**实验报告**

目录

[实验内容 1](#_Toc135525137)

[一、讨论语法的二义性 1](#_Toc135525138)

[二、设计并实现词法分析程序 2](#_Toc135525139)

[三、构造算符优先级关系表 3](#_Toc135525140)

[四、设计并实现语法分析和语义处理程序 4](#_Toc135525141)

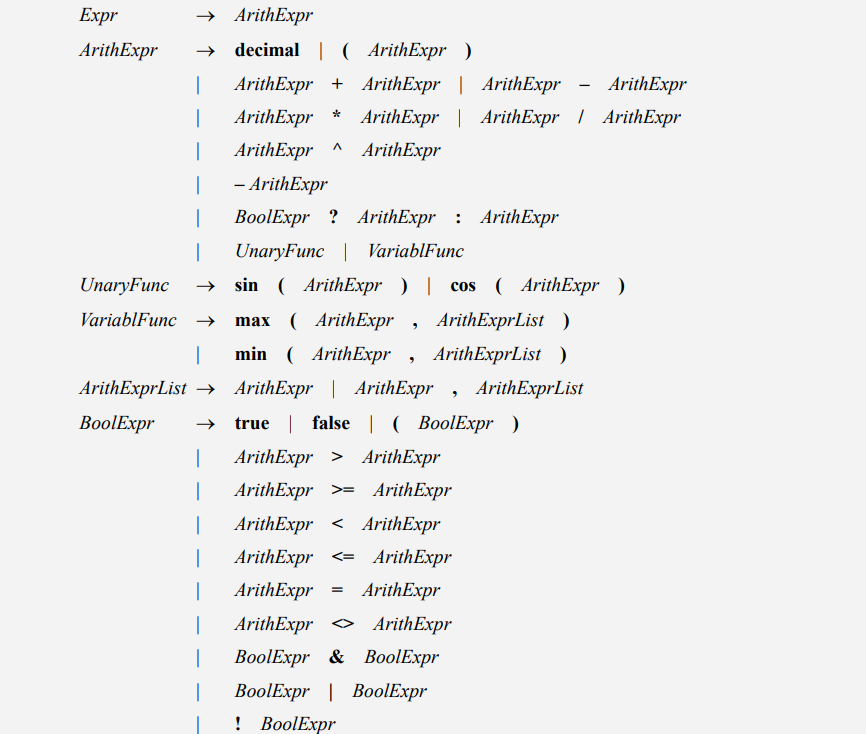
[运行结果 5](#_Toc135525142)

[心得体会 7](#_Toc135525143)

## 实验内容

### 一、讨论语法的二义性

给定的文法如下



显然是有二义性的，比如对于简单的表达式decimal + decimal \* decimal就有两种形式的最左推导

其一

Expr -> ArithExpr

-> ArithExpr + ArithExpr

-> decimal + ArithExpr

-> decimal + ArithExpr \* ArithExpr

-> decimal + decimal \* ArithExpr

-> decimal + decimal \* decimal

其二：

Expr -> ArithExpr

-> ArithExpr \* ArithExpr

-> ArithExpr + ArithExpr \* ArithExpr

-> decimal + ArithExpr \* ArithExpr

-> decimal + decimal \* decimal

-> decimal + decimal \* decima

故该文法是具有二义性的

对于二义性，我们可以通过定义算符的优先级与结合性来解决

### 二、设计并实现词法分析程序

#### 单词分类

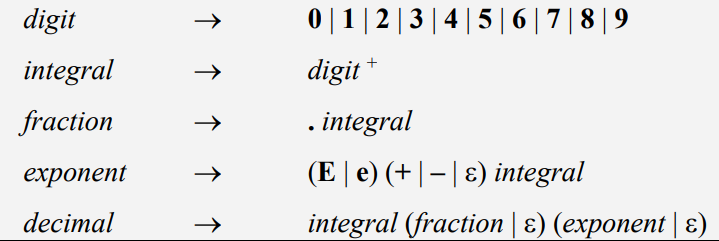
在token包中实现

在说明中，合法的单词种类布尔类型常量、数值类型常量（其中包括科学记数法）、各种算术运算、关系运算、逻辑运算、以及预定义函数

而在实际过程中，简单地将合法单词分成布尔类型常量（Bool），数值类型常量（Decimal），运算符（Operator）（包含算术运算符与关系运算符还有预定义函数），标点符号（Punctuation），还有终结符号（Dollar），一共五类

