|  |
| --- |
| 编译原理  第四章 语法分析 |
|  |
|  |

目 录

[1. 实验题目和要求 2](#_Toc438244562)

[2. 递归调用预测分析实现 自顶向下分析（方法1） 2](#_Toc438244563)

[2.1. 消除文法左递归。 2](#_Toc438244564)

[2.2. 画出状态图得 3](#_Toc438244565)

[2.3. 预测分析程序的实现 3](#_Toc438244566)

[3. LL实现 编程实现教材算法4.2 6](#_Toc438244567)

[4. LR实现 自底向上分析（方法3） 13](#_Toc438244568)

[4.1. 构造识别所有活前缀的DFA。 13](#_Toc438244569)

[4.2. 构造LR分析表。 14](#_Toc438244570)

[4.3. 编程实现算法4.3， 构造LR分析程序。 15](#_Toc438244571)

[4.4. 运行结果截图： 31](#_Toc438244572)

[5. 实验总结 33](#_Toc438244573)

# 实验题目和要求

**题目**：语法分析程序的设计与实现。

**实验内容**：编写语法分析程序，实现对算术表达式的语法分析。要求所分析算术表达式由如下的文法产生。



**实验要求**：在对输入表达式进行分析的过程中，输出所采用的产生式。

方法1：编写递归调用程序实现自顶向下的分析。

方法2：编写LL(1)语法分析程序，要求如下。

1. 编程实现算法4.2，为给定文法自动构造预测分析表。
2. 编程实现算法4.1，构造LL(1)预测分析程序。

方法3：编写语法分析程序实现自底向上的分析，要求如下。

1. 构造识别所有活前缀的DFA。
2. 构造LR分析表。
3. 编程实现算法4.3，构造LR分析程序。

方法4：利用YACC自动生成语法分析程序，调用LEX自动生成的词法分析程序。

# 递归调用预测分析实现 自顶向下分析（方法1）

## 消除文法左递归。

文法改写为：



## 画出状态图得

**化简得**

## 预测分析程序的实现

1. 说明

E定义子程序e()

E`定义子程序g()

T定义子程序t()

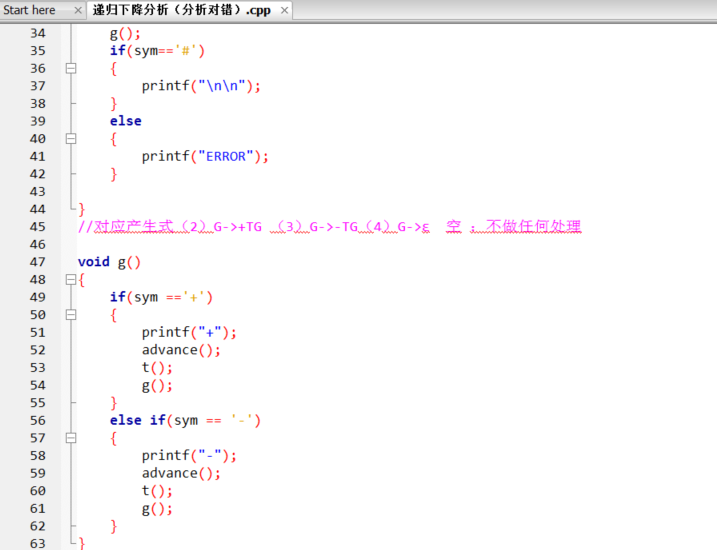
F定义子程序f()

