编译系统课程实验报告

实验 1：词法分析

姓名	卢兑玳	院系	计算机学院系	学号	L170300901
任课教师		指导教师			
实验地点		实验时间			
实验课表现	出勤、表现得分		实验报告得分		实验总分	
	操作结果得分					

一、需求分析

得分

本实验的需求是输入源程序，输出单词符号(token)。即：把构成源程序的字符串转换成“等价的”单词(记号)序列，需要实现的功能有：

1.根据词法规则识别及组合单词

2.对数字常数完成识别

3.查填符号表

4.删去空格字符和注释

5.错误检查

具体而言，词法分析器应该识别：

标识符（由大小写字母、数字以及下划线组成，但必须以字母或者下划线开头）

关键字（①类型关键字：整型、浮点型、布尔型、结构体；②分支结构中的 if 和 else；③循环结构中的 do 和 while；）

运算符（①算术运算符；②关系运算符；③逻辑运算；）

界符（①用于赋值语句的界符，如“=”；②用于句子结尾的界符，如“；”；

常数（无符号整数和浮点数等）

注释（/*……*/形式）

除此之外，还有三项额外需求

（1）要求基于 DFA 技术设计词法分析器。

（2）系统的输入形式：要求能够通过文件导入 FA 转换表和测试用例，可以通过用户界面显示并编辑测试用例。测试用例要涵盖“实验内容”中列出的各类单词，并包含各种单词拼写错误。

（3）系统的输出分为两部分：一部分是打印输出词法分析器的 DFA 转换表。另一部分是打印输出源程序对应的 token 序列

二、文法设计

得分

1. 各类单词的正则文法：

符号：Operator → +|-|*|/|++|+=|-=|*=|/=|%=|%|?|:|^|--|=|==|&&| || | ! | & | | | ~|>|<|>=|<=|>|<|;|,|.|[]|{|}| | \n|\t|\r|#|<<|>>|>>>|letter|

标识符和关键字：

id → letter_ (letter_ | number) *

number → 0|1|2|...|9

letter_ → A|B|…|Z |a | b|…|z|

常数（包括无符号整数、浮点数以及科学计数法形式）：

Numbers = number | oxnumber | oonumber

number = digit+ ((. digit+)| ε) ((E(+|-| ε)digit+)| ε)

digit → 0|1|2|…|9

digit2 → 0|1|2|…|7

digit3 → 0|1|2|…|9|a|b|…|f|

digitt2 → 1|2|…|7

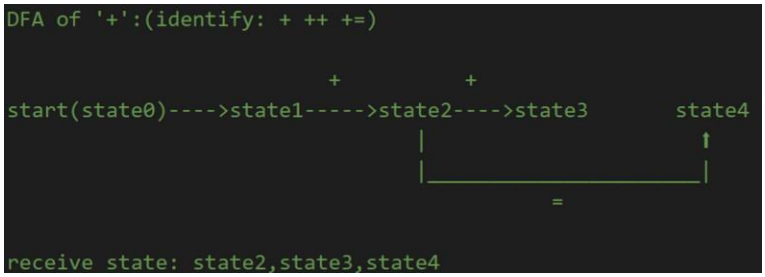
digitt3 → 1|2|…|9|a|b|…|f|

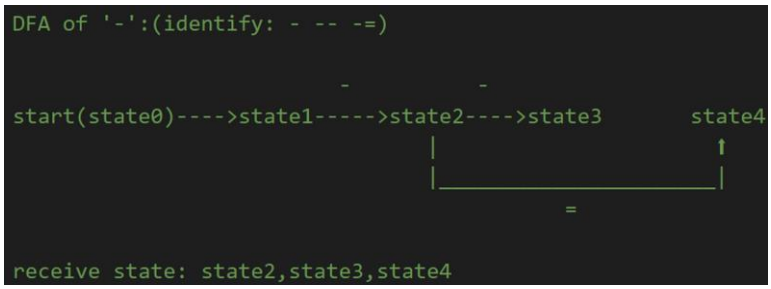
oxnumber = 0x(digitt3(digit3)*|0)

