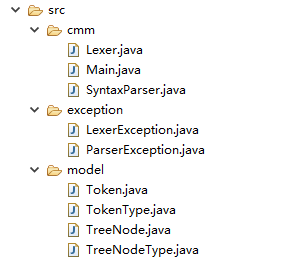
**Checkpoint 3 说明文档**

2015302580121 刘晓宇

1. **任务**

完成CMM语法分析器

1. **代码结构**



1. **功能**

在实验一的基础上，以单词序列为输入，分析单词序列构成的语法结构，给出相应的输出，并进行一定的错误处理。

和实验一进行整合，输入为任意的CMM源程序文件，输出是先序遍历语法树结点到屏幕上。以及实现了一个图形化展示语法树的小程序，鉴于这次不需要GUI界面，故下次交。

1. **主要设计思路**
2. 明确文法，修改文法

Program → Stmt { Stmt }

Stmt → VarDecl | IfStmt | WhileStmt | BreakStmt | AssignStmt |

ReadStmt | WriteStmt | StmtBlock

StmtBlock → **{** {Stmt}**}**

VarDecl → Type VarList;

Type → **int** | **double** | Type **[** **intconstant** **]**

VarList → **ident** { **,** **ident** }

IfStmt → **if** （Expr） [ **else** Stmt ]

WhileStmt → **while** （Expr） Stmt

BreakStmt → **break** **;**

ReadStmt → **read** **(** **ident | ident[intconstant] );**

WriteStmt → **write(**Expr**);**

AssignStmt → Value **=** Expr **;**

Value → **ident** [**intconstant**] | **ident**

Constant → **intconstant | doubleconstant | true | false**

Expr → Expr **+** Expr | Expr **–** Expr | Expr **\*** Expr | Expr **/** Expr |

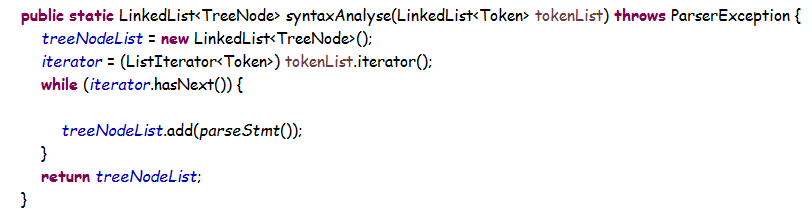
Expr **%** Expr | **−** Expr| Expr <= Expr | Expr < Expr |

