**编译原理**

班级：21计算机科学与计算机

姓名： LOW REN HONG 学号：2021529620004

实验三 基于YACC的TINY语法分析器的构建

1. 实验要求

运用YACC，针对TINY语言，构造一个语法分析器。给出实验方案，实施并描述结果。

1. 实验方案

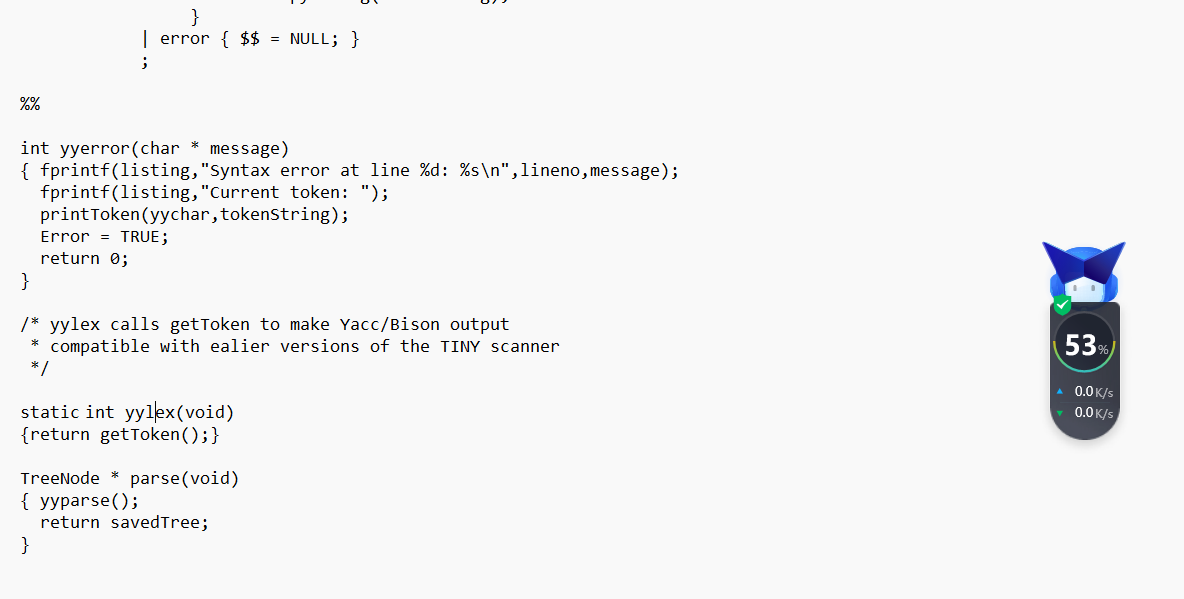
利用教材《编译原理与实践》课后的附录 B 中的例子进行设计

**查找资料**

根据资料，使用Yacc生成的子程序在请求读取下一个单词时会调用yylex()函数。yylex()函数返回一个单词符号，并将相关属性值存储在全局变量yylval中。为了在Lex和Yacc之间进行联合使用，需要在运行Yacc程序时添加-d选项，以生成名为y.tab.h的文件。该文件将包含在Yacc描述文件中通过%tokens定义的所有单词类型。最后，将y.tab.h文件包含在Lex描述文件中。

