编译原理 实验2

161220013 陈金池

1. **简介**

实验2基于实验1完成的c—语言词法、语法分析程序，添加了语义分析功能。该程序考虑了“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”这一要求。

1. **运行方法**

在/Lab/Code文件夹中执行以下指令：

make clean && make

此时已经生成parser可执行文件，可以选择分析测试用例

make test

或分析指定文件

./parser [path-to-cmm-file]

1. **文件结构**

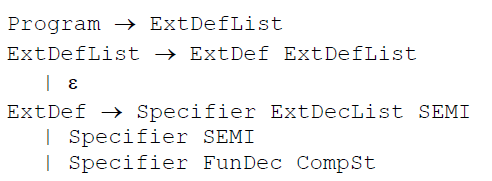
与本次实验相关的文件及其功能如下表所示：

|  |  |
| --- | --- |
| **文件名** | **功能** |
| analyse\_tree [.h/.c] | **语义分析的入口。**   1. 在语法分析完成（即生成Program节点）后，程序会调用该文件下的analyse\_program()函数，开始语义分析。 2. 该文件还包含了许多用于辅助语义分析的函数，如analyse\_exp()等。 |
| SymbolNode [.h/.c] | **定义了符号表节点。**  符号表采用链表的方式构建，该文件中定了链表节点，并提供了查表、改表等操作。 |
| Type [.h/.c] | **定义了类型结构和域结构，并维护一个“已定义结构”的表。**  使用同一种结构体表示基本类型、数组和结构体。域结构使用链表保存。 |
| TreeNode [.h/.c] | **定义了语法树节点。**  语义分析全过程基于语法分析生成的语法树。 |
| util [.h/.c] | **提供了打印错误等工具函数。** |

1. **语义分析简介（对应于analyse\_tree.c文件）**

语法分析生成Program节点后，程序会调用analyse\_program()函数，开始语义分析。该函数的基本流程是：遍历一遍语法树，不断调用各种analyse\_xxx()函数分析各个节点，在该过程中维护保存变量名、函数名的符号表，并将检测到的语义错误输出。

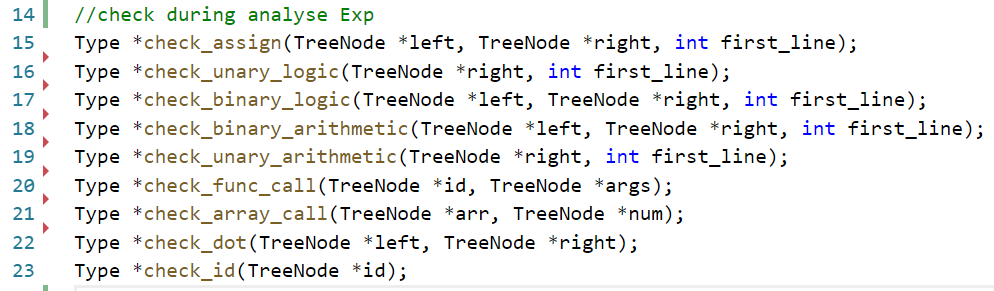
以analyse\_program()函数为例，简述一系列analyse\_xxx()函数的流程。首先，根据语法，Program节点可能有以下的层次结构：



因此，程序首先提取ExtDefList节点，然后用一个while循环遍历ExtDefList的每个子节点，即ExtDef。而对于每个ExtDef节点，一共有三种可能的情况，应该分情况讨论，并在每种情况中调用相应的处理函数，如通过analyse\_fun\_dec()函数完成对FunDec节点的分析。

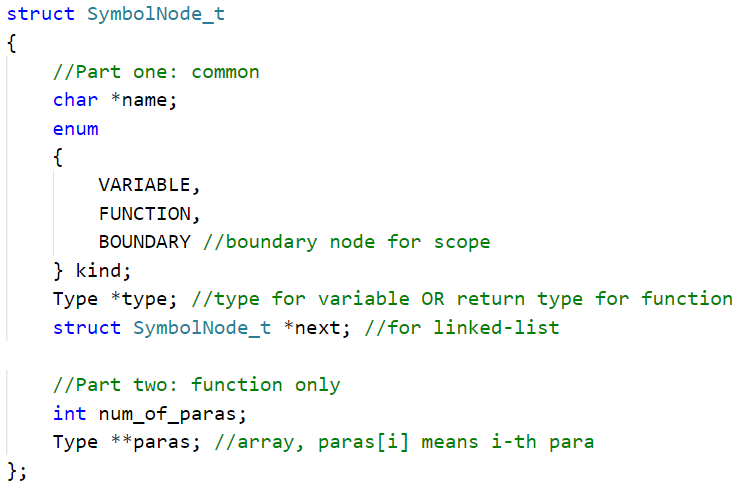
值得一提的是，由于Specifier在分析过程中多次出现，且每个Specifier节点对应了一种类型（如果是数组，还需要考虑Dec中的中括号）。所以，Type.c文件中提供了从Specifier分析得到Type\*类型的函数，即get\_type\_from\_specifier()。

除了一系列analyse函数以外，文件中还提供了一系列check函数，主要用于分析Exp节点过程中的错误检查过程，如下图所示。以check\_id()为例，它检查了表达式中使用的id是否在符号表中，若没有出现则打印一个Error 1.



1. **符号表简介（对应SymbNode.c文件）**

程序使用链表结构来保存符号表，每个节点的结构如图（注释说明了结构中各个域的作用）。值得一提的是图中的BOUNDARY枚举值，该值专门用于表示选做2.2嵌套语句块的边界。每次进入一个语句块之前，会先在符号表中插入一个边界，再进行语句块分析、插入内部变量到符号表等操作。在退出语句块时，会在符号表上搜索边界节点，直到遇到第一个边界停止搜索，并将该过程中的所有节点（即语句块内部的变量定义）删去。



基于上述机制，变量搜索符号表有两种可能的模式：

1. **考虑边界，即只在边界内搜。**如定义变量前，查表以确定是否重复定义。
2. **不考虑边界，即在整个表搜。**如使用变量前，查表以确定变量是否未经定义。
3. **类型系统（对应Type.c文件）**

类型系统使用两个结构体：类型结构体，域结构体。两者均以链表的方式组织。文件中生成了两个Type类型的变量TYPE\_INT和TYPE\_FLOAT，分别表示int和float两种基本类型。在后续分析过程中，保证只使用指向这两个变量的指针表示基本类型。即，系统中只有这两个变量表示基本类型（尽管实际上可以生成多个表示基本类型的Type）。此外，系统维护了一张存放已定义结构体类型的表，并保证每种结构体只使用该表中元素的指针来表示。

数组类型使用了实验教程中的方法，此处不再赘述。

基于上述机制，可以按以下方法使用类型系统：

1. 基本类型、结构类型直接通过“指针相等”表示类型等价。
2. 数组类型先判断维数是否相等，然后判断最低维（即基本类型或结构类型）是否相等即可。