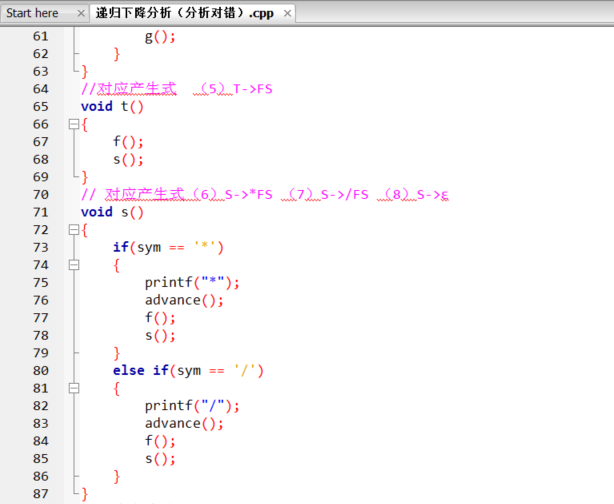
advance()获得下一个字符

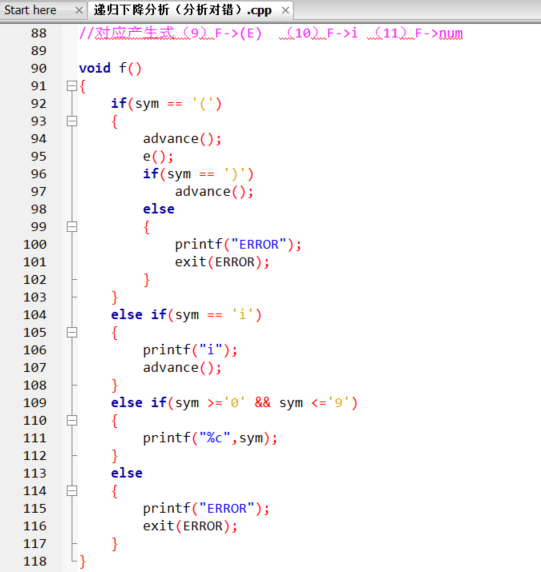
主函数从开始符E出发。

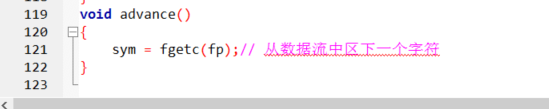
1. 源程序











# LL实现 编程实现教材算法4.2

用动态规划的思想解决FIRST集合和FELLOW集合的计算。

按照规则（代码中有相应的输入提示）

则可以返回出文法的FIRST集合和FELLOW集合

**#include <algorithm>**

**#include <iostream>**

**#include <iomanip>**

**#include <cstring>**

**#include <climits>**

**#include <complex>**

**#include <fstream>**

**#include <cassert>**

**#include <cstdio>**

**#include <bitset>**

**#include <vector>**

**#include <deque>**

**#include <queue>**

**#include <stack>**

**#include <ctime>**

**#include <set>**

**#include <map>**

**#include <cmath>**

**#include <functional>**

**#include <numeric>**

**using namespace std;**

**const int maxn=305;**

**set<string> VT,VN;**

**vector<pair<string,string> > Phi;**

**map<string,int> Map;**

**map<int,string> re;**

**vector<string> G[maxn];**

**set<string> FIRST[maxn],FOLLOW[maxn];**

**int number;**

**string S;**

**int vis[maxn],vis1[maxn];**

**string kesi="kesi";**

**pair<string,string> M[maxn][maxn];**

**set<string> getFIRST(string t)**

**{**

**if(t.length()==1)**

**{**

**int x=Map[t];**

**if(vis[x])return FIRST[x];//如果x的FIRST集已经计算过直接返回**

**vis[x]=1;**

**for(int i=0; i<G[x].size(); i++)**

**{**

**if(G[x][i]==kesi)FIRST[x].insert(kesi);**

**bool flag=true;//表示FIRST1~i-1的都含kesi,初始值设为true则书上两种情况可以放到一起**

**if(G[x][i]==kesi)**

**{**

**FIRST[x].insert(kesi);**

**continue;**

**}**

**for(int j=0; j<G[x][i].length()&&flag; j++)**

**{**

**if(!vis[Map[G[x][i].substr(j,1)]])getFIRST(G[x][i].substr(j,1));**

**int v=Map[G[x][i].substr(j,1)];**

**for(set<string>::iterator it=FIRST[v].begin(); it!=FIRST[v].end(); it++)**

**{**

**string tmp=(\*it);**

**{**

**string B="E";**

**string f="+";**

**if(t==B&&(\*it)==f)cout<<G[x][i]<<" "<<i<<endl;**

**}**

**if(tmp!