SHANGHAI UNIVERSITY

**《编译原理》实验报告**

|  |  |
| --- | --- |
| **学 院** | **计算机工程与科学学院** |
| **组 号** | **10** |
| **实验题号** | **三** |
| **日期** | **2023.4.19** |

|  |  |  |  |
| --- | --- | --- | --- |
| 学号 | 姓名 | 主要工作 | 贡献因子 |
| 20124652 | 邵启承 | 报告撰写，代码测试 | 0.25 |
| 20124698 | 李子浩 | 代码优化，使之优美 | 0.25 |
| 20124699 | 徐驰 | 代码构思，初步实现 | 0.25 |
| 20124703 | 陈世锦 | 代码优化，使之高效 | 0.25 |

**<编译原理>实验报告**

**一、实验目的与要求**

⚫ 给出 PL/0 文法规范，要求编写 PL/0 语言的语法分析程序。

⚫ 通过设计、编制、调试一个典型的语法分析程序，实现对词法分析程

序所提供的单词序列进行语法检查和结构分析，进一步掌握常用的

语法分析方法。

⚫ 选择一种语法分析方法（递归子程序法、LL(1)分析法、算符优先分

析法、SLR(1)分析法）；选择常见程序语言都具备的语法结构，如赋

值语句，特别是表达式，作为分析对象。

**二、实验环境**

微机 CPU P4 以上，256M 以上内存，安装好 C 语言，或 C++，或 Visual

C++。

**三、实验内容**

⚫ 已给 PL/0 语言文法，构造表达式部分的语法分析器。

⚫ 分析对象〈算术表达式〉的 BNF 定义如下：

<表达式> ::= [+|-]<项>{<加法运算符> <项>}

<项> ::= <因子>{<乘法运算符> <因子>}

<因子> ::= <标识符>|<无符号整数>| ‘

(’<表达式>‘

)’

<加法运算符> ::= +|-

<乘法运算符> ::= \*|/

<关系运算符> ::= =|#|<|<=|>|>=

**四、实验内容的设计与实现**

**说明：**选择本实验中最有程序设计技巧或特色的并具有独立功能的源代码片断，并对其算法的技巧或特色进行必要的文字说明。

#include <iostream>//提供了在 C++ 中执行输入和输出操作的功能，用于从标准输入和输出流中读取和写入数据。

#include <string>//提供了在 C++ 中使用字符串的函数和类。

#include <vector>//提供了 std::vector 类，它是一个可以在运行时调整大小的动态数组。

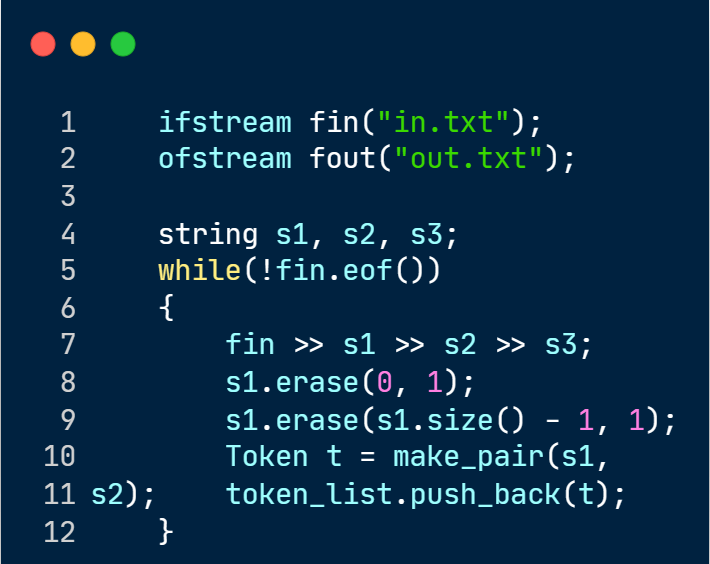
#include <utility>//提供了各种在 C++ 中使用的实用函数和类。

#include <fstream>//提供了在 C++ 中对文件执行输入和输出操作的功能，用于从文件中读取和写入数据。

#include <cstdio>// 标准输入/输出库。它定义了一组函数，提供基本的输入和输出功能。

1. **文件读入和初始化**

代码从文件进行语法分析器的读入。代码定义一个读入流fin，从文件”in.txt”进行读入。其中由于读入的字符串为实验2的输出内容，所以在此处使用三个字符串s1、s2、s3来对被空格分开的字符串进行分别的读入。



