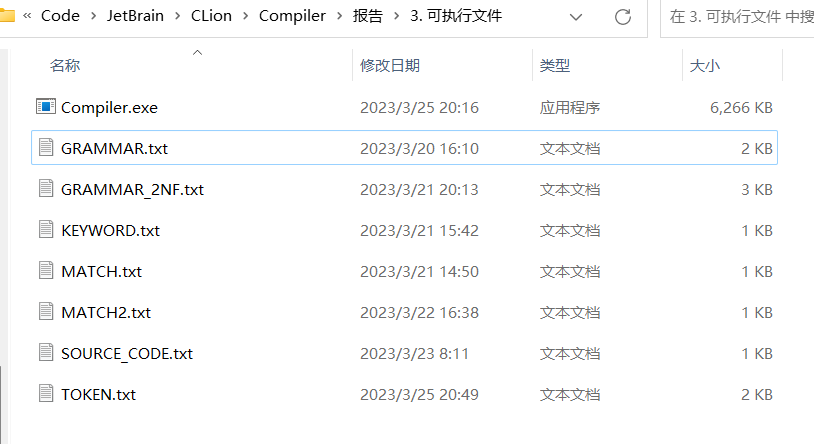
使用实例

运行前准备：编译好的程序，需要的各txt文件



编辑SOURCE\_CODE.txt文件，输入以下示例代码：

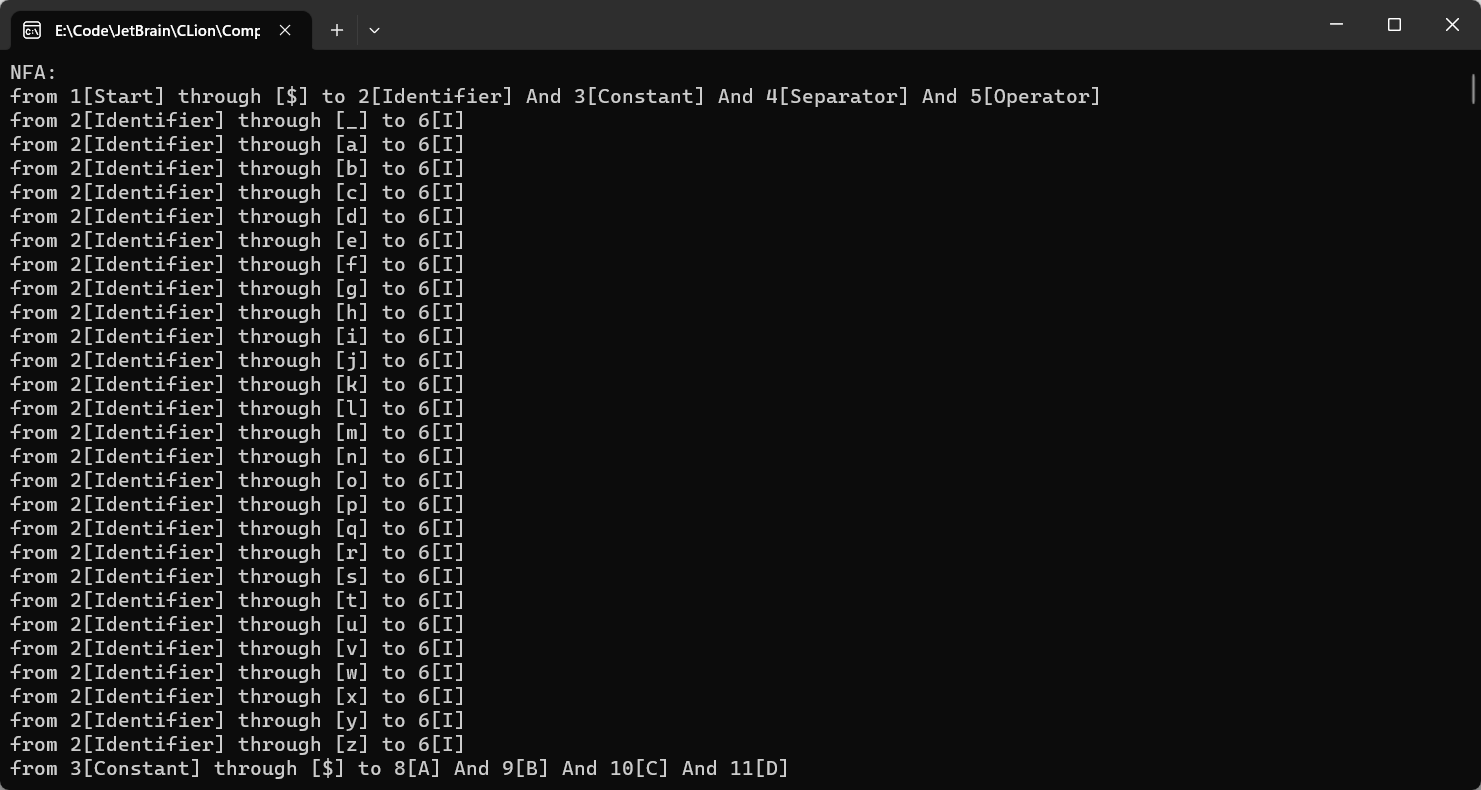
//  
using namespace std;  
void func(){}  
void aaa();  
//这是一条注释  
int main(){  
 int i = 8629;  
 float a = 94.52;//这是一条注释  
 float b = 1.145e+14;

