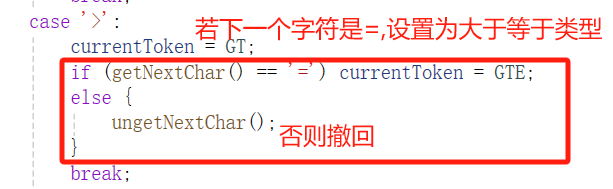




图10 处理新增字符

③对于标识符的定义有所改变，TINY中全部由字母组成，在TINY＋中是首字符为字母，剩下可以是字母也可以是数字的字符串。所以在进入INID状态后，若当前字符既不是数字又不是字母时才结束本次Token的读取。

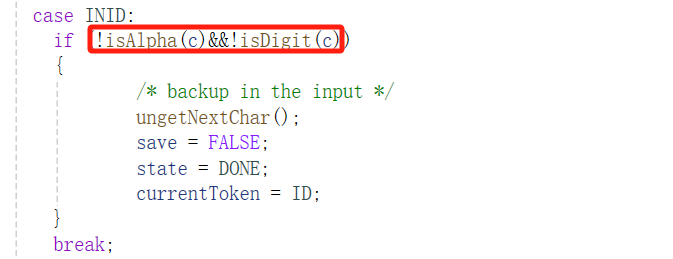


图11 修改标识符的判断

对于字符串类型的判断，需要先有一个单引号，结束时再有一个单引号。所以从START状态读取一个单引号的下一个可能是字符串，也可能是下一个Token的开始。所以增加两个标记变量QUO\_flag和first，first==1代表是第一个单引号，那么下次在初始化为START之后，又再改变为INSTRING状态。

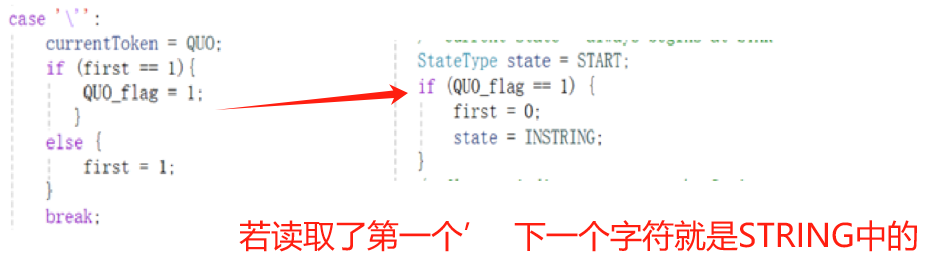


图12 单引号的处理

TINY+中还规定，STRING必须在一行当中，不能跨越多行，因此，在INSTRING状态下若读到了换行符/n、文件结束符EOF，都代表还未读到第二个单引号该行就结束，给currentType设置一个错误的类型QUOERROR ,代表字符串单引号缺失错误。

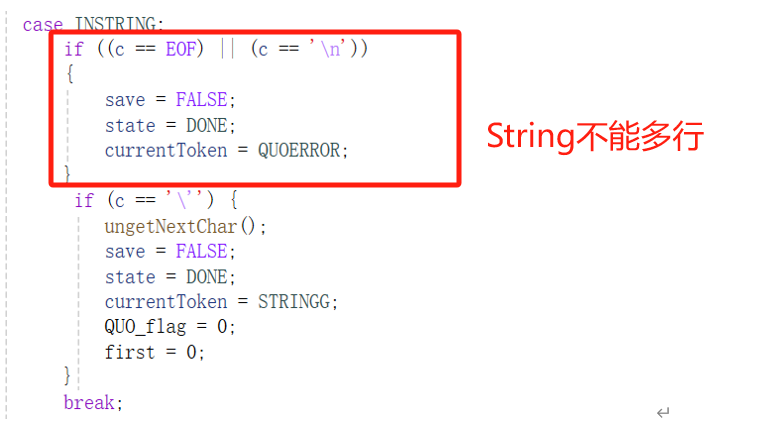


图13 STRING不能跨越多行

④TINY+的注释与TINY的注释大体相同，只多了一点不能嵌套。在原本程序的基础上，只需判断是否嵌套即可。如果已经在INCOMMENT状态中了，在遇到一个左大括号就代表注释中有嵌套，将currentToken设置为COMMENT\_NESTED\_ERROR，代表注释嵌套错误。和字符串的单引号缺失相同，注释也可能会确实右大括号}。在读取到文件结束符EOF都没有读取到}时，就说明缺失}。