* 初始化

s1是单词种类，s2是单词的值，s3是实验2中输出的右括号，在本实验中无用。其中s1利用string函数erase()来删除头部的左括号和尾部的逗号，提取出单词种类，然后和s2进行组合，使其成为一个token，并推入处理队列中等待后续处理。

* 文件操作

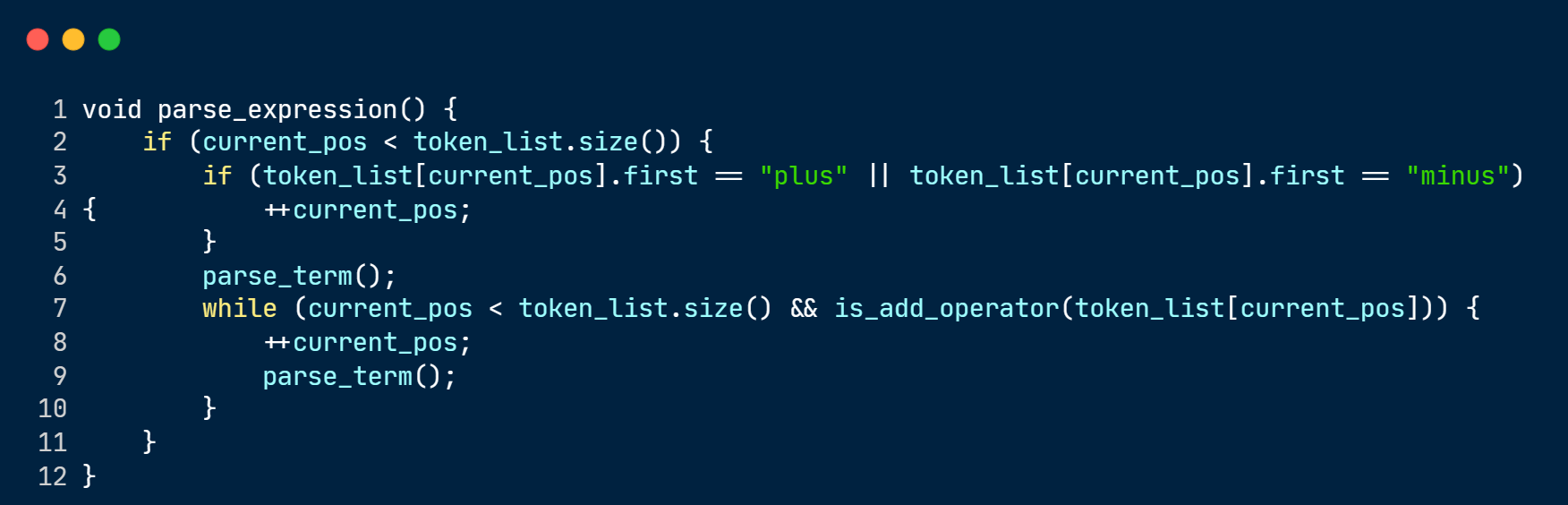
**ofstream outfile("out.txt")** 创建了一个输出文件流对象，将输出结果写入到名为 "out.txt" 的文件中。**ifstream infile("in.txt")** 创建了一个输入文件流对象，读取输入数据来自名为 "in.txt" 的文件。