double c = a + b;//没有  
 complex d = 29+52i;//没有  
 int x = 0;

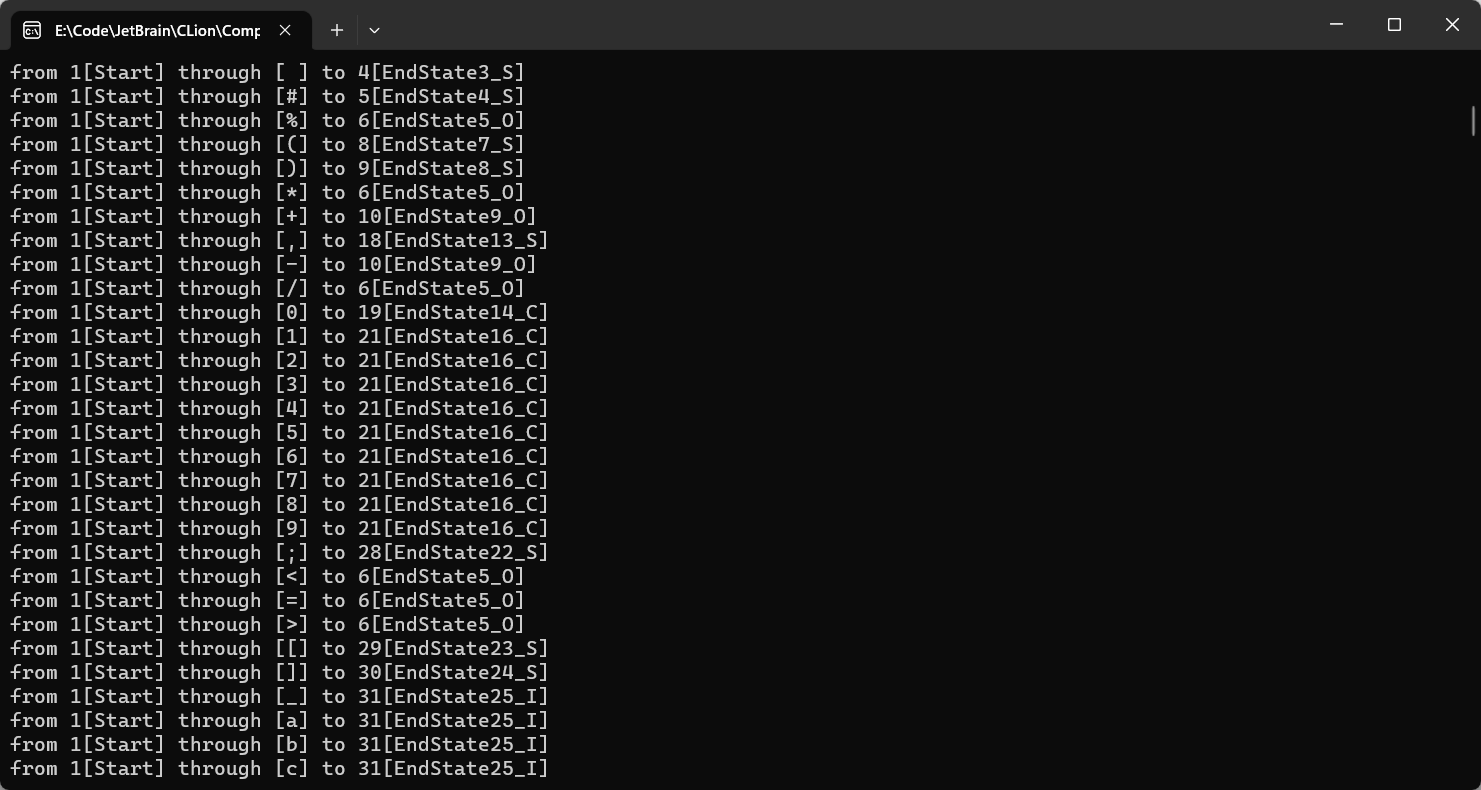
int y = 0;  
 do{  
 if(i == 2952){  
 y += x;  
 i -= 1;  
 //没有  
 }else{  
 y -= x;  
 }  
 }while(x < i);  
 return 0;  
}  
//通过

