# Decaf PA1A 实验报告

##### 一、新加入的数据结构与函数

新加了IfBranch的ilist，是属于List<Tree>的数据结构，其实ilist直接复用slist就可以，但是为了语义上的清晰，单独将ilist列出。

新加入的类包括Scopy、Guard、ArrayConstant、ArrayRepeat、ArrayConcat、SubArray、DynamicAccess、PythonArr、Foreach。这些类分别对应文档中要求的新功能。每个类之内都有一个或多个构造函数，这些构造函数分别对应一个关键字内的不同语法树的不同构造方式，内部用boolean变量分辨，在printTo函数内调用不同的结构即可实现语法树的构建。

##### 二、实现流程

首先在Lexer.l文件中定义新加入的关键字和操作符，作为新的keyword和operator返回，这些新定义的操作符都是Parser类的成员，所以需要在Parser.y中为这些关键字和操作符定义新的token；同时对于一些需要定义优先级才能正常工作的操作符而言，需要在token下方的优先级序列中定义新加入操作符的结合性和优先级次序。靠上方定义的是优先级较低的，越向下优先级越高。

定义好相关的关键字之后，就可以按照readme文档中提示的参考语法在Parser.y中用类似正则表达式定义相应的语法。主要包含了属于Tree类型的stmt、String类型的identifier、Expr类型的expr、List<Tree>和List<Expr>等，这些相关的数据结构需要在Tree.java文件当中定义相关的类来实现。printTo函数可以根据不同的构造函数来打印相关结构的语法树。

##### 三、程序的工作逻辑

Lexer.l文件是JFlex文件的输入脚本，定义了识别关键字、操作符、常数、标识符等的规则。编译成功之后生成Lexer.java文件，Lexer.java需要调用Parser.y编译生成的Parser.java文件来构建关键字等等。

Parser.java文件又需要调用SemValue文件中的new SemValue函数新建语义值、slist等来新建一个关于statement的list……这些语义值都新建成功之后，调用Tree.java的构造函数构造语法树，经由printTo函数即可打印正确的语法树。

##### 四、作业中遇到的问题

在作业当中ArrayConstant（定义一个常数数组）一直报Syntax Error。每次都卡在了[1,2,3]的第一个逗号或者[2]的后括号处，一开始怀疑是Parser.y的语法结构写错了，但是和几个同学讨论之后并没有发现问题。

之后选择将语法结构重新换了一种写法，将逗号从语法的末尾移动到了中间，就正确的通过了。下面列出了语法结构的转换：

转换后正确的代码：

1. ArrayConstant   :   '['  ']'
2. {
3. $$.expr = **new** Tree.ArrayConstant($1.loc);
4. }
5. |   '[' ConstantList ']'
6. {
7. $$.expr = **new** Tree.ArrayConstant($1.loc, $2.elist);
8. }
9. ;
11. ConstantList    :   ConstantList ',' Constant
12. {
13. $$.elist.add($3.expr);
14. }
15. |   Constant
16. {
17. $$ = **new** SemValue();
18. $$.elist = **new** ArrayList<Expr>();
19. $$.elist.add($1.expr);
20. }
21. ;

转换前报告语法错误的代码结构：

1. ArrayConstant   :   '['  ']'
2. {
3. // omitted
4. }
5. |   '[' ConstantList Constant ']'
6. {
7. // omitted
8. }
9. ;
11. ConstantList    :   ConstantList Constant ','
12. {
13. // omitted
14. }
15. |   /\* empty \*/
16. {
17. // omitted
18. }
19. ;

虽然bug调出来了，但是并不清楚出现了什么问题，需要进一步多花时间研究一下。