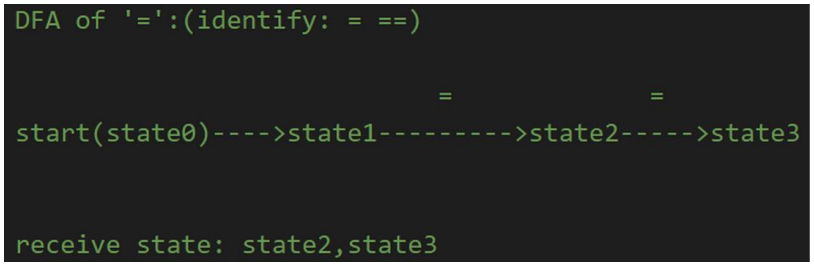
oonumber = o(digitt2(digit2)*|0)

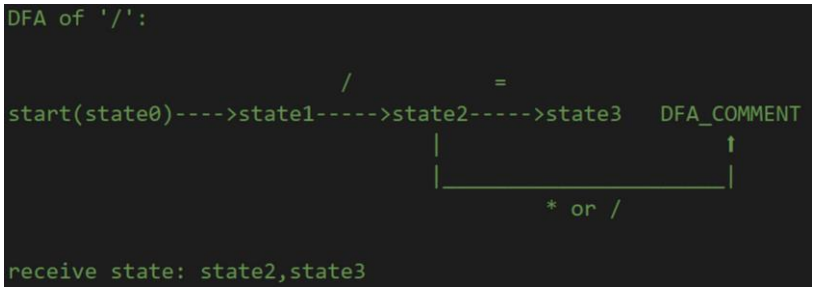
DFA

符号

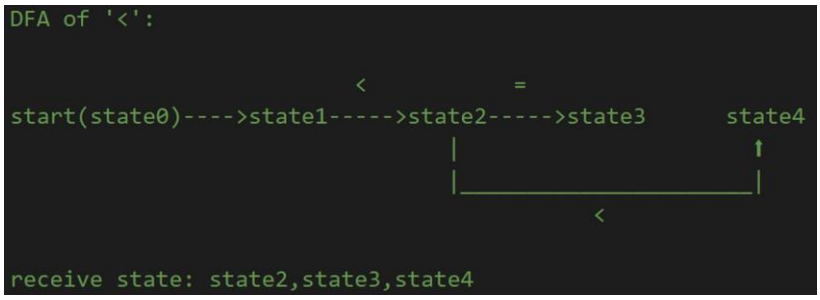


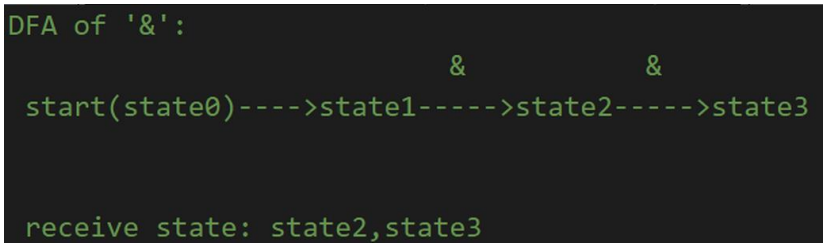


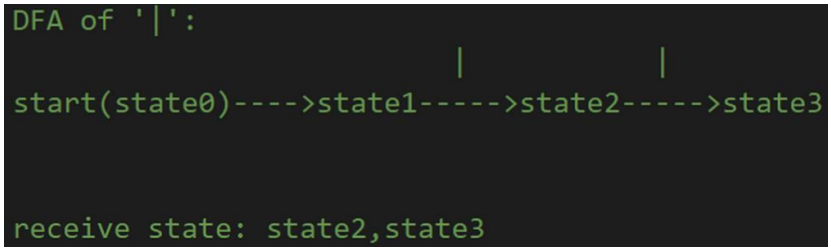

DFA of '=':(identify: = ==)
=
=
start(state0)---->state1--------->state2----->state3
receive state: state2,state3


DFA of '/':
/
=
start(state0)---->state1----->state2----->state3
DFA_COMMENT
* or /
receive state: state2,state3


DFA of '>':(//>: > >= >> >>>(无符号右移))
>
=
>
start(state0)---->state1----->state2----->state3
state4------->state5
>
receive state: state2,state3,state4,state5

