

1.实验过程

只完成了基础部分的实验，通过了 `lexel-[0-3]`。

1. 首先按照[实验一词法分析Wiki](#)中总结的有效token，在 `lexel\lexer.l` 中添加相应的正则表达式和C代码，以及修改模板原本就有的正则表达式，同时也注意过滤 `/* */` 和 `//` 注释，并通过 `lexel-1` 测试是否提取出了正确的token且没有遗漏。

Wiki中提供的有效token并不全，在测试中还发现了新的有效token，如 `do`、`float`，于是添加了新的正则表达式和对应C代码。

2. `lexel-1` 通过后，`lexel-2` 也是直接通过的，因为我直接复制了模板中的 `std::fprintf()` 函数作为正则表达式对应的C代码，仅仅修改了token的名字，因此 `token location` 是不会出错的。

3. 为了识别其他无关字符，也就是 `StartOfLine` 和 `LeadingSpace`：

- 在定义段定义了三个变量：`yystartofline`，int类型，值为1或0，初始值为1，用于判断是否是 `StartOfLine`（值为1）；`yyleadingSpace`，int类型，值为1或0，初始值为0，用于判断是否是 `LeadingSpace`（值为1）；`yyinfo`，char类型字符数组，存储 `StartOfLine` 和 `LeadingSpace` 的信息
- 定义了一个函数 `void getOtherInfo()`，通过 `yystartofline` 和 `yyleadingSpace`，给字符数组 `yyinfo` 赋值。函数的最后给 `yystartofline` 和 `yyleadingSpace` 均赋零
- 将 `getOtherInfo()` 置于除了 `<<EOF>>` 的正则表达式对应的C代码中，并修改 `std::fprintf()`，添加 `yyinfo` 字符数组输出，每一次 `yyinfo` 输出后，都应令 `yyinfo[0] = '\0'`。为了方便修改代码，可将 `getOtherInfo()` 函数和 `std::fprintf()` 函数以及 `yyinfo` 的清空均置于一个函数 `myPrintf()` 中，然后除了 `<<EOF>>` 的正则表达式对应的C代码中只需调用 `myPrintf()` 输出token信息。
- 在正则表达式 `\n` 的C代码中，添加 `yystartofline = 1; yyleadingSpace = 0;`，在正则表达式 `[\t]` 的C代码中，添加 `yyleadingSpace = 1;`

最后通过 `lexel-3` 进行测试，修改。

2.遇到的问题及解决方法

在 `lexel-1` 的测试中，遇到 "counts of tokens are different" 的问题时，我原本通过下述命令对比 `lexel.l` 的token输出和 `clang` 的token输出：

```
1 $HOME/sysu/bin/sysu-lexer < tester/****/*.sysu.c > text1.txt
2
3 ( export PATH=$HOME/sysu/bin:$PATH \
4   CPATH=$HOME/sysu/include:$CPATH \
5   LIBRARY_PATH=$HOME/sysu/lib:$LIBRARY_PATH \
6   LD_LIBRARY_PATH=$HOME/sysu/lib:$LD_LIBRARY_PATH &&
7   clang -E tester/****/*.sysu.c |
8   clang -cc1 -dump-tokens > text2.txt 2>&1 )
```

结果发现 `text1.txt` 中，没有关于头文件的token信息，而 `text2.txt` 中有，我原本以为识别token是忽略头文件的，所以没有在意，删掉了 `clang` 的token输出中关于头文件的部分，然后发现 `lexel.l` 和 `clang` 识别出的token数量是一样的！后来询问同学才知道，头文件也是需要token识别的。

碰巧在课程群中看到了助教发的其他同学关于token数量不一致的解决方法，于是采用了下述命令来输出 `lexel.l` 识别出的token到文本文件中，结果中出现了头文件的token信息：

```
1 ( export PATH=$HOME/sysu/bin:$PATH \  
2   CPATH=$HOME/sysu/include:$CPATH \  
3   LIBRARY_PATH=$HOME/sysu/lib:$LIBRARY_PATH \  
4   LD_LIBRARY_PATH=$HOME/sysu/lib:$LD_LIBRARY_PATH &&  
5   /home/zhou/SysU-lang/preprocessor/sysu-preprocessor /home/zhou/SysU-  
lang/tester/***/**/*.sysu.c |  
6   /home/zhou/sysu/build/lexer/sysu-lexer > test1.txt 2>&1 )
```

所以通过对比 `lexel.1` 的 token 输出和 `clang` 的 token 输出，发现了 token 数量不一致的问题所在，同时也发现了新的有效 token：... 省略号。

3. 自动评测结果

```
zhou@LAPTOP-BOER6N6Q:~/SYsU-lang$ CTEST_OUTPUT_ON_FAILURE=1 cmake --build $HOME/sysu/build -t test  
[0/1] Running tests...  
Test project /home/zhou/sysu/build  
Start 1: lexer-0  
1/10 Test #1: lexer-0 ..... Passed    0.55 sec  
Start 2: lexer-1  
2/10 Test #2: lexer-1 ..... Passed   115.00 sec  
Start 3: lexer-2  
3/10 Test #3: lexer-2 ..... Passed   101.99 sec  
Start 4: lexer-3  
4/10 Test #4: lexer-3 ..... Passed   106.07 sec  
Start 5: parser-0  
5/10 Test #5: parser-0 ..... Passed    0.33 sec  
Start 6: parser-1
```