实验 3 语法分析器-布尔表达式和控制语句

语法分析器分两部分，第一部分为算术表达式，第二部分为布尔表达式和控制语句。

要求

参考课本 4.4，实现递归下降语法分析器。
当语法分析器需要单词符号时，调用词法分析器获取单词符号。

说明：
下列文法中，黑色字体与实验 2 相同，即第一、二步与实验 2 相同；蓝色字体为新增部分，从第三步开始。

文法：

stmts ⟶ *stmt rest0*

rest0 ⟶ *stmt rest0* | *ε*

stmt ⟶ *loc* **=** *expr*;
| **if(***bool***)** *stmt* **else** *stmt*
| **while(***bool***)** *stmt*

loc⟶ **id** *resta*

resta ⟶ [*elist*] | ε

elist ⟶ *expr rest1*

rest1⟶ , *expr rest1* | ε

bool ⟶ *equality*

equality ⟶*rel rest4*

rest4 ⟶ ==*rel rest4* | !=*rel rest4* | **ε**

rel ⟶*expr rop_expr*

rop_expr ⟶ <expr | <=expr | >expr| >=expr | **ε**

expr ⟶ *term rest5*

rest5⟶ **+***term rest5* |**-***term rest5* |**ε**

term ⟶ *unary rest6*

rest6⟶ * *unary rest6* | / *unary rest6* | **ε**

unary ⟶*factor*

factor ⟶ **(***expr***)** | *loc* | **num**

提示

可将以上文法拆解为小的文法分步完成。

第一步：包含乘、除的算术表达式

term ⟶ *unary rest6*

rest6⟶ ***** *unary rest6* | **/** *unary rest6* | **ε**

unary ⟶*factor*

factor ⟶ **num**

输入：

5*2/3

输出：

1) 按推导过程

term ⇒ unary rest6
⇒ factor rest6
⇒ num rest6
⇒ num * unary rest6
⇒ num * factor rest6
⇒ num * num rest6
⇒ num * num / unary rest6
⇒ num * num / factor rest6
⇒ num * num / num rest6
⇒ num * num / num

2) 按使用产生式过程

term ⟶ unary rest6
unary ⟶factor
factor ⟶ num
rest6⟶ * unary rest6
unary ⟶factor
factor ⟶ num
rest6⟶ / unary rest6
unary ⟶factor
factor ⟶ num
rest6⟶ Ɛ

第二步：加入加、减运算

expr ⟶ term rest5

rest5⟶ +term rest5 |-term rest5 |ε

term ⟶ *unary rest6*

rest6⟶ ***** *unary rest6* | **/** *unary rest6* | **ε**

unary ⟶*factor*

factor ⟶ **num**

输入：

9+5*2/3-6

输出：

1) 按推导过程

expr ⇒ term rest5
⇒ unary rest6 rest5
⇒ factor rest6 rest5
⇒ num rest6 rest5
⇒ num rest5
⇒ num + term rest5
⇒ num + unary rest6 rest5
⇒ num + factor rest6 rest5
⇒ num + num rest6 rest5
⇒ num + num * unary rest6 rest5
⇒ num + num * factor rest6 rest5
⇒ num + num * num rest6 rest5
⇒ num + num * num / unary rest6 rest5
⇒ num + num * num / factor rest6 rest5
⇒ num + num * num / num rest6 rest5
⇒ num + num * num / num rest5
⇒ num + num * num / num – term rest5
⇒ num + num * num / num – unary rest6 rest5
⇒ num + num * num / num –factor rest6 rest5
⇒ num + num * num / num –num rest6 rest5
⇒ num + num * num / num –num rest5
⇒ num + num * num / num –num

2) 按使用产生式过程

expr ⟶ term rest5

term ⟶ unary rest6

unary ⟶factor

factor ⟶ num

rest6⟶ Ɛ

rest5⟶ +term rest5

term ⟶ unary rest6

unary ⟶factor

factor ⟶ num

rest6⟶ * unary rest6

unary ⟶factor

factor ⟶ num

rest6⟶ / unary rest6

unary ⟶factor

factor ⟶ num

rest6⟶ Ɛ

rest5⟶ -term rest5

term ⟶ unary rest6

unary ⟶factor

factor ⟶ num

rest6⟶ Ɛ

rest5⟶ Ɛ

第三步：加入关系运算

bool ⟶ equality

equality ⟶rel rest4

rest4 ⟶ ==rel rest4 | !=*rel rest4* | **Ɛ**

rel ⟶expr rop_expr

rop_expr ⟶ <expr | <=expr | >expr| >=expr | **Ɛ**

expr ⟶ term rest5

rest5⟶ **+***term rest5* |**-***term rest5* |**Ɛ**

term ⟶ unary rest6

rest6⟶ ***** *unary rest6* | **/** *unary rest6* | **Ɛ**

unary ⟶factor

factor ⟶ **(***expr***)** | **num**

测试：

1==4<=8

第四步：加入语句和数组

stmts⟶ stmt rest0

rest0 ⟶ stmt rest0 | *Ɛ*

stmt ⟶ loc **=** *expr*;

| **if(***bool***)** *stmt* **else** *stmt*

| **while(***bool***)** *stmt*

loc⟶ **id** *resta*

resta⟶ [*elist*] | Ɛ

elist ⟶ expr rest1

rest1⟶ , expr rest1 | Ɛ

bool ⟶ equality

equality ⟶ *rel rest4*

rest4 ⟶ ==*rel rest4* | !=*rel rest4* | ε

rel ⟶ *expr rop_expr*

rop_expr ⟶ <expr | <=expr | >expr| >=expr | ε

expr ⟶ *term rest5*

rest5 ⟶ **+***term rest5* |**-***term rest5* |ε

term ⟶ *unary rest6*

rest6 ⟶ ***** *unary rest6* | **/** *unary rest6* | ε

unary ⟶ *factor*

factor ⟶ **(***expr***)** | *loc* | **num**

测试：

```
while(a[i]) b[i,j]=10;
```

第五步：测试完整文法

```
while(sum<10000)
    if(a<b)
        sum=sum*(c[10]+10);
    else
        c[10]=sum*c[10]+10;
x[i,j]=sum;
```