

# 中国矿业大学计算机学院 2017级本科生课程设计报告

# 实验二 Flex理论与练习2

| 课程名称 | 系统软件开发实践   |  |
|------|------------|--|
| 报告时间 | 2020.02.19 |  |
| 学生姓名 | 李治远        |  |
| 学 号  | 07172757   |  |
| 专业   | 计算机科学与技术   |  |
| 任课教师 | 席景科老师      |  |

# 目录

| 1 | 实验目的                          | 1        |  |  |
|---|-------------------------------|----------|--|--|
| 2 | 实验任务                          | 1        |  |  |
| 3 | <b>实验代码</b><br>3.1 lex2-1.l代码 | <b>1</b> |  |  |
|   | 3.2 lex2-2.l代码                | 2        |  |  |
| 4 | · 分析程序输出结果                    |          |  |  |
|   | 4.1 分析lex2-1.l实验结果            | 4        |  |  |
|   | 4.2 分析lex2-2.l实验结果            | 5        |  |  |
| 5 | 5 在Windows环境下实验               |          |  |  |
|   | 5.1 lex2-1.l实验结果              | 6        |  |  |
|   | 5.2 lex2-2.l实验结果              | 6        |  |  |
| 6 | 6 在ubuntu环境下实验                |          |  |  |
|   | 6.1 lex2-1.l实验结果              | 7        |  |  |
|   | 6.2 lex2-2.l实验结果              | 7        |  |  |
| 7 | 实验咸粗                          | 8        |  |  |

实验目的 1

# 实验二 Flex理论与练习1

# 1 实验目的

- 1. 阅读《Flex/Bison.pdf》第一章,第二章,掌握Flex基础知识。
- 2. 利用Flex实现用于C语言子集C1的词法分析器。

# 2 实验任务

- 1. 编写Flex代码『lex2-1.1』,实现对上述 $C_1$ 语言的词法分析。要求:输出所有的关键字、专用符号、标识符、整型常数。
- 2. 在实现以上基本功能的基础上,参考《ANSI C grammar (Lex).pdf》,实现以下功能,并另存为『lex2-2.1』:
  - (a) 输出上述标记所在的行号;
  - (b) 忽略注释及其内容,如,注释中的数字/\*123\*/,/123;
  - (c) 增加科学记数法;
  - (d) 十六进制、八进制常数。

# 3 实验代码

#### 3.1 lex2-1.l代码

```
%{
         int line = 1;
     %}
     KWYWORS (else|if|switch|for|int|float|return|void|while)
     L [a-zA-Z_]
     D [0-9]
     ID {L}({L}|{D})*
     NUM [1-9]{D}*
10
     CHARACTER (\+|\-|\*|\/|<|"<="|>|">="|"!="|"|=|;|,|\(|\)|\[|\]|\{|\})
11
12
     delim [ \t\n]
13
     whitespace {delim}+
14
     A [/]
16
     B [*]
17
```

实验代码 2

```
C [^*/]
19
     NOTES1 "/*"(.|\n)*"*/"
     NOTES2 "//".*\n
     NOTES ({NOTES1}|{NOTES2})
     %%
23
     n {line++;}
24
     {whitespace} {}
25
     {NOTES} {/*注释*/ line++;}
     {KWYWORS} {printf("%d\t%20s\t关键字\n", line, yytext);}
     {CHARACTER} {printf("%d\t%20s\t专用符号\n", line, yytext);}
29
     {ID} {printf("%d\t%20s\t标识符\n", line, yytext);}
30
     {NUM} {printf("%d\t%20s\t整数\n", line, yytext);}
31
     . {}
32
     %%
34
     void main(){
35
         printf("Line\t%20s\tPs\n", "Actor");
36
         printf("\n");
37
         yylex();
38
     }
     int yywrap(){
         return 1;
41
42
```

#### 3.2 lex2-2.1代码

此代码基于实验要求1扩展分析。

- 增加专用符号集合;
- 增加关键字集合范围;
- 增加字符串、字符匹配;
- 增加数字类型匹配,整数、八进制、十六进制、科学计数法与浮点类型;

```
1 %{
2 int line = 1;
3 %}
4
5 /*浮点数指数部分*/
6 EXP([Ee][-+]?[0-9]+)
7 /*关键字*/
```