Expr > Expr | Expr >= Expr | Expr != Expr | Expr == Expr |

**ident** | Constant

1. 自顶向下递归下降子程序法

（1） treeNodeList存放语句或者代码块的根节点，即森林中的每棵树的根节点

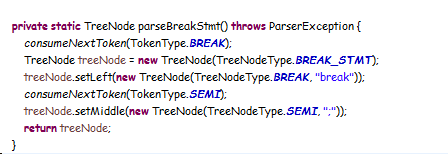


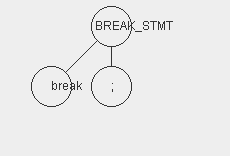
1. 语法分析statement主函数



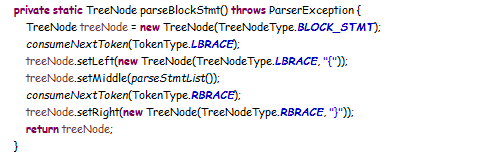
1. 按照文法语法分析各个句型

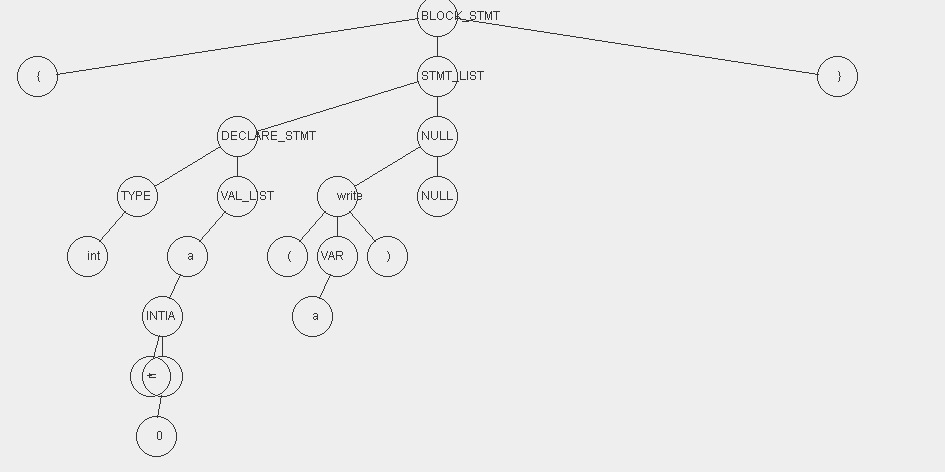
break；



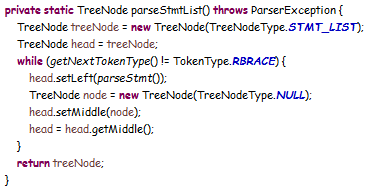


代码块

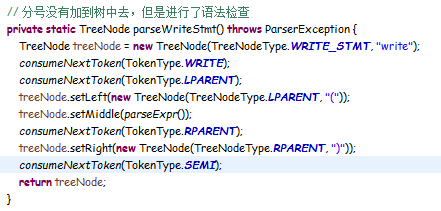


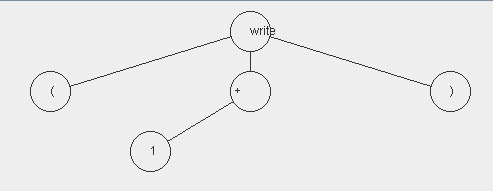


