实验二、自动生成词法分析程序（JFlex）

实验步骤 2.1、总结 Oberon-0 语言的词汇表

一、词汇表

数值常量	十进制	(1-9)+(1-9)*
	八进制	0(0-7)*
标识符		letter(letter | 0-9)
注释		"(*" { 除了"*)"任何字符组合} "*)"
标点符号		; | ,
关键字		integer | write | read | writeln | boolean
保留字		if | then | elsif | while | do | begin | end | of | var | const | array | module | type | procedure | record
运算符	算术运算符	+ | - | * | div | mod
	关系运算符	> | < | # | = | >= | <=
	逻辑运算符	&| or | ~
	复值运算符	:=
	选择运算符	.|[|]
	括号运算符	(|)
	类型运算符	:

二、单词分类的理由说明

在保留字和关键字处理方面，根据实验一总结可知，保留字主要负责代码结构方面，而关键字主要是提供一些预定义的函数以及预声明的类型，而由于观察到保留字均在EBNF中就定义了，而关键字则没出现过在EBNF中，所以对其分类并不困难。

其他的分类遵循以往学习中所了解到的符号的属性，而对于div、mod这种虽然看上去很像关键字的运算符，但考虑到它们的作用以及在产生式中的位置，我还是将其归到运算符一类。

实验步骤 2.2、抽取 Oberon-0 语言的

词法规则

一、根据词汇表可求出以下分类：

```
Number -> Decimal | Octal

Decimal -> [1-9](0-9)*

Octal -> 0(0-9)*

Identifier -> [a-zA-Z]([a-zA-Z] | 0-9)*

Comment -> "(*" ([^\*] | "*"+[^\)])* "*)"

Punctuation -> ";" | ","

ReservedWord -> "If" | "then" | "elsif" | "while" | "do" | "begi
n" | "end" |"of" | "var" | "const " | "record" | "array" | "modu
le" | "type" | "procedure"

Keyword -> "Integer" | "write" | "read" | "writeln" | "boolean"

Operator -> "+"|"-"|"*"|"div"|"mod"|":="| "="|"#"|">"|"<"|">="|"
<="| "("|")"|"&"|"or"|"~"|"["|"]"|"."|":"
```

二、与高级程序设计语言的词法规则的比较

1. 对赋值符和相等符的区分方面，同pascal一样，在oberon-0中，采用 `:=` 表示赋值符，使用 `=` 表示相等符，而在c和java语言中， `=` 表示赋值符， `==` 表示相等符。
2. 而对于取余与除法，在oberon-0中，采用 `DIV` 与 `MOD` ，而在c和java中语言采用 `/` 和 `%`
3. 在oberon-0中，不等于使用 `#` 取表示不等于，取 `&` 表示与，取 `or` 表示或，取 `~` 表示非，而在c与java中，取 `!=` 表示不等于，取 `&&` 表示与，取 `||` 表示或，取 `!` 表示非
4. 在保留字关键字方面，和 pascal 的一样，都有 `begin` 、 `then` 和 `end` 、 `procedure` 、 `module` 的标识，同时也有 `var` 的变量声明，而对比c与java这些高级程序设计语言，关键字除了提供 `integer` 和 `boolean` 外，还提供了更多的

类型例如 `long int`、 `double` 等，功能更加强大。 `Read` 和 `write` 和 `pascal` 基本一样，Java要用system的包才能实现，而 C 则是要用的printf。

5. 注释方面，oberon-0用的是 `(**)` , C 和 java 用的是 `//` 、 `/**/` 等，pascal 用的是 `{}` .

