《编译原理》

第一阶段实验（A）

实验报告

林梓楠 无 37 2013011217

2015.10.24

1 完成内容总述

1、 在 Lexer.l 中加入了 numinstances、fi、do、od 的处理函数，调用 keyword。加入了++、--、|||的处理函数，调用 operator。在 SIMPLER_OPERATOR 宏中加入了？和：。
2、 在 Parser.y 中声明了 SELF_INC、SELF_DEC、NUMINSTANCES、FI、CASESEP、DO、OD、?、:等 token。规定了 SELF_INC、SELF_DEC 的优先级。对 Stmt、Expr 的产生式进行了修改，增加了非终结符 GuardedIfStmt、GuardedDoStmt、GuardedStmts 及其文法。
3、 在 Tree.java 中，增加了类型标识 GUARDEDIF、GUARDEDDO、GUARDEDSTMTS、COND，增加 GuardedBlock、GuardedStmts、Ternary 类。修改了类 Visitor、Unary。
4、 在 SemValue.java 中，增加了关键字 numinstances、fi、do、od 的输出提示，增加了符号++、--、|||的输出提示。

2 实现原理

1、 ++，--：定义了++和—运算符，在 Expr 的产生式中增加了语句（**这部分实现还有一点问题，详见后文**）。对 Tree 中的单元运算符的类 Unary 作了相应的扩展来处理这两个运算符。
2、 三元运算符?:：定义了?和:运算符，在 Expr 的产生式中增加了语句。新实现了 Ternary 类用于处理这个运算符，它继承至 Expr 类。
3、 numinstances：定义了关键词 numinstances，在 Expr 的产生式中增加了语句。新实现了 NumInstances 来处理该运算，它继承至 Expr 类。
4、 串行条件和串行循环卫士语句：由于这两个语句非常的类似，将它们融合在一起实现了。定义了关键词 if、fi、do、od。定义 GuardedIfStmt 用于识别串行卫士语句，定义 GuardedDoStmt 用于识别串行循环卫士语句。它们均是对应的关键词中间套上多个卫士语句串构成的。GuardedStmts 用于识别卫士语句串。在 Tree 中定义了 GuardedBlock 和 GuardedStmts 类，分别对应串行卫士语句和卫士语句串。

3 调试过程

因为我是来自电子系的，做实验时没有人可以交流，遇到了一些棘手的问题花了不少时

间。

做第一题时，原本想采用 LValue SELF_INC 的形式来完成，但是会出现语法冲突，最后结果也不对。调试了半天也没有解决，花了大量时间。最后使用 Expr SELF_INC 的形式。虽然这样能通过数据了，但是其实是不对的，比如 1++这样错误的表达式也能通过。做后面几题倒是蛮顺利的。

在 DDL 快到时，和姚班的一个同学交流，他说我遇到的问题是因为老师提供的 LValue 有错。他发给我了助教提供的新的版本。我换上之后，原本能通过的数据有好几个无法通过了。于是我又开始调试。不知不觉就错过了 DDL。。后来发现，按照新版的 LValue，根本就无法识别出 this.func()这样的语句，因为“this”只能被 Expr 识别，新版的 LValue 并不能识别到“this”。过了几个小时姚班的同学告诉我新的版本在 Call 时确实有问题，于是我又改回了原来的版本。。这时 DDL 已经过去了 1 个小时了，哭啊。。

注：代码仓库地址 https://github.com/fjxmlzn/decaf_compiler