lab2实验报告

151242060 银加

实验环境及编译方法

我使用的是MacOS系统，我在实验的根目录下创建了Makefile文件。在测试时可以在根目录下执行 `make all` 即可完成编译并且生成parser语法分析程序。需要注意的是，与实验指导稍有区别的是在mac上 `-lfl` 需要替换成 `-ll` 才能编译成功，所以在测试时可能需要根据相应操作系统对Makefile进行修改。

进行测试时，可以执行 `make test1` ，... `make test17` , `make testex1` ，... `make testex6` 对测试样例分别进行测试。也可以直接使用 `make testall` 一次性进行所有测试。所有样例均来自实验指导。如果需要添加新的测试样例，可以在根目录下test文件夹中添加新的样例，然后在Makefile中添加相应语句或直接执行即可。

实现功能说明

我的语义分析器实现的功能包括所有的必做内容，加上选做(2.1)和选做(2.3)。

文件结构

在lab1的基础上新增了以下四个文件：

- `symboltable.h` , `symboltable.c` 主要实现了符号表的数据结构，并且定义了符号表上可执行的操作，包括插入、查找、检查等等。
- `semantic.h` , `semantic.c` 主要定义了扫描语法树时扫描到各个类型节点进行的语义分析。

主要功能实现方法

- **符号表**
  我采用的是散列表的方式，如果散列表出现冲突，则在相应数组元素下面挂一个链表。
  我一共维护了两张表，所有变量名组织成一张表 `varTable` （因为我没有实现(2.2)所以没有区分临时变量），所有函数名组织成一张表 `funcTable` 。变量的信息存放在结构体 `FieldList_` 中，函数的信息存放在结构体 `Functype_` 中。
- **类型表示**
  `Type_` 的实现与实验指导中基本一致，不同之处在于对于结构体类型的信息，指导中使用的是 `FieldList structure` ，而我新创建了一种结构体 `Structure_` 来存放结构体类型信息。其中的 `name` 域表示结构体的名称，其中的 `strfield` 域表示结构体的域信息。

重要函数功能说明

- **paramEqual(FieldList f1,FieldList f2)：**
  用于比较两个函数的参数是否一致或者判断两个结构体之间是否结构等价，注意对结构体是将 `Structure_` 的 `strfield` 作为输入参数。比较方式是对链表逐项比较。这一函数可用于实现要求（2.3）。

- **typeEqual(Type t1,Type y2)：**
  用于比较两个类型是否完全等价，对于数组类型需要比较基类型和维数是否相同。对于结构体则调用 `paramEqual` 进行判断。

- **printXXX：**
  以print开头的各函数是为了方便打印错误信息而实现的。

- **insertFunc(Functype f,int type)：**
  此函数的目的是在函数符号表中插入新的函数，其中一个参数为 `type`， `type=0` 表示插入的函数为函数声明， `type=1` 表示插入的函数为函数定义。此函数的返回值有三种情况，返回值为0表示没有错误，返回值为1表示检测到函数已经被定义过，返回值为2表示检测到函数声明之间不一致。这一函数与checkFunc函数一起可用于实现要求（2.1）。

实验总结

本次实验的关键是符号表和类型的表示，其中对于这两者数据结构的实现和操作着实是一项比较复杂的工程，涉及到很多关于链表的操作，需要非常细致才能避免犯错。而对于语义的分析基本是函数之间的相互调用，只要按照产生式一条一条来就好了，不算太复杂。