=kesi)FIRST[x].insert(tmp);**

**}**

**if(FIRST[v].find(kesi)==FIRST[v].end())flag=false;**

**}**

**if(flag)FIRST[x].insert(kesi);**

**}**

**return FIRST[x];**

**}**

**else**

**{**

**bool flag=true;**

**set<string> ret;**

**ret.clear();**

**for(int i=0; i<t.size()&&flag; i++)**

**{**

**int u=Map[t.substr(i,1)];**

**for(set<string>::iterator it=FIRST[u].begin(); it!=FIRST[u].end(); it++)**

**{**

**string t=(\*it);**

**if(t!=kesi)ret.insert(t);**

**}**

**if(FIRST[u].find(kesi)==FIRST[u].end())flag=false;**

**}**

**if(flag)ret.insert(kesi);**

**return ret;**

**}**

**}**

**set<string> getFOLLOW(string B)**

**{**

**int x=Map[B];**

**if(vis1[x])return FOLLOW[x];**

**vis1[x]=1;**

**for(int i=0; i<Phi.size(); i++)**

**{**

**string A=Phi[i].first,omiga=Phi[i].second;**

**int Size=omiga.size();**

**for(int j=0; j<Size; j++)**

**if(omiga.substr(j,1)==B)**

**{**

**int u=Map[omiga.substr(j,1)];**

**set<string> FIRSTt;**

**FIRSTt.clear();**

**if(j+1<Size)**

**{**

**FIRSTt=getFIRST(omiga.substr(j+1,Size-(j+1)));**

**for(set<string>::iterator it=FIRSTt.begin(); it!=FIRSTt.end(); it++)**

**{**

**string t=(\*it);**

**if(t!=kesi)FOLLOW[u].insert(t);**

**}**

**}**

**if(A!=B&&(j==Size-1||FIRSTt.find(kesi)!=FIRSTt.end()))**

**{**

**if(!vis1[Map[A]])getFOLLOW(A);**

**for(set<string>::iterator it=FOLLOW[Map[A]].begin(); it!=FOLLOW[Map[A]].end(); it++)**

**{**

**string t=(\*it);**

**FOLLOW[Map[B]].insert(t);**

**}**

**}**

**}**

**}**

**return FOLLOW[x];**

**}**

**int main()**

**{**

**freopen("input1.txt","r",stdin);**

**// freopen("output.txt","w",stdout);**

**printf("请输入文法G=(VT,VN,S,Phi),其中:\n");**

**printf("请输入终结符号集合VT(以单个符号进行表示),(先输入数目，下一行为终结符号，中间以空格分开):\n");**

**int idx=0;//对终结符与非终结符进行编号**

**cin>>number;**

**while(number--)**

**{**

**string s;**

**cin>>s;**

**if(!Map.count(s))**

**{**

**re[idx]=s;**

**Map[s]=idx++;**

**}**

**VT.insert(s);**