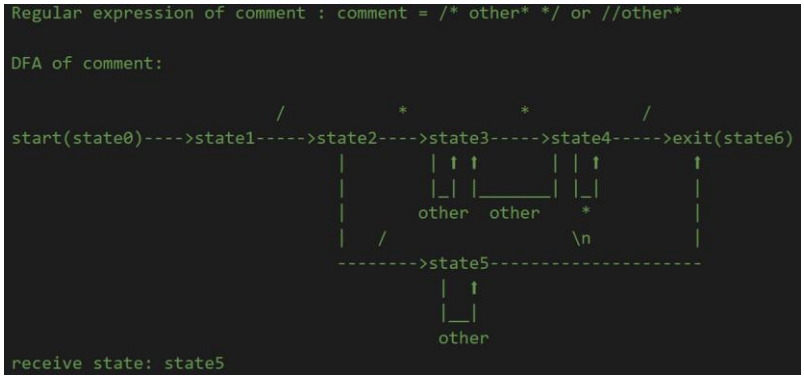

DFA of '<':
<
=
start(state0)---->state1----->state2----->state3
state4
<
receive state: state2,state3,state4


DFA of '&':
&
&
start(state0)---->state1----->state2----->state3
receive state: state2,state3

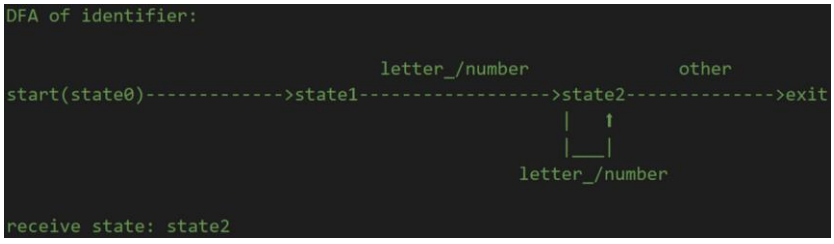

DFA of '|':
|
|
start(state0)---->state1----->state2----->state3
receive state: state2,state3


Regular expression of string : string = " other* "
DFA of string:
"
"
start(state0)
state1
state2
exit(state3)
other(except '\n')
receive state: state3

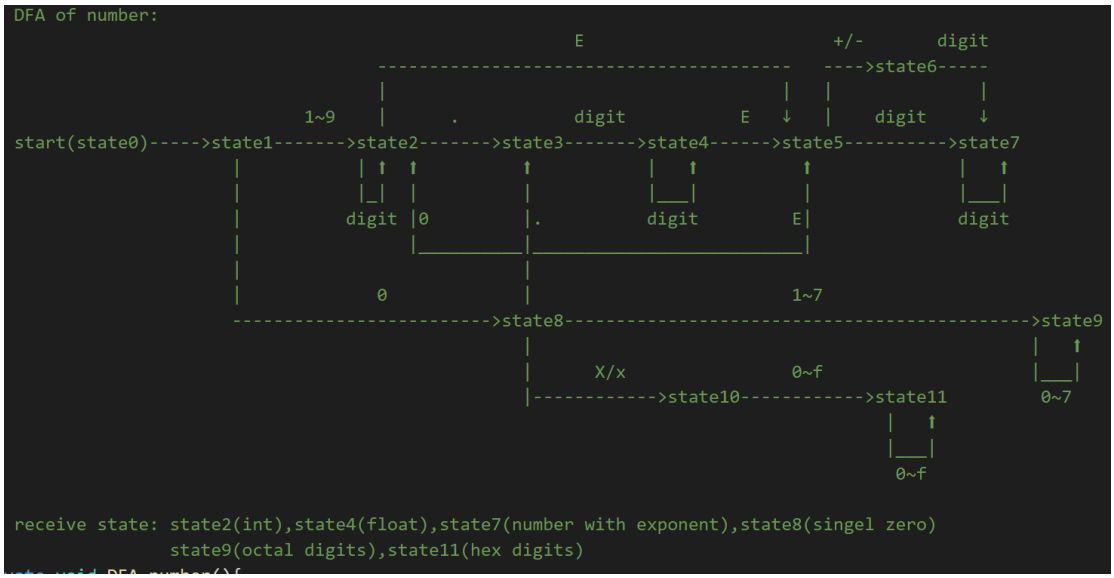

识别字符常量的DFA(仅支持一部分转义字符 '\n' '\t' '\r')
Regular expression of char : char = ' other '
DFA of char:
'
other
'
start(state0)
state1
state2
state3
exit(state5)
r/t/n
state4
receive state: state3