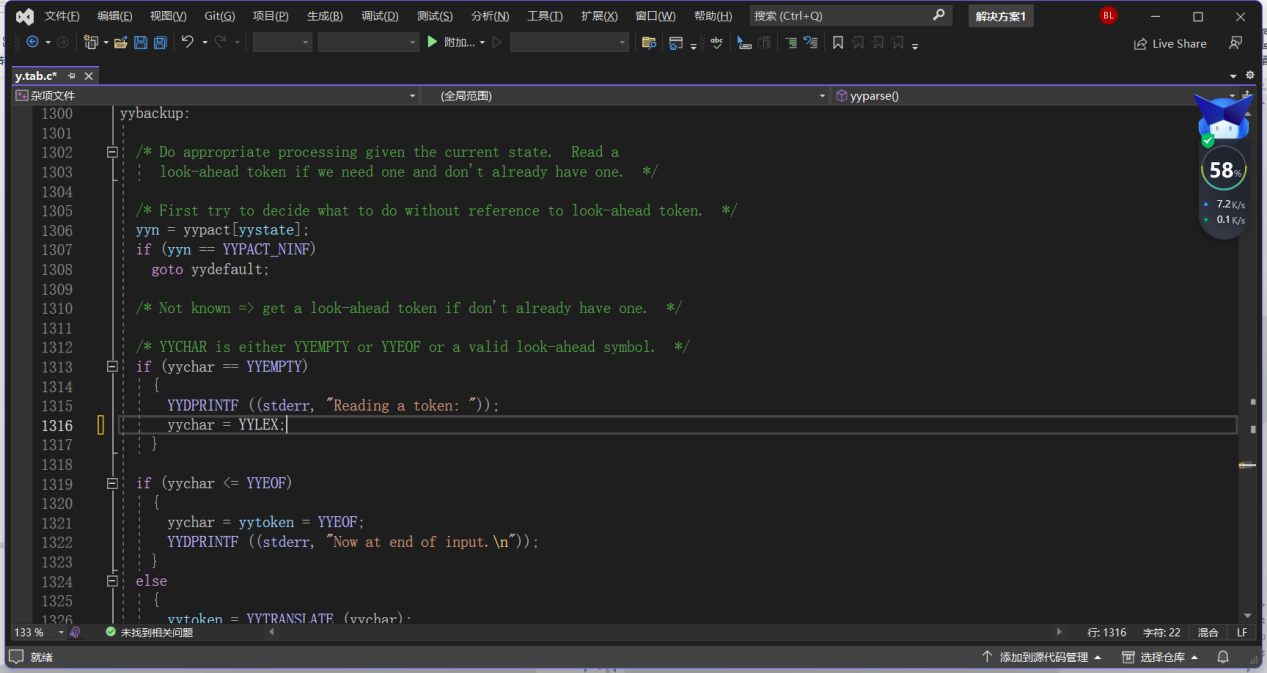
**修改tiny.y**

在附录里的 tiny.y 文件进行修改，将其中的 static int yylex 方法去掉



**修改y.tab.c**

对于其中的 yychar=yylex;修改成 yychar=getToken()



将使用lex方法读到的token送入yacc产生的分析子程序中，实现了两者的使用。

1. 分析表parsing table问题

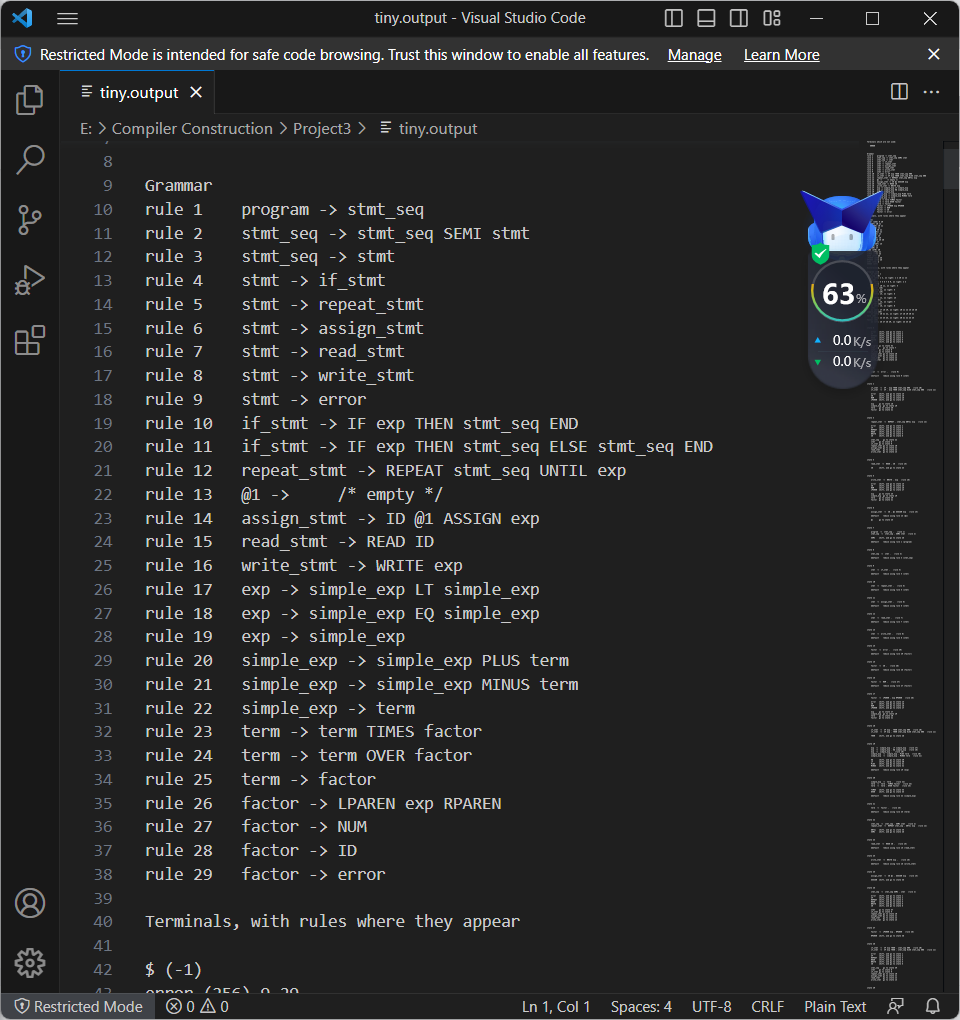
理论和设计（描述parsing table在实验方案中的作用，观察并输出parsing table）

