语法分析器实验：  
LR(0)语法分析器的设计与实现

# 实验目的

熟悉和掌握LR(0)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(0)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《文法构造实验一》、《文法构造实验二》的结果。

# 实验时间

上机课2次4小时；

上机课1次2小时；

5. 分析模型

整体上来说实验要分为3步，第一步需要寻找所有的状态I；第二部需要通过所有的状态I以及I的转换函数，将所有的情况填写到goto表和action表中去；最后一步就是通过action、goto表，将所有的情况进行判断，通过对于字符串的识别程序来判别字符串是否能够通过规则的判断。

项目集构造  
 枚举每个规范句型，然后枚举”⋅”的位置，获得所有的项目项目集规范族构造算法。假定I是文法G’的任一项目集，定义和构造I的闭包CLOSURE(I)的办法是：   
 I的任何项目都属于CLOSURE(I)；   
 若A→α•Bβ属于CLOSURE(I)，那么，对任何关于B的产生式B→γ，项目B→•γ也属于CLOSURE(I)；   
 重复执行上述两步骤直至CLOSURE(I)不再增大为止。

LR(0)分析表构造算法：

假定项目集规范族C={I0,I1,…,In}。令每一个项目集Ik的下标k作为分析器的状态。分析表的ACTION子表和GOTO子表可按如下方法构造：

令那个包含项目S’→•S的集合Ik的下标k为分析器的初态。

（1）若项目A→α•aβ属于Ik且GO(Ik , a)= Ij,a为终结符，置ACTION[k,a]为“把(j,a)移进栈”，简记为“sj”。

（2）若项目A→α•属于Ik，对任何终结符a(或结束符#)，置ACTION[k,a]为“用产生式A→α进行归约”，简记为“rj”（假定产生式A→α是文法G’的第j个产生式）。

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

（4）若GO(Ik , A)= Ij，A为非终结符，则置GOTO[k,A]=j。

（5）分析表中凡不能用规则1至4填入信息的空白格均填上“报错标志”。

LR(0)分析法的分析过程：

遍历输入字符串，对于每一个字符，获取当前状态栈的顶部的状态值，通过查询action表获取的当前的操作是移进、规约还是接受。如果当前操作是移进，将新的状态放入状态栈当中，当移入的字符放入符号栈中。

如果当前操作是规约，那么将需要规约的部分的状态从状态栈中弹出，将规约后的状态放入状态栈，将规约后的左部放入符号栈，当前的字符不向下推进。如果接收，则结束。

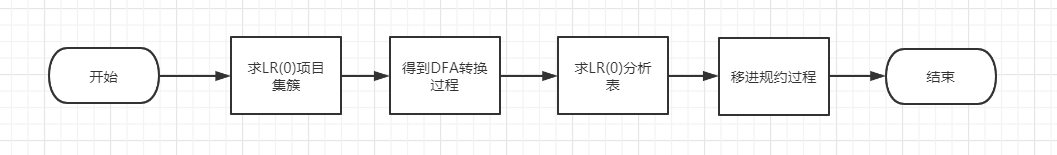
6. 设计模型

在第一步中需要用的模块有：将带有“.”的字符串保存到所规定的数据结构中的模块、找到所有情况的闭包的函数、判断是否和前面重复的函数。最后通过将所有的情况用队列进行存储。通过将新找到的情况寻找到闭包之后进行push，再将队首的进行寻找所有的转换情况，如果不重复再进行入队，以此类推，直到

找到所有没有重复的情况，此时队列为空。

第二步需要很细致对于每一个go进行查找，如果遇到的是终结符，则用s进行填写；如果不是终结符，则填入goto表中。如果已经是末尾，则需要将情况填写进r中。

对字符串分析只需要将所有的情况进行出栈入栈操作。最刚开始将0入栈。对字符串进行判断，如果是s的直接进行移进操作。如果是r类型的则将其规约，即将栈顶几个相关的状态进行弹出，弹出过之后将新的状态进行入栈操作。以此类推，直到遇到了acc符号，则跳出循环对于栈顶进行判断，如果栈顶是0状态，即为接受状态，则直接将返回成功。



7. 系统设计

struct GRP

{

int numofRLT;

int head[10];

int numofTRS[10];

bool end[10][100]; //是否是终结符

int follow[10][100];

int nodepos[10];

