

LAB3 ROSE

ex2

21307099 李英骏

目录

| | | |
|----------|------------------------|----------|
| 1 | Oberson-0 语言词汇表 | 2 |
| 2 | Oberon-0 语言词法规则 | 4 |
| 2.1 | 词法规则的正则定义式描述 | 4 |
| 2.2 | 词法规则异同 | 4 |
| 3 | lex 族工具的输入差异 | 5 |
| 4 | 实验记录: 词法分析器的生成 | 7 |

1 Oberson-0 语言词汇表

| 保留字/关键字 | 说明 |
|-----------|---------------|
| MODULE | 声明程序模块（保留字） |
| BEGIN | 标记程序段开始（保留字） |
| END | 标记程序段结束（保留字） |
| TYPE | 定义自定义类型（保留字） |
| OF | 表示所属关系（保留字） |
| WHILE | 循环语句的开始（保留字） |
| DO | 执行循环体（保留字） |
| IF | 条件判断开始（保留字） |
| THEN | 如果条件满足执行（保留字） |
| ELSIF | 其他条件判断（保留字） |
| ELSE | 条件不满足时执行（保留字） |
| CONST | 声明常量（关键字） |
| VAR | 定义变量（关键字） |
| PROCEDURE | 定义程序子过程（关键字） |
| RECORD | 定义数据结构体（关键字） |
| ARRAY | 数组定义（关键字） |
| OR | 逻辑或运算（关键字） |
| DIV | 进行整数除法（关键字） |
| MOD | 进行模除运算（关键字） |

| 保留字/关键字 | 说明 |
|---------|-------------|
| INTEGER | 定义整数类型（关键字） |
| BOOLEAN | 定义布尔类型（关键字） |

| 运算符 | 说明 |
|-----|-----------|
| + | 执行加法或取正 |
| - | 执行减法或取负 |
| * | 执行乘法 |
| > | 判断大于 |
| >= | 判断大于等于 |
| < | 判断小于 |
| <= | 判断小于等于 |
| # | 判断不等于 |
| = | 判断等于 |
| & | 逻辑与运算 |
|) | 表示右括号 |
| : | 类型声明使用 |
| ; | 结束语句标志 |
| ~ | 执行逻辑非操作 |
| := | 赋值操作 |
| . | 访问结构体成员 |
| [| 左方括号，用于数组 |
|] | 右方括号，用于数组 |
| (| 表示左括号 |
| , | 分隔变量或参数 |
| EOF | 表示源程序的结束 |

| 常量 | 标识符 |
|-----------------------|------------|
| [1-9][0-9]* 0[0-7]* | 字母开头，可包含数字 |

2 Oberon-0 语言词法规则

2.1 词法规则的正则定义式描述

```

Number      -> [1-9][0-9]* | 0[0-7]*
Identifier  -> [a-zA-Z][0-9a-zA-Z]*
WhiteSpace  -> " " | \r | \n | \r\n | \t | \f | \b
Comment     -> \(\*([^\*] | (\*([^\*])))* (\*))* \* \)

```

2.2 词法规则异同

相同点

- 标识符不能以数字开头.
- 八进制数以 0 开头.

不同点

- 标识符规则:
 - C/C++ 和 Java: 标识符可以包含下划线, 并且长度不限.
 - Oberon-0: 标识符不能包含下划线, 并且长度限制为 24 个字符.
- 注释风格:
 - C/C++: 支持 /* */ 块注释和 // 行注释.
 - Oberon-0: 使用 (* *) 作为注释.
- 整数表示:
 - C/C++ 和 Java: 整数表示不受进制影响, 使用二进制表示法.
 - Oberon-0: 整数表示受十进制和八进制的区别影响.
- 运算符:
 - C/C++ 和 Java: 使用
 - Oberon-0: 使用 MOD 进行取模, 使用 DIV 进行除法, 使用 = 进行等于比较, 使用 # 进行不等于比较, 使用 OR 进行或操作, 使用 进 进行非操作.

