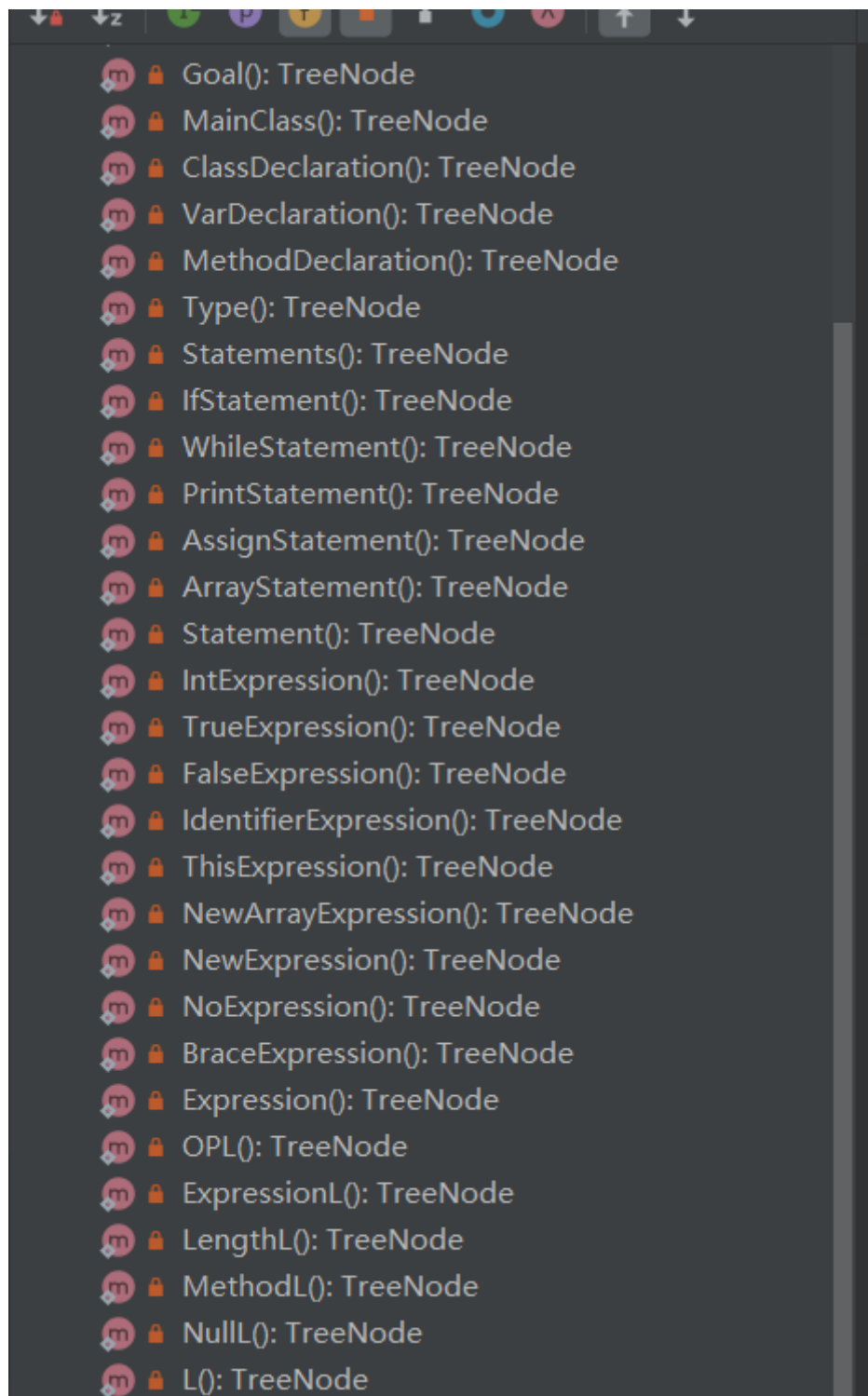


注意事项

1. 在词法分析中未分析的浮点数和1flow1这两中类型在语法分析中处理完成
2. 语法分析时，和IDEA中的提示方式类似，只能提示出第一个语法分析中出现的错误，所以每次查看的时候只看syntaxErrorOut.txt中的第一行，之后更改后，要重新运行词法分析程序，之后在运行语法分析程序。
3. 测试结果在测试文件中，分别显示初始文件测试结果，文件夹：根据错误提示更改过后的正确的测试文件中时改正所有错误后的正确的分析结果，有正确的语法树。
4. 在语法树的输出中，不同的缩进代表树的层词，数的节点只显示了identifier和integerLiteral这两种类型，和最后的EOF。

算法

利用了递归下降的算法，一下是各个函数



在一些局部还利用了LL (1) 的思想

例如：

```
type.setStatement(Statement.TYPE);
getNextWord();
TokenType thisNextToken=nextToken;
switch (currentToken) {
    case INT:
        if (thisNextToken == TokenType.LBRACKET) {
            match(TokenType.INT);
            match(TokenType.LBRACKET);
            match(TokenType.RBRACKET);
        } else {
            match(TokenType.INT);
        }
}
```

```
getNextWord();
TokenType thisNextToken=nextToken;
switch (currentToken) {
    case LBRACE:
        statement.children.add(Statements());
        break;
    case IF:
        statement.children.add(IfStatement());
        break;
    case WHILE:
        statement.children.add(WhileStatement());
        break;
    case SYSTEMOUTPRINTLN:
        statement.children.add(PrintStatement());
        break;
    case IDENTIFIER:
        if (thisNextToken == TokenType.EQUAL) {
            statement.children.add(AssignStatement());
        }
        if (thisNextToken == TokenType.LBRACKET) {
            statement.children.add(ArrayStatement());
        }
        break;
}
```

一部分测试结果

text8中:

The 15 Token : <IDENTIFIER , MyClient> is wrong , The expected token type is "RBRACE".

```