**Parsing Table在实验方案中的作用**

YACC 输出的语法分析函数 int yyparse()在对一个输入文件进行分析时，如果面对某一状

态或输入单词在分析表中找不到相应的动作，则调用yyerror()报错

**利用bison -v来获取parsing table**



1. 内容和步骤

1、针对TINY语言给出 yacc的y文件的代码

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* File: tiny.y \*/

/\* The TINY Yacc/Bison specification file \*/

/\* Compiler Construction: Principles and Practice \*/

/\* Kenneth C. Louden \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

%{

#define YYPARSER /\* distinguishes Yacc output from other code files \*/

#include "globals.h"

#include "util.h"

#include "scan.h"

#include "parse.h"

#define YYSTYPE TreeNode \*

static char \* savedName; /\* for use in assignments \*/

static int savedLineNo; /\* ditto \*/

static TreeNode \* savedTree; /\* stores syntax tree for later return \*/

%}

%token IF THEN ELSE END REPEAT UNTIL READ WRITE

%token ID NUM

%token ASSIGN EQ LT PLUS MINUS TIMES OVER LPAREN RPAREN SEMI

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* File: tiny.y \*/

/\* The TINY Yacc/Bison specification file \*/

/\* Compiler Construction: Principles and Practice \*/

/\* Kenneth C. Louden \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

%{

#define YYPARSER /\* distinguishes Yacc output from other code files \*/

#include "globals.h"

#include "util.h"

#include "scan.h"

#include "parse.h"

#define YYSTYPE TreeNode \*

static char \* savedName; /\* for use in assignments \*/

static int savedLineNo; /\* ditto \*/

static TreeNode \* savedTree; /\* stores syntax tree for later return \*/

%}

%token IF THEN ELSE END REPEAT UNTIL READ WRITE

%token ID NUM

%token ASSIGN EQ LT PLUS MINUS TIMES OVER LPAREN RPAREN SEMI

%token ERROR

%% /\* Grammar for TINY \*/

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* File: tiny.y \*/

/\* The TINY Yacc/Bison specification file \*/

/\* Compiler Construction: Principles and Practice \*/

/\* Kenneth C. Louden \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

%{

#define YYPARSER /\* distinguishes Yacc output from other code files \*/

#include "globals.h"

#include "util.h"

#include "scan.h"

#include "parse.h"

#define YYSTYPE TreeNode \*

static char \* savedName; /\* for use in assignments \*/

static int savedLineNo; /\* ditto \*/

static TreeNode \* savedTree; /\* stores syntax tree for later return \*/

%}

%token IF THEN ELSE END REPEAT UNTIL READ WRITE

%token ID NUM

%token ASSIGN EQ LT PLUS MINUS TIMES OVER LPAREN RPAREN SEMI

%token ERROR

%% /\* Grammar for TINY \*/

program : stmt\_seq

2、给出.l文件的代码

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* File: tiny.l \*/

/\* Lex specification for TINY \*/

/\* Compiler Construction: Principles and Practice \*/

/\* Kenneth C. Louden \*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

%{

#include "globals.h"

#include "util.h"

#include "scan.h"

/\* lexeme of identifier or reserved word \*/

char tokenString[MAXTOKENLEN+1];

%}

%option noyywrap

digit [0-9]

number {digit}+

letter [a-zA-Z]

identifier {letter}+

newline \n

whitespace [ \t]+

%%

"if" {return IF;}

"then" {return THEN;}

"else" {return ELSE;}

"end" {return END;}

"repeat" {return REPEAT;}

"until" {return UNTIL;}

"read" {return READ;}

"write" {return WRITE;}

":=" {return ASSIGN;}

"=" {return EQ;}

"<" {return LT;}

"+" {return PLUS;}

"-" {return MINUS;}

"\*" {return TIMES;}

"/" {return OVER;}

"(" {return LPAREN;}

")" {return RPAREN;}

";" {return SEMI;}

{number} {return NUM;}

{identifier} {return ID;}

{newline} {lineno++;}

{whitespace} {/\* skip whitespace \*/}

"{" { char c;

do

{ c = input();

if (c == EOF) break;

if (c == '\n') lineno++;

} while (c != '}');

}

. {return ERROR;}

%%

TokenType getToken(void)

{ static int firstTime = TRUE;

TokenType currentToken;

if (firstTime)

{ firstTime = FALSE;

lineno++;

yyin = source;

yyout = listing;

}

currentToken = yylex();

strncpy(tokenString,yytext,MAXTOKENLEN);

if (TraceScan) {

fprintf(listing,"\t%d: ",lineno);

printToken(currentToken,tokenString);