各个子类在抽象父类Token的基础上得到相应的标签，值（String类型和double类型）

对于数值常量类型，在实现时根据给定的文法



单独写了一个package **number**来定义不同的表达式如digit、integral等以便区分

#### 词法分析

在scanner包中实现

在词法分析的开始，将输入字符串中的大写字母均转换为小写字母，之后，每次获取当前的 Token。

对于数值常量类型，

若识别到digit，说明有数值常量，继续往后读，直到当前的字符不符合数值常量的定义，而在处理科学计数法的数值时，可以通过判断小数点以及 E 等的关系进行不同的处理，得到相同的常量数值

而在布尔类型或者预定义函数sin‘或者cos或者min或者max

若读到的字符为s或者c或者m，说明可能要出现完整地的函数名了，先判断一下预读后的字符下标（+3或者+4或者+5）是否合法，然后再读取相应长度的字符，判断是否是给定的函数名，若是，继续往后读取数值常量，然后计算值，根据值实例化相应的对象，，否则，直接抛出异常

布尔类型的扫描也是同理，根据读到的字符是不是t或者f进行如上处理

对于空格，在设计时，如识别到空格，直接停止读取，仅将之前的读取到的字串当成token处理，然后忽略空格，继续读取下一个token，若有异常便抛出异常

### 三、构造算符优先级关系表

算符优先关系表在实现上可以视为一个二维数组，其中 T[i][j] 表示当前符号栈栈顶的元素对应的标签为 i ，而当前读取到的符号对应的标签为 j 时，此时应该采取的移入/规约/报错的 action。

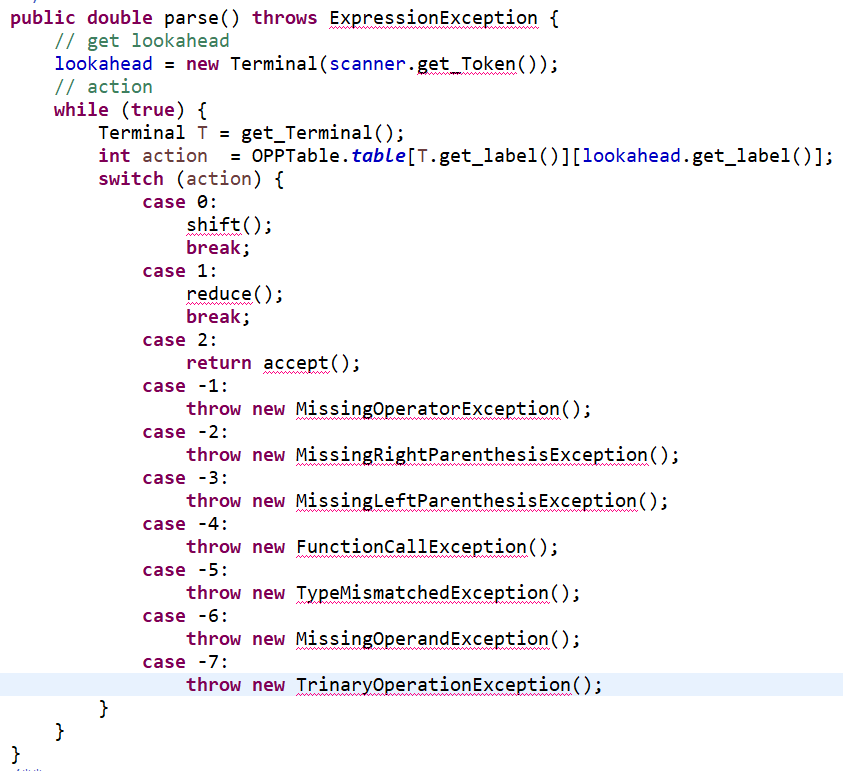
**对于一元取负运算符和二元减法运算符，在前面的scanner就进行判断，根据前一个token的类型，若前一个为数值类型或右括号则这个为二元减法运算符，其余情况为一元取负运算符。**