3 lex 族工具的输入差异

JFlex

- **正则表达式**: JFlex 支持扩展的正则表达式, 包括字符类、组、量词、断言等.
- **状态机**: JFlex 允许定义多个词法分析状态, 并在不同状态之间切换. 这使得处理复杂的词法规则变得更加容易.
- **Unicode 支持**: JFlex 完全支持 Unicode, 可以处理多种语言的输入.
- **用户代码**: JFlex 允许在词法规则中嵌入 Java 代码, 以便在匹配到特定模式时执行.

JLex

- **正则表达式**: JLex 支持基本的正则表达式, 但不如 JFlex. 它主要提供了字符类、组、量词等基本功能.
- **状态机**: JLex 也支持词法分析状态的定义, 但功能和灵活性不及 JFlex.
- **Unicode 支持**: JLex 的 Unicode 支持较为有限, 主要用于处理 ASCII 和基本的 UTF-8 字符集.
- **用户代码**: JLex 允许在词法规则中嵌入 Java 代码, 但嵌入方式较为简单, 缺乏 JFlex 的灵活性.

GNU Flex

- **正则表达式**: GNU Flex 支持广泛的正则表达式, 包括字符类、组、量词、断言等. 正则表达式功能非常强大, 与 JFlex 相当.
- **状态机**: GNU Flex 允许定义多个词法分析状态, 并在不同状态之间切换. 处理复杂的词法规则时更方便.
- **Unicode 支持**: GNU Flex 的 Unicode 支持较为有限, 主要用于处理 ASCII 和基本的 UTF-8 字符集.
- **用户代码**: GNU Flex 允许在词法规则中嵌入 C 代码.

JFlex 的基本框架

- 用户代码
- %% 选项与声明
- 词法规则

JFlex 的选项与声明

- %class 中定义了生成的 Java 文件的类名以及文件名.
- %eofval 中定义了当识别到 EOF 的时候执行的操作.
- %yylexthrow 中定义了当前词法分析程序中要抛出的异常.
- %cup 的声明是为 java_cup 提供接口, 因此生成了 next_token() 函数.
- %line 以及 %column 是声明了可以使用 yyline 以及 yycolumn 获得当前的位置.
- %..... 用户代码.....% 中填写的是用户代码, 一般在这里实现 main 函数.
- <YYINITIAL> 中定义了对于匹配每一个正则表达式后要执行的操作.

比较

- **正则表达式功能**: JFlex 和 GNU Flex 支持更强大的正则表达式功能.
- **Unicode 支持**: 只有 JFlex 提供了更全面的 Unicode 支持.
- **用户代码嵌入**: JFlex 和 GNU Flex 的嵌入方式更为灵活.

4 实验记录: 词法分析器的生成

软装置中似乎并没有文档提及的 Exceptions 和测例. Exception 自行编写, 有几个是复制了上次 LAB2 计算器的, 测例使用 ex1 中构建的测例 (其中前 4 个是词法错误, 详见 ex1 报告). 编写了一个简单的 Main 函数调用生成的词法分析器. 为了方便截图, flex 文件中没有设置%debug, 因此运行 run.bat(即正确的输入) 时不会有任何输出, 而运行 test.bat 有以下输出:

```
(base) PS C:\Users\liyj3\Desktop\Compiler\LAB\LAB3\21307099_李英骏_LAB3\ex2> .\test.bat
Running arithmetic.001
Unexpected exception:
exceptions.IllegalOctalException: LexicalException :
Illegal Octal number.
Illegal Octal number.
    at OberonScanner.yylex(OberonScanner.java:1247)
    at Main.processFile(Main.java:22)
    at Main.main(Main.java:12)
Running arithmetic.002
    at Main.main(Main.java:12)
Running arithmetic.004
Unexpected exception:
exceptions.IllegalSymbolException: LexicalException :
Illegal Symbol.
Illegal Symbol.
    at OberonScanner.yylex(OberonScanner.java:1130)
    at Main.processFile(Main.java:22)
    at Main.main(Main.java:12)
Running arithmetic.005
Running arithmetic.006
Running arithmetic.007
Running arithmetic.008
Running arithmetic.009
Running arithmetic.010
Running arithmetic.obr
```

图 1: Lexical Exceptions Testss

可以看到正确地识别到了词法错误.