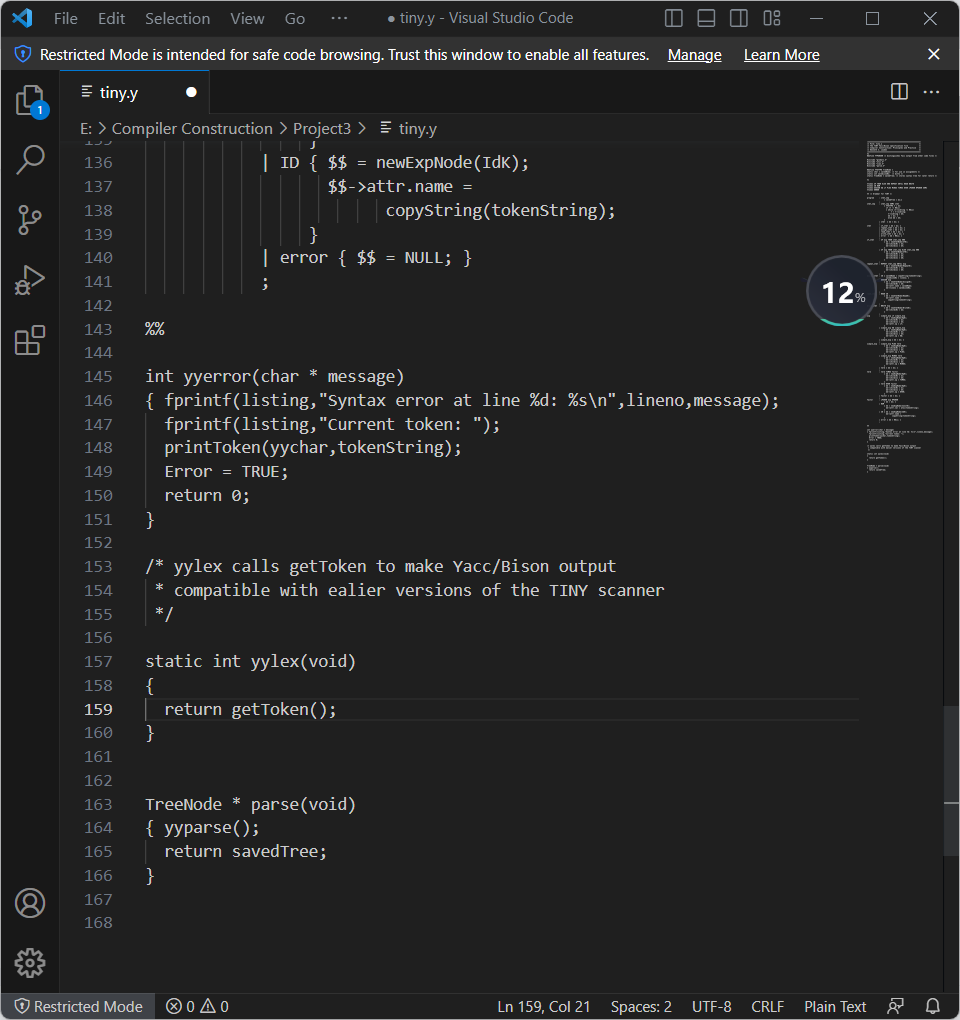
}

return currentToken;