**while (!fin.eof())** 通过 ifstream 对象从输入流中不断读取s1、s2、s3，读到文件结束符EOF为止。该循环用于处理每行中的所有单词。

1. **解析过程**

* 表达式解析

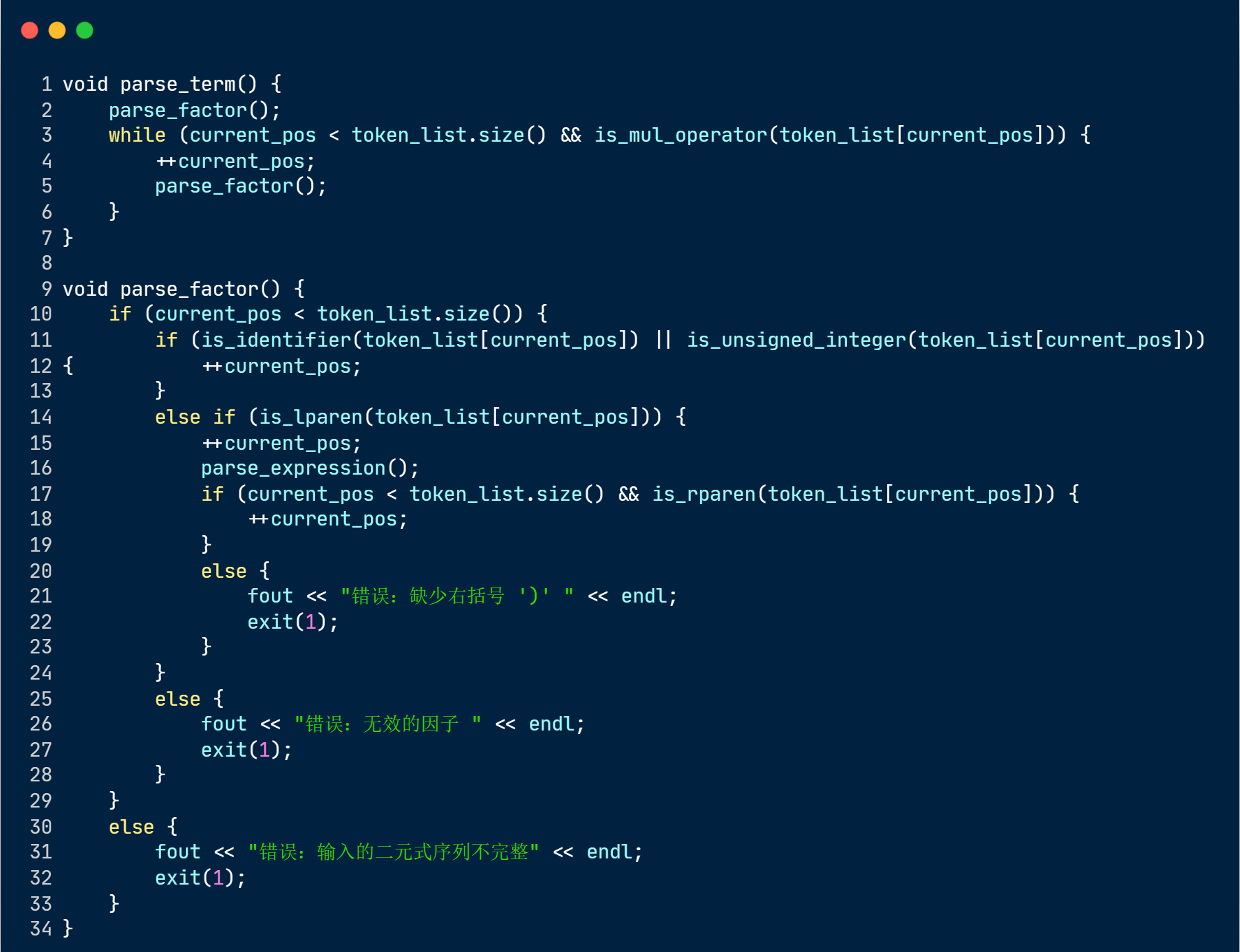
表达式解析函数首先判断表达式的第一个符号是否为正负号，如果是则将解析位置向后移一位。然后解析表达式的第一个项，如果后面还有加减运算符，则继续解析项。如果解析成功，输出正确的表达式，否则输出错误信息。



* 项和因子解析

parse\_term() 函数解析一个项，它被定义为一个或多个因子，由运算符分隔。项解析首先调用 parse\_factor() 函数来解析项中的第一个因子。在循环中检查当前的标记是否为乘法运算符，以及是否还有更多的标记需要解析。如果当前的标记是乘法运算符，它会前进到下一个标记并调用 parse\_factor() 来解析项中的下一个因子。

parse\_factor() 函数负责解析一个因子，它可以是标识符、无符号整数或括在括号中的表达式。它首先检查当前标记是否为标识符或无符号整数。如果是，它会前进到下一个标记。如果当前标记是左括号，它会前进到下一个标记并调用 parse\_expression() 来解析括号内的表达式。最后，如果标记既不是标识符、无符号整数，也不是左括号，它会报告一个错误并退出程序。