双击Compiler.exe运行程序

输出通过语法构建的NFA（由于篇幅限制，仅放出一部分）：



输出NFA确定化后得到的DFA（由于篇幅限制，仅放出一部分）：



输出词法分析结果：

词法分析结果：

line Type value

2 KEYWORD using

2 DELIMITER \0

2 KEYWORD namespace

2 DELIMITER \0

2 KEYWORD std

2 DELIMITER ;

3 KEYWORD void

3 DELIMITER \0

3 IDENTIFIER func

3 DELIMITER (

3 DELIMITER )

3 DELIMITER {

3 DELIMITER }

4 KEYWORD void

4 DELIMITER \0

4 IDENTIFIER aaa

4 DELIMITER (

4 DELIMITER )

4 DELIMITER ;

6 KEYWORD int

6 DELIMITER \0

6 IDENTIFIER main

6 DELIMITER (

6 DELIMITER )

6 DELIMITER {

7 KEYWORD int

7 DELIMITER \0

7 IDENTIFIER i

7 DELIMITER \0

7 OPERATOR =

7 DELIMITER \0

7 CONSTANT 8629

7 DELIMITER ;

8 KEYWORD float

8 DELIMITER \0

8 IDENTIFIER a

8 DELIMITER \0

8 OPERATOR =

8 DELIMITER \0

8 CONSTANT 94.52

8 DELIMITER ;

9 KEYWORD float

9 DELIMITER \0

9 IDENTIFIER b

9 DELIMITER \0

9 OPERATOR =

9 DELIMITER \0

9 CONSTANT 1.145e+14

9 DELIMITER ;

10 KEYWORD double

10 DELIMITER \0

10 IDENTIFIER c

10 DELIMITER \0

10 OPERATOR =

10 DELIMITER \0

10 IDENTIFIER a

10 DELIMITER \0

10 OPERATOR +

10 DELIMITER \0

10 IDENTIFIER b

10 DELIMITER ;

11 KEYWORD complex

11 DELIMITER \0

11 IDENTIFIER d