**}**

**Map["$"]=idx++;**

**printf("请输入非终结符号集合VN(以大写单个字母进行表示)，(先输入数目，下一行为非终结符号，中间以空格分开):\n");**

**cin>>number;**

**while(number--)**

**{**

**string s;**

**cin>>s;**

**if(!Map.count(s))**

**{**

**re[idx]=s;**

**Map[s]=idx++;**

**}**

**VN.insert(s);**

**}**

**printf("请输入文法G的开始符号S:\n");**

**cin>>S;**

**printf("请输入文法G的产生式集合Phi(先输入数目m，之后m行为产生式，形式为alpha beta,表示alpha->beta，中间以空格分开，以kesi表示空):\n");**

**int phinum;**

**cin>>phinum;**

**Phi.clear();**

**while(phinum--)**

**{**

**string alpha,beta;**

**cin>>alpha>>beta;**

**Phi.push\_back(make\_pair(alpha,beta));**

**G[Map[alpha]].push\_back(beta);**

**}**

**for(int i=0; i<Phi.size(); i++)**

**{**

**string alpha=Phi[i].first,beta=Phi[i].second;**

**cout<<alpha<<"->"<<beta<<endl;**

**}**

**//终止符号FIRST集的构造**

**for(set<string>::iterator it=VT.begin(); it!=VT.end(); it++)**

**{**

**string x=(\*it);**

**FIRST[Map[x]].insert(x);**

**vis[Map[x]]=1;//表示X的FIRST集已经计算过，在后面求非终止符号的FIRST里的记忆花搜索有用**

**}**

**//非终止符号FIRST集的构造**

**for(set<string>::iterator it=VN.begin(); it!=VN.end(); it++)**

**{**

**string x=(\*it);**

**FIRST[Map[x]]=getFIRST(x);**

**}**

**for(set<string>::iterator it=VN.begin(); it!=VN.end(); it++)**

**{**

**cout<<"FIRST["<<(\*it)<<"]={";**

**string X=(\*it);**

**bool first=true;**

**for(set<string>::iterator it1=FIRST[Map[X]].begin(); it1!=FIRST[Map[X]].end(); it1++)**

**{**

**if(first)**

**{**

**cout<<(\*it1);**

**first=false;**

**}**

**else cout<<","<<(\*it1);**

**}**

**cout<<"}"<<endl;**

**}**

**FOLLOW[Map[S]].insert("$");**

**for(set<string>::iterator it=VN.begin(); it!=VN.end(); it++)**

**{**

**string B=(\*it);**

**FOLLOW[Map[B]]=getFOLLOW(B);**

**}**

**for(set<string>::iterator it=VN.begin(); it!=VN.end(); it++)**

**{**

**cout<<"FOLLOW["<<(\*it)<<"]={";**

**string X=(\*it);**

**bool first=true;**

**for(set<string>::iterator it1=FOLLOW[Map[X]].begin(); it1!