代码块里的多行语句

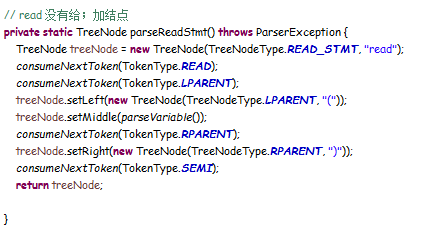


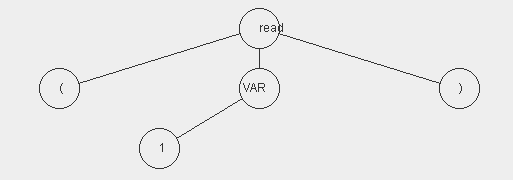
Write语句



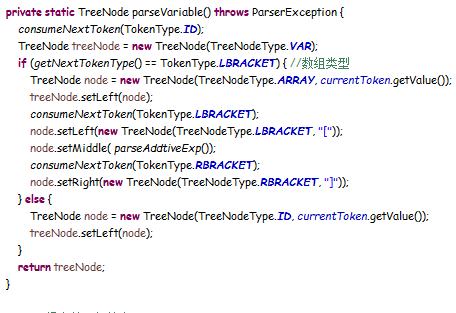


Read语句

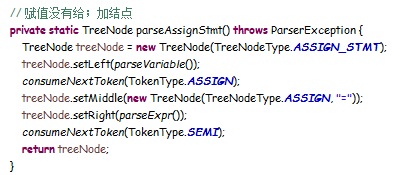


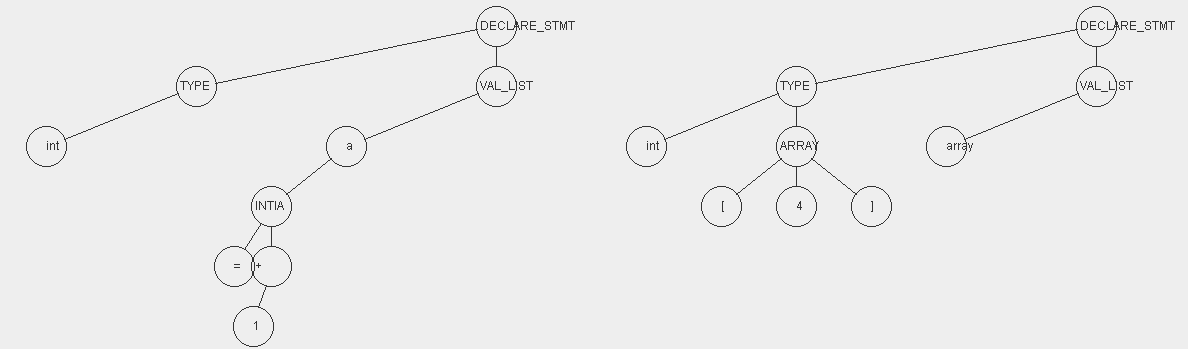


变量名

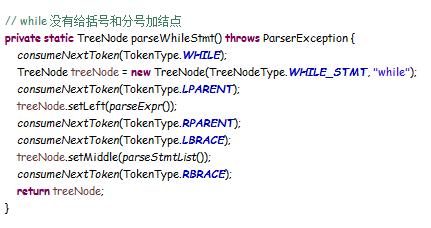


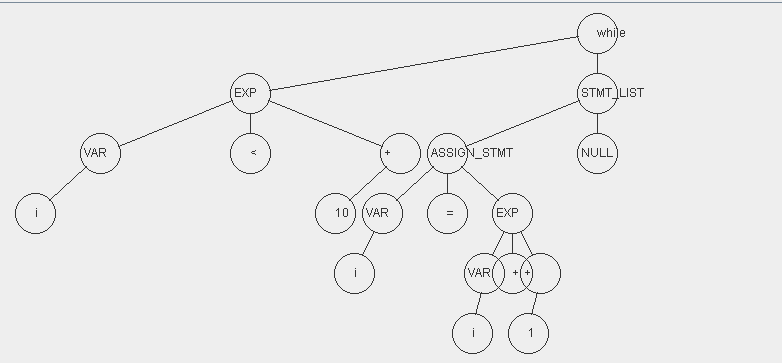
赋值语句





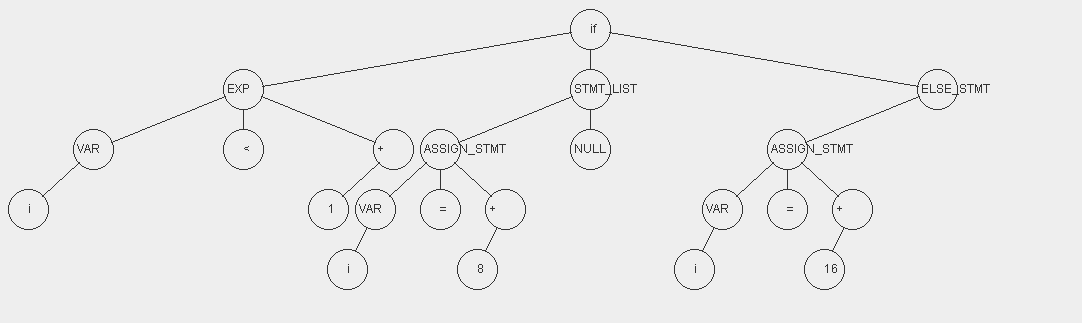
While语句





if语句





关于表达式部分的语法分析取自上学期做过的计算器部分，在此不做展示。

辅助函数



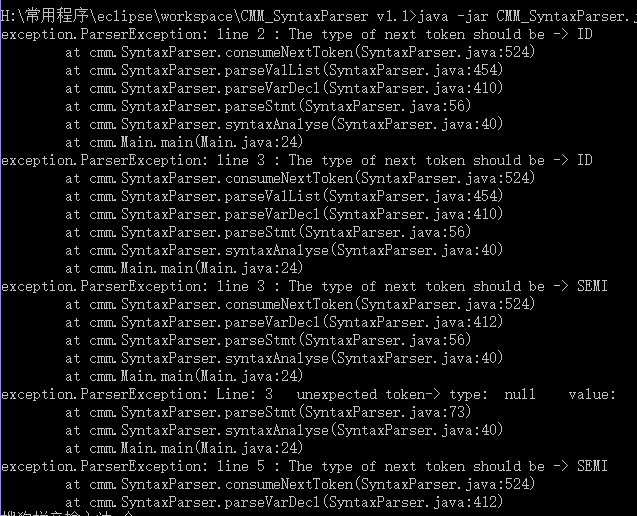