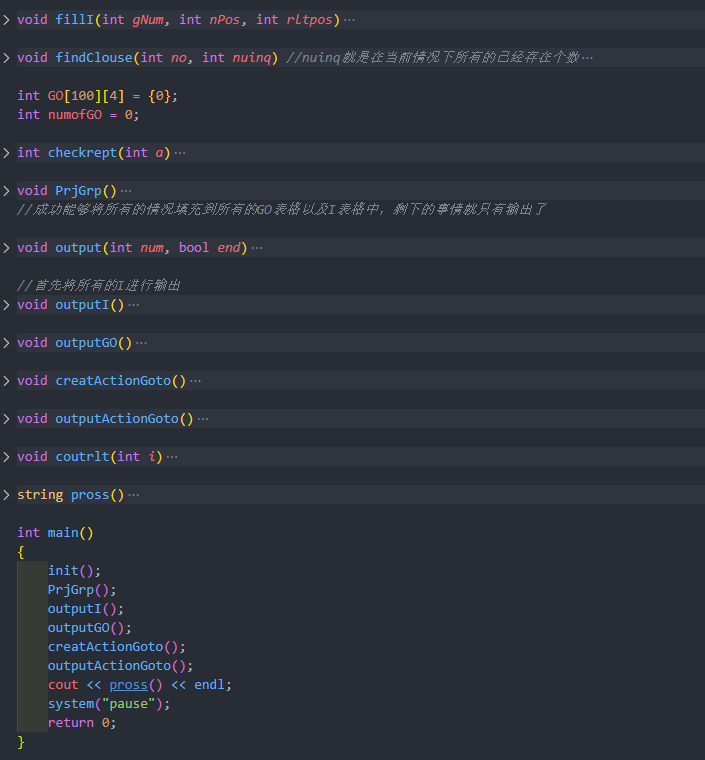
int rltpos[10];

} I[100];

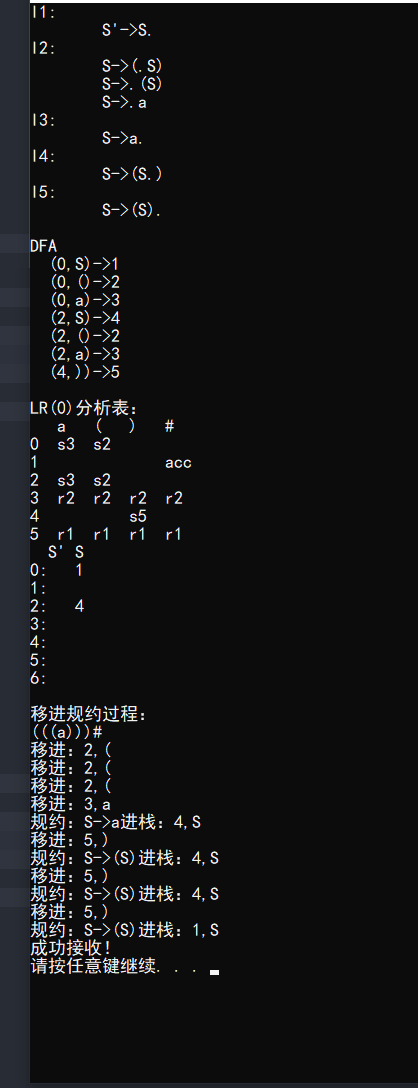
int numofGRP = 0;

//这个是存储的结构体，保存一个情况所有的转换函数，保存每一个转化函数的头，以及跟着的字符。还有nodepos，就是“ .”的位置。

代码整体结构如下：



8.输出结果



9、实验总结

这次实验完成了对于文法的LR(0)分析过程，通过实现计算LR(0)项目集簇，求出DFA转换过程，计算LR(0)分析表以及输出给定字符串的分析过程，实现了LR（0）分析的全过程。

在这次实验过程中，学习到了很多东西。对于LR(0)分析的全过程也有了更深的理解。但同时自身也存在一些不足的地方需要改正。

附件：输入文件的数据格式

## 文件结构：

1. 非终结符个数；
2. 所有非终结符，空格分隔；
3. 终结符个数；
4. 所有终结符，空格分隔；
5. 规则个数；
6. 所有规则，每行一个规则，规则输入格式：左部，右部符号数，右部符号，空格分隔；
7. 开始符号。
8. 待分析的符号串。

## 样例

### 描述：

算术表达式文法G =（VN，VT，P，S）其中：

VN = { S’, S }

VT = { a, (, ) }

P = { S’ → S

S → (S)

S → a }

S’

### 输入：

2

S' S

3

a ( )

3

S' 1 S

S 3 ( S )

S 1 a

S'

( ( ( a ) ) ) #

### 输出：

LR(0) 项目集簇

I0 :

S' -> . S

S -> . ( S )

S -> . a

I1 :

S -> a .

I2 :

S -> . ( S )

S -> ( . S )

S -> . a

I3 :

S' -> S .

I4 :

S -> ( S . )

I5 :

S -> ( S ) .

DFA

状态个数：6

字符表个数：7

状态转换：

(0,2)->1

(0,3)->2

(0,6)->3

(2,2)->1

(2,3)->2

(2,6)->4

(4,4)->5

开始状态：0

结束状态集：{ 0 1 2 3 4 5 }

文法是 LR(0) 文法！

LR(0)分析表

Action：

# a ( )

0 : s1 s2

1 :r2 r2 r2 r2

2 : s1 s2

3 :acc

4 : s5

5 :r1 r1 r1 r1

Goto：

S

0 :3

1 :

2 :4

3 :

4 :

5 :

移进：2, (

移进：2, (

移进：2, (

移进：1, a

规约：S -> a 进栈：4, S

移进：5, )

规约：S -> ( S ) 进栈：4, S

移进：5, )

规约：S -> ( S ) 进栈：4, S

移进：5, )

规约：S -> ( S ) 进栈：3, S

成功接收！