=FOLLOW[Map[X]].end(); it1++)**

**{**

**if(first)**

**{**

**cout<<(\*it1);**

**first=false;**

**}**

**else cout<<","<<(\*it1);**

**}**

**cout<<"}"<<endl;**

**}**

**int delta=VT.size();**

**delta++;**

**for(int i=0; i<VN.size(); i++)**

**for(int j=0; j<VT.size(); j++)M[i][j]=make\_pair("","");**

**for(int i=0; i<Phi.size(); i++)**

**{**

**string A=Phi[i].first,alpha=Phi[i].second;**

**set<string> FIRSTalpha=getFIRST(alpha);**

**cout<<A<<"->"<<alpha<<endl;**

**cout<<"FIRST[Alpha]={";**

**bool first=true;**

**for(set<string>::iterator it=FIRSTalpha.begin(); it!=FIRSTalpha.end(); it++)**

**{**

**if(first)**

**{**

**first=false;**

**cout<<(\*it);**

**}**

**else cout<<","<<(\*it);**

**}**

**cout<<"}"<<endl;**

**cout<<"FOLLOW["<<A<<"]={";**

**first=true;**

**for(set<string>::iterator it=FOLLOW[Map[A]].begin(); it!=FOLLOW[Map[A]].end(); it++)**

**{**

**if(first)**

**{**

**first=false;**

**cout<<(\*it);**

**}**

**else cout<<","<<(\*it);**

**}**

**cout<<"}"<<endl;**

**for(set<string>::iterator it=FIRSTalpha.begin(); it!=FIRSTalpha.end(); it++)**

**{**

**string a=(\*it);**

**if(VT.find(a)!=VT.end())**

**{**

**M[Map[A]-delta][Map[a]]=Phi[i];**

**cout<<A<<' '<<a<<' '<<Phi[i].first<<"->"<<Phi[i].second<<endl;;**

**}**

**}**

**if(FIRSTalpha.find(kesi)!=FIRSTalpha.end())**

**for(set<string>::iterator it=FOLLOW[Map[A]].begin(); it!=FOLLOW[Map[A]].end(); it++)**

**{**

**string b=(\*it);**

**M[Map[A]-delta][Map[b]]=Phi[i];**

**cout<<A<<' '<<b<<' '<<Phi[i].first<<"->"<<Phi[i].second<<endl;;**

**}**

**}**

**return 0;**

**}**

# LR实现 自底向上分析（方法3）

## 构造识别所有活前缀的DFA。

**构造扩展文法**

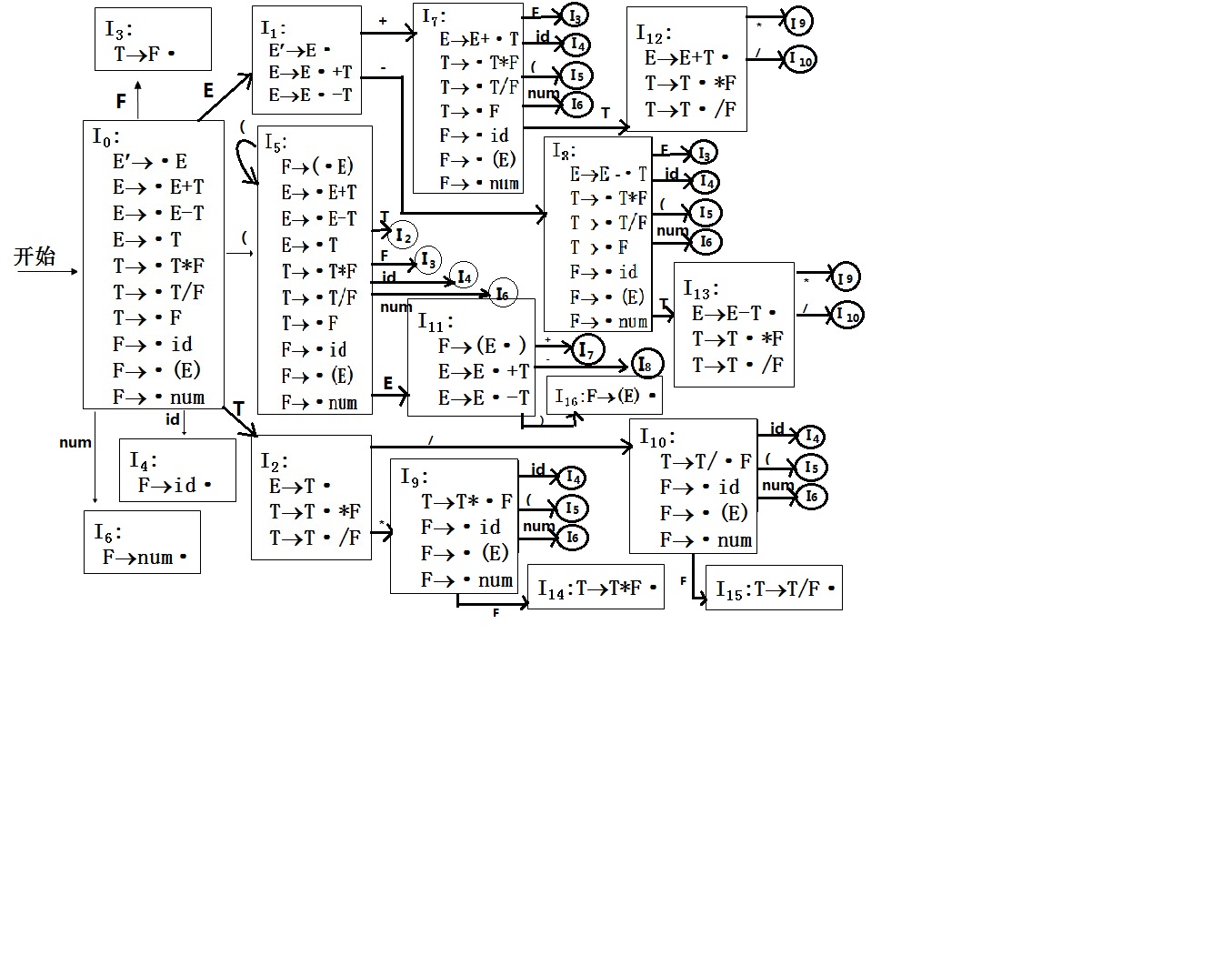




**FIRST和FOLLOW集**如下

|  |  |  |  |
| --- | --- | --- | --- |
|  | **E** | **T** | **F** |
| **FIRST** | id, (, num | id, (, num | id, (, num |
| **FOLLOW** | $, ), +, - | $, ), +, -, \*, / | $, ), +, -, \*, / |

