编译原理 Lab3 实验报告

1 程序功能

实验三实现了对于语法的错误检查和变量重命名，可以对设定的 int、array、function 三种类型进行语法分析和错误检查。

2 程序实现

代码结构：
一个检查错误的方法类 checkVisitor，用于对于语法树的深度遍历访问
基本变量的类型类 Type：包括 int 类型，array 类型，function 类型
创建符号类 Symbol：BaseSymbol 为基础符号类型，functionSymbol 类继承自 BaseScope 同时实现 Symbol 类
符号表类 Scope：基础符号表类 BaseScope，globalScope、functionSymbol 和 localScope 继承自基础符号表类

三个类的关系：Type 中包括了变量类型种类名，Symbol 类中包括了变量名和变量类型，以及是否需要重命名，Scope 为作用域类型，其中包括了作用域名，作用域的返回值以及父作用域

函数重命名时，重命名的判断在第一次遍历树的时候进行判断，需要在 visitor 中记录重命名变量所在的作用域 renameScope，重命名的变量名 rename，并且在 visitor 中再次遍历整个树创建作用域的映射，根据作用域的映射查找实际作用域，最后在遍历终结符节点的时候要将 renameScope 的变量名替换为 rename

代码内容：
首先创建全局作用域，可以在其中定义函数和全局变量，一个作用域指针，指向当前的作用域

分析函数创建语句 funcDef 和 visitFuncFParam，第一个是创建一个函数参数作用域，用于存放函数的参数个数和参数类型、参数名称，这里需要判断参数是否有重定义

其次进入到函数的局部作用域中，这其中包括变量定义和语句两个部分
对于变量定义又包括普通变量定义和常量变量定义。对于普通变量定义，还需要分为有初始值和没有初始值的情况。
对于语句部分，可以分为赋值语句、条件语句和返回语句三个标签进行判断

还有在局部作用域中的局部作用域，这时我们需要对于作用域进行区分，在退出局部作用域时需要移动指针

3 精巧设计

在得到表达式时，为了能够解析嵌套的表达式，需要用一个递归函数对于表达式进行递归，得到最内层的表达式；其次在表达式进行判断的时候，如果是二元表

达式直接对左右两边进行判断会出现重复判断的情况，这时候应该只对右边进行判断，当左边的表达式不含其他二元运算符的时候才对左遍进行判断，这样可以避免重复输出错误。
在打印语法结构树的时候，对于作用域进行了一个简单的映射，仅仅通过作用域名称就可以到第一次检查错误的语法作用域中得到所有符号表进行判断

还将各种错误独立写成函数，这样可以避免大量代码段重复的问题出现，也便于后续 debug

4 印象深的地方

印象深的地方就是得到嵌套关系中表达式的值，通过 exp instanceof XXX 的方式可以递归得到表达式的值。其次便是对于类型、符号、符号表的设计，将数组和函数进行单独的继承，可以避免很多问题