编译原理实验二实验报告

151220055　　李晓霖

一、 我的程序实现了必做的所有内容，并且通过了实验手册里面所有的测试样例。

二、 词法分析和语法分析都是建立在实验一的基础上，因此，完全可以假设对于实验二输入的文本都是没有词法或语法错误的。程序运行开始，依然是先使用实验的 flex 和 bison 工具对输入的文本进行词法分析和语法分析。倘若输入文本没有词法语法错误，则以生成的语法树的根结点开始，向下遍历进行语义分析和类型检查。因此，项目中增加了 semantic.h 和 semantic.c 文件，声明和定义了所需的函数，函数作用如下：

Hash_pjw	计算变量名、结构体、函数在哈希散列表中的位置
Insert	将新增加的变量名、结构体、函数插入到通过 hash_pjw 函数计算所得散列表的位置
Find	通过 hash_pjw 计算得到的位置，在散列表中查找变量名、结构体、函数
Equal	判断两个类型是否等价
Vardec	根据根节点和基本类型，返回该根节点的域表，这个根结点为一个变量
Specifier	根据根结点往下遍历，返回该子树的结构类型，包含结构中的各个域，还有子结构体的各个域

Extdeflist	Extdeflist 函数根据整个程序的语法树进行遍历，通过分析是变量和结构体的定义，还是函数的定义，然后在哈希散列表中查找是否存在已有定义，若有，则重定义，若无，则插入该变量名（或者结构体或者函数）
Compst	Compst 分为定义块和动作块，定义块通过调用 deflist 函数进行分析，动作块通过调用 stmt 函数进行分析
Deflist	根据变量或者结构体的定义语句，在哈希表中查找是否已定义，若有，则重定义，若无，则插入该变量或结构体
Stmt	根据动作语义的不同，进行不同的操作： 若动作为 return，则通过检查 exp 的类型及传递的函数返回类型进行 equal 比较，判断返回类型是否正确； 若动作为另一个 compst 语句块，则交给 compst 函数进行分析； 若动作为一个 exp 语句，则交给 exp 函数进行分析； 若动作为 while 语句块，则首先判断 exp 类型是否为 int，不是 int 则输出错误信息，然后将 stmt 递归调用自己进行分析；

	若动作为 if 且没有配对的 else 语句块，则首先判断 exp 类型是否为 int，不是 int 则输出错误信息，然后将 stmt 递归调用自己进行分析； 若动作为 if 且有配对的 else 语句块，则首先判断 exp 类型是否为 int，不是 int 则输出错误信息，然后将两个 stmt 递归调用自己进行分析；
Exp	根据动作语义的不同，进行不同的操作： 若只有一个 ID，则通过查找哈希表，确定该 ID 是否定义，若无定义，则输出错误信息； 若为 INT 或 FLOAT，则为基本数据类型，返回该数据类型； 若为（exp）、*exp、-exp，则递归调用自己，对 exp 进行分析； 若为+、-、*、/操作，则先递归调用自己，对两个 exp 进行分析，在根据返回的 exp 类型的值，通过 equal 比较，判断类型是否一样，若不一样，则操作数类型不匹配，输出错误信息，若一样，则返回其中任一 exp 的类型； 若为&、|、比较大小运算，则先递归调用自己，对两个 exp 进行分析，在根据返回的 exp 类型的值，通过 equal 比较，判断类型是否一样，若不一样，

	则操作数类型不匹配，输出错误信息，若一样，则返回 INT 类型； 若为赋值操作，则先判断赋值号左边是不是为只有右值的表达式，然后在调用 equal 判断两边的类型是否一样； 若为调用函数，则判断函数是否定义过，并且类型为函数而不是变量或者结构体，以区分错误使用变量和函数未定义两种不同错误，最后根据哈希表中函数的域表和 args 进行比较，若不一样，则函数参数有误； 若为结构体的域，则查找哈希表，判断是否存在该结构体，否则为对“.”的非法使用，若存在该结构体，则根据哈希表中的域表查找该域，查无匹配项则错误； 若为数组，则通过查找查找哈希表，判断是否存在该数组，若不存在，则错误；
Traversetree	通过调用 extdeflist 对整个语法树进行遍历和分析

三、在哈希表中插入符号时，会根据是否为函数进行不同的哈希表计算，这样方便区分在使用 ID LP （Args） RP 时，究竟是错将变量当成函数使用还是函数未定义。