

****

软 件 学 院

《编译技术》实验报告

**题　　目 语 法 制 导 的 三 地 址 代 码 生 成 程 序**

**姓　　名 陈澄**

**学　　号 32420212202930**

**班　　级 软工三班**

**实验时间 2024/05/14**

**2024 年 05 月 14 日**

# 实验目的

掌握计算机语言的语法分析程序设计与属性文法应用的实现方法。

# 实验环境

编写语言：C++

编译环境：Visual Studio2022

# 实验内容

编制一个能够进行语法分析并生成三地址代码的微型编译程序。

# 实验步骤

1、考虑给定的文法，消除左递归，提取左因子。

给定文法G如下：

S -> id = E ;

S -> if C then S1 ;

S -> if C then S1 else S2 ;

S -> while C do S1 ;

C -> E1 > E2

C -> E1 < E2

C -> E1 = E2

E -> E1 + T

E -> E1 - T

E -> T

T -> F

T -> T1 \* F

T -> T1 / F

F -> (E)

F -> id

F -> int8

F -> int10

F -> int 16

其中仅标红部分有左递归，消除左递归，提取左因子后文法G如下：

S -> id = E ;

S -> if C then S S’;

S’ -> ε

S’ -> else S

S -> while C do S ;

C -> E C’

C’ -> > E

C’ -> < E

C’ -> = E

E -> T E’

E’ -> + E E’

E’ -> - E E’

E’ -> ε

T -> F T’

T’ -> \* F T’

T’ -> / F T’

T’ -> ε

F -> (E)

F -> id

F -> int8

F -> int10

F -> int 16

2、编制并化简语法图。

FIRST(S) = {id , if , while}

FIRST(S’) = {else , ε}

FOLLOW(S) = {else , ; , #}

FOLLOW(S’) = {else , ; , #}

FIRST(C) = {( , id , int 8 , int10 , int16}

FIRST(C’) = {> , < , =}

FOLLOW(C) = {do , then}

FOLLOW(C’) = {do , then}

FIRST(E) = {( , id , int 8 , int10 , int16}

FIRST(E′) = {+ , - , ε}

FIRST(T) = {( , id , int 8 , int10 , int16}

FIRST(T′) = {\* , / , ε}

FIRST(F) = {( , id , int 8 , int10 , int16}

FOLLOW(E) = {) , # , > , < , = , ;}

FOLLOW(E′) = {) , # , > , < , = , ;}

FOLLOW(T) = {+ , - , ) , # , > , < , = , ;}

FOLLOW(T′) = {+ , - , ) , # , > , < , = , ;}

FOLLOW(F) = {\* , / , + , - , ) , # , > , < , = , ;}

假设then匹配最近的else则满足LL(1)文法。

语法图（预测分析表）：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | id | if | else | while | 其他 | # |
| S | S-> id = E | S-> if C then S S' |  | S-> while C do S |  |  |
| S’ |  |  | S'-> else S |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | > | < | = | 其他 |
| C | C-> E C' | C-> E C' | C-> E C' | C-> E C' |
| C’ | C'-> +|-|= E | C'-> +|-|= E | C'-> +|-|= E |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | + | - | \* | / | 其他 |
| E | E-> T E' | E-> T E' | E-> T E' | E-> T E' | E-> T E' |
| E’ | E'-> +|- E E' | E'-> +|- E E' |  |  |  |
| T | T-> F T' | T-> F T' | T-> F T' | T-> F T' | T-> F T' |
| T’ |  |  | T'-> \*|/ F T' | T'-> \*|/ F T' |  |

|  |  |  |
| --- | --- | --- |
|  | ( | idn | int8 | int10 | int16 |
| F | F-> E | F-> id | int8 | int10 | int16 |

1. 编制递归子程序的算法。

void matchToken(int expected) {

if (lookahead != expected) {

err();

}

else {

lookahead = getToken();

}

}

Void 某非终结符() {

if(lookahead == token1) {

matchToken(token1);

...(按照产生式匹配后续token)

}

else if(lookahead == token2) {

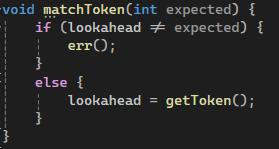
....;

