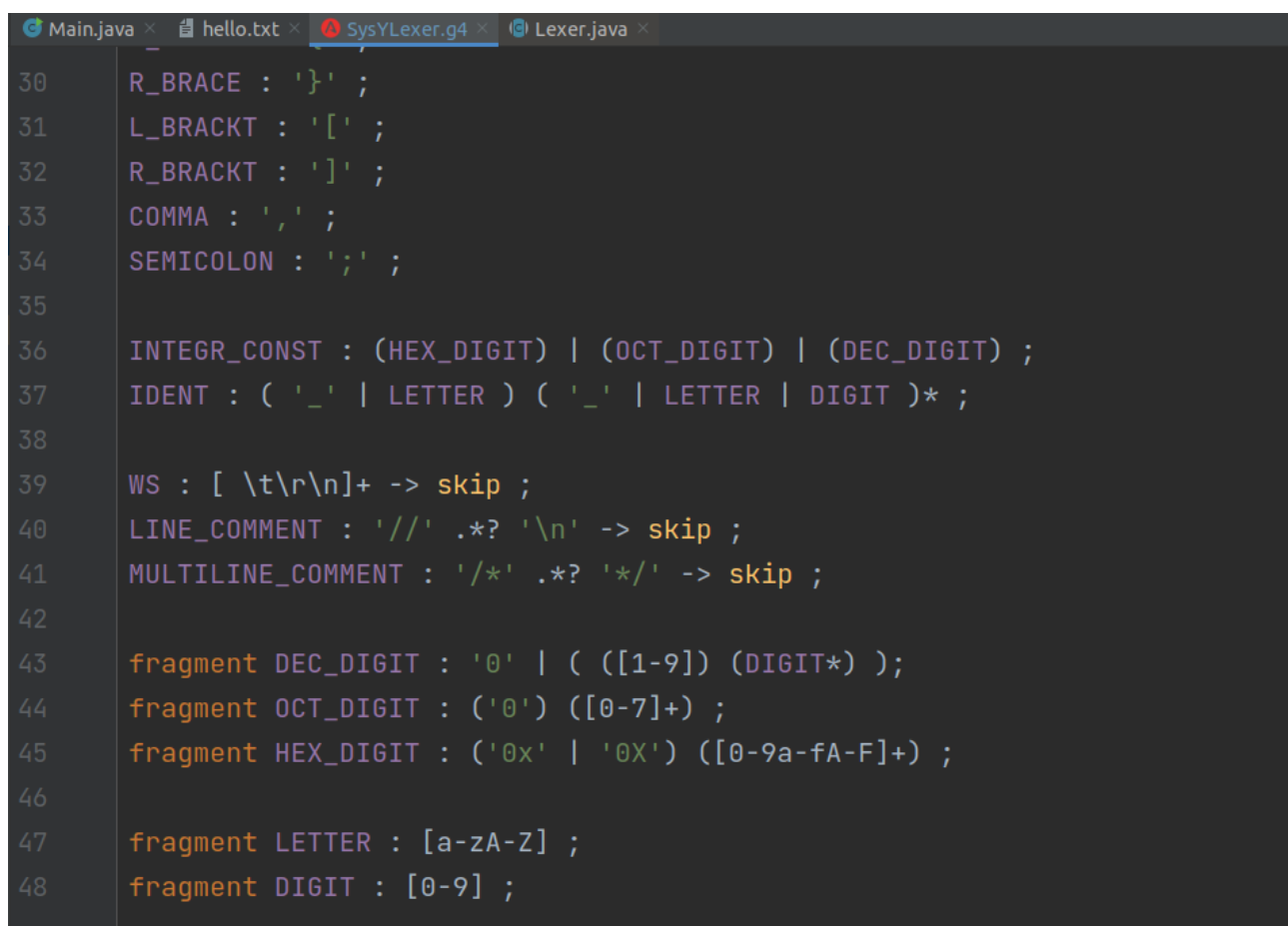


Lab1

201250150 郑寒超

一、实验思路

本次实验主要是写词法分析器，对语法没有要求，所以在 `.g4` 文件中只要写对应的词法即可。



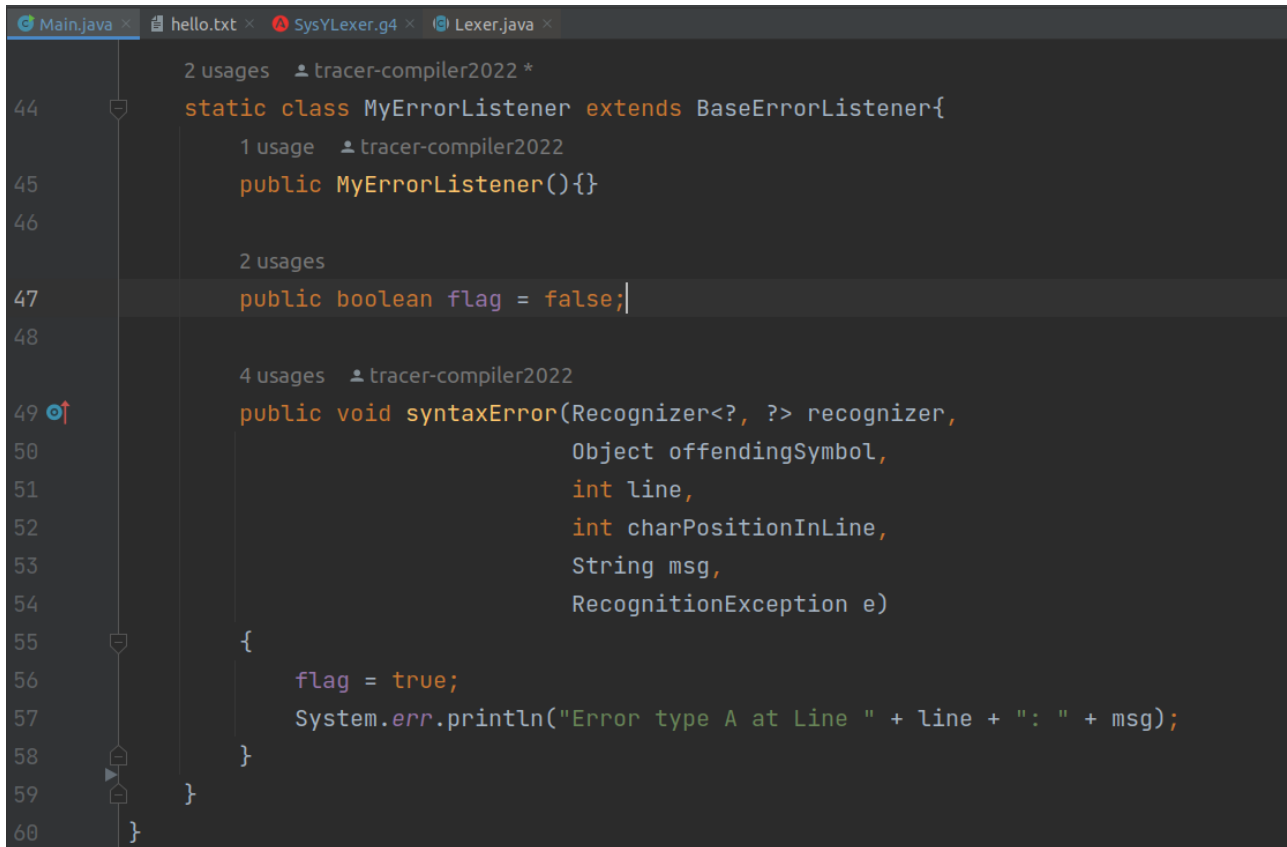
```
30 R_BRACE : '}' ;
31 L_BRACKET : '[' ;
32 R_BRACKET : ']' ;
33 COMMA : ',' ;
34 SEMICOLON : ';' ;
35
36 INTEGR_CONST : (HEX_DIGIT) | (OCT_DIGIT) | (DEC_DIGIT) ;
37 IDENT : ( '_' | LETTER ) ( '_' | LETTER | DIGIT )* ;
38
39 WS : [ \t\r\n ]+ -> skip ;
40 LINE_COMMENT : '//' .*? '\n' -> skip ;
41 MULTILINE_COMMENT : '/*' .*? '*/' -> skip ;
42
43 fragment DEC_DIGIT : '0' | ( ([1-9]) (DIGIT*) );
44 fragment OCT_DIGIT : ('0') ([0-7]+) ;
45 fragment HEX_DIGIT : ('0x' | '0X') ([0-9a-fA-F]+) ;
46
47 fragment LETTER : [a-zA-Z] ;
48 fragment DIGIT : [0-9] ;
```

