**武汉大学国际软件学院**

**实验报告**

**课程名称 解释器构造实践 授课教师 李莉**

**专 业 软件工程 年 级 2014级**

**姓 名**

**学 号**

**协 作 者 无**

**实验学期 2016 — 2017 学年第 一 学期**

**课堂时数 课外时数**

**填写时间 2016 年 11 月 6 日**

|  |
| --- |
| **【实验项目名称】：**  **CMM 语言语法分析** |
| **【实验环境】（使用的软件）：**  IDEA2016.1.3  JDK1.8  Antlr4.5.3  **【参考资料】**  Interpreter Construction Specification-2016  Pragmatic.The Definitive ANTLR 4 Reference.2013  <http://www.antlr.org/api/Java/org/antlr/v4/runtime/tree/ParseTree.html>  《解释器构造任务书-2016》附录  《Pragmatic.The Definitive ANTLR 4 Reference.2013》 |
| **【实验方案设计】：**  **一、编译方法说明**  ** 1.1 给出 CMM 语言的文法。根据所采用的实现语法分析的方法，对文件进行相应的修改。**   1. CMM语言的文法   我使用了作业要求附录中所给出的语法规则，并根据需求的测试文件进行了修改，即：   1. 所能识别出的Tokens:  1. CMM语言具体描述：   (1)语言结构：顺序结构（赋值语句、输入、输出）、选择语句（if-else）、循环结构（while）、循环结构（for）。这些语句结构和C语言的结构一样，允许嵌套。  (2)表达式局限于关系表达式和算术表达式，运算的优先级为：算术运算、关系运算，并服从左结合规则。  算术表达式包括整数和实数上的运算、变量以及“ （） ”、“ \* ”、“ + ”、“ - ”、“ / ”，运算符的优先级顺序为：“ （） ”大于“ \* ”和“ / ”大于“ + ”和“ - ”。  关系运算符包括：“ < ”、“ > ”、“ == ”、“ != ”、“ >= ”、" <= "。  (3)一条语句以“ ; ”结束；程序由一条语句或者由“ { ”和“ } ”嵌套表达的复合语句。  (4)支持多行注释（使用“ /\* ”和“ \*/ ”）  (5)支持数组运算，数组的下标必须是正整数，使用“ [ ”和“ ] ”表示数组下标。  (6)变量的使用之前需要先声明，声明的方式和C语言一样。  **具体文法如下：**  **grammar** CMMPARSER;  *//PROGRAM* prog : stmt\* ;  stmt : varDecl | ifStmt | whileStmt | breakStmt | assignStmt |  readStmt | writeStmt | stmtBlock ; stmtBlock : LEFT\_BRACE stmt\* RIGHT\_BRACE ;  *//DECLARE STMT* varDecl : type varList SEMICOLONE ; type : INT | DOUBLE | type LEFT\_ARRAY\_BRACE expr RIGHT\_ARRAY\_BRACE; varList : ( ID | varAssign | varArray) (COMMA (ID | varAssign | varArray))\* ; varArray : ID LEFT\_ARRAY\_BRACE expr RIGHT\_ARRAY\_BRACE ; varAssign : ( ID | ( ID LEFT\_ARRAY\_BRACE expr RIGHT\_ARRAY\_BRACE ) )  ASSIGNMENT  expr  ;  *//FLOW CONTROL STMT* ifStmt : IF expr stmt ( ELSE stmt )? ; whileStmt : WHILE expr stmt ; breakStmt : BREAK SEMICOLONE ; readStmt : READ LEFT\_LITTLE\_BRACE  ( ID | ( ID LEFT\_ARRAY\_BRACE expr RIGHT\_ARRAY\_BRACE ) )  RIGHT\_LITTLE\_BRACE SEMICOLONE ; writeStmt : WRITE LEFT\_LITTLE\_BRACE expr RIGHT\_LITTLE\_BRACE SEMICOLONE ;  *//ASSIGN STMT* assignStmt : value ASSIGNMENT expr SEMICOLONE ; value : ( ID LEFT\_ARRAY\_BRACE expr RIGHT\_ARRAY\_BRACE ) | ID ; constant : INTCONSTANT | DOUBLECONSTANT | TRUE | FALSE | HEXCONSTANT | HEXFLOATCONSTANT ;  *//BOTTOM EXPR* expr : MINUS expr  | expr ( MULTIPLICATION | DIVISION ) expr  | expr MOD expr  | expr ( PLUS | MINUS ) expr  | expr ( MORETHAN | MORE\_EQUAL | LESSTHAN | LESS\_EQUAL ) expr  | expr ( EQUAL | UNEQUAL ) expr  | ID  | ID LEFT\_ARRAY\_BRACE expr RIGHT\_ARRAY\_BRACE  | constant  | LEFT\_LITTLE\_BRACE expr RIGHT\_LITTLE\_BRACE  ;  *//RESERVE WORD* IF : **'if'**; ELSE : **'else'**; WHILE : **'while'**; READ : **'read'**; WRITE : **'write'**; INT : **'int'**; DOUBLE : **'double'**; TRUE : **'true'**|**'TRUE'**; FALSE : **'false'**|**'FALSE'**; BREAK : **'break'**;  *//OPERATOR* PLUS : **'+'**; MINUS : **'-'**; MULTIPLICATION : **'\*'**; DIVISION : **'/'**; MOD : **'%'** ; ASSIGNMENT : **'='**; LESSTHAN : **'<'**; LESS\_EQUAL : **'<='**; MORETHAN : **'>'**; MORE\_EQUAL : **'>='**; EQUAL : **'=='**; UNEQUAL : **'!