Regular expression of comment : comment = /* other* */ or //other*
DFA of comment:
/
*
*
/
start(state0)
state1
state2
state3
state4
exit(state6)
other
other
*
/
\n
state5
other
receive state: state5

标识符和关键字：


DFA of identifier:
letter_/number
other
start(state0)
state1
state2
exit
letter_/number
receive state: state2

常数：



三、系统设计	得分	

要求：分为系统概要设计和系统详细设计。

（1）系统概要设计：给出必要的系统宏观层面设计图，如系统框架图、数据流图、功能模块图等以及相应的文字说明。

（2）系统详细设计：对如下工作进行展开描述

✓ 核心数据结构的设计

✓ 主要功能函数说明

✓ 程序核心部分的程序流程图

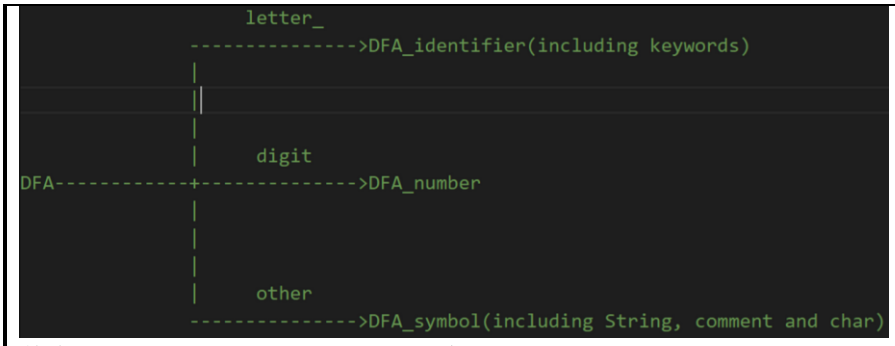
系统概要设计：

本设计中有两个类，分别是实现 GUI 功能和词法分析的功能，用户通过运行 GUI 类的函数，可以调用

对于 DFA 的设计，如下图所示：

程序运行时读入源程序，经过词法分析器，输入单词序列表。

DFA 由如下几部分构成。



其中我们小组定义了如下几种关键字：

```
this.KeyWord_List = List.of(
                    "int"    , "float"    , "double" , "boolean"  , "true"    ,
                    "false"  , "include"  , "char"   , "if"       , "else"    ,
                    "do"     , "while"    , "break"  , "continue" , "for"     ,
                    "main"   , "void"     , "printf" , "class"    , "scanf"   ,
                    "return" , "char"     , "public" , "static"   , "private" );
```

2. 系统的详细设计：s

Scaning 类：实现词法分析功能。

Compiler_gui 类：实现 GUI 操作。

核心数据结构设计：

本组对核心数据结构做了如下设计：

```
private  char  c;                          //当前 DFA 正在处理的字符
        private  int  row;      //当前 DFA 正在处理的字符所处的行号，用于报告错误信息
        private  int  index;        //当前 DFA 正在处理的字符在该行的索引，用于字符指针的前移和后退
        //output variable
        private  List<String[]> token;      //词法分析的分析结果，以{种别码,属性值}的格式存储
      private  List<String[]> error; //词法分析的错误信息，以{错误行号,错误信息}的格式存储
        //input variable
        private  List<String> program;        //被分析的程序文本
        //constant
        private  List<String> KeyWord_List;     //关键字列表，用于识别关键字
```

用一个键值对列表来存储整个自动机：

```
List<Map<String,String>> dfa_map  =new  ArrayList<Map<String,String>>();
```

主要函数说明：

通过读第一个字母判断是那类自动机。

```
public  void  DFA() {
getNextChar();
for(char  firstChar  =  c;firstChar  !=  '$';firstChar = c){
      if(isLetter_(firstChar)){
            DFA_identifier();
      }
```

```
                else if(isDigit(firstChar)){
                        DFA_number();
                }
                else{
                        DFA_symbol();
                }
        }
    }
```

