语法分析器实验：  
LR(1)语法分析器的设计与实现

# 实验目的

熟悉和掌握LR(1)语法分析器的构造，加深对自上而下语法分析方法的理解。

# 实验内容

需要实现的功能：

1）使用LR(1)分析方法构造识别活前缀的DFA；

2）构造文法的分析表（Action表和Goto表）；

3）构造LR语法分析器的总控程序；

3）输入文法（语言语法结构的文法描述存储在文本文件中）；

4）输出文法的项目集簇（标准输出设备）；

5）输出识别活前缀的DFA（标准输出设备）；

6）输出文法的Action表和Goto表（标准输出设备）；

7）对给定的输入串（存储在文本文件中），输出其是否该文法正确句子的判断，并输出文本形式的分析过程。

# 实验要求

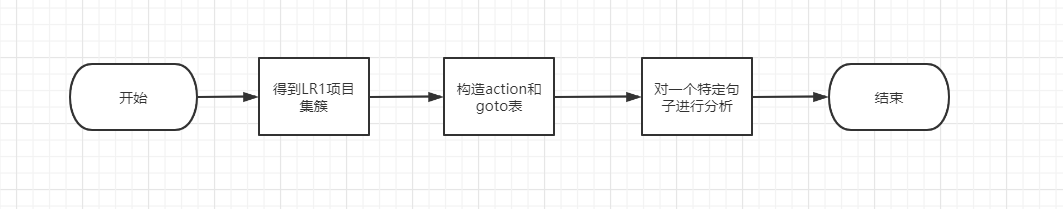
使用《LR0语法分析器的设计与实现》实验的结果。

# 实验时间

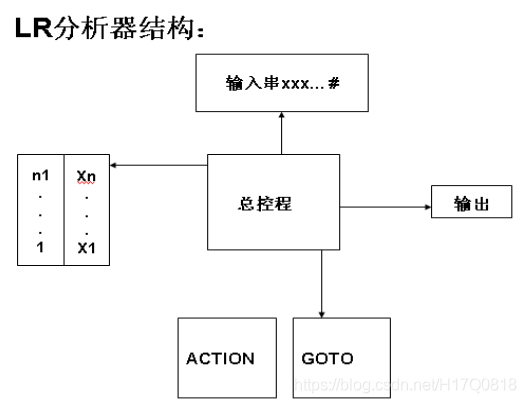
上机课2次4小时；

# 5、模型分析

要构造LR1语法分析器，首先就是要使用LR1分析方法构造识别活前缀的DFA，并输出文法的项目集簇。然后构造文法的action表和goto表。最后输出对一个特定句子的分析过程。流程图即为：



**LR分析器结构：**



**LR1分析法的主要思想：**1、进行最左归约（识别句柄并归约它）

2、将识别句柄的过程划分为由若干状态控制,每个状态控制识别出句柄的一个符号

3、分析栈:存放已识别的文法符号和状态,描述分析过程中的历史和展望信息

4、由一个总控程序来控制整个识别过程

**构造closure：**

a)I的任何项目都属于 CLOSURE(I)；

b) 若项目(A•B,a)属于CLOSURE(I)，B是一个产生式，则对于F IRST(a)中的每个终结符b，如果(B•,b)原来不在CLOSURE(I)中，则把它加进去；

c) 重复步骤 b)直到CLOSURE(I)不再扩大为止。

**LR1分析表的构造：**

假定LR(1)项目集规范族C={I0, I1, …… ，In}，令每个项目集Ik的下标 k为分析器的一个状态，G’的LR(1)分析表含有状态0，1，…… ，n。

令那个含有项目[S’→·S，#]的Ik的下标k为状态0（初态），ACTION 表和GOTO表可按如下方法构造：

1、若项目[A→α·,b]属于Ik，那么置ACTION[k,b]为“用产生式 A →α进行规约”，简记为rj(假定A→α为文法G’的第j个产生式)。

2、若项目[A→α·aβ，b]属于Ik且GO(Ik, a)= Ij,则置ACTION[k,a]为 “把状态j和符号a移进栈”，简记为 sj 。

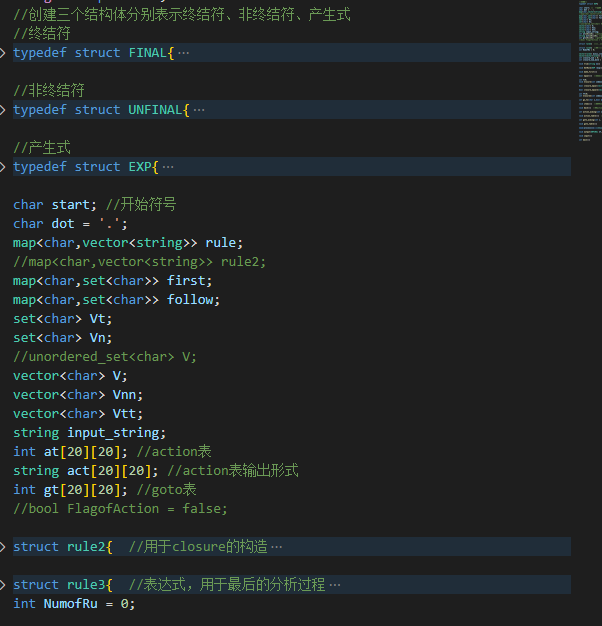
3、若项目[S’→S·,#]属于Ik,则置ACTION[k, #]为“接受”，简记为 acc。

