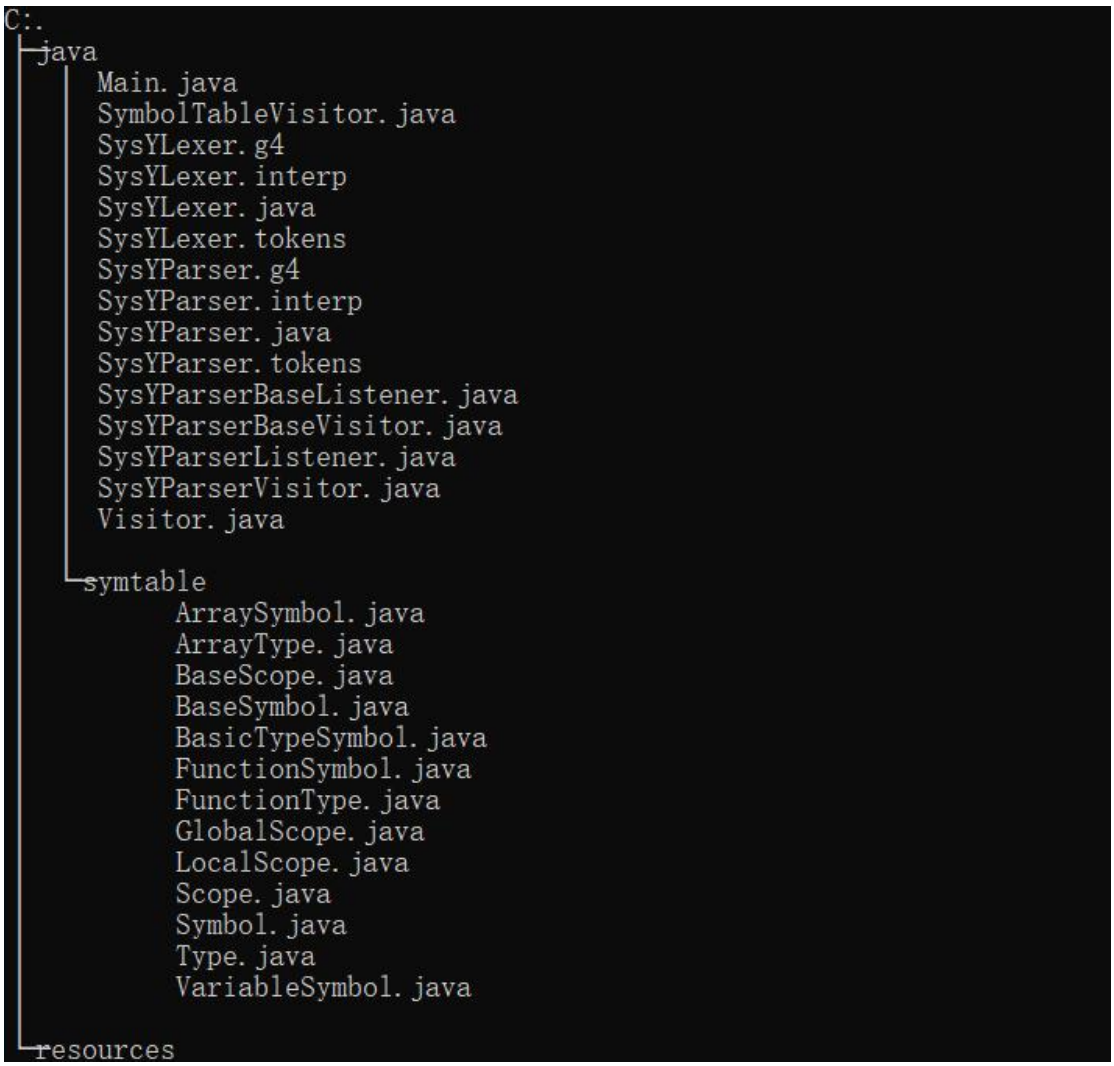
编译原理 lab4

一、作业结构



Symtable 包中类的实现参考老师上课所讲代码。

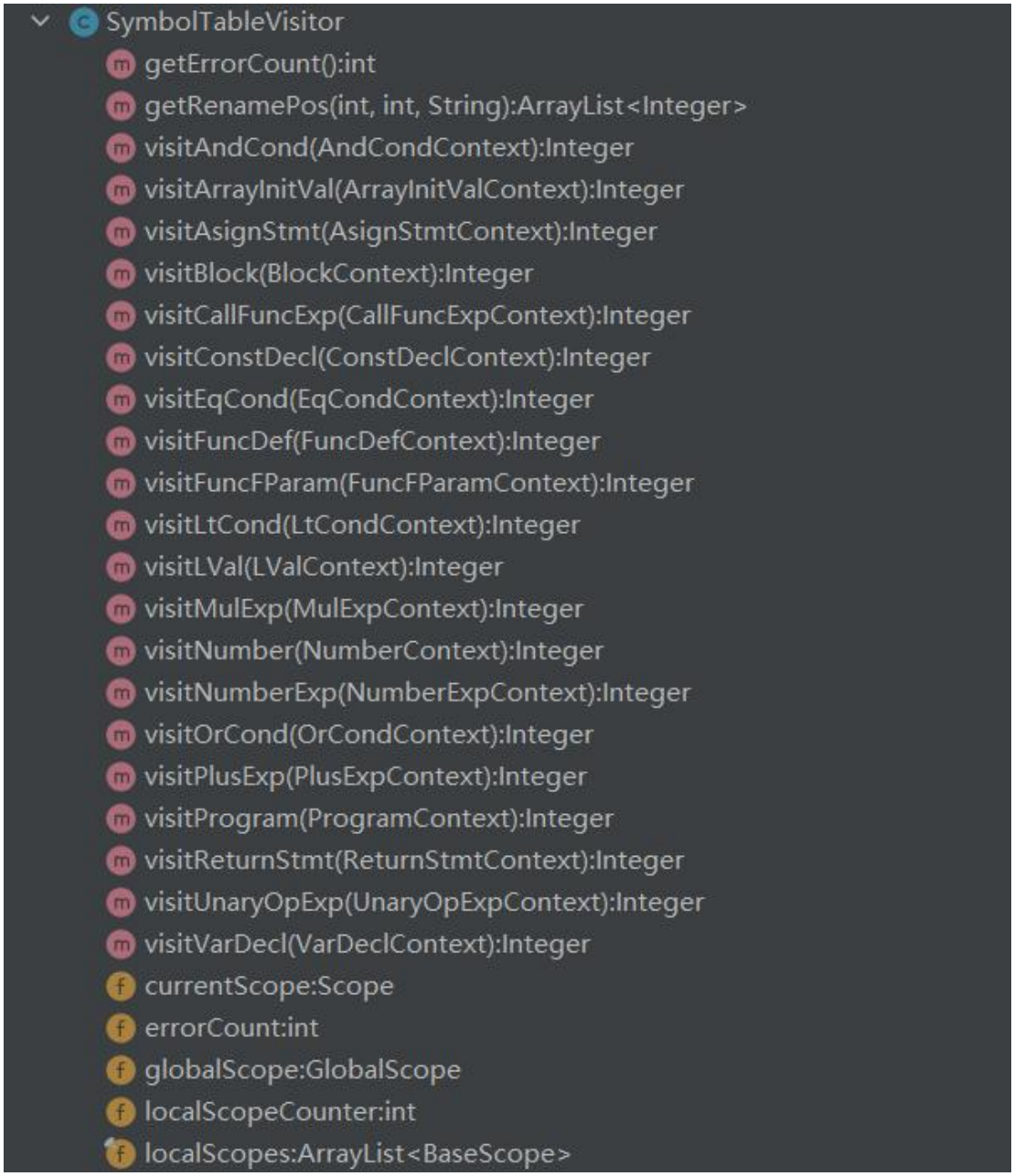
主目录下主要文件有 Visitor.java，SymbolTableVisitor.Java，SysYLexer.g4，SysYParser.g4,以及 mian.java

Visitor.java 是在 lab2 中遍历语法树并打印的方法的基础上改进的，新增了重命名的相关方法。
SymbolTableVisitor.Java 是遍历语法树并生成符号表，顺便进行类型检查，获取重命名变量相关信息的类。
Main 函数以及两个.g4 文件则不过多介绍。

二、实验主要思想：
首先参考老师上课所讲内容构建符号表以及作用域相关的类放在 symtable 包中，其中为了方便之后变量重命名，在 BaseSymbol 类中多加了一个 ArrayList<Integer> pos 来记录符号所出现的位置（行、列号）。
之后则首先按照老师录播所说使用 listener 来构建符号表，但在函数定义错误需要跳过其内

容时，我发现使用 listener 不太好处理，而 lab2 中我已对 visitor 有些理解。故转而使用 visitor 进行遍历构建符号表，以及类型检查。
在经历了不断的调整以及 debug 后（此处省去一万字），终于过了 oj，之后则开始写重命名。
首先我先在 main 函数中遍历 tokens 得到所对应应重命名的变量的符号。
之后我便使用 SymbolTableVisitor 类中新定义的 getRenamePos（）函数，遍历作用域得到需要重命名的变量的所有位置，最后再将该位置通过 main 函数传到 Visitor 类中，在遍历时对所有 Terminal 节点都进行判断，检查其位置是否在应重命名的位置上，若是则改变其名字。

三、具体实现：
SymbolTableVisitor 继承自 SysYParserBaseVisitor<Integer>,返回值设为 Integer，重载了以下方法。绝大多数方法返回值代表其数据结构的维度，比如 0 为常数，1 为 1 维数组，-1 为函数 Null 为 error。具体细节详见代码。



而 Visitor 则是在 lab2 基础上进行改进，其结构相对简单，只在 lab2 基础上多加了给判断是

否需要重命名的判断函数 needReplace，以及设定所需重命名变量的位置的 setRenamePos 函数。