编译原理实验一实验报告

姓名：戴若石 学号：171860004 邮箱：2062499982@qq.com

1. 实验内容

编写一个可以对C--语言源代码进行词法分析和语法分析，构建出一棵语法树的程序。

且完成了全部选做要求。

1.1 词法分析

借助于GUN Flex工具完成，文件lexical.l完成这部分功能。

可以分辨出的词法错误：

- 文法里没有出现的字符
- 以/*开头没有以*/结尾的注释
- 指数形式的浮点型常数，且e之后没有数字

1.2 语法分析

借助于GUN Bison工具完成，文件syntax.y完成这部分功能。

1.2.1 语法分析树的构建与输出

多叉树的结点包含两个指针，left指针指向自己的最左边的子结点，right指针指向自己右边的兄弟结点。

构建过程位于bison代码的规则部分，定义非终结符号的yylval属性类型为树的结点指针，在每个产生式后的语义动作中进行树的构建。

在初始符号program的产生式对应的语义动作中调用多叉树的输出函数。输出函数以递归遍历整棵语法分析树并输出相应信息。

1.2.3 优先级和结合性

- 在bison文件的定义部分定义运算符的优先级以消除Exp对应产生式的二义性。
- 定义一个优先级低于ELSE的运算符以解决悬空else问题

1.2.4 错误恢复

错误恢复过程有如下几条思路：

- 声明语句（Def）出错了则匹配到SEMI后再同步。
- 赋值语句（Stmt）出错了，考虑如下几种情况：
  - 若是运算语句（Exp）或RETURN语句都匹配到SEMI为止，
  - 若是if语句分别考虑不同位置出错的情况（IF后的条件出错、条件后的第一部分语句出错、ELSE后的第二部分语句出错），
  - 若是WHILE语句则考虑条件出错的情况。
- 添加产生式 ExtDefList->error ExtDefList使得上一个函数的错误不影响到下一个函数的分析。
- 由于语法分析树的构建在产生式后的语句中完成，故错误恢复时添加的产生式后也须为产生式左端新建结点，否则会出现Segment fault。

1.3 输出

- 若C--源文件中存在语法或词法错误，则会将错误类型和错误位置（位于源文件第几行）以红色字符输出到stderr中。
- 若C--源文件中不存在语法或词法错误，则会将语法分析树按照要求输出到stdout中。