**实验 1 表达式翻译器**

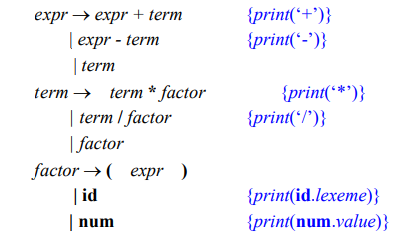
--22920212204144 梁丽丽

# 实验目的

构造一个从中缀表达式到后缀形式的表达式翻译器，初步了解递归下降语法分析 原理及语法制导翻译的过程。

# 实验内容

1. 表达式文法及其翻译模式定义如下：



2.说明：

1）num 为无符号整数；

2）在后缀形式输出时，为了能区分每个单词，单词之间用空格间隔，例如，

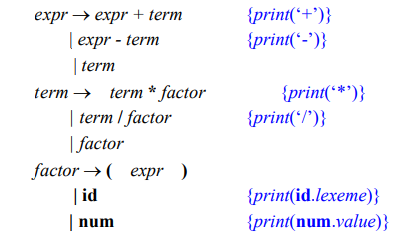
输入：a+18\*b

输出：a 18 b \* + 3）

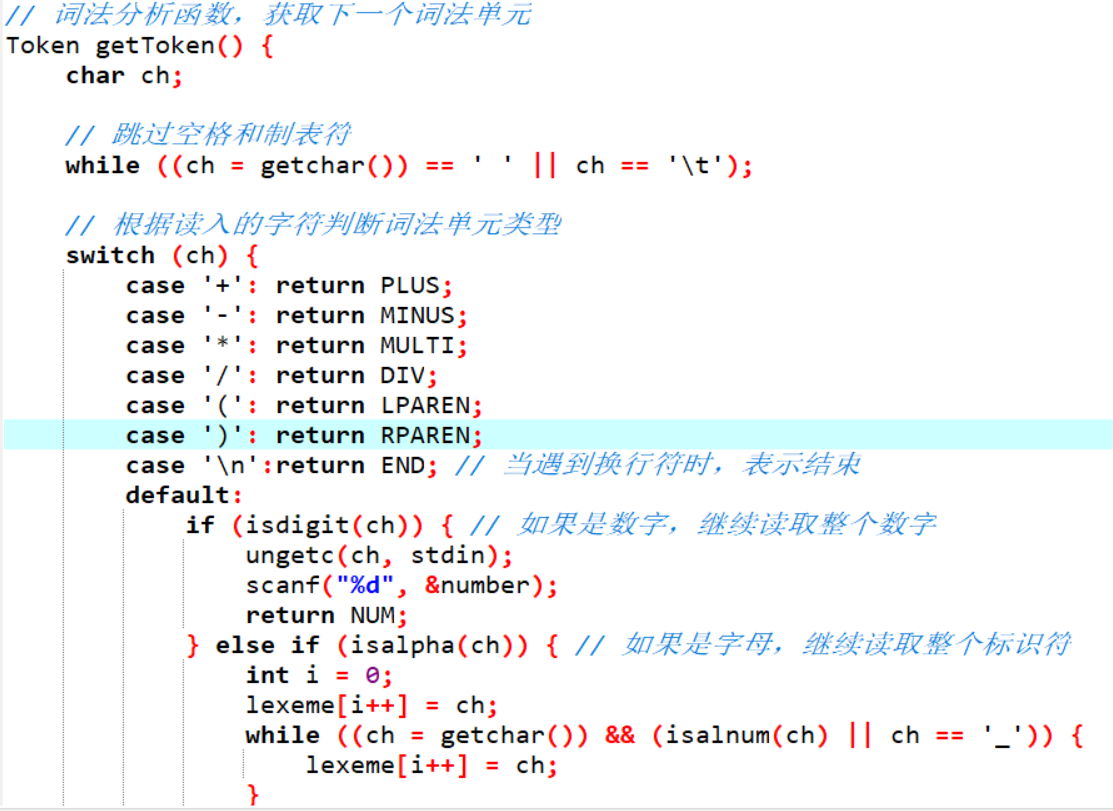
3）改写文法，消除该文法的左递归，然后使用语法制导翻译方案方法实现表达式的翻译；