**（五） 运行分析**

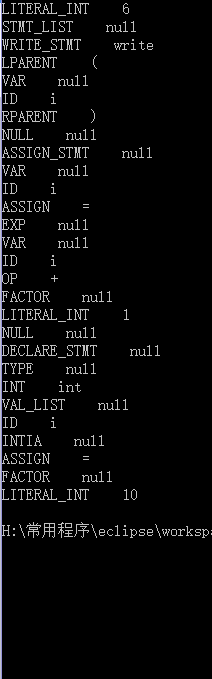
将程序导出jar包 ：CMM\_SyntaxParser.jar，命令行参数有一个，是位于与src同级的test文件夹下的文件路径。

命令行键入命令 java -jar CMM\_SyntaxParser.jar test/error1\_ID.cmm

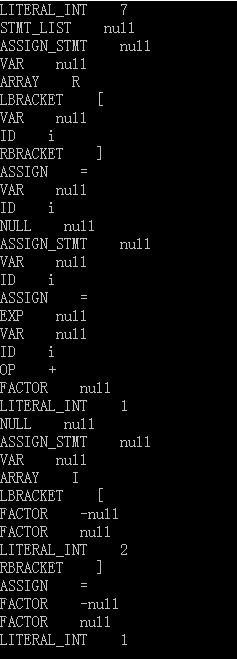
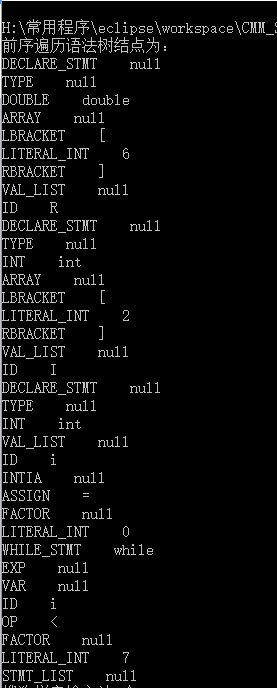
程序执行后，会打印抛出的异常及语法分析结果，表现为打印语法树结点。以下为测试用例的运行结果：

（1）.error1\_ID.cmm

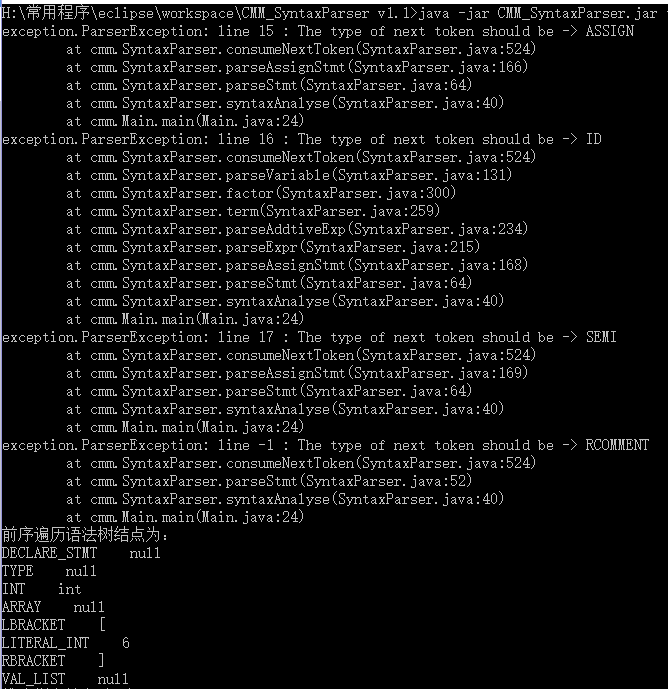


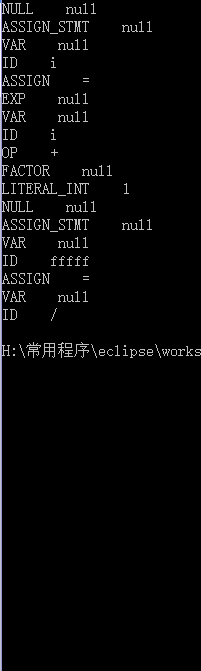
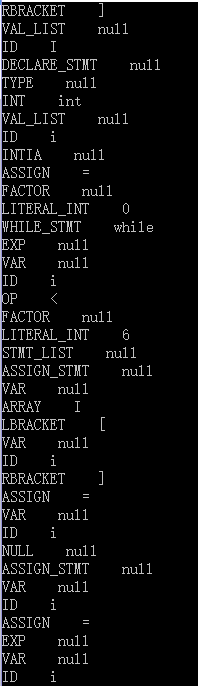


