LAB 1 实验报告

裴明亮　151242033

一．实验环境及编，译方法

使用的是 Windows 系统，在实验的根目录下创建了 Makefile 文件；进行测试时，执行 Make test 即可执行相应测试样例的输出。

二. 实验功能说明

1. 识别八进制数和十六进制数

```
oct 0[0-7]*
oct_illegal 0[0-9]*
hex ("0x"|"0X")[0-9A-Fa-f]*
hex_illegal ("0x"|"0X")[0-9A-Za-z]*
```

我在实现词法时，增加了识别正确八进制和十六进制的功能，也增加了识别错误八进制和十六进制的功能，识别的正则表达式如上图，_illegal 表示识别错误的数
可通过样例 5 和 6 进行识别。

2. 区分浮点数和整数

```
float1 {decimal}"."[0-9]+
float2 [1-9]"."[0-9]*[Ee][+-]?[0-9]+
float_ill1 ({decimal}".")|("."{decimal})
float_ill2 [0-9]*"."[0-9]*[Ee][+-]?
```

float1 是正常形式的浮点数，float2 是指数形式的浮点数，_ill1 和_ill2 后缀是相应的错误形式。
float1_ill1 能识别小数点前或者后没有数字的；float2_ill2 能够识别样例中 e 后面缺失指数形式，识别错误后会报错。
可通过样例 7 和 8 进行识别。

3. 识别行注释和块注释

```
comment_line "//"[^\n]*
comment_line_s "/*"[^"*/"]*"*/"
```

comment_1line 是别的是行注释，comment_2line 是别的是块注释，
识别错误后会报错。
可通过样例 7 和 8 进行识别。

4. 语法树的实现

在 NODE.h 和 NODE.c 文件中实现语法树的构造。

NODE.h 文件主要定义了 Node 这种结构类型，属性 name 用来记录语法符号，属性 value 用于记录实际正则表达式匹配的字符串，属性 lineno 用于记录行号，child 指针指向子字节，bro 指针指向兄弟字节

NODE.c 文件则实现 initNODE 函数，addChild 和 printTree 三个函数，其中 initNODE 用于生成语法树上的新节点，出现在词法单元或语法单元的 action 中

```
{yylval.node = initNode("ASSIGNOP",yytext);return ASSIGNOP;}
{yylval.node = initNode("COMMA",yytext);return COMMA;}
```

addChild 用于将 child 设为 parent 的子节点，对右边产生式有多个词法单元的情况下添加顺序

```
Specifier SEMI {$$ = initNode("ExtDef","\0");addChild($$,$2);addChild($$,$1);}
Specifier FunDec CompSt {$$ = initNode("ExtDef","\0");addChild($$,$3);addChild($$,$2);addChild($$,$1);}
```

printTree 用于按照顺序输出符合格式要求的语法树

三. 总结

实验的完成略为匆忙，部分错误恢复的功能存在缺陷，出现异常，需要后续的调整。