编译原理 Lab1 实验报告

a) 我的程序使用 flex+bison 实现了 C–语言的词法分析与语法分析，能够输出抽象语法树，并能给出程序中的词法/语法错误。

b) 通过在 Code 文件夹下运行 `make` 可以生成 `parser` 文件,运行 `./parser/[test_sample]` 可以对样例进行测试。

c) 具体内容

1. 首先设计 ast, 在 ast.h 文件中定义了一个 struct Node, 内部存放 type, son, succ, val, lineno 等信息，以二叉树的形式存在（这里的二叉树两个节点分别命名为 son, succ，son 是该节点第一个儿子, succ 则是该节点在同一级上的后继节点，也就是说这里的二叉树实际上模拟了一个多叉树，son 指向儿子的开头，而儿子们则以链表形式存在，用 succ 连接）
2. 词法分析部分使用了 flex，按照手册所给要求书写了正则表达式，识别到对应 token 后会创建一个代表终结符的 Node, 并将其类型和字符串复制进 Node 信息中。（这里调用的 newNode 函数会根据 token 类型的不同选择不同的生成方式,）
3. 语法分析部分沿用了手册上的规则，在匹配到规则的时候新建一个规约到的项，将被规约的所有项变成这个项的儿子。（newNode 支持可变参数，但必须要传入一个 NULL 作为结尾的判断标志，目前没有找到什么方法可以规避这一点）
4. 打印语法树只需要根据对应的 node 类型，做一遍 DFS 输出即可。这里使用了一些宏定义，来实现定义出枚举类型的同时，定义出用于打印的字符串。
5. 错误恢复部分是比较痛苦的一部分（因为不知道 OJ 会测什么错误，而且相关资料也相对比较欠缺）在不断的摸索之下，添加了很多规则，才完成这一部分内容（面向 OJ 调试）