编译原理 L1 实验报告

实现功能：

使用词法分析工具 GNU Flex，对使用 C—语言书写的源代码进行词法分析，并打印分析结果。

实现步骤：

1. lexical.l 的定义 definitions 部分，定义复杂或者比较长的正则表达式（include 一些头文件，这次实验无需）。定义正则有难度的只有 integer 和 float。Integer 要求能识别八进制和十六进制，float 要求能识别科学计数法（包括.5e-1 和 4.E1 这样的特别写法，小数点可以出现在 e 前的数字串的任意位置）。
2. lexical.l 的规则 rules 部分，为匹配到各个正则表达式时所做的操作。匹配到单行和多行注释时，用 input()消耗掉其中的内容。因为 OJ 运行的是 makefile 中写的最终生成的 parser，所以，匹配到词素 tokens 时，return 一个数字（用这个数字唯一识别一个 token），所需做的打印操作在 main.cpp 中完成（当 main 函数调用 yylex()函数时就可以根据匹配到的正则表达式返回不同的数字）。
3. lexical.l 的 user subroutines 无需书写，main 函数在 main.cpp 中完成。
4. main.cpp 首先需要 extern 引入 yyin、yylineno、yytext 和 yylex()函数。
5. main.cpp 的 main 函数从控制台接收源文件名，然后在 while(true)中调用 yylex()函数，获取 lexical.l 中匹配到正则表达式后返回的数字，直到返回的数

字为 0 表明匹配工作结束。由于如果源文件有错误，则只输出错误，所以用 errOut 和 corrOut 两个字符串记录，根据 wrongFlag 决定最终用 stderr 输出哪个字符串。

6. main.cpp 中难点是以 yytext 为参数写 getNum 获取十进制的数字和 getFloat 获取普通（非科学计数法）的小数。前者分十进制、八进制、十六进制三种情况写。后者可以用#include<sstream>，stringstream(yytext)，用>>转换为普通小数；或者用 atof(yytext)来转换。

印象深刻的点：

1. 将 yytext 所指向的科学计数法小数转换为普通小数时，如果用 stringstream，要声明变量为 double 类型（atof 返回的是 double），声明为 float 精度不够。
2. 将 lexical.l 与 main.cpp 连接起来的方式：在 lexical.l 的 rules 部分匹配到正则表达式，不写处理，而是 return 可唯一识别该正则的数字。
3. lexical.l 的 rules 部分正则的书写顺序：关键字要在 id 前
4. id 中可以有下划线，而不是只有数字和字符
5. 正则表达式书写|符号中间不能有空格