实验代码 3

```
KWYWORS (void|signed|unsigned|short|long|int|float|double|char
              |enum|struct|union|typedef|const|volatile|auto|static|extern
              |register|sizeof|goto|return|break|continue|if|else|switch|case
              |default|do|while|for)
12
     L [a-zA-Z_]
13
     D [0-9]
14
     ID {L}({L}|{D})*
15
     /*整数*/
     NUM 0|[1-9]{D}*
17
     /*专用符号*/
     CHARACTER (#|\+|-|\*|\/|<|"<="|>|">="|"=="
19
                  |"!="|=|;|,|\(|\)|\[|\]|\{|\}|!|',|\")
20
     delim [ \t\n]
21
     whitespace {delim}+
     /*注释*/
     A [/]
24
     B [*]
25
     C [^*/]
26
     NOTES1 "/*"(.|\n)*"*/"
27
     NOTES2 "//".*\n
28
     NOTES ({NOTES1}|{NOTES2})
     /*科学计数法*/
     SCIENTIFIC [0-9]*(\.)?[0-9]*{EXP}?
31
     /*八进制*/
32
     OCTAL 0[0-7]*
33
     /*十六进制*/
34
     HEX O[Xx][0-9a-fA-F]+
35
     /*字符串、字符*/
     STRING \".*\"
37
     CHAR \'.*\'
38
39
     %%
40
     \n {line++;}
41
     {whitespace} {}
42
     {NOTES} {/*注释*/ line++;}
43
     {NUM} {printf("%d\t%20s\t整数\n", line, yytext);}
44
     {HEX} {printf("%d\t%20s\t十六进制\n", line, yytext);}
45
     {OCTAL} {printf("%d\t%20s\t/\进制\n", line, yytext);}
46
     {SCIENTIFIC} {printf("%d\t%20s\t科学计数法\n", line, yytext);}
     {STRING} {printf("%d\t%20s\t字符串\n", line, yytext);}
     {CHAR} {printf("%d\t%20s\t字符\n", line, yytext);}
49
```

分析程序输出结果 4

```
{KWYWORS} {printf("%d\t%20s\t关键字\n", line, yytext);}
50
     {CHARACTER} {printf("%d\t%20s\t专用符号\n", line, yytext);}
51
     {ID} {printf("%d\t%20s\t标识符\n", line, yytext);}
52
     %%
     void main(){
55
         printf("Line\t%20s\tPs\n", "Actor");
56
         printf("\n");
57
         yylex();
     }
     int yywrap(){
         return 1;
61
     }
62
```

# 4 分析程序输出结果

#### 4.1 分析lex2-1.l实验结果

2-1.cpp测试代码如下所示。

第1行,#符号不在子集 $C_1$ 中,不匹配,不输出; include 匹配标识符,输出; <符号匹配专用符号,输出; iostream 匹配标识符,输出; >符号匹配专用符号,输出。

第2行, using 匹配标识符,输出; namespace 匹配标识符,输出; std 匹配标识符输出。

第3行, int 匹配关键字, 输出; main 匹配标识符, 输出; (匹配专用符号, 输出; ) 匹配专用符号, 输出。

第4行, { 匹配专用符号, 输出。

分析程序输出结果 5

第5行, cout 匹配标识符,输出; < 两次匹配专用符号,输出; "不在集合,不匹配,不输出; Hello 匹配标识符,输出;! 不在集合,不匹配,不输出; "不在集合,不匹配,不输出; < 两次匹配专用符号,输出; endl 匹配标识符,输出。; 匹配专用符号输出。

第6行, cout 匹配标识符,输出; < 两次匹配专用符号,输出; "不在集合,不匹配,不输出; Welcome 匹配标识符,输出; to 匹配标识符,输出; c 匹配标识符,输出; + 两次匹配专用符号,输出; ! 不在集合,不匹配,不输出; "不在集合,不匹配,不输出; < 两次匹配专用符号,输出; endl 匹配标识符,输出。; 匹配专用符号输出。