**构造识别所有活前缀的DFA**如下



## 构造LR分析表。

** (1)  (4)  (7)**

** (2)  (5)  (8)**

** (3)  (6)  (9)**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **状态** | **action** | | | | | | | | | **goto** | | |
| **+** | **-** | **\*** | **/** | **id** | **num** | **(** | **)** | **$** | **E** | **T** | **F** |
| **0** |  |  |  |  | **s4** | **S6** | **S5** |  |  | **1** | **2** | **3** |
| **1** | **s7** | **s8** |  |  |  |  |  |  | **acc** |  |  |  |
| **2** | **r3** | **r3** | **s9** | **s10** |  |  |  | **r3** | **r3** |  |  |  |
| **3** | **r6** | **r6** | **r6** | **r6** |  |  |  | **r6** | **r6** |  |  |  |
| **4** | **r7** | **r7** | **r7** | **r7** |  |  |  | **r7** | **r7** |  |  |  |
| **5** |  |  |  |  | **s4** | **s6** | **s5** |  |  | **11** | **2** | **3** |
| **6** | **r9** | **r9** | **r9** | **r9** |  |  |  | **r9** | **r9** |  |  |  |
| **7** |  |  |  |  | **s4** | **s6** | **s5** |  |  |  | **12** | **3** |
| **8** |  |  |  |  | **s4** | **s6** | **s5** |  |  |  | **13** | **3** |
| **9** |  |  |  |  | **s4** | **s6** | **s5** |  |  |  |  | **14** |
| **10** |  |  |  |  | **s4** | **s6** | **s5** |  |  |  |  | **15** |
| **11** | **s7** | **s8** |  |  |  |  |  | **s16** |  |  |  | **16** |
| **12** | **r1** | **r1** | **s9** | **s10** |  |  |  | **r1** | **r1** |  |  |  |
| **13** | **r2** | **r2** | **s9** | **s10** |  |  |  | **r2** | **r2** |  |  |  |
| **14** | **r4** | **r4** | **r4** | **r4** |  |  |  | **r4** | **r4** |  |  |  |
| **15** | **r5** | **r5** | **r5** | **r5** |  |  |  | **r5** | **r5** |  |  |  |
| **16** | **r8** | **r8** | **r8** | **r8** |  |  |  | **r8** | **r8** |  |  |  |