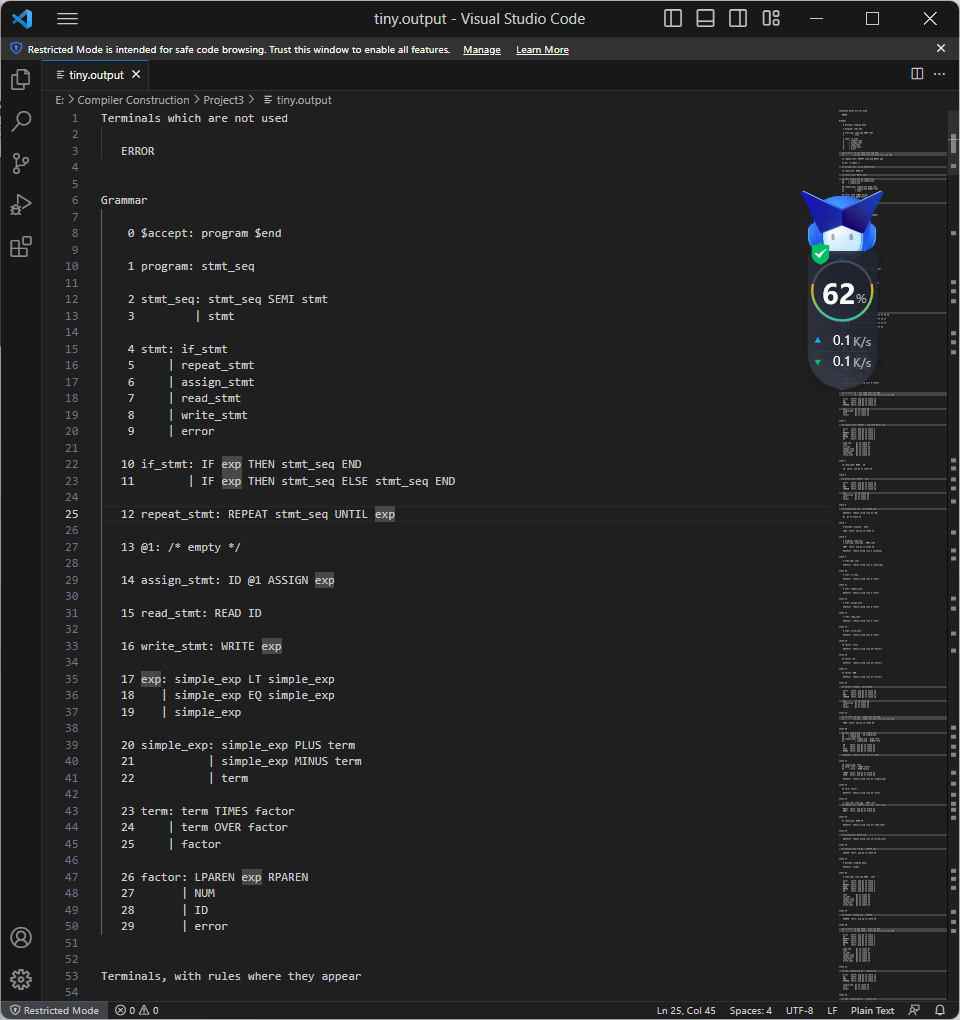
}

3、实验具体步骤

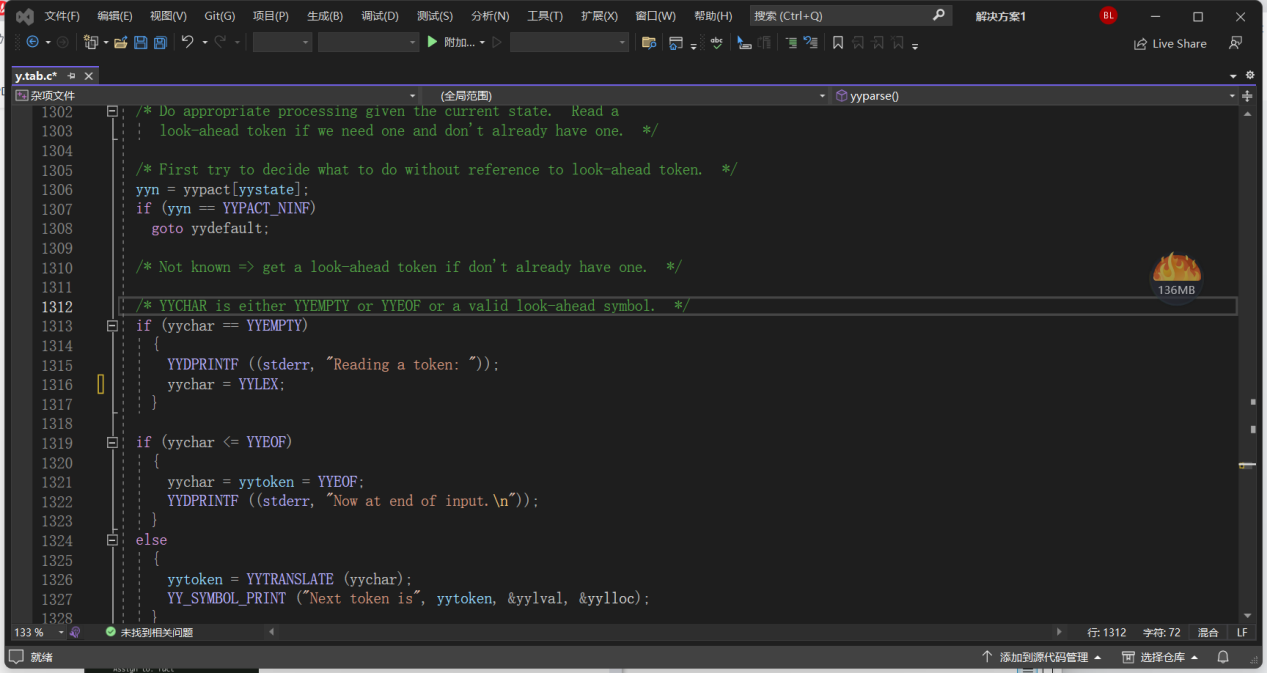
**1）将tiny.y中的yylex方法去掉**

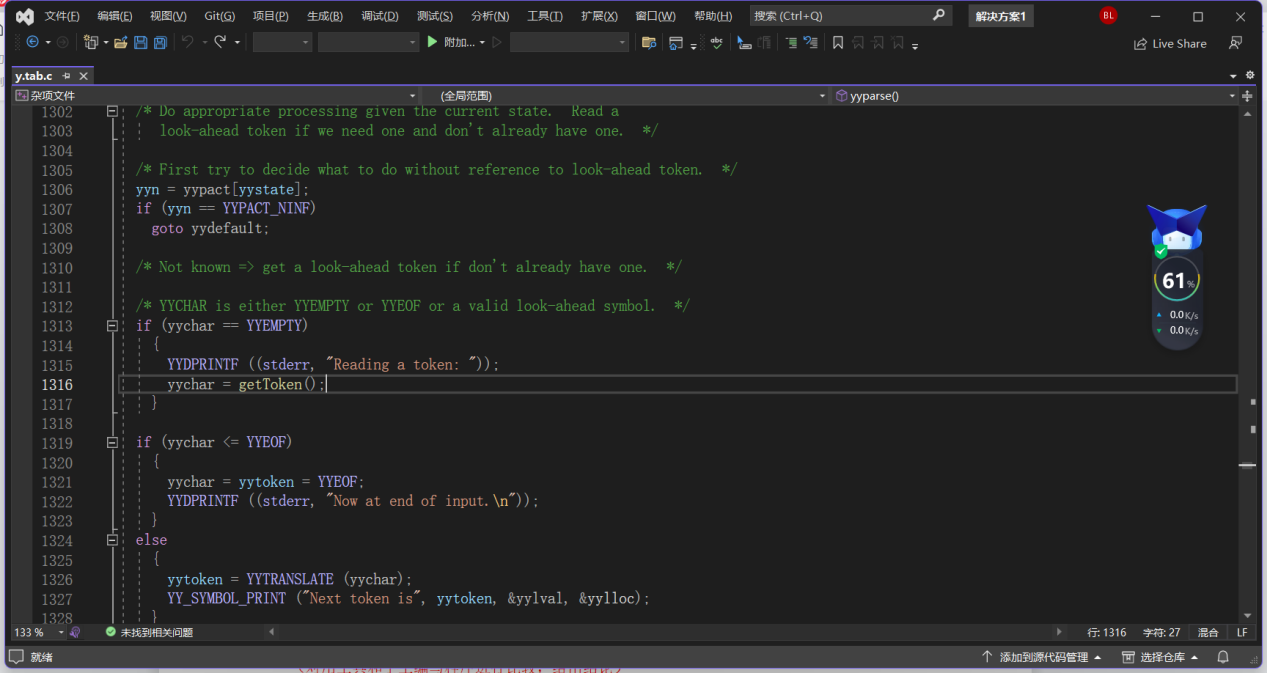


1. **输入命令 bison -v tiny.y 生成tiny.output**