对各种异常进行处理，根据异常的具体类型选择终止，指针前移或后移。比如：像字符串中插入\n，我们小组采取了跳过此符号，继续往后读的方法。字符类型如果有多于一个字符，将会指针前移，只读第一个字符。

```
    private void exception_handler(int exc_code,String s) {
        //获取一个完整的错误单词
        //Identifier format error
        if(exc_code == 1){
            while(isLetter_(c) || isDigit(c)){
                s += c;
                getNextChar();
            }
            error.add(new String[]{row+"", "Incorrect identifier format:"+s});
        }
        ‘’ ‘’ 此部分代码省略，详情见源程序 ‘’ ‘’
        //Hex Number format error:
        else if(exc_code == 11){
            while(isLetter_(c) || isDigit(c)){
                s += c;
                getNextChar();
            }
            error.add(new String[]{row+"", "Incorrect Octal Number format:"+s});
        }
    }
```

此函数作为识别标识符和关键字 DFA:

```
private void DFA_identifier() {
        int state = 0;
        String indentifier = "";
        while (true) {
            // start(state0)
            if (state == 0)
                state = 1;
            // state1
        ‘’ ‘’ 此部分代码省略，详情见源程序 ‘’ ‘’
                if(KeyWord_List.contains(indentifier)){
                    receive_handler("关键字",indentifier);
                }
```

```
                    return;
                }
            }
        }
```

由于 symbol 种类过多，因此在设计时为识别 symbol 的 DFA 设计了许多子函数，

```
private  void  DFA_plus()
```

```
private  void  DFA_minus()
```

识别+= ++ + - -- -=

```
private  void  DFA_mult_percent_not()
```

```
identify:* % ! *= %= !=
```

```
private  void  DFA_equal()
```

```
identify: = ==
```

```
private  void  DFA_slash()
```

```
identify: /
```

```
private  void  DFA_greater()
```

```
identify: >: > >= >> >>>
```

```
private  void  DFA_less()
```

```
identify: <: < <= <<
```

```
private  void  DFA_and()
```

```
identify: && &
```

```
private  void  DFA_or()
```

```
identify: || |
```

```
private  void  DFA_string()
```

```
identify: string
```

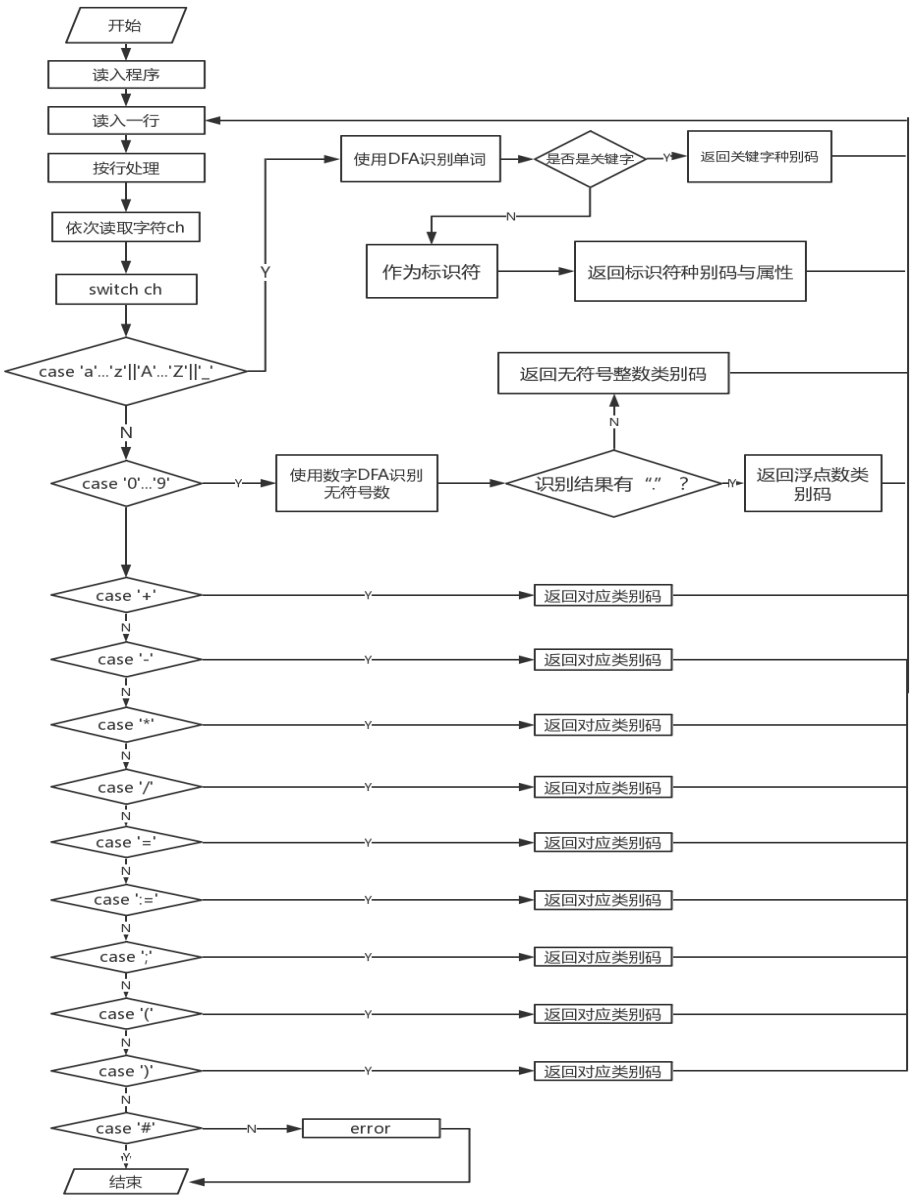
```
private  void  DFA_char()
```

```
identify: char(including \t \r \n)

private  void  DFA_comment()

identify: comment
```