这两个函数使用 current\_pos 变量来跟踪标记列表中的当前位置。如果在解析完整个表达式之前到达标记列表的末尾，它会报告一个错误并退出程序。

* 辅助函数

这些函数是用于判断标记类型的函数，它们返回一个布尔值，指示给定的标记是否是特定类型的标记。

is\_add\_operator() 函数接受一个标记作为参数，如果标记是 "plus" 或 "minus"，则返回 true，否则返回 false。

is\_mul\_operator() 函数接受一个标记作为参数，如果标记是 "times" 或 "divide"，则返回 true，否则返回 false。

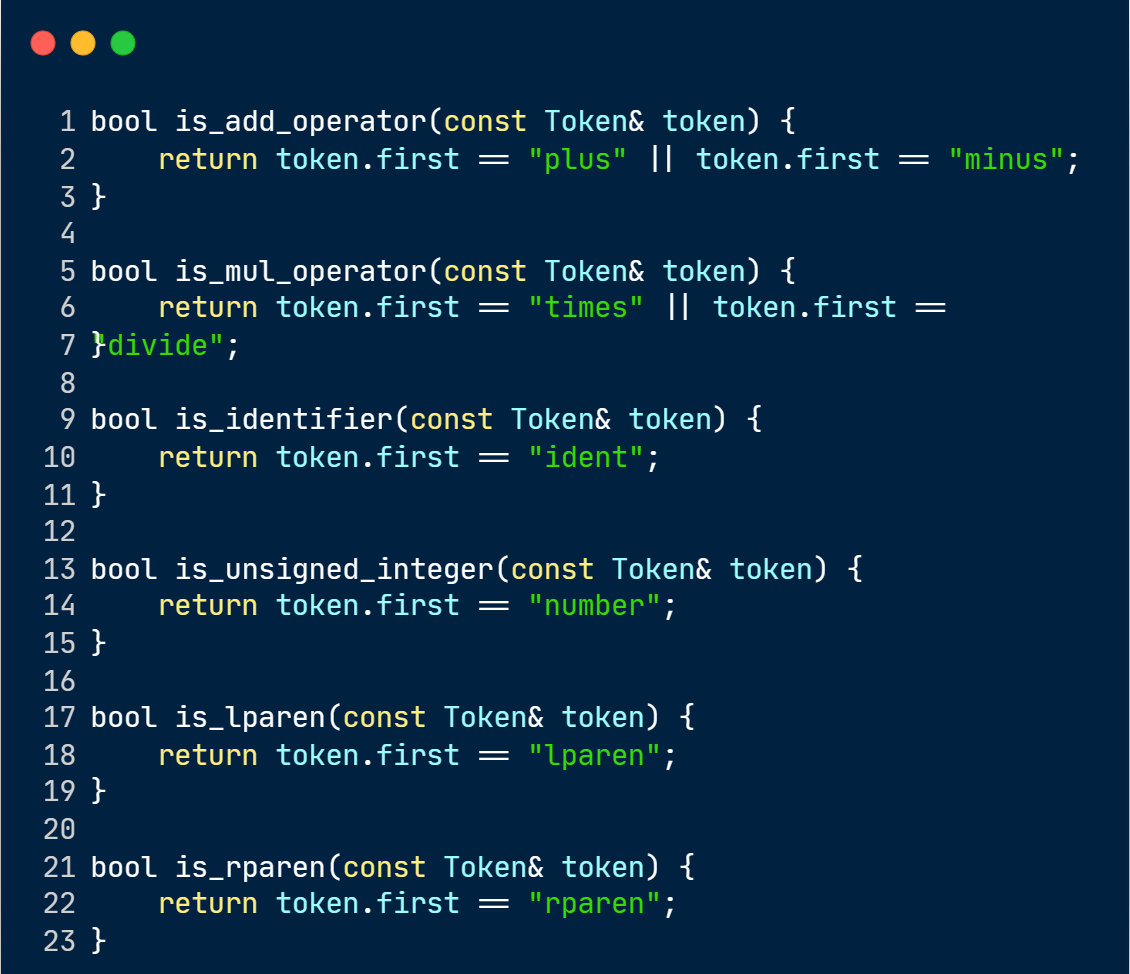
is\_identifier() 函数接受一个标记作为参数，如果标记是 "ident"，则返回 true，否则返回 false。

is\_unsigned\_integer() 函数接受一个标记作为参数，如果标记是 "number"，则返回 true，否则返回 false。

is\_lparen() 函数接受一个标记作为参数，如果标记是 "lparen"，则返回 true，否则返回 false。

is\_rparen() 函数接受一个标记作为参数，如果标记是 "rparen"，则返回 true，否则返回 false。

这些函数的作用是在解析器中用于确定标记的类型。在解析器中，标记的类型对于正确解析语言非常重要。



1. **判断语法的正确性**

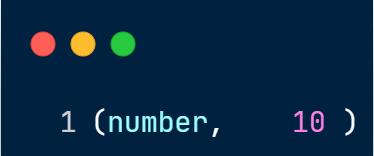
在函数的主体部分，首先检查输入的标记列表是否为空，如果是，则输出一个错误消息并返回。然后调用 parse\_expression() 函数来解析整个表达式。其中解析和判断的过程如以上所提到的一样。如果表达式解析成功，即 current\_pos 等于标记列表的大小，那么输出一个正确的消息。否则，输出一个解析错误的消息。

