实验二 语义分析

姓名：张子谦　　学号：151220166

院系：计算机科学与技术系　联系方式：zhangziqian1011@126.com

一、完成的功能

1. 检测 19 种语义错误(1-17 为必做-要求 1，18,19 为选做-要求 2.1)；

2. 实现要求 2.2，使变量的定义受可嵌套作用域的影响，外层语句块中定义的变量可在内层语句块中重复定义(但此时在内层语句块中就无法访问到外层语句块的同名变量)，内层语句块中定义的变量到了外层语句块中就会消亡，不同函数体内定义的局部变量可以相互重名；

3. 实现要求 2.3，将结构体间的类型等价机制由名等价改为结构等价；

4. 总体来说完成了所有必做部分和额外要求；

二、实现方法

1. 维护符号表和函数表，从根结点往下遍历一次，对于每个节点分析时找出语义错误；

2. 符号表运用 Imperative Style，每个符号表的节点内除了符号之外还存有哈希表和栈中的前后节点，便于操作：

```
typedef struct SymbolNode {
        Symbol *symbol;
        struct SymbolNode *preHashList, *preStack;
        struct SymbolNode *nextHashList, *nextStack;
} SymbolNode;
```

每个符号所存的信息如下：

```
typedef struct Symbol {
        enum { VAR, STRUCTS, FUNC } kind;
        union {
                Type *type;
                Func *func;
        };
        int depth;
        char *name;
} Symbol;
```

其中 depth 记录作用域深度，Func 为函数所保存的信息，结构如下：

```
typedef struct Func{
        Type *returnType;
        FieldList *arg;
        int isDefined;
} Func;
```

其中 isDefined 变量用来判断函数是否定义过。

3.额外要求实现方式

要求 2.1：在原有的 syntax.y 中添加了函数声明这一产生式，并通过 Func 的 isDefined 判断是否被定义过。

要求 2.2：通过 depth 变量和 CompSt 构成的栈的弹出变量来实现，为了防止函数参数类型在弹出时被删除，用 copyType 来复制形参的 Type。

要求 2.3：只比较 structure 的域即可。

三、运行方式

1. 在 Code 文件夹运行 make 可以在项目目录中产生可执行文件 parser，./parser testfile 可以对 testfile 进行词法语法分析并在屏幕上输出结果。

2. make test 可以测试 testcase/文件夹下面的所有.cmm 文件并输出到对应的.out 文件中。

3. make clean 清除所有 make 生成的文件。