='** | **'<>'**;  *//DELIMITER* LEFT\_LITTLE\_BRACE : **'('**; RIGHT\_LITTLE\_BRACE : **')'**; SEMICOLONE : **';'**; COMMA : **','**; LEFT\_BRACE : **'{'**; RIGHT\_BRACE : **'}'**;  LEFT\_ARRAY\_BRACE : **'['**; RIGHT\_ARRAY\_BRACE : **']'**;  *//IDENTIFIER* ID : LETTER|(LETTER(LETTER|DecDigit|**'\_'**)\*(LETTER|DecDigit)) ;  WS : [ \t\n\r]+ -> skip ; LINE\_COMMENT : **'//'** ~[\r\n]\* (**'\r'**? **'\n'**|EOF) -> skip ; COMMENT : **'/\*'** .\*? **'\*/'** -> skip ;  *//CONSTANT* **fragment** LETTER  : [A-Z]  | [a-z]  ; INTCONSTANT : (**'0'** | [1-9](DecDigit)\*) ; HEXCONSTANT : **'0'** (**'x'**|**'X'**) HexDigit+ ; **fragment** DecDigit : [0-9]; **fragment** HexDigit : ([0-9]|[a-f]|[A-F]) ;  DOUBLECONSTANT  : INTCONSTANT **'.'** (DecDigit)\*  | (**'0'**..**'9'**)+ **'.'** ;  HEXFLOATCONSTANT  : HEXCONSTANT **'.'** HexDigit\*  ;   **1.2说明使用的语法分析方法，如递归下降法、预测分析表法或 LR 方法，给出所采用方法的一些分析过程。**   1. 本次语法分析采用递归向下分析法，根据上面的CMM语言的BNF范式编写递归调用的过程，总共有9个递归调用过程，通过它们之间的调用完成CMM代码的语法分析过程，最后生成一个LISP形式的语法分析树。 2. 预测分析表：  |  |  | | --- | --- | | 1 | Prog:=Statement(Statement)\* | | 2 | Statement:=Assignstate | | 3 | Statement:=Ifstate | | 4 | Statemen:=Whilestate | | 5 | Statement:=Forstate | | 6 | Statement:=Readstate | | 7 | Statement:=Writestate | | 8 | Assignstate:=Type Assignstatement | | 9 | Assignstate:=Assignstatement | | 10 | Assignstatement:=para B | | 11 | B:=ε | | 12 | B:="="resultstate";" | | 13 | Resultstate:=Expressionstate"; | | 14 | Resultstate:=read"("Expressionstate")" | | 15 | Expressionstate:=Expressionstatemen (Operator ExpressionStatement)\*";" | | 16 | Expressionstatement:=factor | | 17 | Expressionstatement:="("Expressionstate")" | | 18 | Operator:="+" | | 19 | Operator:="-" | | 20 | Operator:="\*" | | 21 | Operator:="/" | | 22 | Ifstate:=if"("Conditionstate")""{"Statement"}"A | | 23 | A:=ε | | 24 | A:=else"{"statement"}" | | 25 | Conditionstate:=Factor Relationaloperator Factor | | 26 | Relationaloperator:="<" | | 27 | Relationaloperator:="<=" | | 28 | Relationaloperator:=">" | | 29 | Relationaloperator:=">=" | | 30 | Relationaloperator:="==" | | 31 | Relationaloperator:="!=" | | 32 | Whilestate:=while"("Conditionstate")""{"Statement"}" | | 33 | Forstate:=for"("Assignstate Conditionstate Assignstate")""{"Statement"}" | | 34 | Readstate:=read"("Expressionstate")"";" | | 35 | Writestate:=write"("expression")"";" | | 36 | Factor:=constant | | 37 | Factor:=para | | 38 | Type:=int | | 39 | Type:=double |  1. **程序结构说明，包括程序的总体结构，以及主要的数据结构、算法说明。**   (1)程序的结构图    (2)设计及实现说明  ANTLR生成的解析器叫做递归下降解析器（recursive-descent parser），属于自顶向下解析器（top-down parser）的一种。根据CMM语言的文法规则的BNF范式编写递归调用的代码，parser()方法是程序的入口，它调用的stmt()完成整个语法分析的工作，stmt的背后是9个互相调用的方法。  