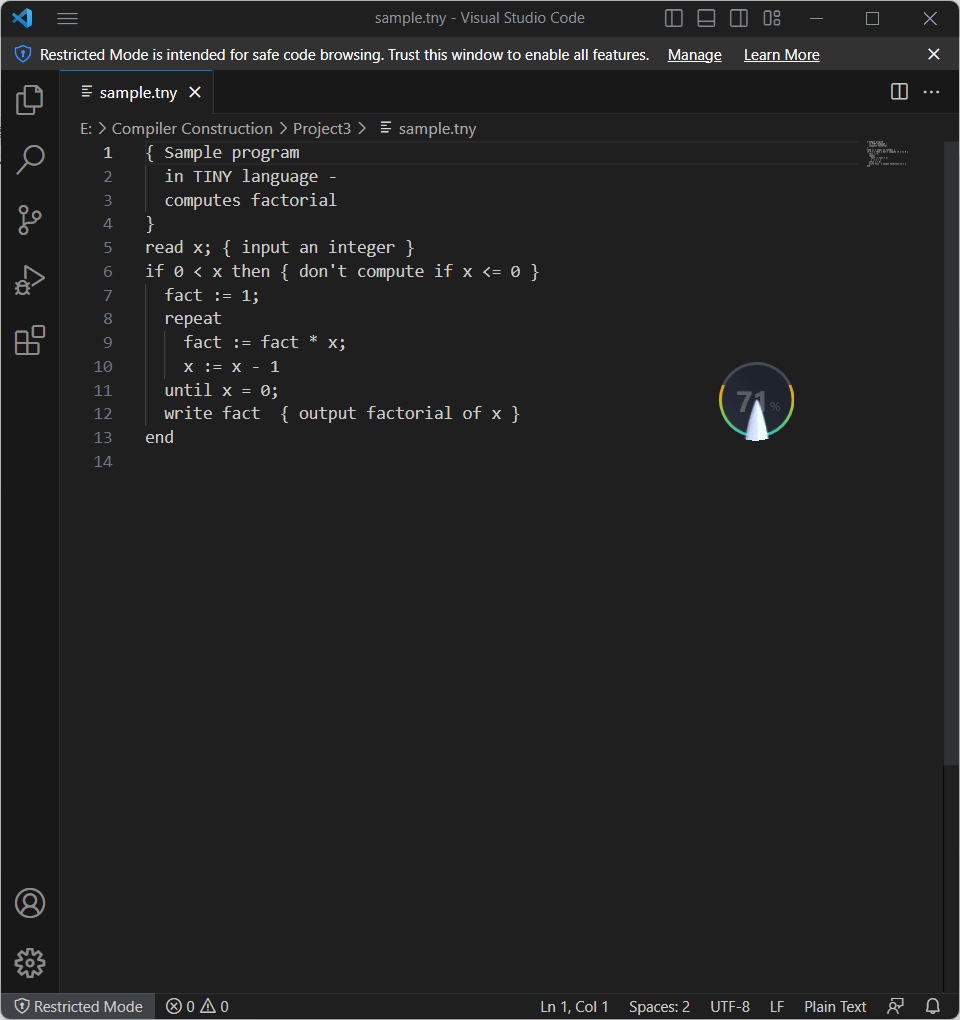


1. **输入命令lex tiny.l和yacc -d tiny.y生成y.tab.c,y.tab.h,lex.yy.c文件**
2. **修改 y.tab.c 将 yychar=yylex()修改成 yychar=getToken()**