}

}

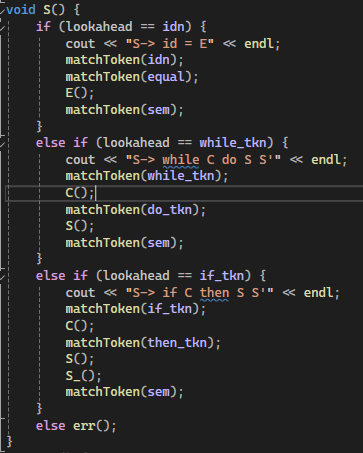
1. 编制各个递归子程序函数。

matchToken()方法如下：

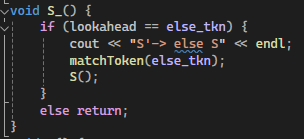


各个递归子程序函数：

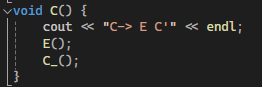
S：



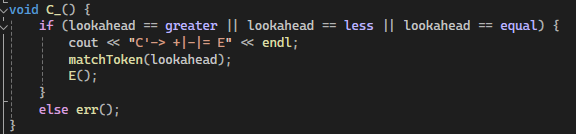
S’：



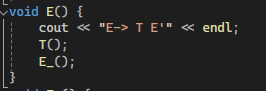
C：



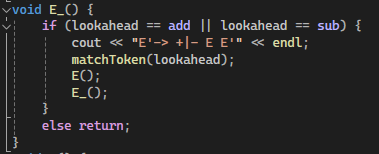
C’：



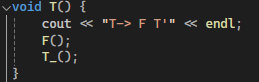
E：



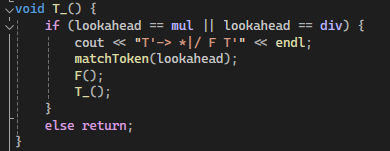
E’：



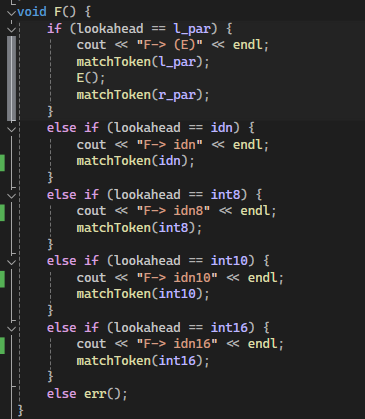
T：



T’：

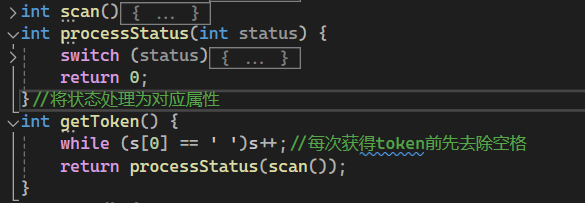


F：

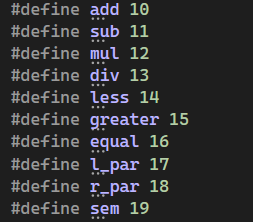
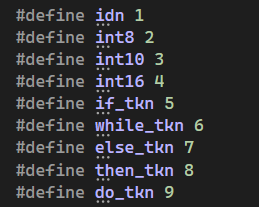