如果在解析期间发生异常，则 catch 块捕获异常并输出异常消息。

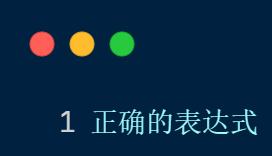


1. **测试用例与实验结果**

**测试用例1：**



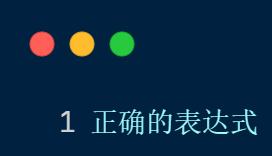
**运行结果：**



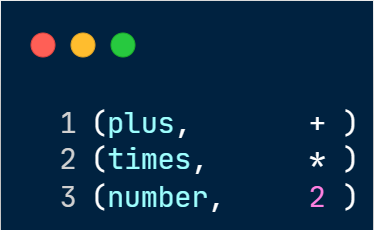
**测试用例2：**



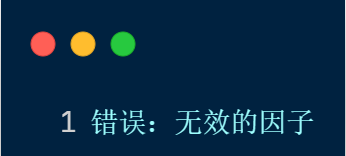
**运行结果：**



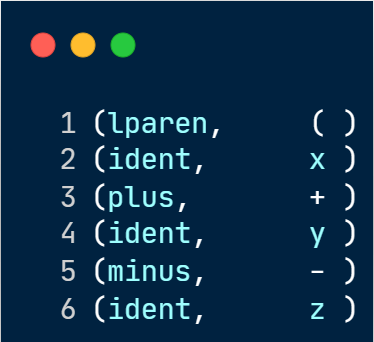
**测试用例3：**



**运行结果：**



**测试用例4：**



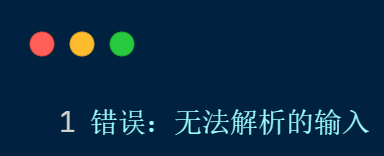
**运行结果：**



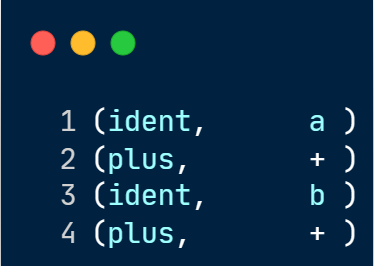
**测试用例5：**



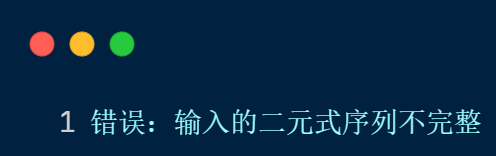
**运行结果：**



**测试用例6：**



**运行结果：**



**五、收获与体会**

徐驰：在这次实验中，我深刻认识到了递归子程序方法的优点和不足之处。递归子程序方法的优点在于简单易懂，易于实现，容易调试和维护。但是，递归子程序方法也有一些不足之处，比如容易产生左递归和回溯，可能会导致性能问题。因此，在实现语法分析器时，需要权衡这些因素，并选择最适合的方法。另外，在实验过程中，我也学会了如何将一个语法规则转化为递归子程序，并如何使用语法分析器来识别输入的符号串是否符合某个语法规则。

邵启承：通过这次实验，我更深刻地认识到了语法分析器的重要性和实现方法。语法分析器是编译器中的重要组成部分，用于对源代码进行语法分析，识别其中的语法错误和语义错误。而递归子程序方法是一种常见的语法分析方法，其原理是将一个语法规则转化为一个对应的递归函数。在实现语法分析器时，需要根据语法规则编写相应的递归函数，并使用递归调用来实现语法分析。这次实验让我对编译原理有了更深入的理解，也让我更加熟练地掌握了递归子程序方法。

陈世锦：在这次实验中，我学会了如何使用递归子程序方法来进行语法分析。递归子程序方法是一种基于递归调用的语法分析方法，其优点在于简单易懂，容易实现和调试。但是，递归子程序方法也有一些不足之处，如可能产生左递归和回溯。在实现语法分析器时，需要根据语法规则编写相应的递归函数，并考虑如何处理左递归和回溯的问题。通过这次实验，我深入了解了语法分析器的实现原理和方法，并学会了如何使用递归子程序方法来进行语法分析。

李子浩：通过这次实验，我深入了解了递归子程序方法的原理和实现方法，并学会了如何将一个语法规则转化为递归子程序。在实现语法分析器时，我遇到了一些挑战。为了解决这些问题，我学习了大量的知识。通过这次实验，我不仅学会了递归子程序方法的实现，也学会了如何权衡不同的实现方法，并选择最适合的方法来实现语法分析器。

**2023.4.19**