第7行, return 匹配标识符,输出; 0 不匹配定义中的数字,不输出; ; 匹配专用符号,输出。

第8行,} 匹配专用符号,输出。

注:此分析输出仅仅基于实验要求  $C_1$  集合,是常规C语言的一个子集。所有的空格、换行符与注释均不匹配,不输出。

#### 4.2 分析lex2-2.l实验结果

```
#include <iostream>
     using namespace std
     int main(){
          int a 123 023 0x23ff 34830.34E+4
4
          char 'a'
         // comment1
         /*
         comment2
         123 456 int
          */
10
11
          cout<<"Welcome to c++! " endl;</pre>
12
          return 0;
13
     }
14
```

第1行, #匹配专用符号,输出; include 匹配标识符,输出; <符号匹配专用符号,输出; iostream 匹配标识符,输出; >符号匹配专用符号,输出。

第2行, using 匹配标识符,输出; namespace 匹配标识符,输出; std 匹配标识符输出。

第3行, int 匹配关键字, 输出; main 匹配标识符, 输出; (匹配专用符号, 输出; ) 匹配专用符号, 输出; { 匹配专用符号, 输出。

第4行, int 匹配关键字, 输出; a 匹配标识符, 输出; 123 匹配整数, 输出; 023 匹配八进制输出; 0x23ff 匹配十六进制, 输出; 34830.34E+4 匹配科学计数, 输出。

第5行, char 匹配标识符,输出; 'a' 匹配字符,输出。

第6行, cout 匹配标识符,输出; <两次匹配专用符号,输出; "Welcome to c++! "匹配字符串,输出; endl 匹配标识符,输出。; 匹配专用符号输出。

第7行, return 匹配标识符, 输出; 0 匹配整数, 输出; ; 匹配专用符号, 输出。

第8行,} 匹配专用符号,输出。

# 5 在Windows环境下实验

#### 5.1 lex2-1.l实验结果

在Windows环境下lex2-1.1实验编译结果如图1所示,实验结果如图3所示。

```
C:\Users\yuan\Documents\课程\系统软件开发实践\代码\ex2>flex lex2-1.1
C:\Users\yuan\Documents\课程\系统软件开发实践\代码\ex2>cl lex.yy.c
用于 x86 的 Microsoft (R) C/C++ 优化编译器 19.24.28316 版版权所有(C) Microsoft Corporation。保留所有权利。
lex.yy.c
Microsoft (R) Incremental Linker Version 14.24.28316.0
Copyright (C) Microsoft Corporation. All rights reserved.
/out:lex.yy.exe
lex.yy.obj
```

图 1: lex2-1.l编译结果

#### 5.2 lex2-2.l实验结果

在Windows环境下lex2-2.1实验编译结果如图2所示,实验结果如图4所示。

```
C:\Users\yuan\Documents\课程\系统软件开发实践\代码\ex2>flex lex2-2.1
C:\Users\yuan\Documents\课程\系统软件开发实践\代码\ex2>cl lex.yy.c
用于 x86 的 Microsoft (R) C/C++ 优化编译器 19.24.28316 版
版权所有(C) Microsoft Corporation。保留所有权利。
lex.yy.c
lex.yy.c
lex.yy.c(1): warning C4819: 该文件包含不能在当前代码页(936)中表示的字符。请将该文件保存为Microsoft (R) Incremental Linker Version 14.24.28316.0
Copyright (C) Microsoft Corporation. All rights reserved.
/out:lex.yy.exe
lex.yy.obj
```