13      ), RPAREN
14      {, LBRACE
15      MyClient, IDENTIFIER
16      mc, IDENTIFIER
17      ;, SEMICOLON
```

```

MainS.java x tokenOut8.txt x test8.txt x SyntaxAnalysis.java x Main.java x
class WhatHappen {
    public static void main(String[] args) {
        MyClient mc;
        int handle;

        mc = new MyClient();
        while(!false){
            handle = mc.start(10, 10);
        }
    }
}

```

The 22 Token : <WHILE , while> is wrong , The expected token type is "RBRACE".

```

20    ), RPAREN
21    ; , SEMICOLON
22    while , WHILE
23    ( , LPAREN
24    ! , EXCLAMATION
25    false , FALSE
26    ) , RPAREN

```

```

1    class WhatHappen {
2        public static void main(String[] args) {
3
4
5        mc = new MyClient();
6        while(!false){
7            handle = mc.start(10, 10);
8        }
9    }
10 }

```

The 93 Token : <RBRACE , }> is wrong , The expected token type is "RETURN".

```

88      .,FULLSTOP
89      Juggling,IDENTIFIER
90      (,LPAREN
91      ),RPAREN
92      ;;SEMICOLON
93      },RBRACE
94      public,PUBLIC
95      int,INT

```

```

    public int run(int host, int port){
        int handle;
        handle = this.Juggling();
    }

```

The 312 Token : <RBRACE , > is wrong , The expected token type is "RETURN".

```

309      false,FALSE
310      ),RPAREN
311      ;;SEMICOLON
312      },RBRACE
313      },RBRACE
314      class,CLASS

```

```

5
6     public boolean HolyLight(){
7         in = in + 1;
8         out = out - 1;
9         System.out.println(false);
10
11     }
12 }

