lab2 语义分析　　121220026 谷雨

一、实现的功能

实现了全部基本功能及要求 2.1（函数声明）,2.2（作用域），没有实现 2.3（2.3 的要求似乎与 C 不一致）。

实现函数声明需要在 ExtDef 的文法中加上一条 Specifier FunDec SEMI 表示声明，同时程序中用于描述函数的类型中需要有一个域用来记录函数是只被声明了还是已经被定义了。

作用域的实现：采用实验讲义 PDF 中的十字链表。每次访问一个 CompSt 时，首先要将作用域层数（在代码中用 scope 表示）加一，在访问一个 CompSt 的末尾需要将该层作用域内的变量全部从变量表中删除然后再将 scope －后离开该 CompSt。值得一提的是，全局作用域层数为 0， 函数的形参作用域层数为 1（如果函数有定义，这些形参就相当于定义在函数主体的局部变量）。

符号表结构：整个符号表结构分为三张符号表，变量表，函数表，结构体表，其中只有变量表有作用域的问题，故只有变量存储时要用到十字链表。符号表均采用哈希表，采用的哈希算法是我在 DJB 哈希算法的基础上进行了一些微调得到的哈希算法。

亮点：至于实现的亮点，本次实验的全部代码都由我独立思考完成，没有对外界做任何的参考，所以很多地方的设计都耗费了一些心血。如果非要说亮点的话，个人感觉我的 *visit_DefList_in_struct* 等几个递归函数的设计还是比较巧妙的。

缺点与不足：缺点倒是很清楚，由于时间比较紧张（碍于前段时间各种杂事，我从四天前才开始着手本次实验，结果还是超过了 ddl 两个多小时），所以有些地方做的比较投篮。比如三张不同符号表的存储类型都为我定义的结构体类型 Item，Item 中的一些项只对变量有用，比如 name（虽然函数与结构体也有 name，但是在他们的 Type 中已经存了一次），scope（只有变量才有作用域），这样是很浪费的，以后有时间我会改为使用泛型来实现。此外，测试的样例并不够多，不能保证程序的健壮性，不过至少可以保证可以通过 PDF 上的全部测试样例。

二、编译方式

编写了 makefile

直接 make 就 OK 了，重新编译的话先 make clean

三、一两句题外话

这次实验完全独立完成的话，收获还是挺大的。不过能从中得到这种收获的人并没有想象的那么多。个别托管平台起到了不该有的消极作用。