程序核心部分流程图如下（GUI 并未包括在核心部分之内）：



四、系统实现及结果分析

得分

要求：对如下内容展开描述。

（1） 系统实现过程中遇到的问题；

（2） 针对某测试程序输出其词法分析结果；

（3） 输出针对此测试程序对应的词法错误报告；

（4） 对实验结果进行分析。

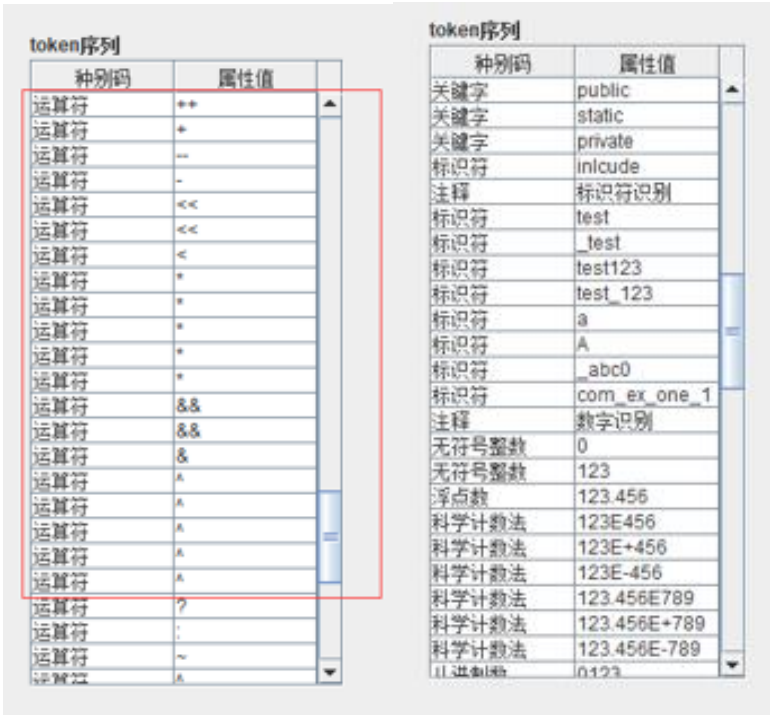
注：其中的测试样例自行产生。

（1）错误回收时格式出错，解决时，把“’”换行情况忽略，并判断为错，两个错误的 char 中间一个标识符吧

对于===这种符号不知如何识别，最终解决方法为根据最长匹配原则，识别成一个连等号和一个赋值号。

有些异常确实永远不会被触发，原因是有些 DFA 能选中他们的时候 就说明已经验证过第一个字符了，但是他们的格式错误也只会由第一个字符的错误触发

（2）将 test.txt 作为输入，可以得到如下图所示的输出结果，可以看出，已经成功地实现了对 token 序列的输出。注释部分已经全部删除。对于各种运算符、标识符、数据都做出了正确的判断。