```

32321
 The 319 Token : <IDENTIFIER , extend> is wrong , The expected token type is "LBRACE".
 The 320 Token : <LBRACE , {> is wrong , The expected token type is "LBRACE".

```

317     class, CLASS
318     MyClient, IDENTIFIER
319     extend, IDENTIFIER
320     {, LBRACE
321     public, PUBLIC
322     int, INT

```

```

83
84     class MyClient extend{
85
86         public int start(int host, int port){
87             int handle;
88             handle = this.run()
89             return handle;
90         }
91     }
92

```

The 343 Token : <RETURN , return> is wrong , The expected token type is "SEMICOLON".

```

339      .,FULLSTOP
340      run,IDENTIFIER
341      (,LPAREN
342      ),RPAREN
343      return,RETURN
344      handle,IDENTIFIER
345      ;,SEMICOLON

```

```

83
84 class MyClient extends Client{
85
86     public int start(int host, int port){
87         int handle;
88         handle = this.run()
89         return handle;
90     }
91 }

```

text12:

```

The 224 Token : <FULLSTOP , .> is wrong , The expected token type is "SEMICOLON".

```

```

=,EQUAL
2,INTEGERLITERAL
.|,FULLSTOP
0,INTEGERLITERAL
;,SEMICOLON
tmp2,IDENTIFIER

```

```

59         int tmp1;
60         int tmp2;
61         int tmp3;
62         tmp1 = 2.0;
63         tmp2 = 3;
64         tmp3 = 4;

```

text13:

The 53 Token : <NEW , new> is wrong , The expected token type is "SEMICOLON".

```

50         ,,SEMICOLON
51         messagelist,IDENTIFIER
52         =,EQUAL
53         new,NEW
54         [,LBRACKET

```

```

12     public boolean init(){
13         index = 0;
14         messagelist = new [10];
15         in = 0;

```

text15:

The 28 Token : <STATIC , static> is wrong , The expected token type is "IDENTIFIER".

```

26     {,LBRACE
27     public,PUBLIC
28     static,STATIC
29     void,VOID

```



```

7      class NewHappend {
8          public static void main(String[] args) {
9              MyClient mc;
10             int handle;
11
12             mc = new MyClient();
13             while(!false){
14                 handle = mc.start(488, 388);
15             }
16         }
17     }

```

text16:

The 9 Token : <INT , int> is wrong , The expected token type is "STRING".

```

7      main,MAIN
8      (,LPAREN
9      int,INT
10     argc,IDENTIFIER
11     ),RPAREN
12     {,LBRACE
13     mc,IDENTIFIER

```

```

1      class WhatHappen {
2          public static void main(int argc) {