4、若GOTO(Ik, A)= Ij，A为非终结符，则置 GOTO(k, A)=j，分析表中凡不能用规则1至5填入信息的空白格均置上“出错标志 ”(空)。

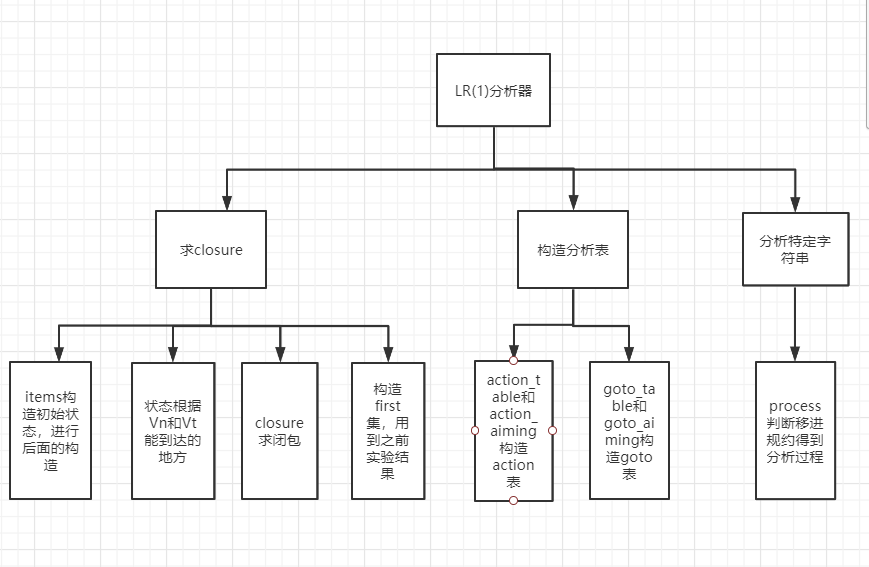
按上述算法构造的含有ACTION和GOTO两部分的分析表，如果每个入口不含多重定义，则称它为文法G的一张规范的LR(1)分析表。

# 6、模型构建

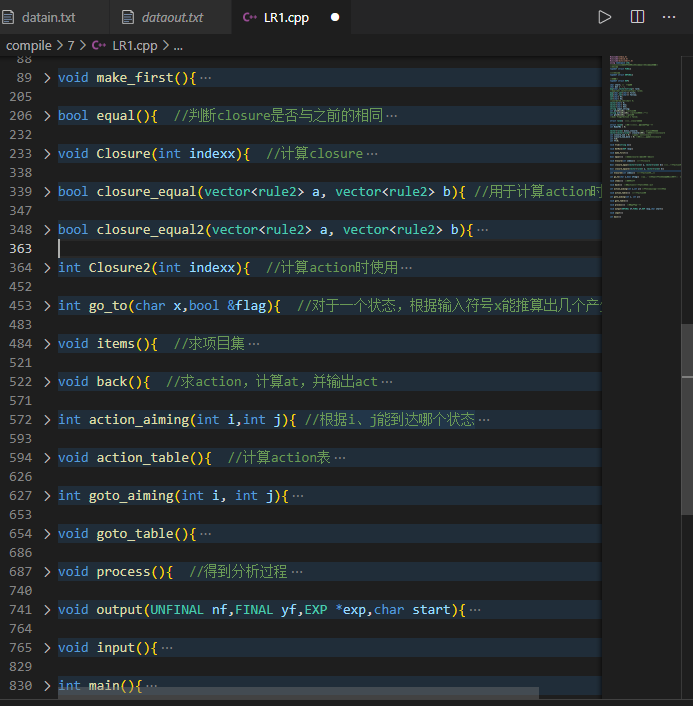
对于该程序，用到了很多数据结构来存储里面的很多信息，包括保存的信息和输出的信息，大致有：