图14 处理注释

⑤在以上的完善修改程序中，我们新增了COMMENTERROR, QUOERROR, NOCHARERROR, COMMENT\_NESTED\_ERROR等错误的Token类型，所以在TokenType中也要记得新增。并且要修改printToken函数，新增TokenType中所有新增的类型对应的输出情况，同时也要将输出修改为（种别码，属性值）的形式。

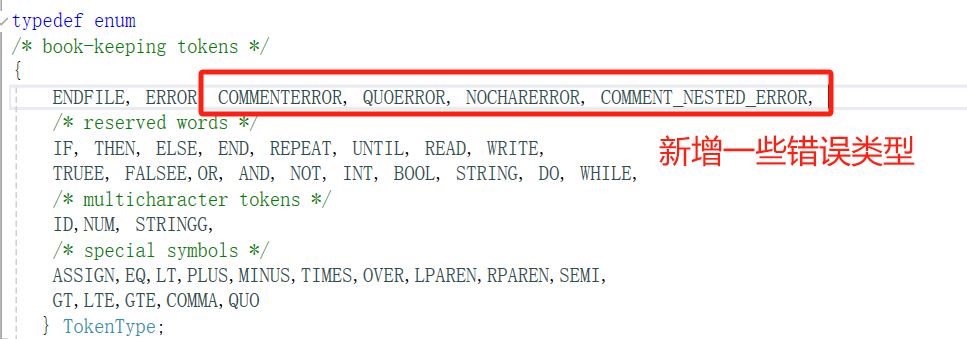


图15 新增错误的类型

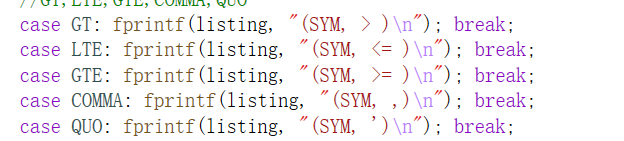
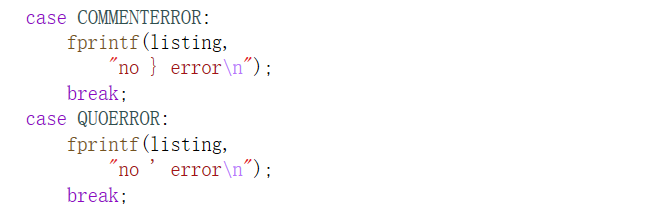
 

图16 修改printToken函数

⑥处理中文字符，有时读取中文字符程序会中断，经过调试定位到是isalpha和isdigit函数不允许传入参数的范围在0——255之外，因此自己重新编写两个函数isAlpha和isDigit替换，这样程序就可以正常运行下去。

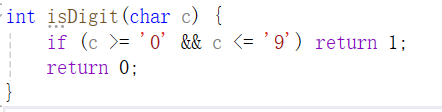
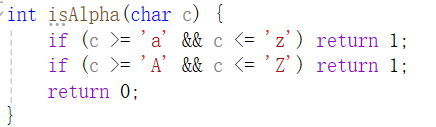
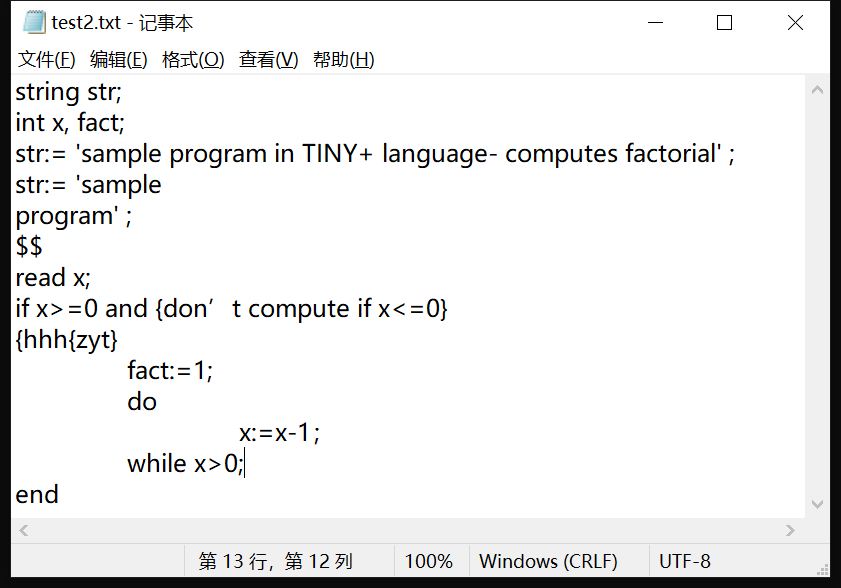


