编译原理lab1

专业	学号	姓名
计算机科学与技术	19335209	吴浩岚

实验环境

- debian v11.1
- clang v11.0.1-2
- gcc v10.2.1

实验过程

1.词素的提取

通过不断地添加新的词素，即可。在实验过程中可以通过测试的提示，或是通过对比我们的编译器输出和clang的输出，来发现哪些词素没有判断出来。

```
int {
  lexPrint("int");
  return ~YYEOF;
}

char {
  lexPrint("char");
  return ~YYEOF;
}

return {
  lexPrint("return");
  return ~YYEOF;
}

const {
  lexPrint("const");
  return ~YYEOF;
}
```

其中一些值得注意的是，"0x"十六进制数，“&&”等，这些符号如果处理不当，测试过程中不会进行明确的报错，而仅仅是提示token的数量不对。需要进行更仔细地排查。

2.计算行号与列号

其中yyleng是字符串的长度，yycolumn是指当前读取到的位置的列号，yyrow是指当前读取到的位置的行号。

所以我们只需要将yycolumn减去yyleng，就可以得到字符串首字符所在的行号和列号啦。

需要注意的是，我们还需要引入yycolpre，pre_col，pre_row等来处理特殊情况。比如换行之后，需要改变：

```
\n {
  ++yyrow;
  yycolpre = yycolumn;
  pre_col = 1;
  yycolumn = 1;
}
```

3.处理头文件

经过预处理后的头文件如下所示：

```
# 1 "<stdin>"
# 1 "<built-in>" 1
# 1 "<built-in>" 3
# 321 "<built-in>" 3
# 1 "<command line>" 1
# 1 "<built-in>" 2
# 1 "<stdin>" 2
# 1 "sylib/sylib.h" 1
# 10 "sylib/sylib.h"
```

其中，第一个空格后是行号，第二个空格后是路径

所以，我们首先读取前面的数字，并更新行号与列号；

之后读取后面的文件路径，并保存在全局的path中。

4.判断StartOfLine，LeadingSpace

根据行号、列号和前一个列号来判断即可：

```
if(pre_row != yyrow){
    start=" [StartOfLine]";
  }
  if(pre_col != yycolumn - yyleng){
    leading = " [LeadingSpace]";
  }
```

实验结果

如下所示，lexer相关的测试都顺利通过了。