1. 连接实验一的词法分析函数scan( )，进行测试。

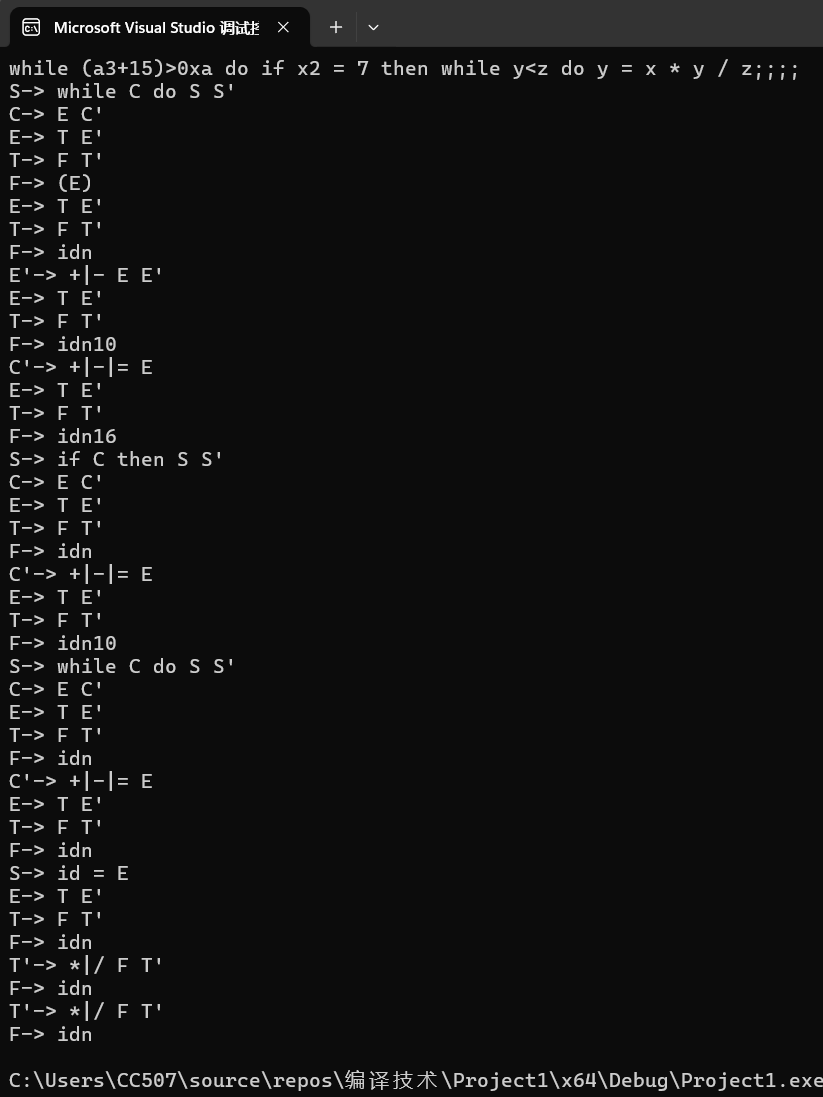
连接实验一的词法分析器编写getToken()方法：



token定义：



试运行结果：



可见语法分析没有问题，但是原语法有匹配多个分号的问题，故将原语法改为：

S0 -> S;

S -> id = E

S -> if C then S S’