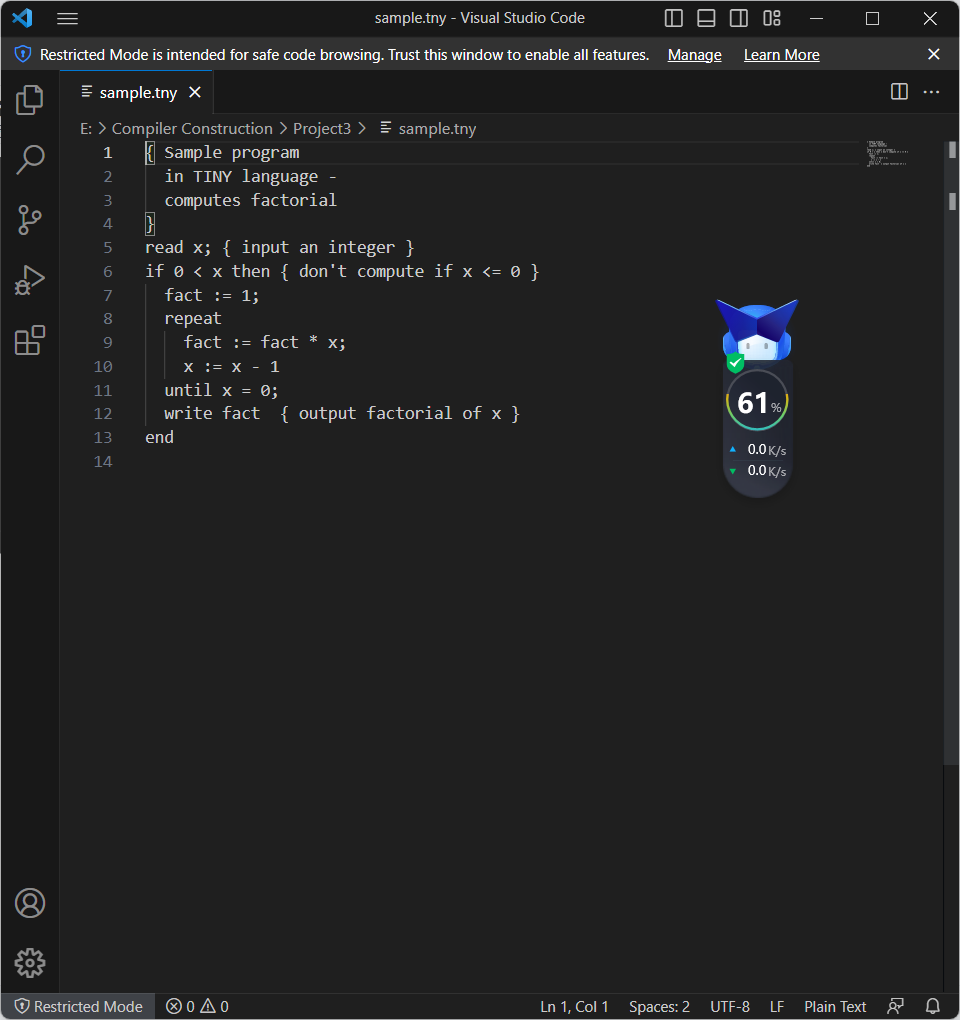


1. **输入 make 命令进行构建生成 tiny.out 可执行文件**
2. **以 sample.tny 作为输入执行 tiny.out 文件**

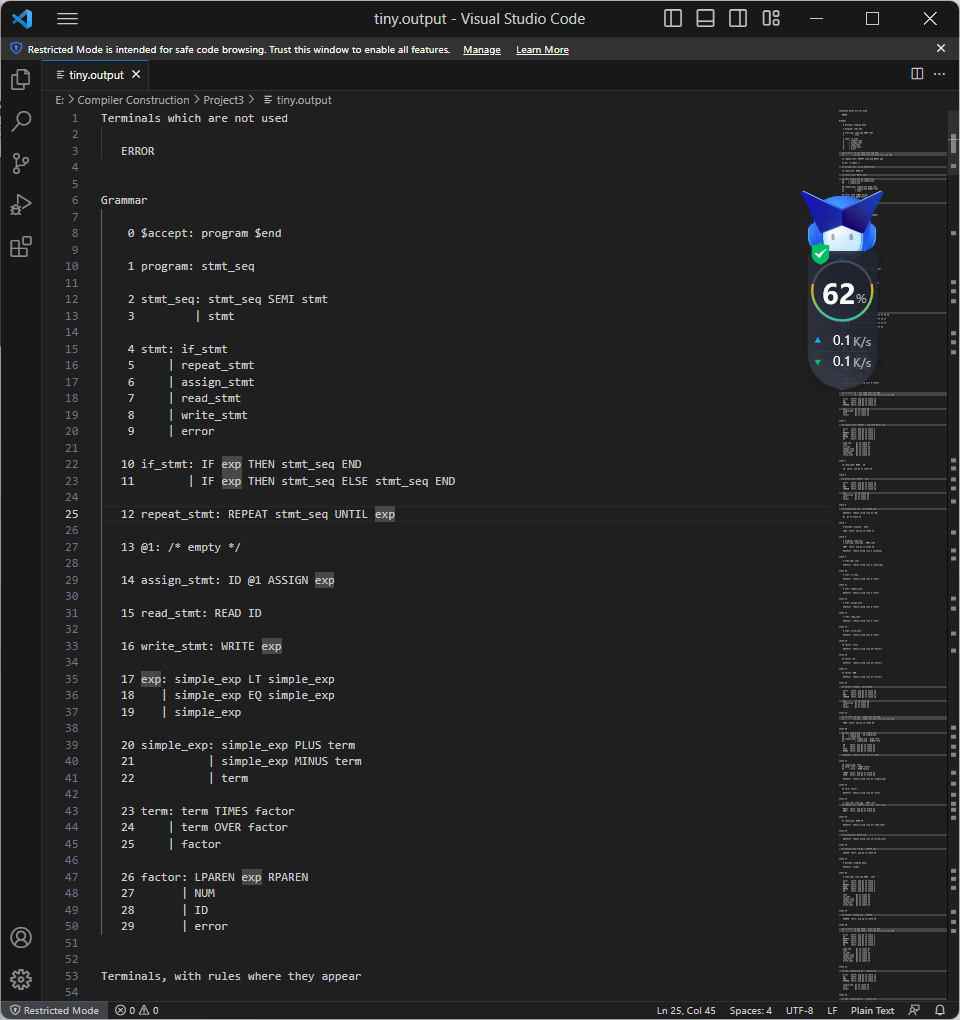


1. 实验结果:

截图

**输出**  


**分析表**



1. 实验结论:

1 、实验结论

（是否能够准确描述实验的结论）

通过这次的实验我们使用lex和yacc工具，对于输入的内容lex会进行词法分析然后yacc对于lex词法分析后再进行语法分析。通过调用tiny.out文件进行语法分析。通过bison -v命令生成tiny.out分析表

2、分析和总结

（对用工具和手工编写程序进行比较，给出结论）

1. 工具

根据lex和yacc工具给出的格式进行编写就可以完成词法分析和语法分析。

1. 手写编写程序

优点：手写编程的话会比较全面，对于输入，输出以及token的定义等等可以进行更加详细的设计。

缺点：写起来很耗时

3、实验中出现的冲突及解决过程描述

（理论，描述有哪些可能的冲突，操作中，描述具体遇到的冲突和解决方案）

1. 没有对tiny.y和y.tab.c中文件进行处理直接进行lex和yacc之后make需要先对tiny.y中的yylex方法去除再进行yacc -d tiny.y 生成y.tab.c再对y.tab.c中的yychar=yyylex进行修改替换
2. 对于yychar=yylex()的修改可能会改错位置因为在tiny.y中有许多yychar字符，而yychar=yylex()
3. tiny.y中去掉yylex方法可能没去掉干净yylex 方法需要全部去除，包括函数的定义和实现内容
4. 在lex和yacc之后，生成lex.yy.c和y.tab.c和y.tab.h之后再进行make