图 2: lex2-2.l编译结果

| Line                               |           | Actor Ps       |
|------------------------------------|-----------|----------------|
|                                    |           |                |
| 1                                  | include   | Identifier     |
| 1                                  |           | Special symbol |
| 1                                  | iostream  | Identifier     |
| 1                                  |           | Special symbol |
| 2                                  | using     | Identifier     |
| 2                                  | namespace | Identifier     |
| 2                                  | std       | Identifier     |
| 3                                  | int       | Keyword        |
| 3                                  | main      | Identifier     |
| 3                                  |           | Special symbol |
| 3                                  |           | Special symbol |
| 4                                  |           | Special symbol |
| 5                                  | cout      | Identifier     |
| 5                                  |           | Special symbol |
| 5                                  |           | Special symbol |
| 5                                  | Hello     | Identifier     |
| 5                                  |           | Special symbol |
| 5                                  |           | Special symbol |
| 5                                  | endl      | Identifier     |
| 5                                  |           | Special symbol |
| 6                                  | cout      | Identifier     |
| 6                                  |           | Special symbol |
| 6                                  |           | Special symbol |
| 6                                  | Welcome   | Identifier     |
| 6                                  | to        | Identifier     |
| 6                                  |           | Identifier     |
| 6                                  |           | Special symbol |
| 1122233334555555555666666666666677 | end1      | Identifier     |
| 6                                  |           | Special symbol |
| 7                                  | return    | Keyword        |
|                                    |           | Special symbol |
| 8                                  |           | Special symbol |

Developer Command Prompt for VS 2019 \Users\yuan\Documents\课程\系统软件开发实践\代码\e Special symbol Identifier Special symbol Identifier Special symbol Identifier Identifier Identifier Keyword Identifier std int Special symbol Special symbol Special symbol Keyword Identifier a 123 Integer Octal 34830. 34E+4 Scientific notation Char Identifier Special symbol Special symbol String Identifier Special symbol Keyword Integer Special symbol Special symbol

图 3: lex2-1.l实验结果

图 4: lex2-2.l实验结果

# 6 在ubuntu环境下实验

#### 6.1 lex2-1.l实验结果

在ubuntu环境下lex2-1.l实验编译结果如图5所示,实验结果如图7所示。

```
yuan@ubuntu:~/桌面/System So文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
/uan@ubuntu:~/桌面/System Software/ex2$ flex lex2-1.l
/uan@ubuntu:~/桌面/System Software/ex2$ cc lex.yy.c
```

图 5: lex2-1.l编译结果

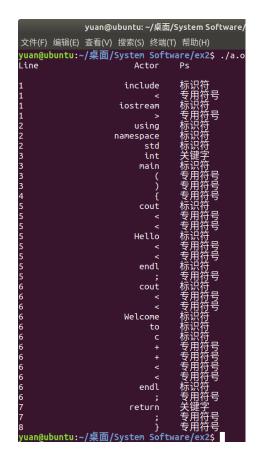
#### 6.2 lex2-2.l实验结果

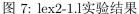
在ubuntu环境下lex2-2.1实验编译结果如图6所示,实验结果如图8所示。

```
yuan@ubuntu:~/桌面/Syste
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
yuan@ubuntu:~/桌面/System Software/ex2$ flex lex2-2.l
yuan@ubuntu:~/桌面/System Software/ex2$ cc lex.yy.c
```

图 6: lex2-2.l编译结果

实验感想





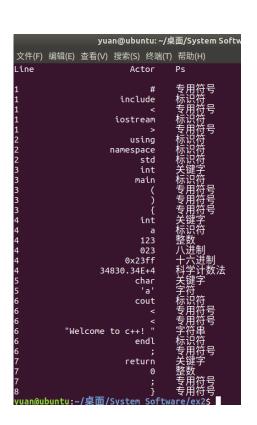


图 8: lex2-2.l实验结果

# 7 实验感想

由于在编译原理课程中有过类似的实验,所以这次实验没有遇到较大的问题,只是遇到了编码错误问题,在 ubuntu 中编写好的代码在 windows 环境中,中文会出现乱码。在 windows 环境下重新编写,任然出现此问题,所以将所有中文输出改为英文,解决了此问题。

虽然我完成了这次实验,但是程序分析的效果还没有达到预期的效果,比如说注释没有算在代码行数中,这在实际的编译过程中是不可能,因为这样会造成出错位置的行数不对,无法快速找到错误的代码。

经过这次实验,我对正则表达式的书写更加清晰,在编写 flex 代码也更加熟练,对于 flex 如何分析代码程序也更加理解。