对于三元运算符，其优先级最低，只有在当’:’ 遇到终结符’$’ 时，才会进行归约

而对于四种预定义函数名，之后必须紧跟着括号，所以只有遇到’(’ 才是合法的，进行移入； 遇到其他任意运算符都抛出异常。

### 四、设计并实现语法分析和语义处理程序

在读入完整的输入串后，读取token，然后根据token和终结符查移进规约表，进行相应的操作，



对于shift操作也较为简单，将当前的lookahead入栈并读取下一个token在进行接下来的操作

对于reduce，可以细分为单目运算归约，双目运算归约，三目运算归约, 函数运算以及括号匹配归约。

在单目运算中，数值常量类型中，直接转换成实际的十进制数，然后再入栈即可

双目运算符，对于加减乘除等操作，先判断合不合法，合法取出相应的token，进行相应的计算得到十进制数，删除取出的token并将结果入栈即可

取负、取非再要求中也属于一元运算符，，简单地先删除栈顶字符“-”，然后对目前的栈顶字符，取反或取非，然后再压回栈中即可

三目运算符，只有 ……？ ……：……，先判断表达式是否合法，若合法，往右读取字符，

以上遇到的括号匹配是在读取到右括号’)’ 时执行的，从栈中弹出字符，然后执行相应的归约动作，直到遇到左括号’(’ 。当弹出“(“后，会从栈中再读取字符（即预期函数名），如果该词法单元的类型是函数，那么就执行相应的函数动作， 否则抛出异常，将相应的结果入栈，或者是为了保证优先级的括号，简单地记录即可，然后重复出栈，入栈操作即可

对于accept，直接返回栈中最后的结果就行

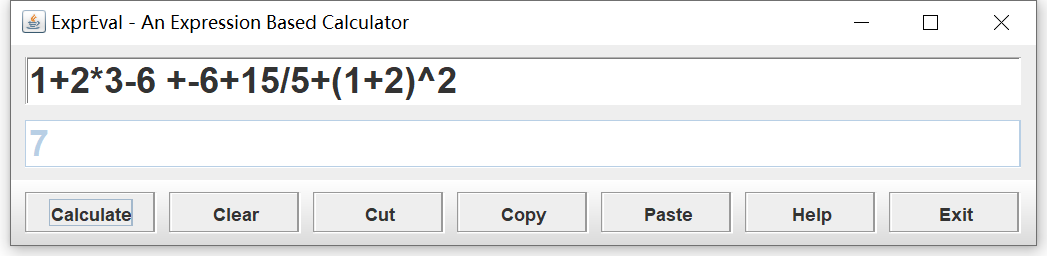
在语义处理部分，在调用 scanner 获取到token时，会给每个token赋予一个标签（特地建立了一张映射表map），以后便可以根据标签得到相应的token类型。在reduce时，会根据token所需要的相关token的类型，然后比对从栈中读取出来的token的标签，如果相同就继续reduce，否则，抛出异常。