```

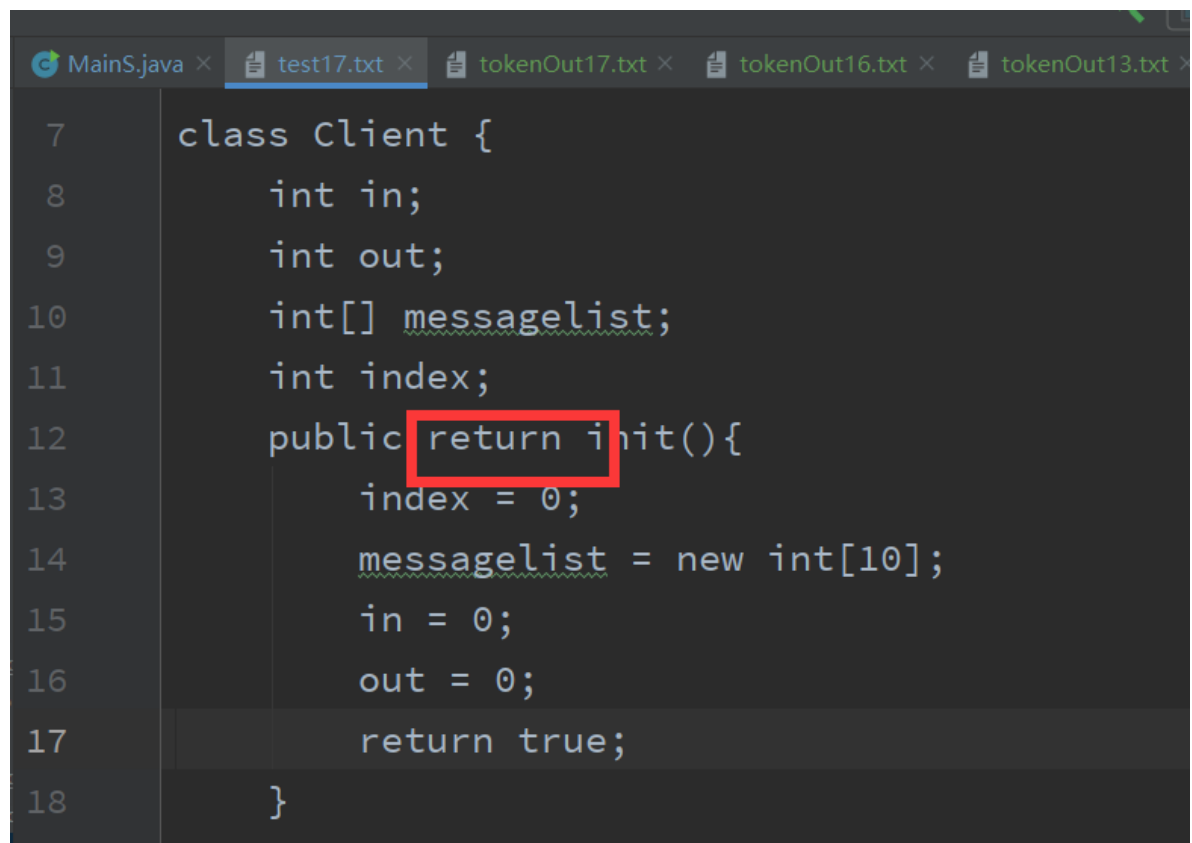
17

The 42 Token : <RETURN , return> is wrong , The expected token type is "IDENTIFIER".

```

39     index,IDENTIFIER
40     ;,SEMICOLON
41     public,PUBLIC
42     return,RETURN
43     init,IDENTIFIER
44     (,LPAREN

```



```
7 class Client {
8     int in;
9     int out;
10    int[] messagelist;
11    int index;
12    public return init(){
13        index = 0;
14        messagelist = new int[10];
15        in = 0;
16        out = 0;
17        return true;
18    }
```

text18中:

The 192 Token : <RETURN , return> is wrong , The expected token type is "ELSE".

```

177 ;,SEMICOLON
178 if,IF
179 (,LPAREN
180 0,INTEGERLITERAL
181 <,LESSTHAN
182 messagelist,IDENTIFIER
183 .,FULLSTOP
184 length,LENGTH
185 ),RPAREN
186 {,LBRACE
187 flag,IDENTIFIER
188 =,EQUAL
189 false,FALSE
190 ;,SEMICOLON
191 },RBRACE
192 return,RETURN
193 flag,IDENTIFIER

```

```

4 public boolean isVoid(){
5     boolean flag;
6     if(0 < messagelist.length){
7         flag = false;
8     }
9     return flag;
10 }

```

```

25      (,LPAREN
26      ),RPAREN
27      ;;SEMICOLON
28      while,WHILE
29      (,LPAREN
30      !,EXCLAMATION
31      false, FALSE

```

```

1      class WhatHappen {
2          public static void main(String[] args) {
3              mc = new MyClient();
4              while(!false);
5          }
6      }

```

text20:

The 65 Token : <SEMICOLON , ;> is wrong , The expected token type is "Expression".

```

60      =,EQUAL
61      0,INTEGERLITERAL
62      ;;SEMICOLON
63      out,IDENTIFIER
64      =,EQUAL
65      ;;SEMICOLON

```

```

2          public boolean init(){
3              index = 0;
4              messagelist = new int[10];
5              in = 0;
6              out =|;
7              return true;
8          }

