编译原理实验二实验报告

- 李润泽 211275011
- 211275011@smail.nju.edu.cn

程序功能

实现了对c--语言的语义分析，除去基本要求的17个语义错误，完成了选做3.3的内容，即在结构体结构等价的基础上完成对17个语义错误的判断
采用的数据结构为散列表存储变量符号的方法。Hash 函数使用了讲义中提供的P.J.Weinberger提出的hash函数。
相关数据结构定义如下

```
struct Type_
{
Kind kind;
union
{
// 基本类型
BasicType basic;
// 数组类型信息包括元素类型与数组大小构成
struct { pType elem; int size; } array;
// 结构体类型信息是一个链表（变量与定义共享）
pFieldList structure;
//函数信息包括函数名，参数个数，参数名，返回值类型
struct {
int argc;
pFieldList argv;
pType returnType;} function;
} u;
}Type;

struct FieldList_
{
char* name;  // 域的名字
pType type;  // 域的类型
pFieldList tail;  // 下一个域
}FieldList;


typedef struct tableItem {
    pFieldList field;
    pItem nextitem;

} TableItem;

typedef struct hashTable {
   pItem *hashArray;
} HashTable;
```

为区分结构体变量的定义和结构体类型的定义，在Kind枚举中新加入一个类型STRUCTURE_DEF，但其type.u仍用pFieldList structure;

对于匿名结构体，采用了随机生成256位由大小写字母和数字组成的字符串作为其name，极大概率避免了与其他符号的命名冲突；

代码中将结构体看做oop语言中的类，对每个结构体都封装了对应的操作方法，包括基本的new(构造新的结构体) copy(深度拷贝结构体）以及部分结构体特有的方法，比如Type 的 checkType (类型检测)等。通过封装会频繁调用的功能函数来方便后面的语义分析。

语义分析中，当程序碰到符号需要处理时必定是进入了ExtDef后，所以程序遍历语法树，遇到ExtDef节点便从该节点开始进行语义分析，然后对ExtDef 展开会得到的所有非终结符节点进行处理对符号表进行操作以及查找语义错误。

程序运行

使用用提供的Makefile文件编译运行即可