11 DELIMITER \0

11 OPERATOR =

11 DELIMITER \0

11 CONSTANT 29+52i

11 DELIMITER ;

12 KEYWORD int

12 DELIMITER \0

12 IDENTIFIER x

12 DELIMITER \0

12 OPERATOR =

12 DELIMITER \0

12 CONSTANT 0

12 DELIMITER ;

13 KEYWORD int

13 DELIMITER \0

13 IDENTIFIER y

13 DELIMITER \0

13 OPERATOR =

13 DELIMITER \0

13 CONSTANT 0

13 DELIMITER ;

14 KEYWORD do

14 DELIMITER {

15 KEYWORD if

15 DELIMITER (

15 IDENTIFIER i

15 DELIMITER \0

15 OPERATOR ==

15 DELIMITER \0

15 CONSTANT 2952

15 DELIMITER )

15 DELIMITER {

16 IDENTIFIER y

16 DELIMITER \0

16 OPERATOR +=

16 DELIMITER \0

16 IDENTIFIER x

16 DELIMITER ;

17 IDENTIFIER i

17 DELIMITER \0

17 OPERATOR -=

17 DELIMITER \0

17 CONSTANT 1

17 DELIMITER ;

19 DELIMITER }

19 KEYWORD else

19 DELIMITER {

20 IDENTIFIER y

20 DELIMITER \0

20 OPERATOR -=

20 DELIMITER \0

20 IDENTIFIER x

20 DELIMITER ;

21 DELIMITER }

22 DELIMITER }

22 KEYWORD while

22 DELIMITER (

22 IDENTIFIER x

22 DELIMITER \0

22 OPERATOR <

22 DELIMITER \0

22 IDENTIFIER i

22 DELIMITER )

22 DELIMITER ;

23 KEYWORD return

23 DELIMITER \0

23 CONSTANT 0

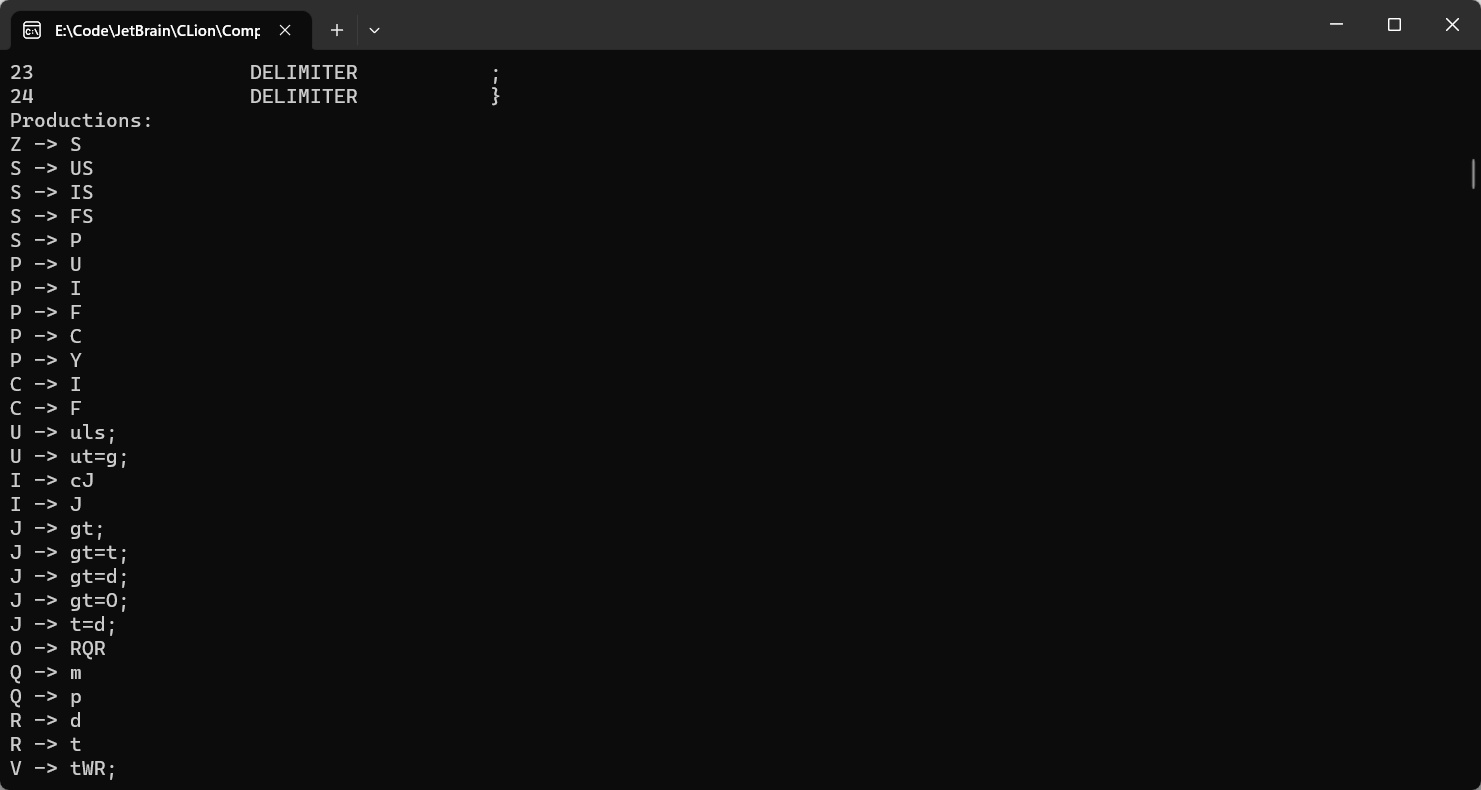
23 DELIMITER ;

