

实验二、 自动生成词法分析程序（JFlex）

19335019 陈浣

Oberon-0 语言的词汇表

以表格形式列出 Oberon-0 语言的词汇表：

类别	内容
数值常量	(1-9)+(1-9)* 0(0-7)*
算术运算符	+, -, *, DIV, MOD
关系运算符	=, #, <, <=, >, >=
逻辑运算符	&, OR, ~
其他运算符	:=, :, (,), [,]
标识符	letter {letter digit}*
标点符号	;, .
保留字	MODULE, PROCEDURE, BEGIN, END, IF, THEN, ELSIF, ELSE, WHILE, DO, OF
关键字	INTEGER, BOOLEAN, Read, Write, WriteLn, CONST, ARRAY, VAR, TYPE, RECORD
注释	(* , *)

（1）单词分类的理由：主要依据在Oberon-0 语言充当的角色进行分类。 另外， 以下四类其实可以归属为运算符这一大类， 为了精细划分才拆分为4类。

运算符类	内容
算术运算符	+, -, *, DIV, MOD
关系运算符	=, #, <, <=, >, >=
逻辑运算符	&, OR, ~
其他运算符	:=, :, (,), [,]

（2）对于保留字与关键字的分类：主要还是依据实验一中的思路， 与程序功能性相关的归属到关键字， 与划分程序结构层次相关的归属到保留字。

Oberon-0 语言的词法规则

用正则定义式描述 Oberon-0 语言词法规则； 若你使用纯文本书写正则定义式， 其中的元符号“定义为”使用“->”表示， 空串用“epsilon”表示。

```

1 digit -> [0-9]
2 letter -> [a-z A-Z]
3
4 Number -> Decimal | Octal
5 Decimal -> [1-9](digit)*
6 Octal -> 0[0-7]*
7
8 Identifier -> letter( letter | digit )*
9
10 ReservedWord -> "module" | "procedure" | "begin" | "end" | "if" | "then" |
    "elseif" | "else" | "while" | "do" | "of"
11 Keyword -> "Integer" | "write" | "read" | "writeln" | "boolean" | "const" |
    "array" | "var" | "type" | "record"
12
13 Comment -> "(*" ([^*] | "*" + "[^\\)]" ) * "*)"
14
15 Punctuation -> ";" | ","
16
17 Operator -> "+" | "-" | "*" | "div" | "mod" | ":" | "=" | "<" | ">" | "<=" | ">=" |
    "(" | ")" | "&" | "or" | "~" | "[" | "]" | "." | ":"

```

Oberon-0 语言与其他高级语言的词法规则的异同比较

与 Pascal、C/C++、Java 等常见高级程序设计语言的词法规则相比，Oberon-0 语言的词法规则有何异同？

(1) 不同之处

- 部分运算对应的运算符不同：例如 or 表示或，~ 表示非，:= 为赋值，DIV 为除法，MOD 取余，# 表示不相等等。
- 注释的方式不同。
- 标识符中不可以有下划线
- 数据只能是整数类型和 Boolean 类型，且不支持书写 BOOLEAN 类型的常量，没有浮点数类型、字符串类型等其他类型。

(2) 相同之处：除了上述提及的区别之外，其他的词法规则与 Pascal、C/C++、Java 等常见高级程序设计语言的词法规则基本保持一致。

3 种不同 lex 族软件工具的输入文件中，词法规则定义的差异或特点

1. 运行的环境不同：GUN Flex 运行环境是 C 语言平台，jFlex 和 JLex 运行环境是 java 平台。
2. 词法和语法规则上存在差异。
3. jflex 与 jlex 需要将代码写入 `%{` 和 `%}` 之间；gnu flex 则直接编写代码，没有另外的格式要求。
4. 源文件的结构不同：jflex 与 jlex 分为用户代码、选项声明、词法规则三个部分，GNU Flex 则分为定义段 (definitions)、规则段 (rules)、用户代码段 (user code) 三部分。

