词法实验

实验目标

借助flex工具实现一个词法分析器，将SysY代码中的单词和符号进行分类，然后按照单词符号出现顺序依次输出：原始单词符号、种类、出现在源程序的位置（行数和列数）。

具体类别如下：

标识符	类别
K	关键字
I	标识符
C	常量
O	算符
D	界符
T	其他

文件夹内容

- `1-词法实验.pdf`：本实验指导。
- `demo-code/`：文件夹下包含一个demo代码，用于理解flex的使用方法，在终端执行`run .sh`即可运行程序
  - `example.l`：一个简单的示例代码
  - `run.sh`：运行示例代码的脚本
  - `test.sy`：用于分析的sysy代码
- `测试用例/`：包含6个sy文件，用于测试代码的正确性。选择plo的同学，请根据这6个测试文件编写相应的plo测试文件，对于sysy中有但是plo中没有的语法，可以不用写到测试文件当中。
- `example/`：包含一个运行示例，仅供参考

- `example.sy`：示例sysy代码
- `example.out`：示例的词法分析输出

实验指导

（1）flex的使用

```
%option noyywrap
%{
声明
%}
辅助定义
%%
识别规则
%%
用户子程序
```

- 声明：该部分用于编写需要使用的**头文件、全局变量**等，可以为空，代码语法同C。
- 辅助定义：该部分可以使用一个名字代表一个正规式，以便于后续规则的编写。特别地，后面的辅助定义也可以使用前面的辅助定义。

  ```
  NAME EXPRESSION
  ```

  辅助定义每行两个字符串，第一个（NAME）表示该正规式的名字，第二个（EXPRESSION）为正规式。
- 识别规则：该部分由正规式和相应的动作组成。表示匹配到每个正规式之后，需要进行的操作。该部分的正规式可以使用辅助定义中的内容（使用{NAME}使用名字为NAME的辅助定义）。

  ```
  EXPRESSION {
    CODE
  }
  ```

  其中，EXPRESSION表示正规式，CODE表示匹配到这个表达式之后需要进行的操作。CODE部分的代码为C代码。
- 用户子程序：该部分编写包括main函数在内的所有需要用到的函数，flex会将其插入到生成的C代码中。需要注意的是，main函数中需要调用yylex函数，执行正规式的匹配。

  对于文件的输入输出，仅需要将yyin赋值为所需要的文件指针（也可以是stdin），其余的yylex会自动处理。

编写完程序之后，使用如下指令运行flex：

```
flex PATH_TO_FLEX_CODE
```

其中PATH_TO_FLEX_CODE为编写的flex程序的路径。

运行结束之后，会生成lex.yy.c文件。编译并运行该文件即可进行测试。

```
gcc lex.yy.c -o a
./a
```

（2）规则编写

以下列举几个编写规则时需要注意的重难点：

- 行数、列数统计，以便于输出错误信息。
- 对于程序语法错误的处理，即其他类（T类）。
- 对于注释的处理，包括单行注释和成段注释。
- windows和linux下换行符的差别。
- 包含数字的变量和数字之间的区分。

（3）正规式

下面是可能使用到的正规式的语法规则，更多的规则可以自行百度。

正规式	匹配的内容
x	匹配一个字母x（匹配完全相同的内容）
.	匹配**除换行符**外的所有符号
"abc"	完全匹配abc
[abc]	匹配a或b或c（匹配中括号中的**任意一个**字符）
[a-z]	匹配a或b或……或z的任意一个字符
[^a-z]	**不匹配**a或b或……或z的任意一个字符
r*	正规式r出现**零次或多次**
r+	正规式r出现**一次或多次**
r?	正规式r出现**零次或一次**
r{1,2}	正规式r出现**1-2次**
r{1,}	正规式r出现**1次或多次**
r{1}	正规式r出现**1次**
r|s	匹配正规式r或正规式s
rs	**顺序**匹配正规式r和正规式s

同时，可以使用**反斜杠（\）**或**双引号**对符号进行转义。例如，可以使用 \] 或者 "]" 匹配右括号。