每个方法在最后会返回一个构造好的节点\_localctx，节点t给调用它的方法，然后调用它的方法用这个返回的节点继续构造，最后语法分析完成后会返回一棵完全构造好了的树节点，然后用tree.toStringTree（parser）将解析树按照 LISP 风格打印出来。   1. **程序调试，设计充分的测试数据，调试的数据及结果。**   （1）首先要对文法进行调试，我用的测试例子是老师给的测试脚本，并结合自己的文法做了相应的修改：  **测试1**：test1.cmm:  /\*\*  \* 测试 变量声明  \* 2010-11-13  \* 邱忠磊  \*  \* 测试变量声明是否合法 主要是：  \* 1.在相同作用域内不允许出现同名变量  \* 2.不同作用域内隐藏上级同名变量  \*/  //测试标识符是否合法  int a\_2;  double r\_2\_r;  double i,j=23,k;  //测试不同作用域内同名变量隐藏  int c = 23;  double b[2];  b[0] = 23;  b[1] = 0;  if(b[1] == 0)  {  double c = 23.5;  if(c == 23.5)  write(1);  else  write(0);  double b[1];  b[0] = 0.05;  if(b[0] == 0.05)  write(1);  else  write(0);  }  /\*output:  1  1  \*///:~**测试结果：（图片太长就分解了）**  (prog (stmt (varDecl (type int) (varList a\_2) ;)) (stmt (varDecl (type double) (varList r\_2\_r) ;)) (stmt (varDecl (type double) (varList i , (varAssign j = 23) , k) ;)) (stmt (varDecl (type int) (varList (varAssign c = 23)) ;)) (stmt (varDecl (type double) (varList (varArray b [ 2 ])) ;)) (stmt (assignStmt (value b [ 0 ]) = (expr (constant 23)) ;)) (stmt (assignStmt (value b [ 1 ]) = (expr (constant 0)) ;)) (stmt (ifStmt if (expr ( (expr (expr b [ 1 ]) == (expr (constant 0))) )) (stmt (stmtBlock { (stmt (varDecl (type double) (varList (varAssign c = 23.5)) ;)) (stmt (ifStmt if (expr ( (expr (expr c) == (expr (constant 23.5))) )) (stmt (writeStmt write ( (expr (constant 1)) ) ;)) else (stmt (writeStmt write ( (expr (constant 0)) ) ;)))) (stmt (varDecl (type double) (varList (varArray b [ 1 ])) ;)) (stmt (assignStmt (value b [ 0 ]) = (expr (constant 0.05)) ;)) (stmt (ifStmt if (expr ( (expr (expr b [ 0 ]) == (expr (constant 0.05))) )) (stmt (writeStmt write ( (expr (constant 1)) ) ;)) else (stmt (writeStmt write ( (expr (constant 0)) ) ;)))) })))))  **测试2：test2.cmm:**  /\*\*  \* ���� һ�������ֵ  \* 2010-11-13  \* ������  \*   \* ���� һ�������ֵ ��Ҫ�ǣ�  \* 1.����ʱ��ֵ  \* 2.�������㸳ֵ  \* 3.read��ֵ  \* 4.�������ݣ�����һ��������ֵ������һ������  \* �����漰������ת������  \*/  ///////////////����ʱ��ֵ int aa = 23,ab = (-4); double b ,c =2.55;   /////////////�������㸳ֵ int a = 15 - 3 \* ( 150 / 3 / 10); if(a == 0) {  write(1); } else  write(0);  /////////////read��ֵ double r; read(r); write(r);  //�������� double r2 = 5; int a2 = 3; r2 = a2; if(r2 == 3) {  write(1); } else  write(0);  r2 = a2 ; write(r2); /\*output: 1 233 #������������� 233.0 1 3.0 \*///:~  **测试结果：**  **(prog (stmt (varDecl (type int) (varList (varAssign aa = 23) , (varAssign ab = (expr ( (expr - (expr (constant 4))) )))) ;)) (stmt (varDecl (type double) (varList b , (varAssign c = 2.55)) ;)) (stmt (varDecl (type int) (varList (varAssign a = (expr (expr (constant 15)) - (expr (expr (constant 3)) \* (expr ( (expr (expr (expr (constant 150)) / (expr (constant 3))) / (expr (constant 10))) )))))) ;)) (stmt (ifStmt if (expr ( (expr (expr a) == (expr (constant 0))) )) (stmt (stmtBlock { (stmt (writeStmt write ( (expr (constant 1)) ) ;)) })) else (stmt (writeStmt write ( (expr (constant 0)) ) ;)))) (stmt (varDecl (type double) (varList r) ;)) (stmt (readStmt read ( r ) ;)) (stmt (writeStmt write ( (expr r) ) ;)) (stmt (varDecl (type double) (varList (varAssign r2 = 5)) ;)) (stmt (varDecl (type int) (varList (varAssign a2 = 3)) ;)) (stmt (assignStmt (value r2) = (expr a2) ;)) (stmt (ifStmt if (expr ( (expr (expr r2) == (expr (constant 3))) )) (stmt (stmtBlock { (stmt (writeStmt write ( (expr (constant 1)) ) ;)) })) else (stmt (writeStmt write ( (expr (constant 0)) ) ;)))) (stmt (assignStmt (value r2) = (expr a2) ;)) (stmt (writeStmt write ( (expr r2) ) ;)))**  **测试3：test3.cmm:**  /\*\*  \* �������  \* 2010-11-13  \* ������  \*   \* ������� ��Ҫ�ǣ�  \* 1.�����±�  \* 2.���鸳ֵ  \* �������鸳ֵ����   \* 1)ֱ�Ӹ�ֵ;   \* 2)�������㸳ֵ;   \* 3)������ֵ;   \* 4)read��ֵ  \*/  double realArray[6]; //����ʵ������ int intArray [2]; //������������ double a = 2.0; int b = 0;  //�����鸳ֵ realArray[b] = 2; realArray[1] = a; realArray[2] = intArray[0]; a = 0.9; realArray[3] = a; realArray[4] = 5 - 2 / (4.0 - 3) \* 2.5 + 0.01;    read(realArray[5]);   //��ӡ���� int x = 0; while(x < 6) {  write(realArray[x]);  x = x + 1; }  /\*output: 23.33 #������������� 2.0 2.0 0.0 0.9 0.01 23.33 \*///:~  **测试结果：（局部）**  (prog (stmt (varDecl (type double) (varList (varArray realArray [ 6 ])) ;)) (stmt (varDecl (type int) (varList (varArray intArray [ 2 ])) ;)) (stmt (varDecl (type double) (varList (varAssign a = (expr (constant 2.0)))) ;)) (stmt (varDecl (type int) (varList (varAssign b = (expr (constant 0)))) ;)) (stmt (assignStmt (value realArray [ b ]) = (expr (constant 2)) ;)) (stmt (assignStmt (value realArray [ 1 ]) = (expr a) ;)) (stmt (assignStmt (value realArray [ 2 ]) = (expr intArray [ 0 ]) ;)) (stmt (assignStmt (value a) = (expr (constant 0.9)) ;)) (stmt (assignStmt (value realArray [ 3 ]) = (expr a) ;)) (stmt (assignStmt (value realArray [ 4 ]) = (expr (expr (expr (constant 5)) - (expr (expr (expr (constant 2)) / (expr ( (expr (expr (constant 4.0)) - (expr (constant 3))) ))) \* (expr (constant 2.5)))) + (expr (constant 0.01))) ;)) (stmt (readStmt read ( realArray [ 5 ] ) ;)) (stmt (varDecl (type int) (varList (varAssign x = (expr (constant 0)))) ;)) (stmt (whileStmt while (expr ( (expr (expr x) < (expr (constant 6))) )) (stmt (stmtBlock { (stmt (writeStmt write ( (expr realArray [ x ]) ) ;)) (stmt (assignStmt (value x) = (expr (expr x) + (expr (constant 1))) ;)) })))))  **测试4：test4.cmm:**  /\*\*  \* �����������  \* 2010-11-13  \* ������  \*   \* ����������� ��Ҫ�ǣ�  \* 1.�������ȼ�  \* 2.����ʱ����ת��  \* 3.��������  \* 4.