191098328 计算机科学与技术系 张世茂

邮箱：191098328@smail.nju.edu.cn

在本次实验中，我借助 Flex 和 Bison 工具实现了针对 C--语言词法分析和文法分析的 C 程序，该程序可以针对给定的 C--代码文件以及相应的语言文法进行词法和语法分析，在出现词法错误时会报告 Type A 类型的错误，在遇到语法错误时会报告 Type B 类型的错误。在整个程序没有错误时，则将整个程序进行语法分析时生成的语法树按照规则的缩进格式进行输出，并针对不同类型的语法单元打印相应的单元值的信息或行号。此外，针对附加任务 1.2 的要求，我还词法分析中增加了对指数形式浮点数的识别。

要实现对程序的词法分析，由于要将读入的字符流识别为不同类型的词法单元，因此首先要设计 C--文法中所需的词法单元对应的正则表达式，而针对这些我们设计出的匹配模式，在接下来可以让程序在识别到某个模式时执行特定的动作。在我的程序中当识别到某一个词法单元时就生成该词法单元的语法树节点并将节点指针值赋给 yylval，然后将识别到的词法单元通过 return 进行返回，这些操作为之后的语法分析打下基础。在语法分析中，首先对 C—文法中要用到的非终结符进行声明，此外由于语法分析涉及到一些算式符号，因此还需要对这些符号的优先级和左右结合性进行规定。为了进行语法分析，主体是实现上下文无关文法，利用已知的 C--文法可以方便地实现，在每个文法匹配规则之后的动作里，利用设计好的函数和已有的数据创建上层节点，并将节点的指针值通过 return 返回向上传递，这里是将所有产生式中的语法单元的数据类型设置为自定义的数据类型 node*。为了使语法分析程序在遇到语法错误时不至于在报错后直接退出，在文法中增加一些 error 单元，其作用就是在遇到错误时可以从栈中弹出一部分元素，将 error 压入后不断向后读取直到找到通过一个定义好的文法式可以对应上的单元再继续压入和归并，进行再同步。为了区分最终是否应当输出语法树，设立初值为 0 的标志位变量 mistakes 即可，若遇到词法或语法错误则将其置 1。在这些都实现完成后，在 main 函数中对函数接口进行适当的调用即可。

有关语法树的实现主要是相关的数据结构以及树节点的创建和传递。因为孩子的数目不固定，因此我创建了一个多叉树类型来生成语法树，在每个节点中都存储了其孩子的数目以及一个 node**类型的指针，指向一个存储 node*类型数据的动态数组，数组中存储的即为分别指向其所有孩子节点的指针。节点类型 node 中存储了语法单元的行号、当前节点孩子的数目、指向储存孩子节点指针的数组的指针、当前节点对应的语法单元的类型、当前节点对应的语法单元的值。一个 create_node 函数就是创建一个动态节点类型并进行对应的赋值，

最后将指向该节点的指针返回。一个 print_tree 函数就是对创建的语法树进行先序遍历输出，根据实验手册要求对不同类型的节点进行区分后进行对应的输出，根据调用函数时传入的深度参数数值进行缩进，先输出当前节点后再遍历孩子递归调用函数即可。

编译该程序的方法为：

1.将文件解压后进入到 Code 文件夹下；

2.在终端中输入 make clean 命令并回车，防止有冗余残留的多余文件；

3.在终端中输入 make 命令并回车，对程序进行编译；

4.生成了最终的可执行文件。

在本次实验中，我在代码中主要的具有特殊性和代表性的工作体现在语法树数据结构的设计以及在上下文无关文法归并后进行的操作设计上。在设计语法树时，为了方便对每一层归并得到的父节点方便统计和遍历其子节点，即语法产生式中产生的语法单元，我将语法树设计为多叉树进行存储，在每一个节点中除存储必要的数据信息外还存储一个存储孩子数量的变量以及一个存储指向孩子节点的指针的动态数组的头指针，这样不仅从理解上更加直观，也可以更方便地进行遍历和相关函数的设计。在上下文无关文法进行匹配和归并时，将语法单元的值的数据类型设定为定义的结构体类型变量 node 的指针变量 node*，这样就可以对值的传递进行统一，且将所有需要的值直接储存在节点中然后将指针进行传递，不仅可以只通过简单的赋值就统一化地完成数据的整合，同时也只需要通过简单的赋值就可以完成多叉树的构建。