1. erroe2\_array.cmm

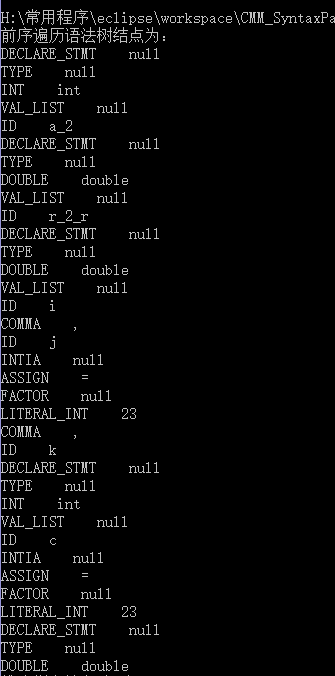


(3) error3\_comment.cmm

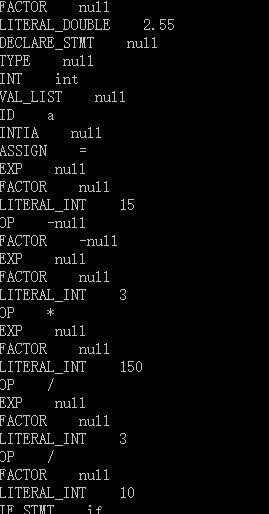
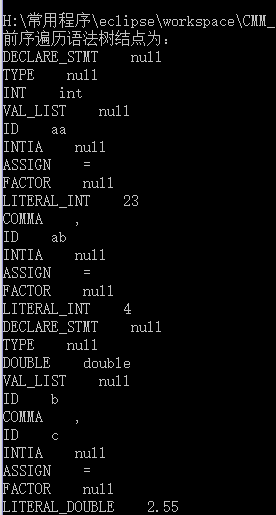




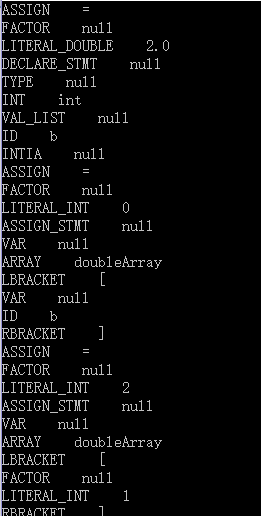
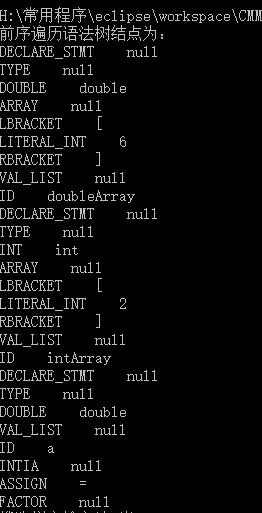
1. test1\_变量声明.cmm（太长，截取部分）



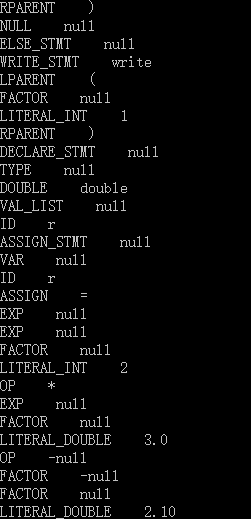
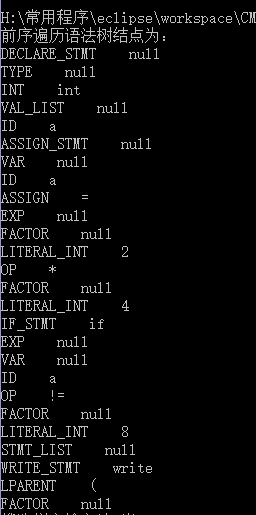
1. Test2\_一般变量赋值.cmm（太长，截取部分）