24 DELIMITER }

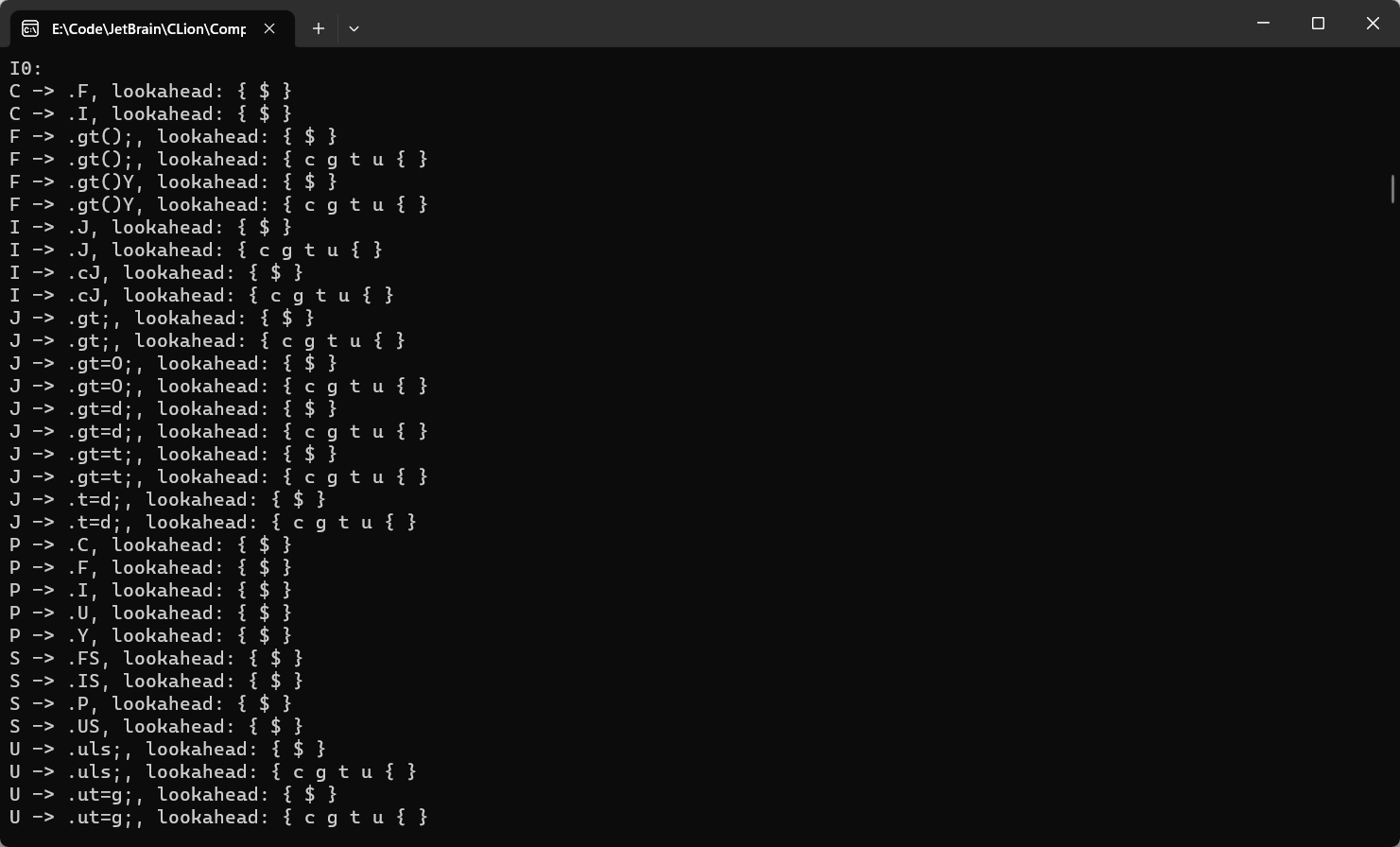
词法分析到此结束，此时得到了Token表

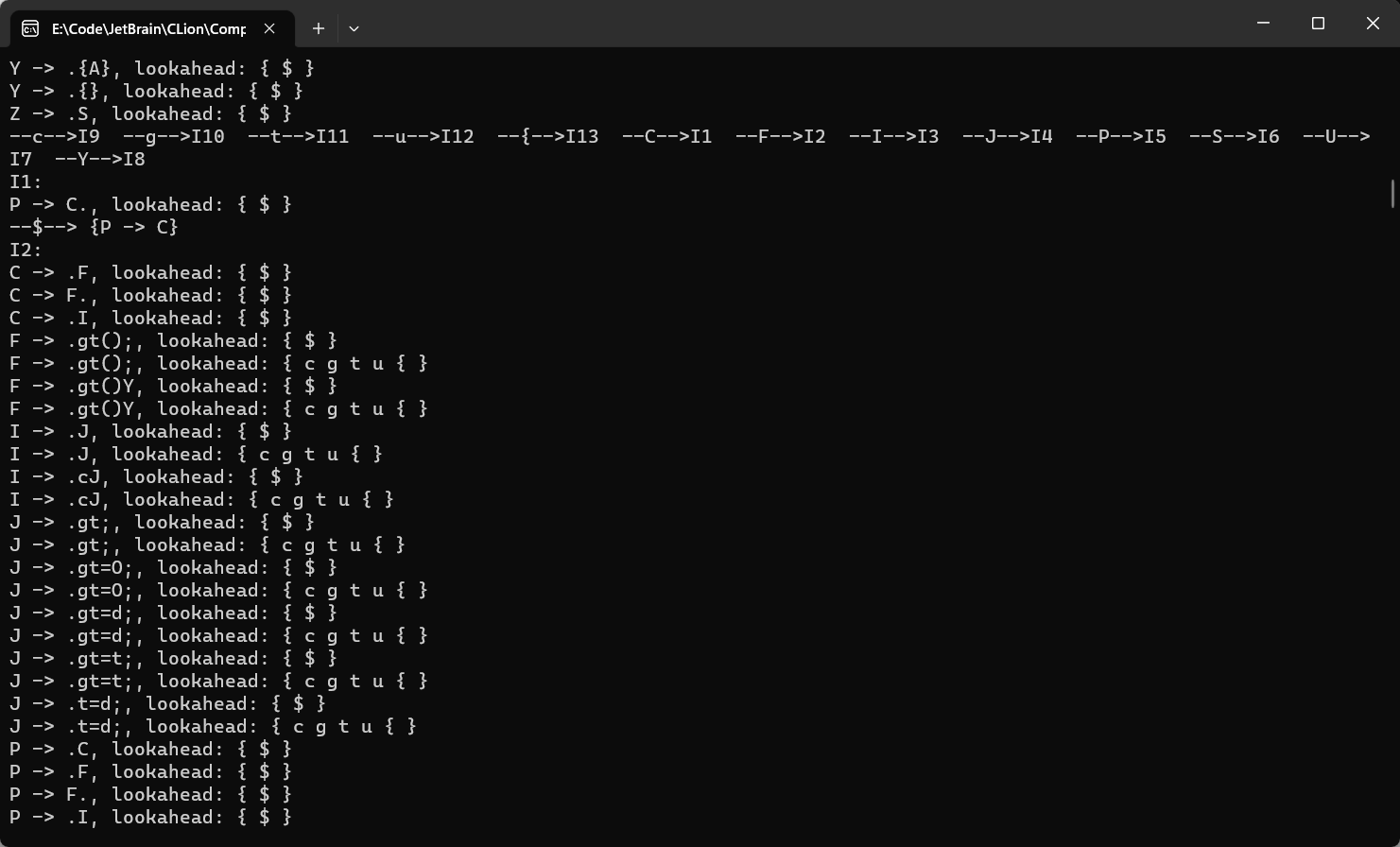
语法分析：

输出读取的产生式：

