实验 1　　191220154　　张涵之

a) 我的程序实现了哪些功能？
   a) 检查词法错误（错误类型 A）；
      在 lexical.l 中实现，设全局变量 lexicalErrorCnt 为词法错误计数，所有词法单元正则表达式后定义 MYSCHA（mysterious character）修改计数器并输出错误信息。
   b) 检查语法错误（错误类型 B）；
      在 syntax.y 中实现，设全局变量 syntaxErrorCnt 为语法错误计数，在 yyerror 中修改计数器并输出错误信息，每次匹配到 error 调用时进行。
   c) 构建和输出正确程序的语法树。
      在 syntax.y 和 main.c 中实现，定义结构体 Node 作为语法树的节点，记录词法单元的名字、取值、所在行号、类型，展开时的子节点和同层下一节点。
      定义枚举类型 Type，其中 COMLPLEX_为所有 lexical.l 中不涉及的复杂类型。
      函数 createNode 接收名字、取值、行号和类型并返回生成的节点指针。
      函数 modifyTree 接收当前被操作的和若干个待插入节点，从而构造语法树。
      函数 printNode 和 printTree 用于递归打印语法树。
      全局变量 Node* treeRoot 用于标记语法树的根，仅在 Program 中进行赋值。
      在 lexical.l 中匹配词法单元时调用 createNode 生成节点，syntax.y 中产生式根据附录中的 C--语言文法，生成 Type 为 COMPLEX_的节点并完成 modifyTree 操作。
      各种 error 被匹配后调用 yerror，输出不同的错误信息。
      在 main.c 读取文件，执行 yyparse 后，若此时统计到的词法错误和语法错误数量均为零，即程序正确无误，则调用 printTree 输出语法树，否则不输出。

b) 你的程序应该如何被编译？
   使用 makefile 进行编译。直接 cd 到 Code 目录下 make 即可。
   编译完成后，输入命令./parser test 即可对 test 文件进行词法和语法分析。