## 编程实现算法4.3， 构造LR分析程序。

**#include<iostream>**

**#include<stack>**

**#include<queue>**

**#include<fstream>**

**#include<cstring>**

**using namespace std;**

**string AnalyzeChart[16][11]=**

**{//LR文法的分析表**

**"s5", "e1", "e1", "s4", "e2", "e1", "1", "2", "3", "e1", "e1",**

**"e3", "s6", "e1", "e3", "e2", "acc", "error", "error", "error", "s7", "e1",**

**"e6", "r2", "s8", "e6", "r2", "r2", "error", "error", "error", "r2", "s9",**

**"e6", "r4", "r4", "e6", "r4", "r4", "error", "error", "error", "r4", "r4",**

**"s5", "e1", "e1", "s4", "e2", "e1", "10", "2", "3", "e1", "e1",**

**"e5", "r6", "r6", "e3", "r6", "r6", "error", "error", "error", "r6", "r6",**

**"s5", "e1", "e1", "s4", "e2", "e1", "error", "11", "3", "e1", "e1",**

**"s5", "e1", "e1", "s4", "e2", "e1", "error", "12", "3", "e1", "e1",**

**"s5", "e1", "e1", "s4", "e2", "e1", "error", "error", "13", "e1", "e1",**

**"s5", "e1", "e1", "s4", "e1", "e1", "error", "error", "14", "e1", "e1",**

**"e3", "s6", "e1", "e3", "s15", "r3", "error", "error", "error", "s9", "e1",**

**"e6", "r1", "s8", "e6", "r1", "r1", "error", "error", "error", "r1", "s9",**

**"e6", "r7", "s8", "e6", "r7", "r7", "error", "error", "error", "r7", "s9",**

**"e6", "r3", "r3", "e6", "r3", "r3", "error", "error", "error", "r3", "r3",**

**"e6", "r8", "r8", "e6", "r8", "r8", "error", "error", "error", "r8", "r8",**

**"e6", "r5", "r5", "e6", "r5", "r5", "error", "error", "error", "r5", "r5",**

**};**

**stack<float> digit;**

**queue<float> digit1;**

**queue<int> inputid;**

**char \*FinalSymbol[6]=**

**{**

**"i","+","\*","(",")","#"//终结符**

**};**

**char \*UnfinalSymbol[5]=**

**{**

**"E","E'","T","T'","F"//非终结符**

**};**

**stack<char> ufstack;//状态栈**

**void initialize()**

**{**

**ufstack.push('0');**

**inputid.push(0);**

**}**

**char \* keyword[6] = {"for","if","then","else","while","do"};**

**int flag1=-1,flag2=-1;**

**struct sign**

**{**

**int line;**

**int token;**

**char temp[10];**

**int i;**

**};**

**void reset(sign & s)**

**{**

**s.i = 0;**

**memset(s.temp,'$',10);**

**s.token = -1;**

**}**

**bool is\_key\_word(sign & s)**

**{**

**int i,r;**

**for(i=0;i<6;i++)**

**{**

**r = memcmp(s.temp,keyword[i],s.i);**

**if(r == 0)**

**{**

**s.token = i+1;**

**return true;**

**break;**

**}**

**}**

**return false;**

**}**

**bool is\_id(sign & s)**

**{**

**int i;**

**bool b = false;**

**if( (s.temp[0]>='a' && s.temp[0]<='z') | (s.temp[0]>='A' && s.temp[0]<='Z') )**

**b = true;**

**else**

**return b;**

**for(i=1;i<strlen(s.temp);i++)**

**{**

**if( (s.temp[i]>='a' && s.temp[i]<='z') | (s.temp[i]>='A' && s.temp[i]<='Z') | (s.temp[i]>='0' && s.temp[i]<='9'))**

**;**

**else**

**{**

**b = false;**

**return b;**

**}**

**}**

**return b;**

**}**

**bool is\_num(sign & s)**

**{**

**digit.push(0);**

**bool b = true;**

**char c;**

**int state = 12;**

**for(int i=0;i<strlen(s.temp);i++)**

**{**

**c = s.temp[i];**

**//int tn;**

**switch(state)**

**{**

**case 12:**

**{**

**if(c>='0' && c<='9' || c=='-')**

**{**

**state = 13;**

**}**

**else**

**return false;**

**break;**

**}**

**case 13:**

**{**

**if(c>='0' && c<='9') state = 13;**

**else if(c=='.') state=14;**

**else if(c=='E' || c=='e') state=16;**

**else return false;**

**break;**

**}**

**case 14:**

**{**

**if(c>='0' && c<='9') state=15;**

**else return false;**

**break;**

**}**

**case 15:**

**{**

**if(c>='0' && c<='9') state=15;**

**else if(c=='E' || c=='e') state=16;**

**else return false;**

**break;**

**}**

**case 16:**

**{**

**if(c=='+' || c=='-') state=17;**

**else if(c>='0' && c<='9') state=18;**

**else return false;**

**break;**

**}**

**case 17:**

**{**

**if(c>='0' && c<='9') state=18;**

**else return false;**

**break;**

**}**

**case 18:**

**{**

**if(c>='0' && c<='9') state=18;**

**else return false;**

**break;**

**}**

**}**

**}**

**return b;**

**}**

**int handle(sign & s)**

**{**

**s.temp[s.i] = '\0';**

**if(strlen(s.temp)==0)**

**return 0;**

**if(is\_key\_word(s))**

**{**

**cout<<"("<<s.token<<","<<keyword[s.token-1]<<")";**

**inputid.push(s.token);**

**return s.token;**

**}**

**else if(is\_id(s))**

**{**

**cout<<"(10,"<<s.temp<<")";**

**inputid.push(10);**

**return 10;**

**}**

**else if(is\_num(s))**

**{**

**cout<<"(11,"<<s.temp<<")";**

**inputid.push(11);**

**int i=0;**

**float n=0;**

**int n1=0;**

**while(i<s.i)**

**{ n1=n1\*10;**

**if(s.temp[i]=='.')