生成 Oberon-0 语言的词法分析程序

(1) 测试实验一中自己写的源代码 `account.obr` :

```
1  ReservedWord : MODULE
2  Identifier : program
3  PUNCTUATION : ;
4  COMMENT : (* define the type--AccountRecord *)
5  Keyword : TYPE
6  Identifier : AccountRecord
7  OPERATOR : =
8  Keyword : RECORD
9  Identifier : id
10 OPERATOR : :
11 Keyword : INTEGER
12 PUNCTUATION : ;
13 Identifier : salary
14 OPERATOR : :
15 Keyword : INTEGER
16 ReservedWord : END
17 PUNCTUATION : ;
18 COMMENT : (*Find the max salary among n accounts *)
19 ReservedWord : PROCEDURE
20 Identifier : FindMax
21 PUNCTUATION : ;
22 Keyword : VAR
23 Identifier : x
24 PUNCTUATION : ,
25 Identifier : n
26 PUNCTUATION : ,
27 Identifier : i
28 OPERATOR : :
29 Keyword : INTEGER
30 PUNCTUATION : ;
31 Identifier : accounts
32 OPERATOR : :
33 Keyword : ARRAY
34 NUMBER : 100
35 ReservedWord : of
36 Identifier : AccountRecord
37 PUNCTUATION : ;
38 ReservedWord : BEGIN
39 Keyword : Read
40 OPERATOR : (
41 Identifier : n
42 OPERATOR : )
43 PUNCTUATION : ;
44 Identifier : i
45 OPERATOR : :=
46 NUMBER : 1
47 PUNCTUATION : ;
48 Identifier : x
49 OPERATOR : :=
50 NUMBER : 0
51 PUNCTUATION : ;
52 ReservedWord : if
53 Identifier : n
54 OPERATOR : >
55 NUMBER : 0
```

```
56 Reservedword : THEN
57 Identifier : accounts
58 OPERATOR : [
59 NUMBER : 0
60 OPERATOR : ]
61 PUNCTUATION : .
62 Identifier : id
63 OPERATOR : =
64 NUMBER : 0
65 PUNCTUATION : ;
66 Identifier : x
67 OPERATOR : :=
68 Keyword : Read
69 OPERATOR : (
70 Identifier : accounts
71 OPERATOR : [
72 NUMBER : 0
73 OPERATOR : ]
74 PUNCTUATION : .
75 Identifier : salary
76 OPERATOR : )
77 Reservedword : END
78 PUNCTUATION : ;
79 Reservedword : WHILE
80 OPERATOR : (
81 Identifier : i
82 OPERATOR : <
83 Identifier : n
84 OPERATOR : &
85 Identifier : i
86 OPERATOR : <
87 NUMBER : 100
88 OPERATOR : )
89 Reservedword : DO
90 Identifier : accounts
91 OPERATOR : [
92 Identifier : i
93 OPERATOR : ]
94 PUNCTUATION : .
95 Identifier : id
96 OPERATOR : =
97 Identifier : i
98 PUNCTUATION : ;
99 Keyword : Read
100 OPERATOR : (
101 Identifier : accounts
102 OPERATOR : [
103 Identifier : i
104 OPERATOR : ]
105 PUNCTUATION : .
106 Identifier : salary
107 OPERATOR : )
108 PUNCTUATION : ;
109 Reservedword : if
110 Identifier : accounts
111 OPERATOR : [
112 Identifier : i
113 OPERATOR : ]
```

```
114 PUNCTUATION : .
115 Identifier : salary
116 OPERATOR : >
117 Identifier : x
118 ReservedWord : THEN
119 Identifier : x
120 OPERATOR : :=
121 Identifier : accounts
122 OPERATOR : [
123 Identifier : i
124 OPERATOR : ]
125 PUNCTUATION : .
126 Identifier : salary
127 ReservedWord : ELSE
128 Identifier : x
129 OPERATOR : :=
130 Identifier : x
131 ReservedWord : END
132 PUNCTUATION : ;
133 Identifier : i
134 OPERATOR : :=
135 Identifier : i
136 OPERATOR : +
137 NUMBER : 1
138 PUNCTUATION : ;
139 ReservedWord : END
140 PUNCTUATION : ;
141 Keyword : write
142 OPERATOR : (
143 Identifier : x
144 OPERATOR : )
145 PUNCTUATION : ;
146 Keyword : write
147 OPERATOR : (
148 Identifier : n
149 OPERATOR : )
150 PUNCTUATION : ;
151 Keyword : write
152 OPERATOR : (
153 Identifier : i
154 OPERATOR : )
155 PUNCTUATION : ;
156 Keyword : writeLn
157 ReservedWord : END
158 Identifier : FindMax
159 PUNCTUATION : ;
160 ReservedWord : END
161 Identifier : program
162 PUNCTUATION : .
163 EOF :
164 Scanning Finished. No Error Found.
```

(2) 测试词法错误的类型以及测试结果:

1. Illegal Symbol Exception: 识别单词时遇到不合法的输入符号(譬如@、\$等符号)则抛出该异常。
出错内容: `sal@ary: INTEGER;`

```
Identifier : sal
..\src\testcases\account.001 : Compiler error. (details:Lexical error. (details:IllegalSymbol error. (details:IllegalSymbol error.)))
Line 19, Column 34: @
```

2. Illegal Octal Exception: 当 0 开头的整数常量中含有 0~7 之外的符号(包括 8 和 9 或其他字母)时抛出该异常。出错内容: `x := 08;`

```
Identifier : x
OPERATOR :=
..\src\testcases\account.002 : Compiler error. (details:Lexical error. (details:IllegalOctal error. (details:IllegalOcta
1 error.)))
Line 16, Column 6: 08
```

3. 含有不合法的常量：常量中数字的个数超出限制。出错内容：`i :=`

```
100000000000000000000000000000000000001;
```

```
Identifier : i  
OPERATOR :=  
..\src\testcases\account.003 : Compiler error. (details:Lexical error. (details:IllegalIntegerRange error. (details:Ille  
galIntegerRange error.)))  
Line 15, Column 6: 100000000000000000000000000000000
```

4. 标识符长度不允许超过 24 个字符。出错内容: `VAR x, n, izxcvbnmasdfghjklqwert:`

```
INTEGER;  
Keyword : VAR  
Identifier : x  
PUNCTUATION : ,  
Identifier : n  
PUNCTUATION : ,  
.\src\testcases\account.004 : Compiler error. (details:Lexical error. (details:IllegalIdentifierLength error. (details:  
IllegalIdentifierLength error.)))  
Line 11, Columne 11: izxcvbnmasdfghjklqwertyuiop
```

实验心得与体会

实验二内容相比实验一多了一些。通过实验二，我熟悉了 jflex 的使用，了解了从生成词汇表到确定词法规则，再到利用现有工具生成词法分析器的生成过程。另外，通过与其他高级语言的比较，也加深了对几种语言的认识与辩证性思考。总之，在这一过程中有一定的收获。