## 运行结果

自己的测试样例

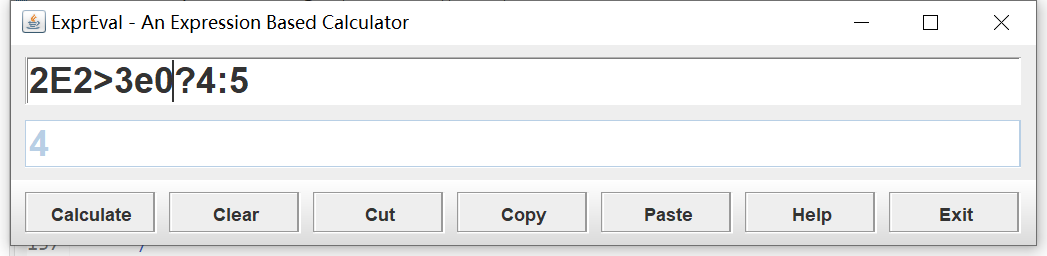
一

如计算1+2\*3-6 +-6+15/5+(1+2)^2



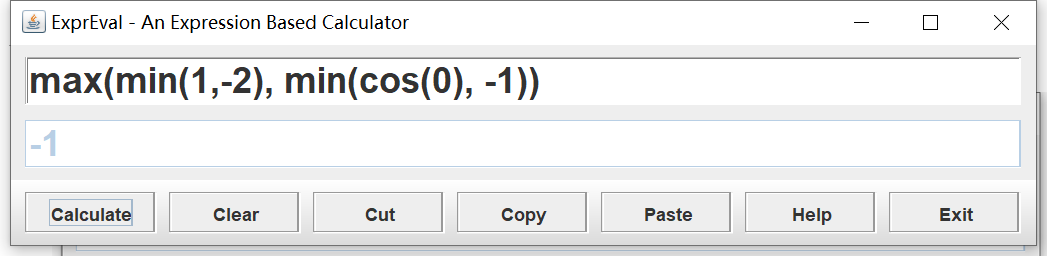
结果为7，符合实际结果，对于加减乘除，乘方，运算无误

二



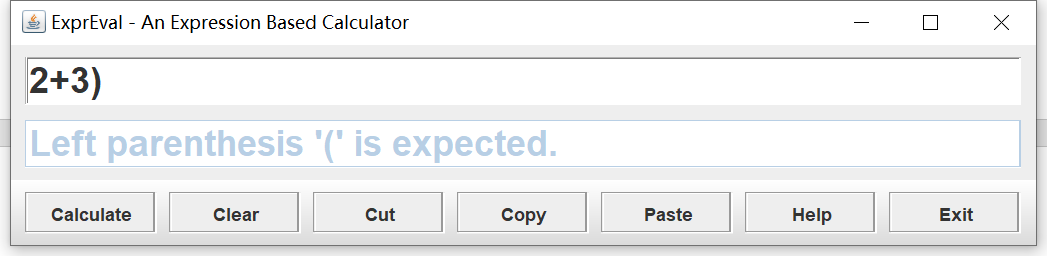
结果也是一致的，对于三目运算符，科学计数法无误

三



结果符合预期，对于max，min，cos等函数使用无误