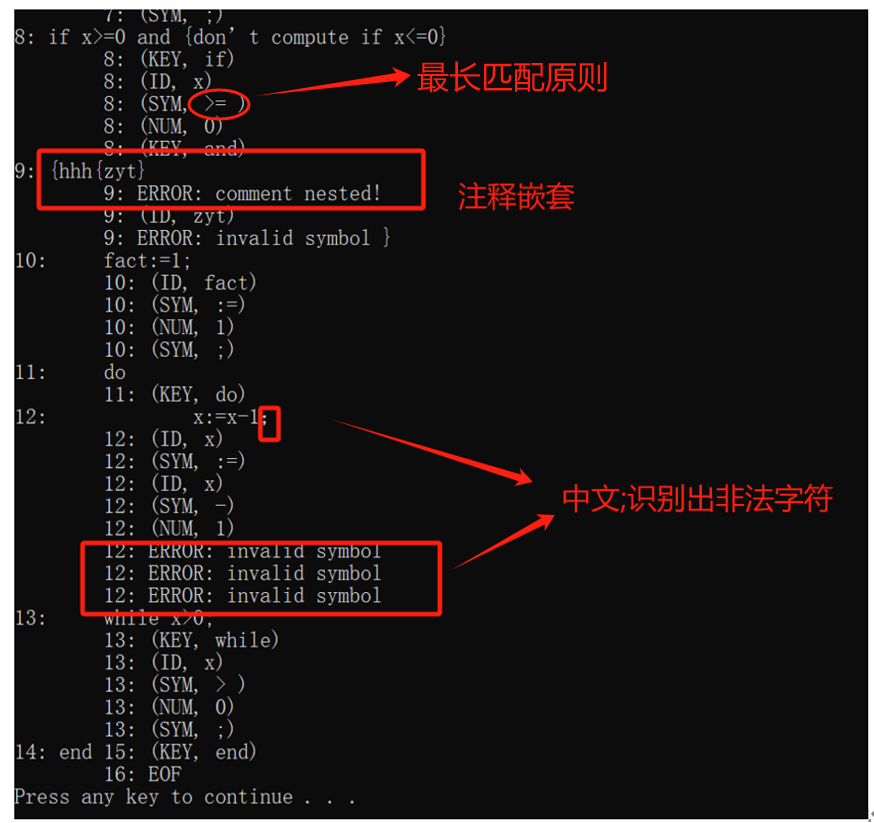
图17 编写函数isAlpha和isDigit

四、实验结果



编译之后，将test2.txt作为参数传入 main.exe 读取，之后效果如下：





自己编写测试样例test2.txt，TINY+词法分析器可以识别出的 token 序列； 词法分析器以最长匹配为原则，识别出>=而不是>；Token 以（种别码，属性值）表示。并且可以识别出非法字符、字符串匹配错误、注释嵌套等词法错误。

五、实验总结

本次实验目的主要是深入理解TINY语言的词法及其词法分析器的实现原理，并在此基础上，基于已有的TINY词法分析器，开发语言TINY+的词法分析器。最终实现了一个可以识别出token序列、识别出基本此法错误的词法分析器，成就感十足。通过本次实验，我掌握词法分析的一些基本方法，对DFA的相关概念如状态转移有了更深的理解，也彻底明白了自动机识别语言的运作过程。

|  |
| --- |
| **指导教师批阅意见：**  **成绩评定：**  **指导教师签字：**  年 月 日 |
| 备注： |