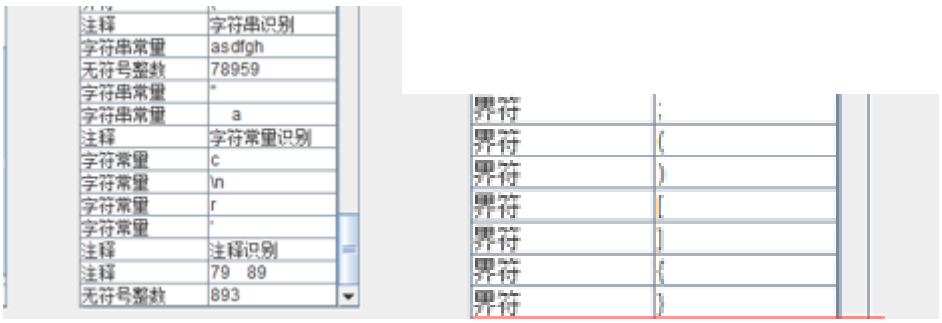


token序列

种别码	属性值
运算符	++
运算符	+
运算符	--
运算符	-
运算符	<<
运算符	<<
运算符	<
运算符	*
运算符	*
运算符	*
运算符	*
运算符	*
运算符	&&
运算符	&&
运算符	&
运算符	^
运算符	^
运算符	^
运算符	^
运算符	^
运算符	?
运算符	:
运算符	~

token序列

种别码	属性值
关键字	public
关键字	static
关键字	private
标识符	inlcude
注释	标识符识别
标识符	test
标识符	_test
标识符	test123
标识符	test_123
标识符	a
标识符	A
标识符	_abc0
标识符	com_ex_one_1
注释	数字识别
无符号整数	0
无符号整数	123
浮点数	123.456
科学计数法	123E456
科学计数法	123E+456
科学计数法	123E-456
科学计数法	123.456E789
科学计数法	123.456E+789
科学计数法	123.456E-789



注释	字符串识别
字符串常量	asdfgh
无符号整数	78959
字符串常量	"
字符串常量	a
注释	字符常量识别
字符常量	c
字符常量	\n
字符常量	r
字符常量	'
注释	注释识别
注释	79 89
无符号整数	893

界符	;
界符	(
界符	)
界符	[
界符	]
界符	{
界符	}

（3）对于 test 异常中的各种设置的异常情况也进行了输出，如下图所示：

Error 具体信息如下所示：

16

Incorrect number format:1test

24

Incorrect Octal Number format:01238

25

Incorrect Octal Number format:0x123g8

26

Incorrect number format:123.

27

Incorrect number format:123.1B

28

Incorrect number format:123.1EB

29

Incorrect number format:123.456E-789B

30

Incorrect number format:123.456E-B

31

Incorrect number format:123.456E-

32

Incorrect Octal Number format:0xx

47

" is not closed:

50

Incorrect char format,mismatch of another '