S’ -> ε

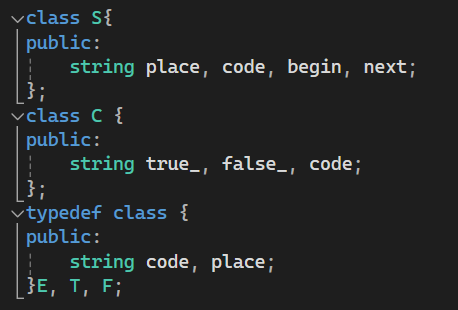
S’ -> else S

S -> while C do S

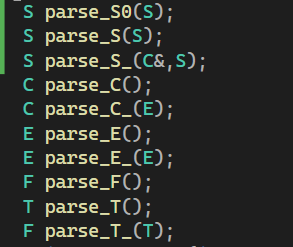
开始符号改为S0

1. 设计三地址代码生成的数据结构和算法。

数据结构如下设计：



函数名以及传参修改：



算法：

E,T,F中place存储当前变量名，code存储当前代码

算术表达式计算时place作为之前所有运算得到的值，并传入下一个递归子程序与下一个变量进行运算，最终传出并生成三地址代码存储在code中。

C中true\_、false\_存储当前布尔表达式的跳转标号，code存储当前代码

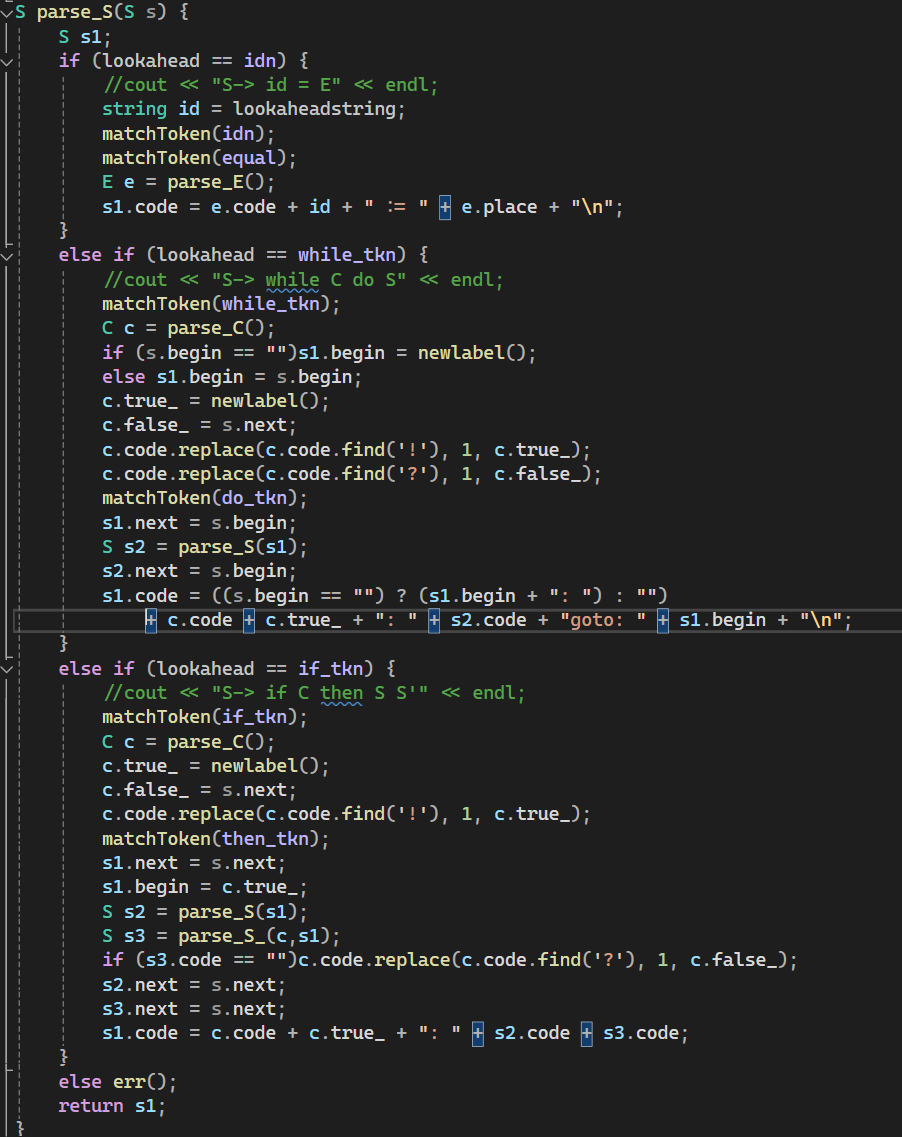
布尔表达式计算时true\_存入code时先使用“!”进行填入，等到整个语句分析完后再进行回填，false\_同理使用“?”先行填入。

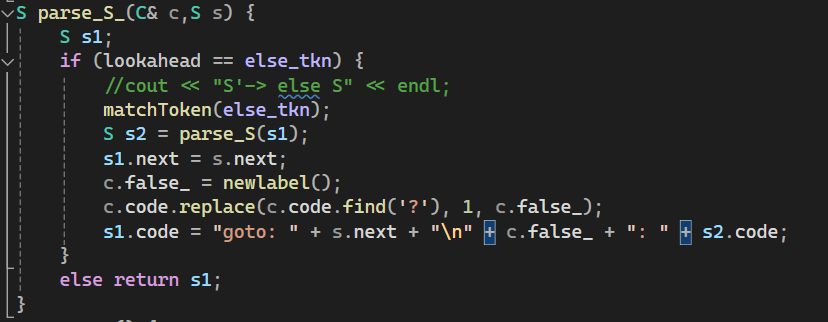
S中begin存储当前语句开始标号，next存储下一语句标号，code存储当前代码。