�������  \*/  int a ;  a = 2 \* 4; //a = 8 if( a<> 8)  write(0); else  write(1);  double r ; r = 2 \* (3.0 - 2.10) - 0.9 \* (2.50 / 1.25 ); //r = 0.0  if( r <> 0) {  write(0); } else  write(1);  double b = 4.000001; write(b);  int x = 60 \* 60 \* 24 ; int y = 60 \* 60 ; write(x / y);   /\*output: 1 0 4.000001 24 \*///:~  **结果：**  **(prog (stmt (varDecl (type int) (varList a) ;)) (stmt (assignStmt (value a) = (expr (expr (constant 2)) \* (expr (constant 4))) ;)) (stmt (ifStmt if (expr ( (expr (expr a) <> (expr (constant 8))) )) (stmt (writeStmt write ( (expr (constant 0)) ) ;)) else (stmt (writeStmt write ( (expr (constant 1)) ) ;)))) (stmt (varDecl (type double) (varList r) ;)) (stmt (assignStmt (value r) = (expr (expr (expr (constant 2)) \* (expr ( (expr (expr (constant 3.0)) - (expr (constant 2.10))) ))) - (expr (expr (constant 0.9)) \* (expr ( (expr (expr (constant 2.50)) / (expr (constant 1.25))) )))) ;)) (stmt (ifStmt if (expr ( (expr (expr r) <> (expr (constant 0))) )) (stmt (stmtBlock { (stmt (writeStmt write ( (expr (constant 0)) ) ;)) })) else (stmt (writeStmt write ( (expr (constant 1)) ) ;)))) (stmt (varDecl (type double) (varList (varAssign b = (expr (constant 4.000001)))) ;)) (stmt (writeStmt write ( (expr b) ) ;)) (stmt (varDecl (type int) (varList (varAssign x = (expr (expr (expr (constant 60)) \* (expr (constant 60))) \* (expr (constant 24))))) ;)) (stmt (varDecl (type int) (varList (varAssign y = (expr (expr (constant 60)) \* (expr (constant 60))))) ;)) (stmt (writeStmt write ( (expr (expr x) / (expr y)) ) ;)))**  测试文件5：  test5.cmm:  /\*\*  \* IF-ELSE ����  \* 2010-11-13  \* ������  \*   \* IF-ELSE ���� ��Ҫ�ǣ�  \* 1.�����ж�  \* 2.���ִ��  \* 3.IF-ELSE Ƕ��  \*/  int a; a=1; if(a \* 5 == 5) {  double r ;  r = 2.0;  if(r)  {  write(r);  } } else {  a=5;  write(a); }   //if Ƕ�� int aa = 3; if(aa < 4)  if(2 < aa)  if(aa <> 3)  write(aa);  else  write(aa-2);   /\*output: 2.0 1 \*///:~  **Output:**  (prog (stmt (varDecl (type int) (varList a) ;)) (stmt (assignStmt (value a) = (expr (constant 1)) ;)) (stmt (ifStmt if (expr ( (expr (expr (expr a) \* (expr (constant 5))) == (expr (constant 5))) )) (stmt (stmtBlock { (stmt (varDecl (type double) (varList r) ;)) (stmt (assignStmt (value r) = (expr (constant 2.0)) ;)) (stmt (ifStmt if (expr ( (expr r) )) (stmt (stmtBlock { (stmt (writeStmt write ( (expr r) ) ;)) })))) })) else (stmt (stmtBlock { (stmt (assignStmt (value a) = (expr (constant 5)) ;)) (stmt (writeStmt write ( (expr a) ) ;)) })))) (stmt (varDecl (type int) (varList (varAssign aa = (expr (constant 3)))) ;)) (stmt (ifStmt if (expr ( (expr (expr aa) < (expr (constant 4))) )) (stmt (ifStmt if (expr ( (expr (expr (constant 2)) < (expr aa)) )) (stmt (ifStmt if (expr ( (expr (expr aa) <> (expr (constant 3))) )) (stmt (writeStmt write ( (expr aa) ) ;)) else (stmt (writeStmt write ( (expr (expr aa) - (expr (constant 2))) ) ;)))))))))  **测试文件6：**  Test6.cmm:  /\*\*  \* WHILE ����  \* 2010-11-13  \* ������  \*   \* WHILE ���� ��Ҫ�ǣ�  \* 1.�����ж�  \* 2.