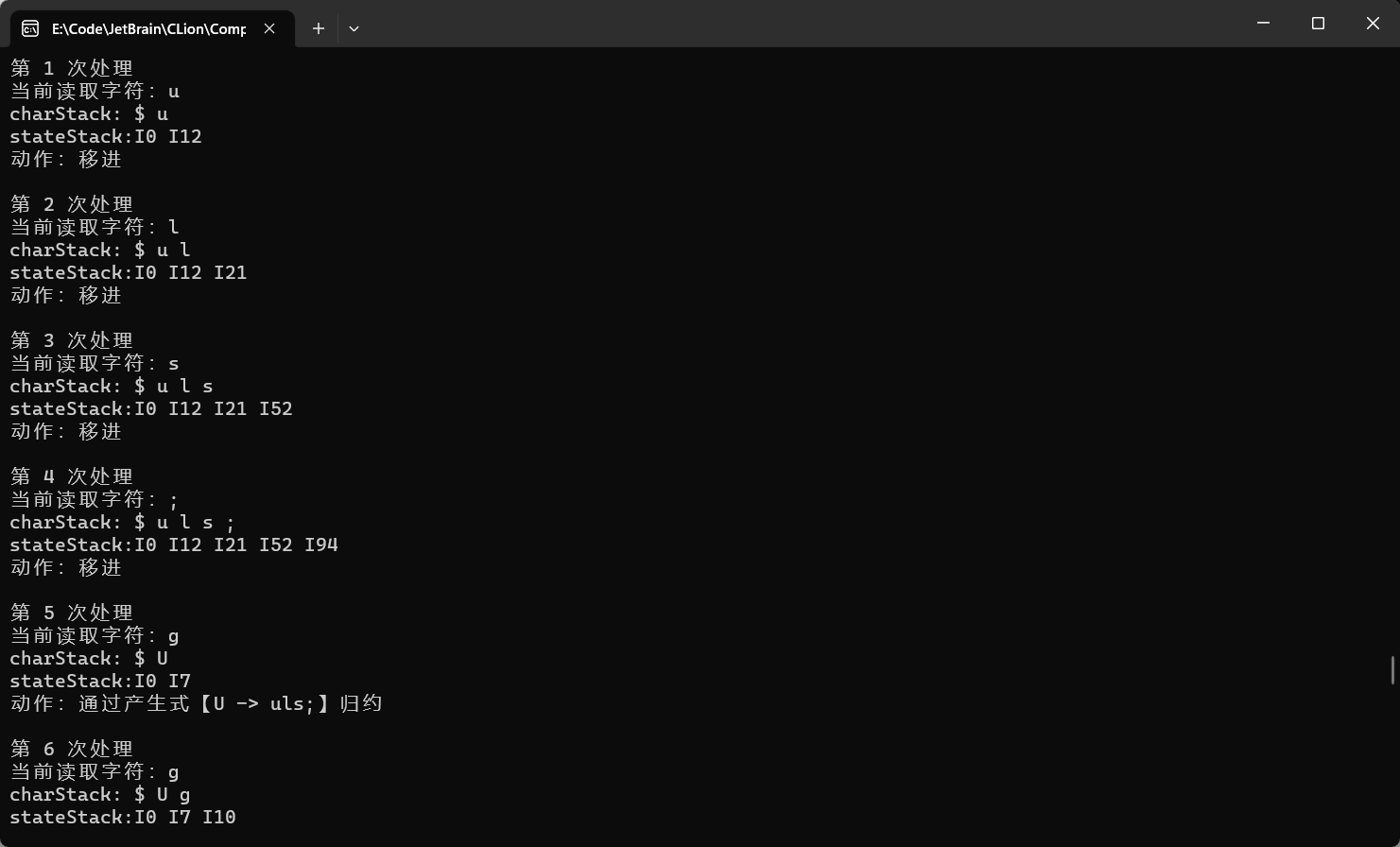


输出通过产生式构建的项集族以及对应的状态转移关系（包含展望符、ACTION及GOTO表，篇幅原因仅展示部分）





输出词法分析过程：

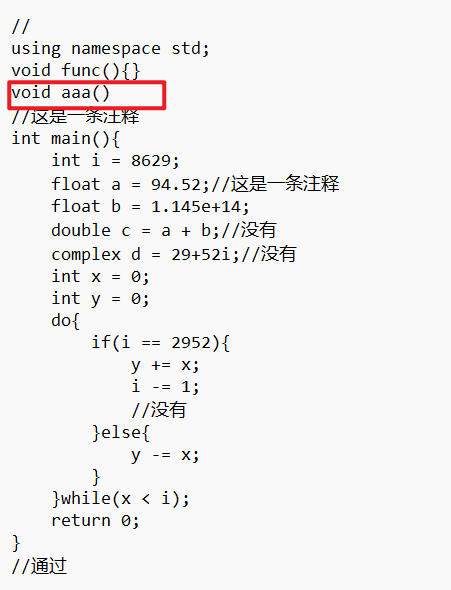