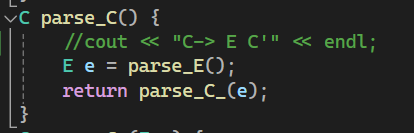
在递归子程序中生成的三地址代码会累加到code中最终返回。

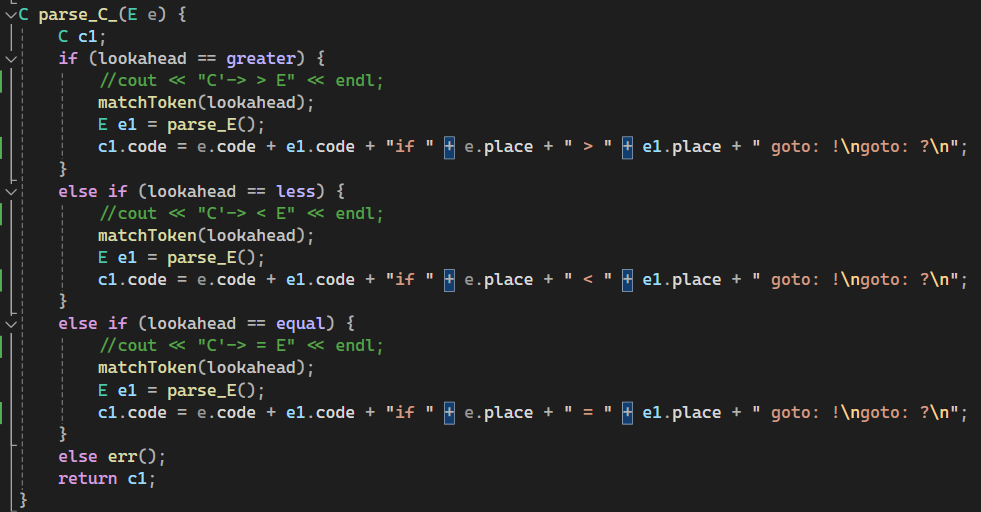
1. 将各个递归子程序函数改写为代码生成函数。

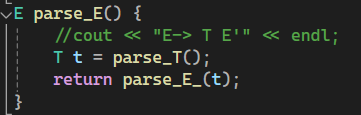
为各个递归子程序加入语义分析：

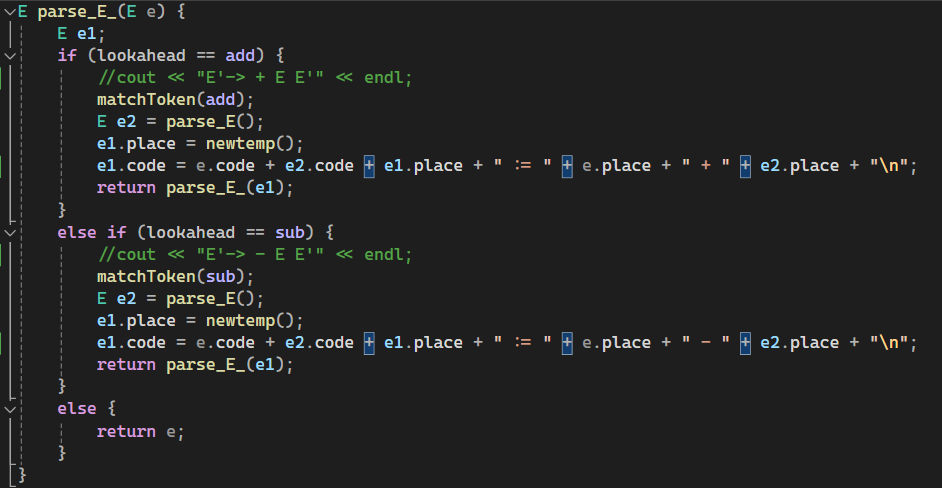


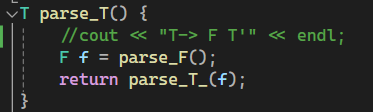


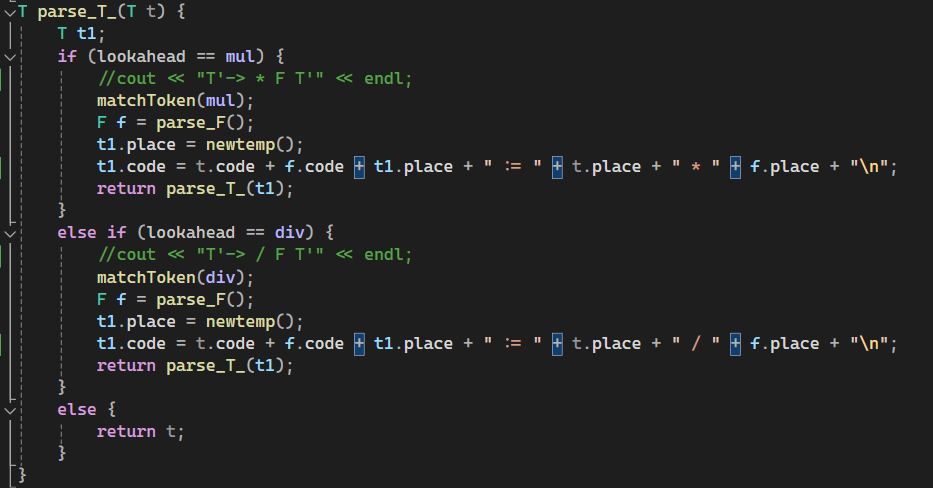


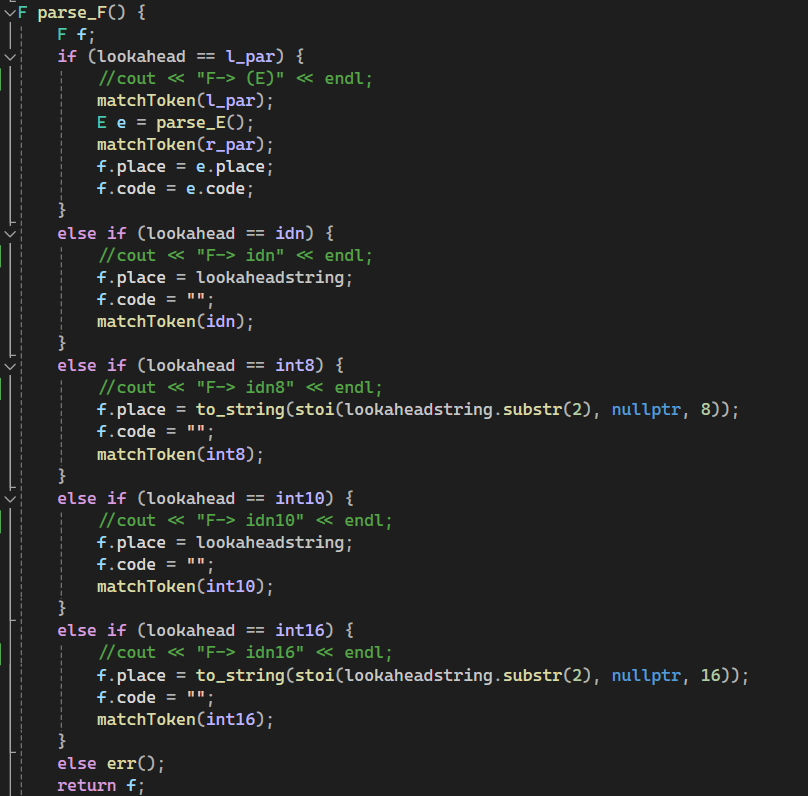


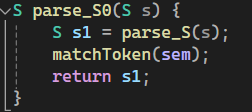










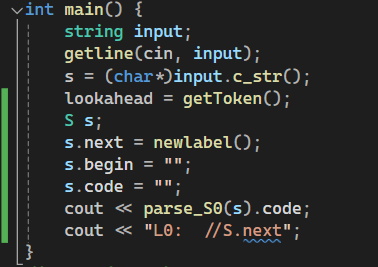


