**实验3 递归下降法的语法分析器**

**姓名：李波 学号：22920202204570**

**一、实验目的**

掌握语法分析器的构造原理，掌握递归下降法的编程方法。

**二、实验内容**

**将原文法进行提取左公因子和消除左递归处理后得到的文法如下：**

*program* → *block*

*block*→ **{** *stmts* **}**

*stmts* → *stmt* *stmts* | ε

*stmt* → **id****=** *expr* **;**

| **if** **(***bool***)** *stmt A*

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

*|* **do** *stmt* **while (***bool***)** *;*

*|* **break** *;*

*| block*

*A → else stmt | ε*

*bool → expr B*

*B →* **relop** *expr | ε*

*expr* → *term expr’*

*expr’ → C expr’ | ε*

*C* → **+** *term* | **-** *term*

*term* → *factor term’*

*term’ → D term’ | ε*

*D* → **\*** *factor |* **/** *factor*

*factor* →  **( *e****xpr* **)** | **id**| **num**

根据文法编写程序。

**三、实验结果**

**图形用户界面

中度可信度描述已自动生成**

文本

描述已自动生成

**四、实验遇到的问题和解决方式**

问题：

修改文法时，没有注意到有些生成式带有分号，而有些生成式没有分号，所以在写程序时没有写match(“;”)语句，导致结果错误。

解决方法：

通过理解原文法中生成式的意义，判断哪些生成式最后需要加上分号，如“id = expr;”和“break;”等生成式需要加上分号，而以stmt结尾的生产式和block等生成式不需要加上分号。最后根据判断结果决定代码中是否需要写match(“;”)语句。