实验步骤 2.3、下载词法分析程序自动生成工具 JFlex

一、安装及配置

1.修改bin/jflex.bat

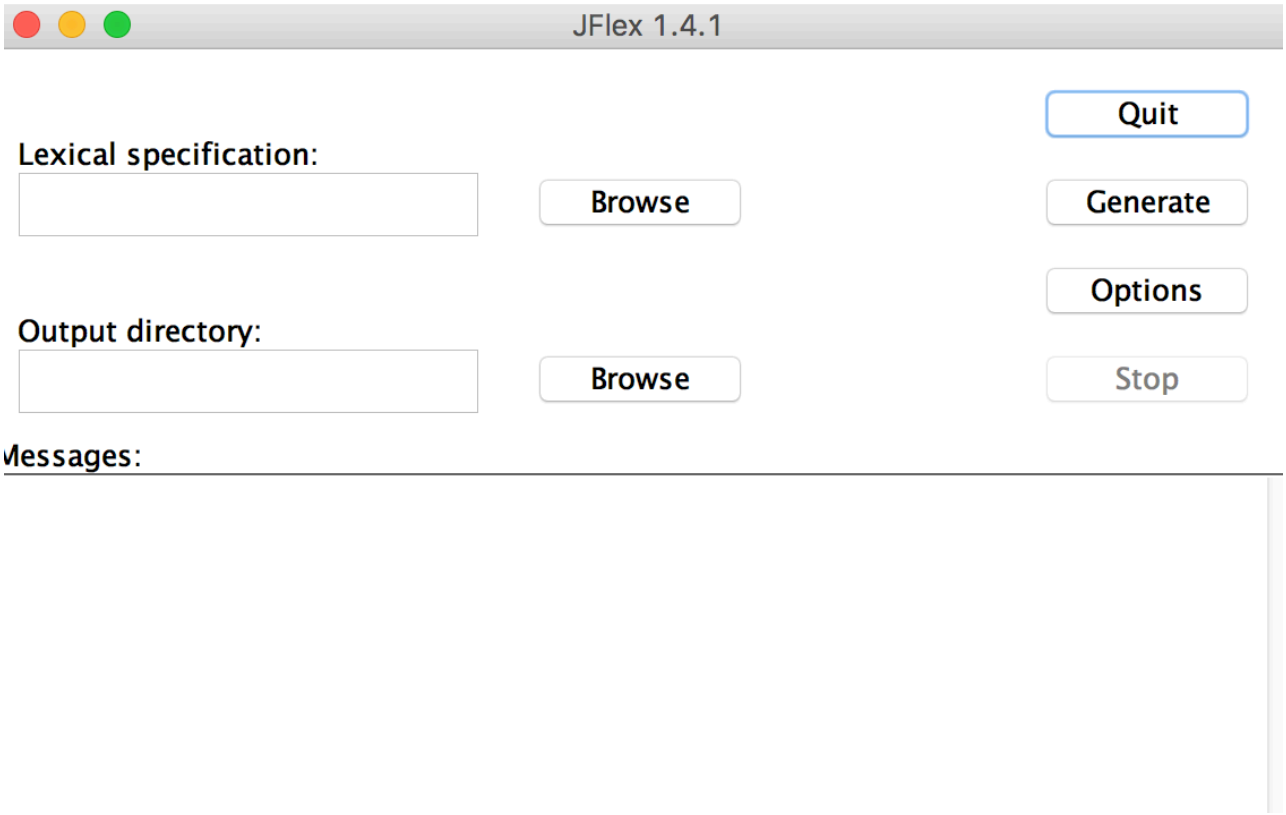
```
set JFLEX_HOME=/Users/Aaron-Qiu/Desktop/大师之路/黑科技/jflex-1.4-2.1

REM only needed for JDK 1.1.x:
set JAVA_HOME=/usr/libexec/java_home
```