1. 编制测试程序（main函数）。

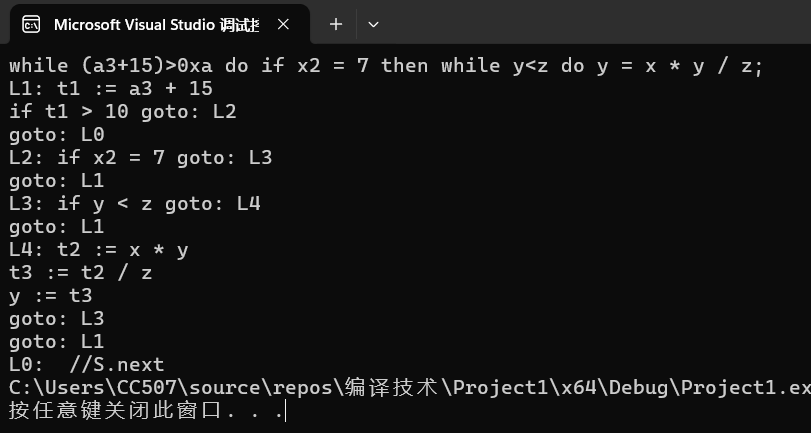
main函数中读取输入串，

获取第一个token,

调用parse\_S0()进行三地址代码生成并输出。



1. 调试程序：输入一个语句，检查输出的三地址代码 。



# 思考题

1. 生成的三地址代码可否直接输出（不采用数据结构来实现属性code）？

答：

不可直接输出，因为条件语句涉及代码的标号问题，跳转的标号需要整条语句识别完整之后才能确定，如果直接输出会导致无法确定标号。

1. 如何保证四则运算的优先关系和左结合性？

答：

优先关系：

文法中T代表乘除运算，E为加减运算，F为括号运算，文法中F的递归深度大于T的递归深度大于E意味着识别时会优先识别括号然后是乘除最后才进行加减。

左结合性：

匹配token时从左到右识别即可保证左结合性。

1. 如何采用代码段相对地址代替三地址代码序列中的标号？

答：

遍历一遍生成的三地址代码，按顺序为每条代码生成一个相对地址即可。