53

Incorrect char format:\a

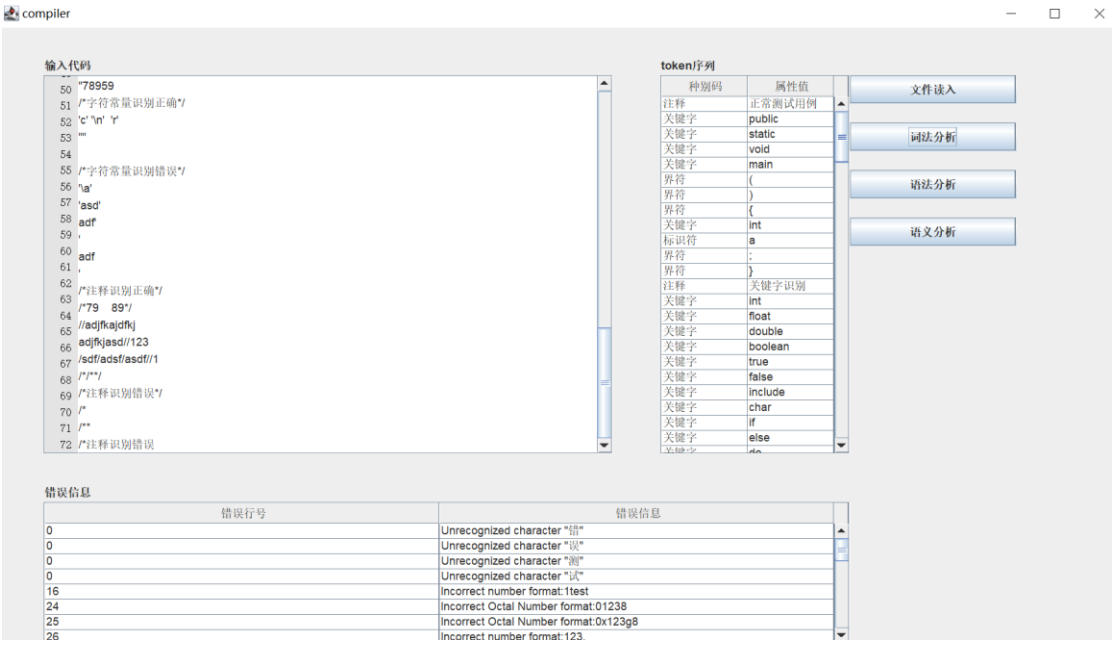
54

Incorrect char format:asd

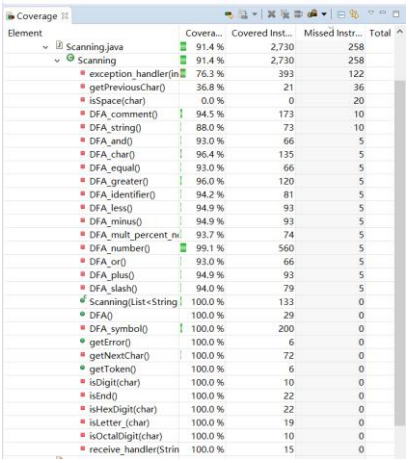
55

Incorrect char format,mismatch of another '
56
Incorrect char format,mismatch of another '
58
Incorrect char format,mismatch of another '
66
Comment is not closed,mismatch another */
67
Comment is not closed,mismatch another */
68
Comment is not closed,mismatch another */
68
Unrecognized character "注"
68
Unrecognized character "释"
68
Unrecognized character "识"
68
Unrecognized character "别"
68
Unrecognized character "错"
68
Unrecognized character "误"

（4）整个 GUI 运行结果如下图所示：



除此之外，对程序进行了优化，使得程序覆盖度高达 91.4%


Coverage
Element
Covera...
Covered Inst...
Missed Instr...
Total
Scanning.java 91.4 % 2,730 258
Scanning 91.4 % 2,730 258
exception_handler(in 76.3 % 393 122
getPreviousChar() 36.8 % 21 36
isSpace(char) 0.0 % 0 20
DFA_comment() 94.5 % 173 10
DFA_string() 88.0 % 73 10
DFA_and() 93.0 % 66 5
DFA_char() 96.4 % 135 5
DFA_equal() 93.0 % 66 5
DFA_greater() 96.0 % 120 5
DFA_identifier() 94.2 % 81 5
DFA_less() 94.9 % 93 5
DFA_minus() 94.9 % 93 5
DFA_mult_percent_n 93.7 % 74 5
DFA_number() 99.1 % 560 5
DFA_or() 93.0 % 66 5
DFA_plus() 94.9 % 93 5
DFA_slash() 94.0 % 79 5
Scanning(List<String 100.0 % 133 0
DFA() 100.0 % 29 0
DFA_symbol() 100.0 % 200 0
getError() 100.0 % 6 0
getNextChar() 100.0 % 72 0
getToken() 100.0 % 6 0
isDigit(char) 100.0 % 10 0
isEnd() 100.0 % 22 0
isHexDigit(char) 100.0 % 22 0
isLetter_(char) 100.0 % 19 0
isOctalDigit(char) 100.0 % 10 0
receive_handler(Strin 100.0 % 15 0

指导教师评语：

日期：