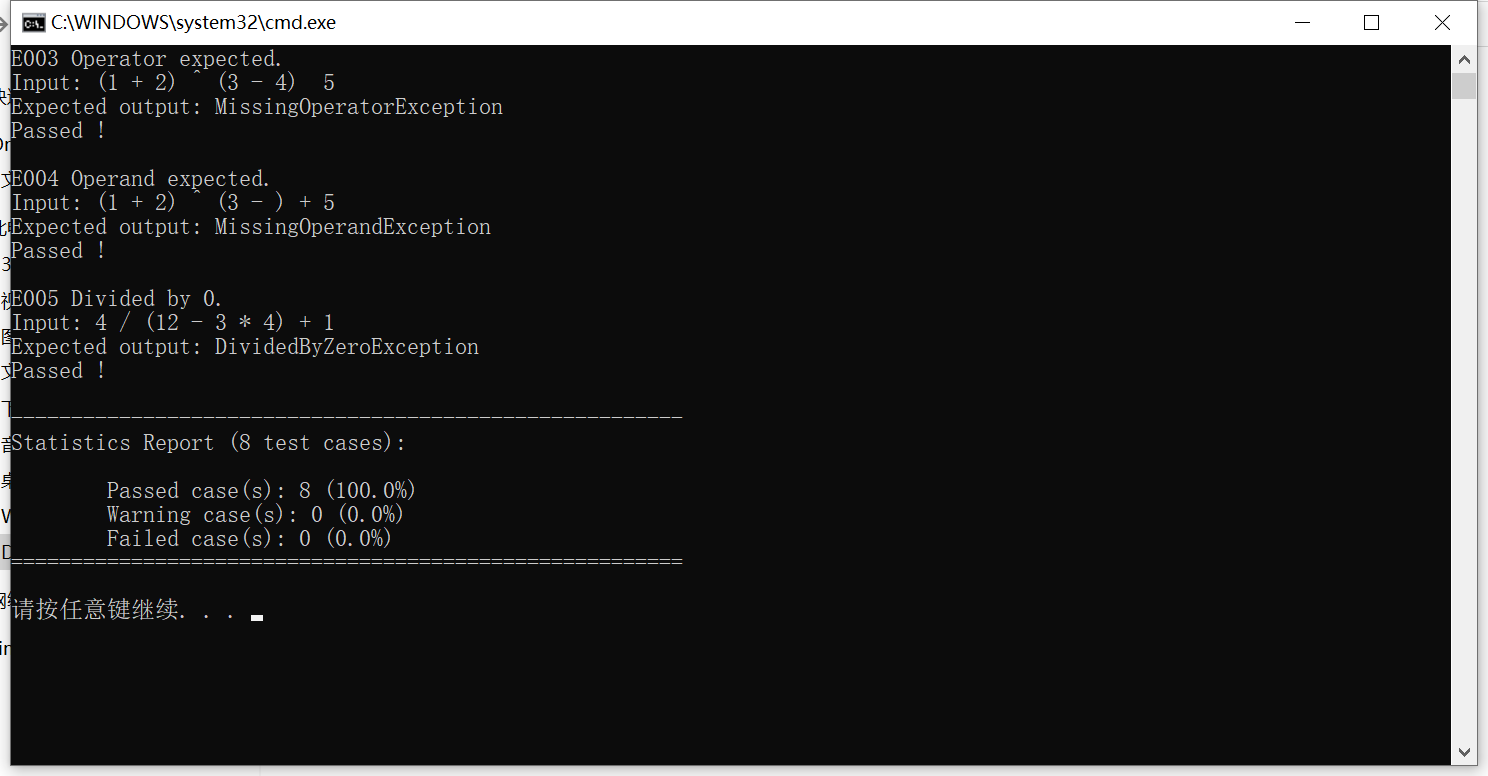
四



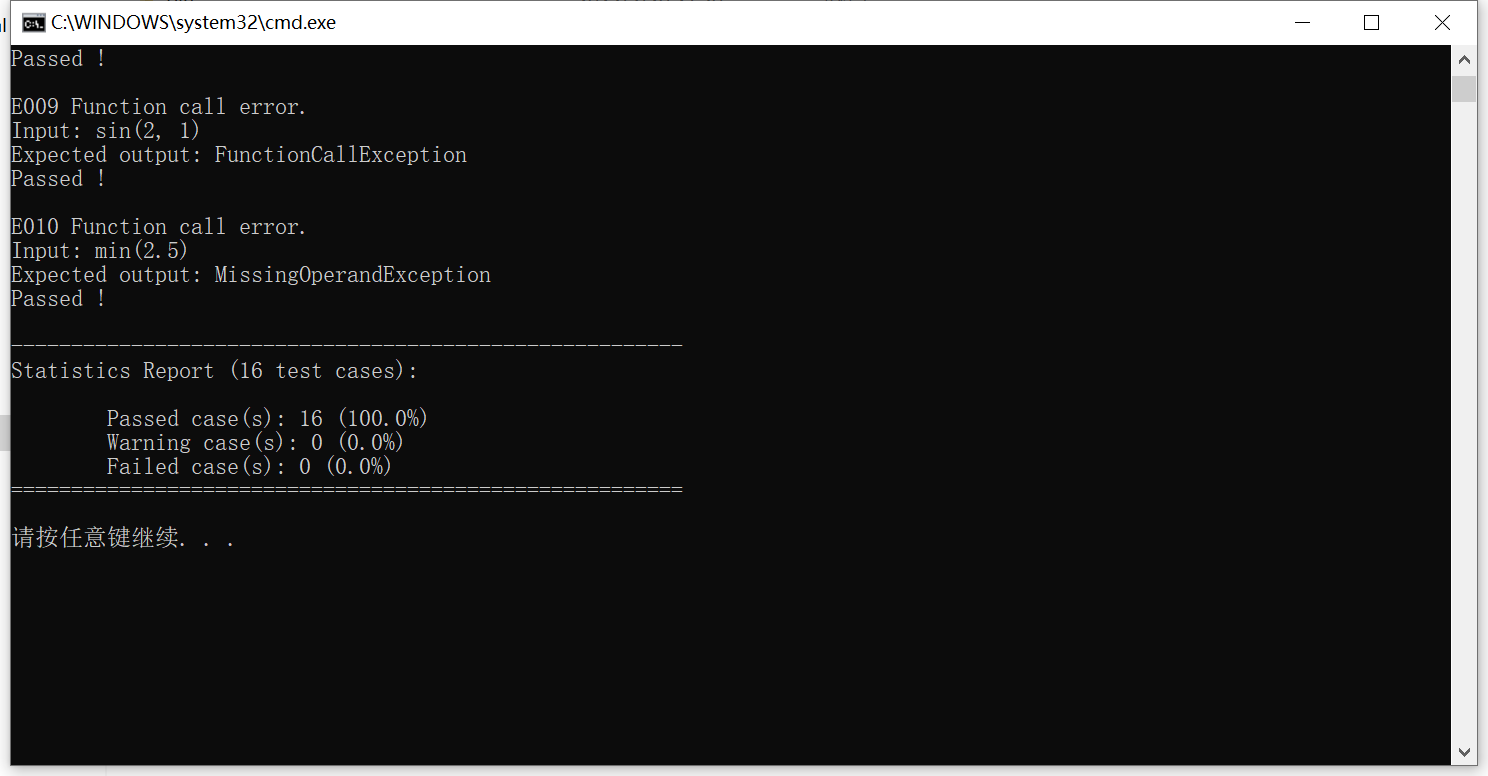
符合预期，正确处理括号匹配问题

以下是给定的样例的结果

test\_simple



test\_standard



两者的计算都无误

## 心得体会

在这次实验中，内容比较多，需要处理的细节比较繁琐，在写的时候，会时不时忘记写一点标签，或者及时更新标签、输入串当前下标等，但是最终还是慢慢一点点修正过来。这次实验的重点就是canner，parser的处理，OPP表的构建，在做这个实验的时候已经忘得差不多了，算是重新温习了一遍