```

text21:

The 147 Token : <EQUAL , => is wrong , The expected token type is "RBRACKET".

```
144 messagelist, IDENTIFIER
145 [, LBRACKET
146 index, IDENTIFIER
147 =, EQUAL
148 tmp, IDENTIFIER
149 ;, SEMICOLON
150 index, IDENTIFIER
```

```
}
if(index < 10){
    messagelist[index = tmp;
    index = index + 1;
}
```

text22:

The 35 Token : <IDENTIFIER , messagelist> is wrong , The expected token type is "RBRACKET".

```
31 out, IDENTIFIER
32 ;, SEMICOLON
33 int, INT
34 [, LBRACKET
35 messagelist, IDENTIFIER
36 ;, SEMICOLON
37 int, INT
```

```
class Client {
    int in;
    int out;
    int[ messagelist;
    int index;
```

text23:

```
The 210 Token : <SEMICOLON , ;> is wrong , The expected token type is "IDENTIFIER".
```

```
208 {, LBRACE
209 boolean, BOOLEAN
210 ;, SEMICOLON
211 int, INT
```

```
public int Juggling(){
    boolean;
    int tmp1;
```

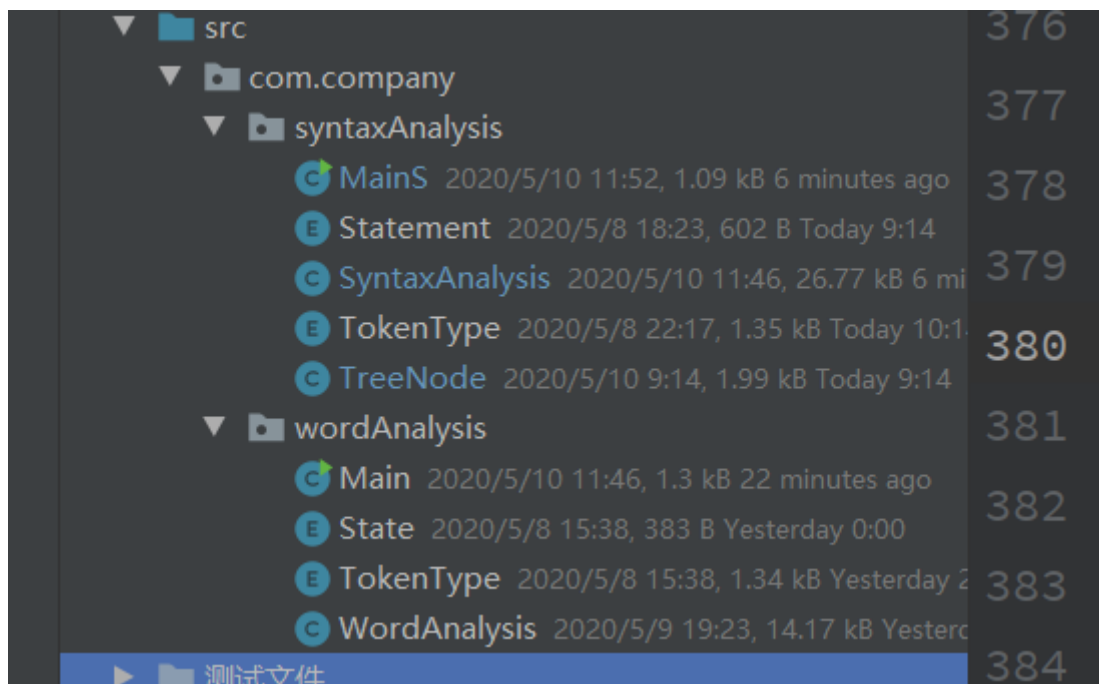
再text23中自己添加的:

```
The 337 Token : <INTEGERLITERAL , 1> is wrong , The expected token type is "IDENTIFIER".
```