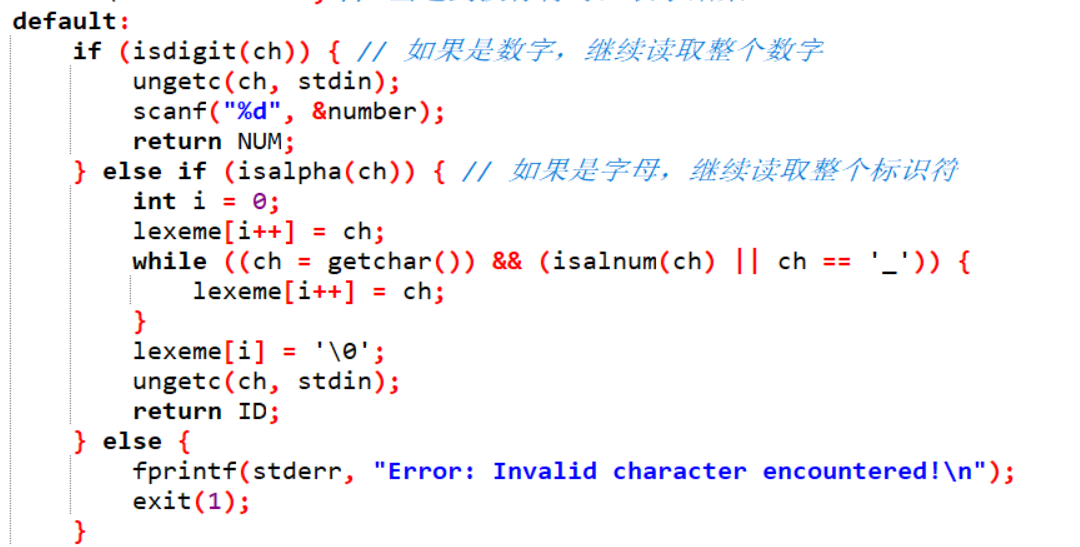
4）考虑简单的报错，如非法变量名、非法常量、非法字符、缺少操作数、括号不匹配等。