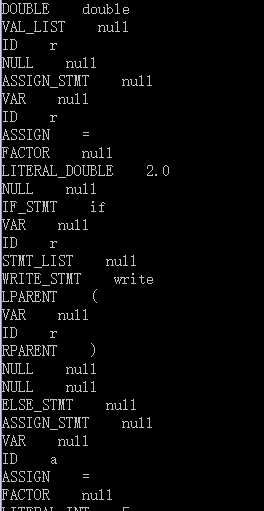
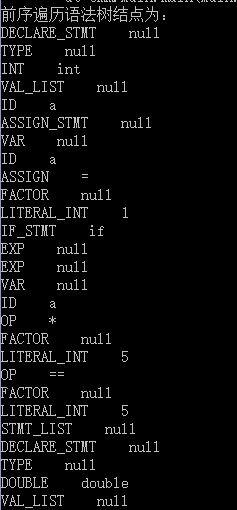
1. Test3\_数组.cmm（太长，截取部分）



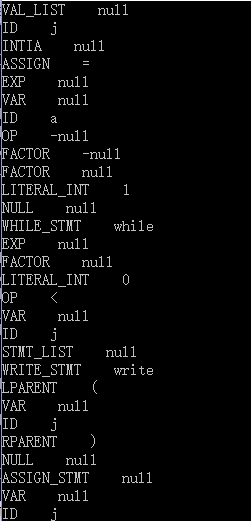
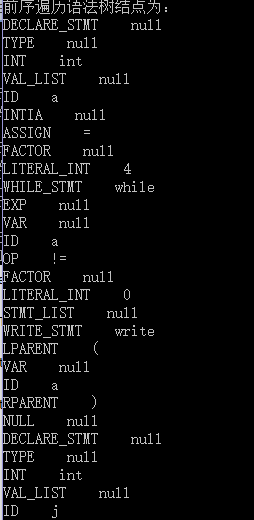
1. Test4\_算术运算（太长，截取部分）



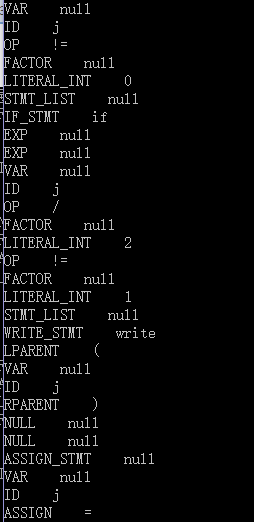
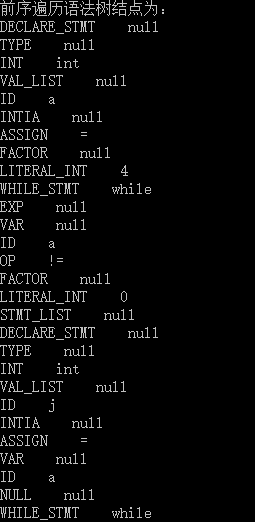
1. test5\_IF\_ELSE.cmm（太长，截取部分）



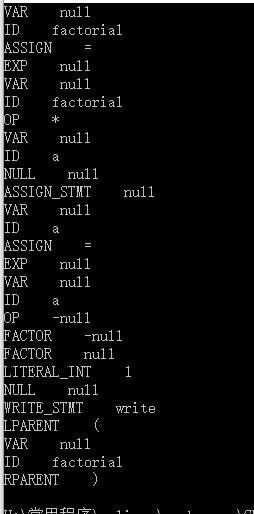
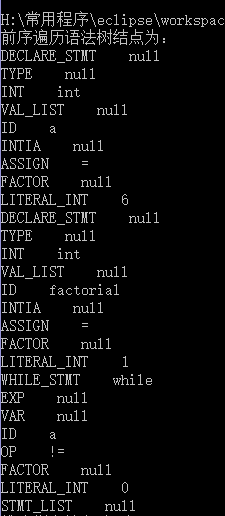
1. Test6\_WHILE（太长，截取部分）



1. Test7\_IF\_ELSE与WHILE（截取部分）



（11）test8\_阶乘.cmm



（12）test9\_数组排序.cmm