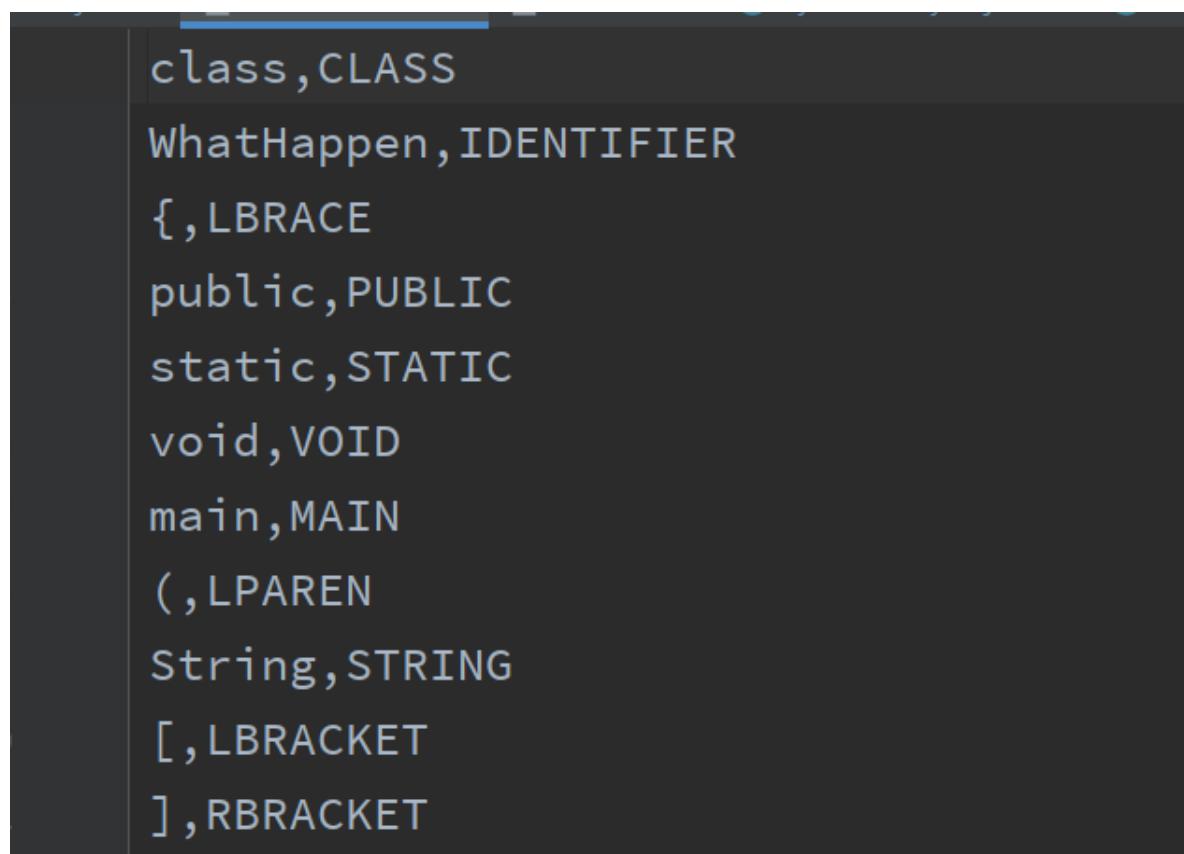
```
336 int, INT
337 1, INTEGERLITERAL
338 flow1, IDENTIFIER
339 ;, SEMICOLON
```

```
81 class MyClient extends Client{
82
83     public int start(int host, int port){
84         int handle;
85         int lflow1;
86         handle = this.run();
87         return handle;
88     }
```

代码结构



在第一次实验的基础上对词法分析的结果进行了一些修改，输出的样子为



syntaxAnalysis

Statement是所有语法树的根节点的状态，

SyntaxAnalysis：语法分析类

TokenType 是词的类型

TreeNode是语法树的结构