可以看到接收状态为YES，语法分析通过

此时我们修改代码，人为制造一些错误：

去除函数声明后的”;”，此时语法分析将不通过

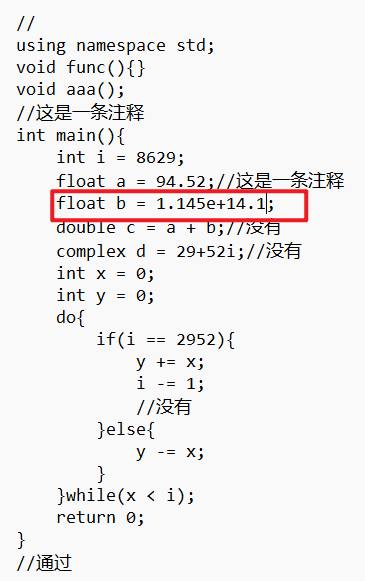


运行结果：

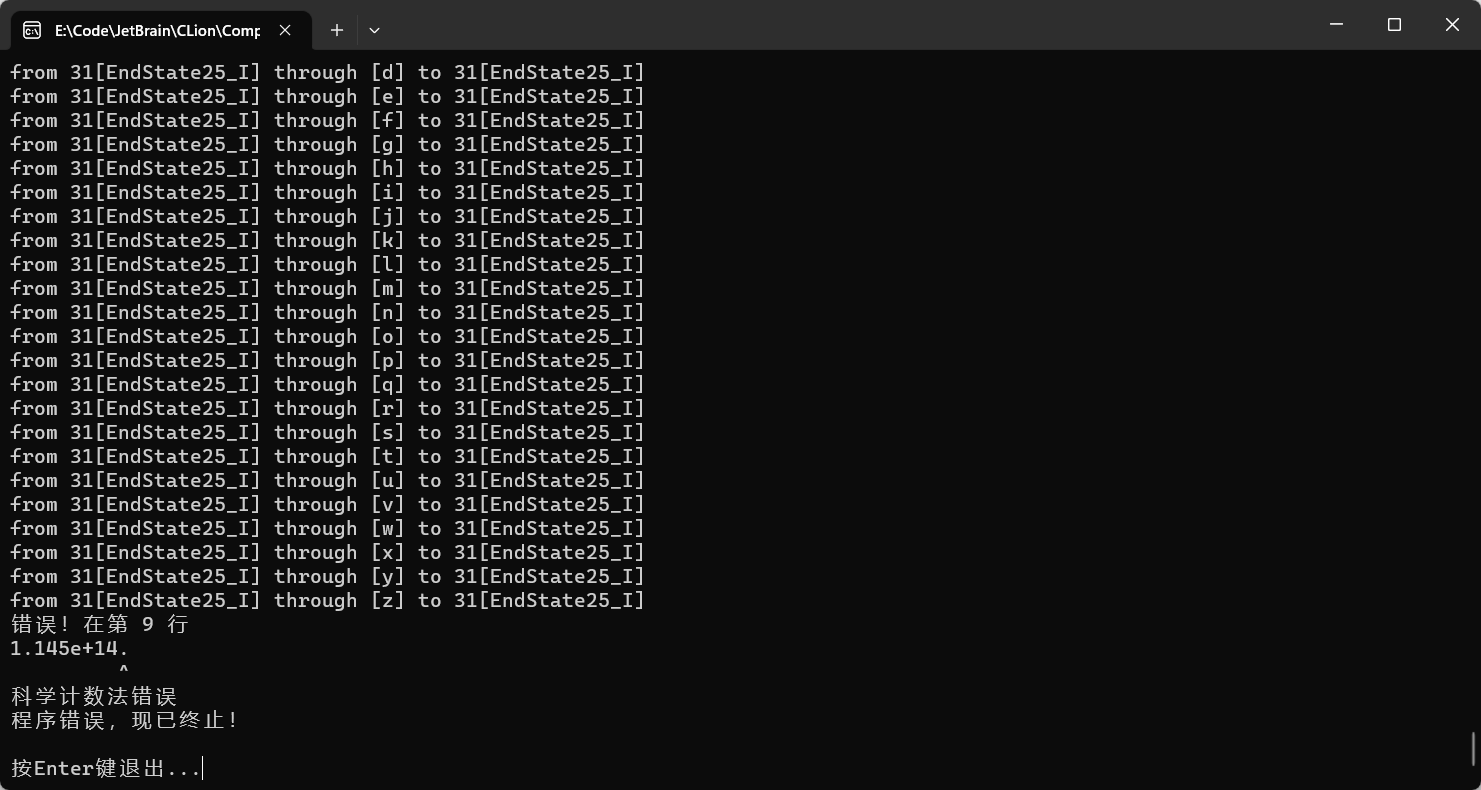


提示不被接受，函数声明后需要接字符”;”或者”{”

科学计数法指数部分不能为小数：



运行结果：



输出了错误处第9行的内容，并指向错误位置，表示输入字符不被接受，并给出大概的错误解释。

标识符必须字母或者下划线开头：