然后是在 `Main` 函数中处理逻辑，相应的代码在手册中都已给出，主要需要实现是

1. 自己的 `MyErrorListener`
2. 处理 `tokens` 的输出

MyErrorListener

首先需要知道代码是如何调用这个类的，所以我在实现的时候先将 **BaseErrorListener** 中的函数全部拷贝到 **MyErrorListener** 中，然后对每个函数增加一个 **System.out.println(...)**，这样就可以知道在出现错误是会调用哪一个函数，最后发现是 **syntaxError** 函数，然后就可以去 **Lexer** 类中找到对应调用位置，查看是如何调用这个函数，最后编写这个函数。

A screenshot of an IDE window showing the implementation of the `MyErrorListener` class. The class is a static class that extends `BaseErrorListener`. It has a constructor `MyErrorListener()` and a static variable `boolean flag = false;`. The `syntaxError` method is implemented with parameters `Recognizer<?, ?> recognizer`, `Object offendingSymbol`, `int line`, `int charPositionInLine`, `String msg`, and `RecognitionException e`. Inside the method, `flag` is set to `true` and a message is printed: `System.err.println("Error type A at Line " + line + ": " + msg);`. The IDE shows line numbers from 44 to 60. The tabs at the top are `Main.java`, `hello.txt`, `SysYLexer.g4`, and `Lexer.java`.

```
2 usages  1 tracer-compiler2022 *
44         static class MyErrorListener extends BaseErrorListener{
45             1 usage  1 tracer-compiler2022
46                 public MyErrorListener(){}
47
48             2 usages
49                 public boolean flag = false;
50
51             4 usages  1 tracer-compiler2022
52                 public void syntaxError(Recognizer<?, ?> recognizer,
53                                         Object offendingSymbol,
54                                         int line,
55                                         int charPositionInLine,
56                                         String msg,
57                                         RecognitionException e)
58                 {
59                     flag = true;
60                     System.err.println("Error type A at Line " + line + ": " + msg);
61                 }
62             }
```

tokens 输出

首先确定 `getAllTokens` 函数返回的类型=>`List<? extends Token>`

在遍历 `token` 列表时调用 `Token` 类中对应函数来获取行号、类型等相关信息，再做输出。

注意：出现错误时，只输出错误信息。

```
23 List<? extends Token> allTokens = sysYLexer.getAllTokens();
24 // occur error
25 if (myErrorListener.flag) return;
26
27 for (Token token : allTokens) {
28     int typeId = token.getType();
29     String type = SysYLexer.VOCABULARY.getSymbolicName(typeId);
30     String text = token.getText();
31     if (typeId == SysYLexer.INTEGR_CONST){
32         if (text.equals("0")) {
33             text = "0";
34         } else if (text.startsWith("0x") || text.startsWith("0X")){
35             text = Integer.parseInt(text.substring(beginIndex: 2), radix: 16) + "";
36         } else if (text.startsWith("0")){
37             text = Integer.parseInt(text.substring(beginIndex: 1), radix: 8) + "";
38         }
39     }
40     int lineNo = token.getLine();
41     System.err.println(type + " " + text + " at Line " + lineNo + ".");
42 }
43 }
```

二、碰到的问题

1、输出格式

注意输出格式要严格按照手册要求。

2、INTEGR_CONST的处理

注意八进制数和十六进制数的处理，输出 `token` 时是需要输出其十进制值。