在之后将主程序分为三个模块，第一个是求closure构造自动机，第二个是构造LR(1)分析表，第三个是对给定的字符串进行LR(1)分析。结构图如大致如下：

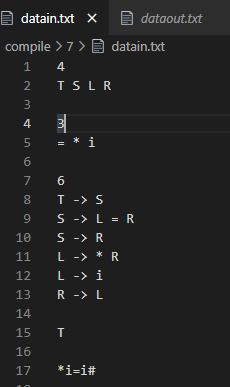


期间还有一些判断相等、控制输出格式之类的函数，整个程序的框架如下：

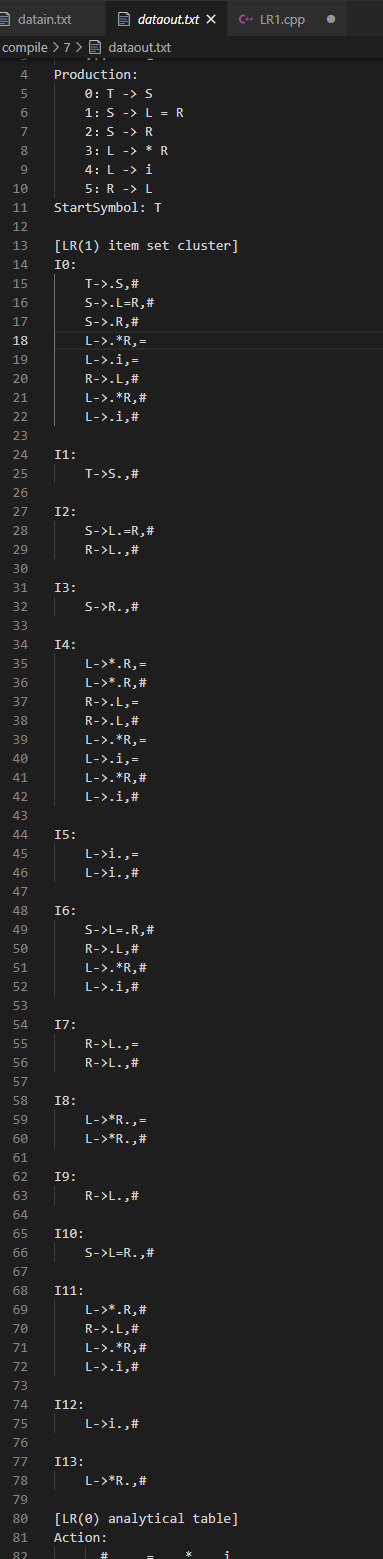


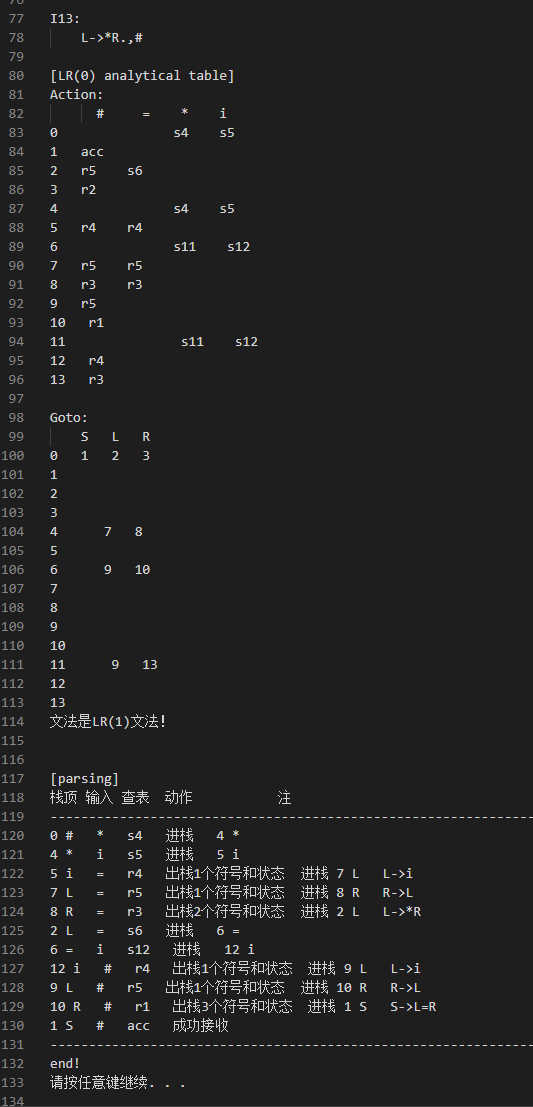
# 7、实验结果

输入：



该程序输出结果如下：





# 8、总结反思

①、个人感觉这次实验难度还是挺大的，也是花了不少的时间来完成。但同时也发现自己在模型设计时出现的一些失误，一些数据结构没有创建的很好，导致后面要花更多的代码来进行数据的修改和完善。这也是自己的一些失误吧。

②、通过这次试验使自己对于LR1文法的构造有了很深刻的理解，自己动手将其实现出来，虽然过程很辛苦，但是有了结果的时候还是挺欣慰的。另外，在这次试验中还是有一些不是特别完善的地方，有些地方比较繁琐不尽人意。还会慢慢修改的。

③、这门课的学习快要接近尾声，在这门课上自己学习到了很多东西，关于知识，关于动手实验，也关于计算机学习中的一些方法与经验。包括老师说的编程时很重要的三点：问题分析、模型构建以及代码实现。“纸上得来终觉浅，绝知此事要躬行”，在以后的学习中也要多动手，多思考。将在这门课上学到的东西带到以后的学习中去。