# 技术方法



**代码解析：**





getToken()：

该函数负责词法分析，它从输入中读取字符并确定标记类型。

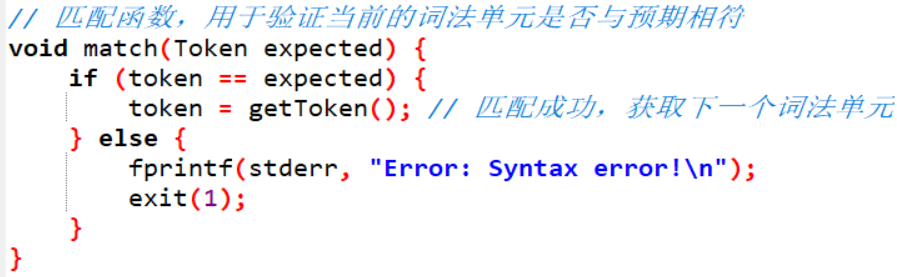
它跳过空格和制表符。

它使用switch语句在不同字符之间切换以识别标记类型，如加号、减号、数字、标识符等。

对于数字，它读取整个数字并将其存储在全局变量number中。

对于标识符，它读取整个字符串并将其存储在全局数组lexeme中。

它返回标记类型。

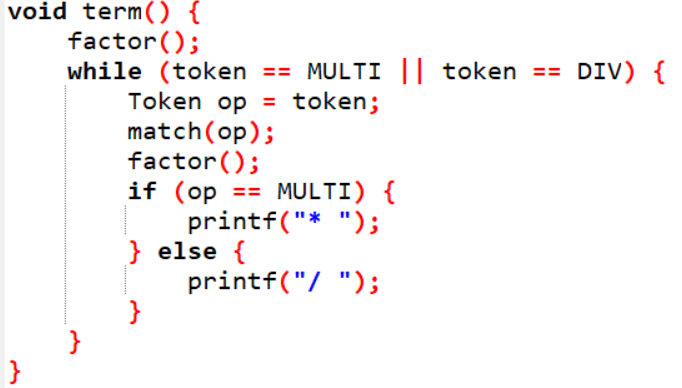


match(Token expected)：

该函数用于验证当前标记是否与预期标记匹配。

如果匹配成功，它使用getToken()获取下一个标记。

如果没有匹配成功，它打印错误消息并退出程序。



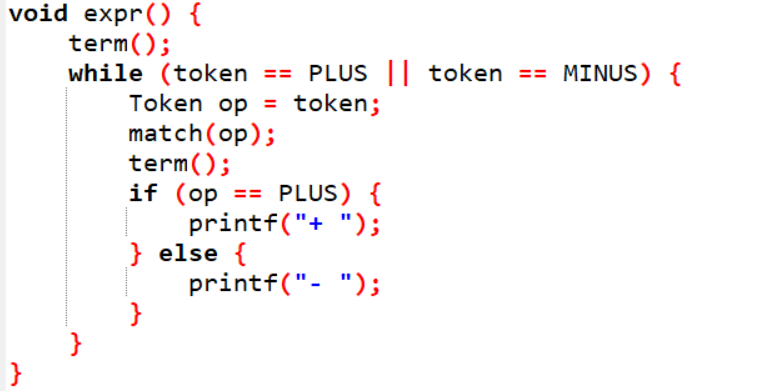
term()：

该函数负责解析表达式中的项。

它调用factor()解析因子。

如果有连续的乘法或除法操作，它打印相应的符号。

它循环直到找不到更多的乘法或除法操作为止。



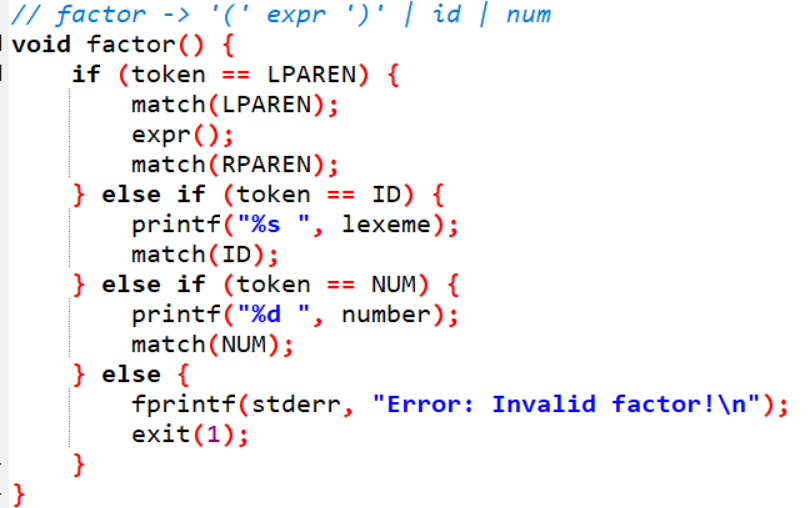
expr()：

该函数负责解析表达式。

它调用term()解析一个项。

如果有连续的加法或减法操作，它打印相应的符号。

它循环直到找不到更多的加法或减法操作为止。



factor()：

该函数负责解析表达式中的因子。

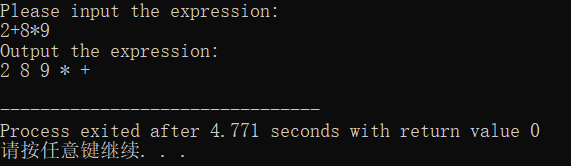
它可以是括号表达式、标识符或数字。

它检查当前标记类型并调用适当的解析函数或打印标识符/数字。

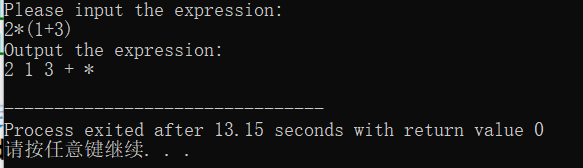
如果没有找到预期的标记类型，则打印错误消息并退出程序。

# 完成功能

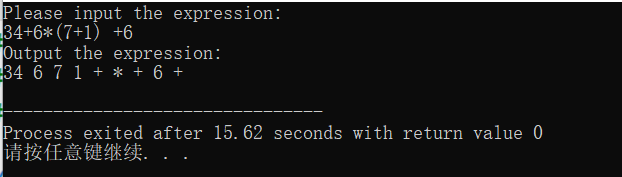
实现符号优先级的后缀表达式：



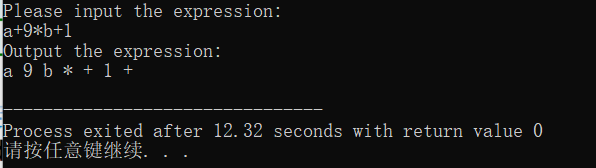
实现（）优先的后缀表达式：



能识别表达式中的空格、多位无符号整数



能识别表达式中的变量



# 解决问题

* 问题1： 在对表达式进行语法分析时，如何正确判断括号的优先级。
* 解决方案：在factor() 函数中，如果是左括号，则调用 expr() 函数处理括号内的表达式。
* 问题2： 在处理\*、/和+、-的优先级时，如何编程
* 解决方案：当解析器遇到一个乘法或除法运算符时，它会调用 term() 函数来处理乘法或除法部分，其中会调用 factor() 函数来处理因子。这样可以确保先处理乘法或除法，保证了它们的优先级。

# 实验心得体会

通过本次实验，我加深了对编译原理中词法分析和语法分析的理解，以及如何将理论知识转化为实际的代码实现。同时，我也学会了如何处理错误、封装函数等编程技巧，这对提高程序的质量和可维护性非常重要。