���ִ��  \* 3.WHILEǶ��  \*/  int a = 4; while(a <> 0) {  write(a);  int j=a-1;  while(0 < j)  {  write(j);  j=j-1;  }  a=a-1; } /\*output: 4 3 2 1 3 2 1 2 1 1 \*///:~  **Output:**  (prog (stmt (varDecl (type int) (varList (varAssign a = (expr (constant 4)))) ;)) (stmt (whileStmt while (expr ( (expr (expr a) <> (expr (constant 0))) )) (stmt (stmtBlock { (stmt (writeStmt write ( (expr a) ) ;)) (stmt (varDecl (type int) (varList (varAssign j = (expr (expr a) - (expr (constant 1))))) ;)) (stmt (whileStmt while (expr ( (expr (expr (constant 0)) < (expr j)) )) (stmt (stmtBlock { (stmt (writeStmt write ( (expr j) ) ;)) (stmt (assignStmt (value j) = (expr (expr j) - (expr (constant 1))) ;)) })))) (stmt (assignStmt (value a) = (expr (expr a) - (expr (constant 1))) ;)) })))))  **测试文件7：**  Test7.cmm:  /\*\*  \* IF-ELSE �� WHILE ���Ƕ�� ����  \* 2010-11-13  \* ������  \*   \* IF-ELSE �� WHILE ���Ƕ�� ���� ��Ҫ�ǣ�  \* 1.�����ж�  \* 2.���ִ��  \* 3.IF-ELSE �� WHILE ��� Ƕ��  \*/  int a = 4; while(a <> 0) {  int j = a;  while(j <> 0)  {  if(j/2 <> 1)  write(j);  j = j-1;   }  if( a < 2)  {  write(a);  }  else  write(a+3);  a = a -1; } /\*output: 4 1 7 1 6 1 5 1 1 \*///:~  **Output:**  (prog (stmt (varDecl (type int) (varList (varAssign a = (expr (constant 4)))) ;)) (stmt (whileStmt while (expr ( (expr (expr a) <> (expr (constant 0))) )) (stmt (stmtBlock { (stmt (varDecl (type int) (varList (varAssign j = (expr a))) ;)) (stmt (whileStmt while (expr ( (expr (expr j) <> (expr (constant 0))) )) (stmt (stmtBlock { (stmt (ifStmt if (expr ( (expr (expr (expr j) / (expr (constant 2))) <> (expr (constant 1))) )) (stmt (writeStmt write ( (expr j) ) ;)))) (stmt (assignStmt (value j) = (expr (expr j) - (expr (constant 1))) ;)) })))) (stmt (ifStmt if (expr ( (expr (expr a) < (expr (constant 2))) )) (stmt (stmtBlock { (stmt (writeStmt write ( (expr a) ) ;)) })) else (stmt (writeStmt write ( (expr (expr a) + (expr (constant 3))) ) ;)))) (stmt (assignStmt (value a) = (expr (expr a) - (expr (constant 1))) ;)) })))))  **测试文件8：**  Test8.cmm:  /\*\*  \* �׳˲���  \* 2010-11-13  \* ������  \*   \* �׳˲��� ��Ҫ��  \* a = N!,����NΪ����  \*/   int a = 6; int factorial = 1; while( a <> 0 ) {  factorial = factorial \* a;  a = a -1; } write( factorial );  /\*output: factorial : 720 \*///:~  **Output：**  **(prog (stmt (varDecl (type int) (varList (varAssign a = (expr (constant 6)))) ;)) (stmt (varDecl (type int) (varList (varAssign factorial = (expr (constant 1)))) ;)) (stmt (whileStmt while (expr ( (expr (expr a) <> (expr (constant 0))) )) (stmt (stmtBlock { (stmt (assignStmt (value factorial) = (expr (expr factorial) \* (expr a)) ;)) (stmt (assignStmt (value a) = (expr (expr a) - (expr (constant 1))) ;)) })))) (stmt (writeStmt write ( (expr factorial) ) ;)))**  **测试文件9：**  **Test9.cmm**  double R[6] ;  R[1] = -0.99;  R[2] = -1.0;  R[3] = 5;  R[4] = 4.01;  R[5] = 3.0;   int i;  int j;  int swap;  i = 6;  while (i <> 1)  {  swap = 0;  j = 1;  while (j < i-1)  {  //��������Ԫ��  if ( R[j+1] < R[j])  {  swap = 1;  R[0] = R[j];  R[j] = R[j+1];  R[j+1] = R[0];  }  j = j + 1;  }  i = i - 1;  if(swap <> 1)  i = 1;  }   int k = 1;  while(k<6)  {  write(R[k]);  k = k +1;  } /\*output: -1.0 -0.99 3.0 4.01 5.0  \*///:~  **output：**  **(prog (stmt (varDecl (type double) (varList (varArray R [ (expr (constant 6)) ])) ;)) (stmt (assignStmt (value R [ (expr (constant 1)) ]) = (expr - (expr (constant 0.99))) ;)) (stmt (assignStmt (value R [ (expr (constant 2)) ]) = (expr - (expr (constant 1.0))) ;)) (stmt (assignStmt (value R [ (expr (constant 3)) ]) = (expr (constant 5)) ;)) (stmt (assignStmt (value R [ (expr (constant 4)) ]) = (expr (constant 4.01)) ;)) (stmt (assignStmt (value R [ (expr (constant 5)) ]) = (expr (constant 3.0)) ;)) (stmt (varDecl (type int) (varList i) ;)) (stmt (varDecl (type int) (varList j) ;)) (stmt (varDecl (type int) (varList swap) ;)) (stmt (assignStmt (value i) = (expr (constant 6)) ;)) (stmt (whileStmt while (expr ( (expr (expr i) <> (expr (constant 1))) )) (stmt (stmtBlock { (stmt (assignStmt (value swap) = (expr (constant 0)) ;)) (stmt (assignStmt (value j) = (expr (constant 1)) ;)) (stmt (whileStmt while (expr ( (expr (expr j) < (expr (expr i) - (expr (constant 1)))) )) (stmt (stmtBlock { (stmt (ifStmt if (expr ( (expr (expr R [ (expr (expr j) + (expr (constant 1))) ]) < (expr R [ (expr j) ])) )) (stmt (stmtBlock { (stmt (assignStmt (value swap) = (expr (constant 1)) ;)) (stmt (assignStmt (value R [ (expr (constant 0)) ]) = (expr R [ (expr j) ]) ;)) (stmt (assignStmt (value R [ (expr j) ]) = (expr R [ (expr (expr j) + (expr (constant 1))) ]) ;)) (stmt (assignStmt (value R [ (expr (expr j) + (expr (constant 1))) ]) = (expr R [ (expr (constant 0)) ]) ;)) })))) (stmt (assignStmt (value j) = (expr (expr j) + (expr (constant 1))) ;)) })))) (stmt (assignStmt (value i) = (expr (expr i) - (expr (constant 1))) ;)) (stmt (ifStmt if (expr ( (expr (expr swap) <> (expr (constant 1))) )) (stmt (assignStmt (value i) = (expr (constant 1)) ;)))) })))) (stmt (varDecl (type int) (varList (varAssign k = (expr (constant 1)))) ;)) (stmt (whileStmt while (expr ( (expr (expr k) < (expr (constant 6))) )) (stmt (stmtBlock { (stmt (writeStmt write ( (expr R [ (expr k) ]) ) ;)) (stmt (assignStmt (value k) = (expr (expr k) + (expr (constant 1))) ;)) })))))**  **测试10：**  **Error1.cmm:**  /\*\*  \* ��־������ ����  \* 2010-11-13  \* ������  \*   \* ���Ա��������Ƿ�Ϸ� ��Ҫ�ǣ�  \* 1.��ʶ���Ƿ����� �����淶(����ĸ��  \* ���֡��»�����ɣ�ֻ������ĸ��ͷ  \* ���Ҳ������»��߽���)  \* 2.����ͬ�������ڲ��������ͬ������  \*/   //���Ա�ʶ���Ƿ�Ϸ� int \_a; double 3\_a; //���ֿ�ͷ int a\_; //�»��߽�β int a@com; //�Ƿ��ַ�  int i=0; while(i<6) {  write(i);  i=i+1; }  //��ͬ�������ڶ���ͬ������ int i=10;  /\*output ����������Ϣ���ܸ��� \*///:~  **Output:**  line 15:4 token recognition error at: '\_'  line 16:8 token recognition error at: '\_'  line 16:7 extraneous input '3' expecting {'[', ID}  line 17:5 token recognition error at: '\_'  line 18:5 token recognition error at: '@'  line 18:6 extraneous input 'com' expecting {';', ','}  (prog (stmt (varDecl (type int) (varList a) ;)) (stmt (varDecl (type double 3) (varList a) ;)) (stmt (varDecl (type int) (varList a) ;)) (stmt (varDecl (type int) (varList a com) ;)) (stmt (varDecl (type int) (varList (varAssign i = (expr (constant 0)))) ;)) (stmt (whileStmt while (expr ( (expr (expr i) < (expr (constant 6))) )) (stmt (stmtBlock { (stmt (writeStmt write ( (expr i) ) ;)) (stmt (assignStmt (value i) = (expr (expr i) + (expr (constant 1))) ;)) })))) (stmt (varDecl (type int) (varList (varAssign i = (expr (constant 10)))) ;)))  **测试11：**  **Error2.cmm**  /\*\*  \* ���鱨�� ����  \* 2010-11-13  \* ������  \*   \* ���鱨�� ���� ��Ҫ�ǣ�  \* 1.����Խ��  \* 2.�����±�  \*/  double R[6]; //����ʵ������ int I[2]; //������������  int i=0;  //����Խ�� while(i<7) {  R[i] = i;  i=i+1; }  //�±���Ч I[-2] = -1;  /\*output: ����������Ϣ���ܸ��� \*///:~  **Output：**  (prog (stmt (varDecl (type double) (varList (varArray R [ (expr (constant 6)) ])) ;)) (stmt (varDecl (type int) (varList (varArray I [ (expr (constant 2)) ])) ;)) (stmt (varDecl (type int) (varList (varAssign i = (expr (constant 0)))) ;)) (stmt (whileStmt while (expr ( (expr (expr i) < (expr (constant 7))) )) (stmt (stmtBlock { (stmt (assignStmt (value R [ (expr i) ]) = (expr i) ;)) (stmt (assignStmt (value i) = (expr (expr i) + (expr (constant 1))) ;)) })))) (stmt (assignStmt (value I [ (expr - (expr (constant 2))) ]) = (expr - (expr (constant 1))) ;)))  **测试12：**  **Error3.cmm**  /\*\*  \* ע�ͱ��� ����  \* 2010-11-13  \* ������  \*   \* ע�ͱ��� ���� ��Ҫ�ǣ�  \* 1.ע��Ƕ��  \* 2.����ע���޽�β  \*/  int I[6];  int i=0; while(i<6) {  I[i] = i;  i=i+1; }  /\*output: ����������Ϣ���ܸ��� \*///:~  //ע��Ƕ�� /\* ffff /\* mmmm \*/ fffff \*/ //����ע���޽�β /\* ����...//  **Output：**  这是报错的注释在java中的显示，第一个是匹配到最近的\*/，所以之后的文字会报错，第二个是只由/\*开头而缺少结尾的\*/会把后面的所有文字都注释掉。  line 25:25 mismatched input '\*' expecting '='  line 27:3 token recognition error at: '�'  line 27:4 token recognition error at: '�'  line 27:5 token recognition error at: '�'  line 27:6 token recognition error at: '�'  line 27:7 token recognition error at: '.'  line 27:8 token recognition error at: '.'  line 27:9 token recognition error at: '.'  (prog (stmt (varDecl (type int) (varList (varArray I [ (expr (constant 6)) ])) ;)) (stmt (varDecl (type int) (varList (varAssign i = (expr (constant 0)))) ;)) (stmt (whileStmt while (expr ( (expr (expr i) < (expr (constant 6))) )) (stmt (stmtBlock { (stmt (assignStmt (value I [ (expr i) ]) = (expr i) ;)) (stmt (assignStmt (value i) = (expr (expr i) + (expr (constant 1))) ;)) })))) (stmt (assignStmt (value fffff) \* / / \*)))  **【结论】（结果）：**  结论见上。源码及工程附在压缩文件里。  这里毕竟只是语法分析的测试，所以涉及语义分析的部分还没有进行编码，所以涉及一些数组之类的如error2.cmm中这样本该报错的语义错误，因为只涉及到语法分析，所以并没有报错，请自觉忽略所有和语义分析有关的部分。  谢谢！ |
| **【小结】：**  通过这次试验，在实验一的基础上，加强了动手能力，以及对IDEA的使用。我在编写文法上有了更深的体会。有时候以为文法写起来会比较简单，其实有很多疏忽和错误。此外，更加深入的理解了词法语法分析的过程，学到了很多。还有就是，对程序中错误的调试掌握水平又上升了一个台阶。 |
| **指导教师评语及成绩** |
| **【评语】：**        **成绩：**           **指导教师签名：**  **批阅日期：** |