2.修改环境变量~/. bash_profile

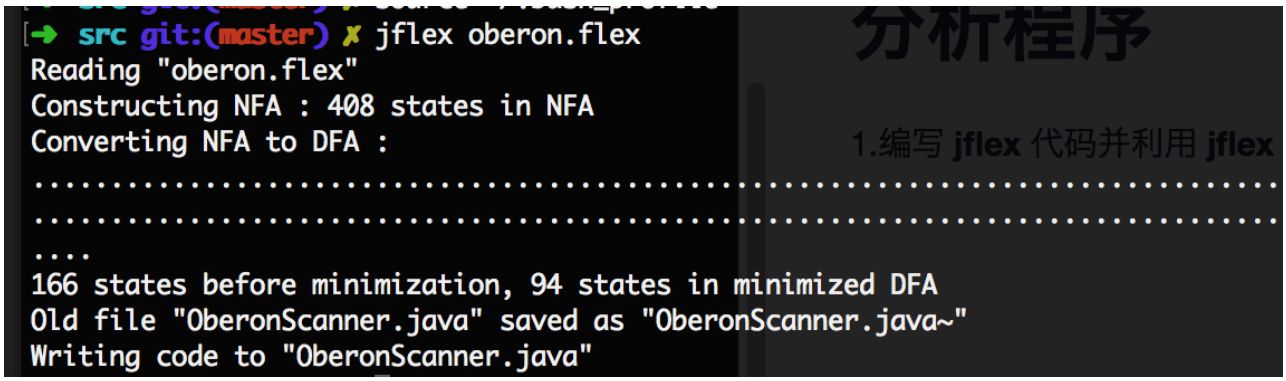
```
# add jflex
export PATH=/Users/Aaron-Qiu/Desktop/大师之路/黑科技/jflex-1.4-2.1/bin:$PATH
```

3.查看版本（验证是否安装陈功）



实验步骤 2.4、生成 Oberon-0 语言的词法分析程序

一、编写 jflex 代码并利用 jflex 生成词法分析器。



二、运行无词法错误样例

```
ReservedWoed : END
Punctuation : ;
ReservedWoed : END
Identifier : StudentsInfo
Punctuation : ;
ReservedWoed : BEGIN
Identifier : Factorial
Operator : (
Operator : )
Punctuation : ;
Identifier : StudentsInfo
Operator : (
Operator : )
Punctuation : ;
ReservedWoed : END
Identifier : SomeFuncs
Operator : .
EOF :
../src/testcases/SomeFuncs.obr : No Lexical error!
```

三、运行错误样例（由于错误样例太多，该处只给出 第一个错误样例的运行效果）

```
➜ bin git:(master) ✗ java Main ../src/testcases/SomeFuncs.001
ReservedWoed : Module
Identifier : SomeFuncs
Punctuation : ;
Identifier : PRICEDURE
Identifier : Factorial
Punctuation : ;
ReservedWoed : VAR
comment : (* IllegalSymbolException *)
Identifier : n
Punctuation : ,
Identifier : i
Punctuation : ,
Identifier : result
Operator : :
Keyword : INTEGER
Punctuation : ;
../src/testcases/SomeFuncs.001 : LexicalException :
Illegal Symbol.
Line 4, Colume 24: @
```

实验步骤 2.5、讨论不同词法分析程序

生成工具的差异

由于只实际操作过jFlex，所以对 JLex 和 GNU Flex 的讨论只可以根据查看其文档来观察差异，个人认为主要有以下差异：

1. 运行环境不同，jFlex 和 JLex 运行在 java 平台，而 GUN Flex 运行在 C 语言平台。
2. 文件分割不一样，虽然它们都是将文件分为三部分，都用到了 `%%` 符号去分割，但 jflex 与 jlex将 3部分分为用户代码、选项声明、词法规则，而GNU Flex 将3部分分为定义段 (definitions) 、规则段 (rules)、用户代码段 (user code)。由于编程语言的不同，gnu flex 中还可以直接写上 main 函数。
3. 而在用户代码定义上，jflex 与 jlex 都采用了 `%{ <code> %}` 的格式。而gnu flex无需特殊格式，直接在相应部分编写代码即可。
4. gnu flex中规则部分与 jflex 和 jlex 也有很大不同。gnu flex的定义例子 `[a-zA-Z]+ { str += yytext; return 1; }`；jflex的定义例子 `Identifier = [:jletter:]+ [:jletterdigit:]*`