**

**{**

**n1=1;**

**i++;**

**}**

**else**

**{**

**n=n\*10+(s.temp[i]-'0');**

**i++;**

**}**

**}**

**if(n1!=0)**

**n=n/n1;**

**// cout<<s.i<<" "<<s.temp<<" "<<n<<endl;**

**digit1.push(n);**

**n=0;**

**reset(s);**

**return 11;**

**}**

**else**

**{**

**return 0;**

**}**

**reset(s);**

**}**

**void scan(FILE \* fp)**

**{**

**sign mark = {1,-1,{'$','$','$','$','$','$','$','$','$','$'},0};**

**char ch;**

**do**

**{**

**ch = fgetc(fp);**

**if(ch=='\n')**

**{**

**mark.line++;**

**handle(mark);**

**}**

**else if(ch==' ')**

**handle(mark);**

**else if(ch=='+')**

**{**

**handle(mark);**

**cout<<"(13,"<<ch<<")";**

**inputid.push(13);**

**}**

**else if(ch=='-')**

**{**

**handle(mark);**

**cout<<"(14,"<<ch<<")";**

**inputid.push(14);**

**}**

**else if(ch=='\*')**

**{**

**handle (mark);**

**cout<<"(15,"<<ch<<")";**

**inputid.push(15);**

**}**

**else if(ch=='/')**

**{**

**handle (mark);**

**cout<<"(16,"<<ch<<")";**

**inputid.push(16);**

**}**

**else if(ch==':')**

**{**

**handle (mark);**

**mark.temp[mark.i++] = ch;**

**cout<<"(17,"<<ch<<")";**

**inputid.push(17);**

**}**

**else if(ch=='<')**

**{**

**handle (mark);**

**cout<<"(20,"<<ch<<")";**

**inputid.push(20);**

**}**

**else if(ch=='>')**

**{**

**handle (mark);**

**cout<<"(23,"<<ch<<")";**

**inputid.push(23);**

**}**

**else if(ch=='=')**

**{**

**if(mark.temp[mark.i-1]==':')**

**{**

**cout<<"(18,:=)";**

**reset(mark);**

**inputid.push(18);**

**}**

**else**

**{**

**handle(mark);**

**cout<<"(25,"<<ch<<")";**

**inputid.push(25);**

**}**

**}**

**else if(ch==';')**

**{**

**handle (mark);**

**cout<<"(26,"<<ch<<")";**

**inputid.push(26);**

**}**

**else if(ch=='(')**

**{**

**handle (mark);**

**cout<<"(27,"<<ch<<")";**

**inputid.push(27);**

**}**

**else if(ch==')')**

**{**

**handle (mark);**

**cout<<"(28,"<<ch<<")";**

**inputid.push(28);**

**}**

**else**

**{**

**mark.temp[mark.i++] = ch;**

**handle(mark);**

**}**

**}while(ch!='#');**

**cout<<endl;**

**}**

**int main(int argc ,char \*argv[])**

**{**

**initialize();//初始化stack**

**FILE \* fp;**

**fp = fopen("Input.txt","r");**

**scan(fp);**

**//cout<<inputid.size()<<endl;**

**string temp;**

**int h=0;**

**int z=30;**

**inputid.pop();**

**inputid.push(0);**

**while((!inputid.empty()) && z>0)**

**{**

**z--;**

**while(true)**

**{**

**int state=-1,sflag2=0;**

**char temp1=ufstack.top();**

**int sflag1=inputid.front();**

**//cout<<sflag1<<endl;**

**{**

**if(sflag1==11)**

**flag1=0;**

**else if(sflag1==13)**

**flag1=1;**

**else if(sflag1==15)**

**flag1=2;**

**else if(sflag1==27)**

**flag1=3;**

**else if(sflag1==28)**

**flag1=4;**

**else if(sflag1==0)**

**flag1=5;**

**else if(sflag1==14)**

**flag1=9;**

**else if(sflag1==16)**

**flag1=10;**

**else flag1=-1;**

**}**

**if(flag1==-1)**

**{**

**cout<<"Unexpected symbol!"<<endl;**

**}**

**state=temp1-'0';**

**// cout<<state<<"VVVVV"<<flag1<<endl;**

**char sta=0;**

**char t1=ufstack.top();**

**ufstack.pop();**

**if(!ufstack.empty() && ufstack.top()>='0' && ufstack.top()<='9')**

**{**

**sta=ufstack.top()-'0';**

**if(sta>0)**

**state=state+sta\*10;**

**}**

**else**

**ufstack.push(t1);**

**cout<<"("<<state<<", "<<flag1<<") ------"<<AnalyzeChart[state][flag1].c\_str ()<<"-----"<<endl;**

**if(strcmp(AnalyzeChart[state][flag1].c\_str (),"s5")==0)//移近**

**{**

**ufstack.push('i');**

**ufstack.push('5');**

**cout<<"shiftid"<<endl;**

**//ip++;**

**digit.push(digit1.front());**

**digit1.pop();**

**cout<<inputid.front();**

**inputid.pop();**

**}**

**else if(strcmp(AnalyzeChart[state][flag1].c\_str (),"s4")==0)//移近**

**{**

**ufstack.push('(');**

**ufstack.push('4');**

**cout<<"shift ( "<<endl;**

**//ip++;**

**inputid.pop();**

**}**

**else if(strcmp(AnalyzeChart[state][flag1].c\_str (),"s6")==0)//移近**

**{**

**ufstack.push('+');**

**ufstack.push('6');**

**cout<<"shift + "<<endl;**

**//ip++;**

**inputid.pop();**

**}**

**else if(strcmp(AnalyzeChart[state][flag1].c\_str (),"s7")==0)//移近**

**{**

**ufstack.push('-');**

**ufstack.push('7');**

**cout<<"shift - "<<endl;**

**//ip++;**

**inputid.pop();**

**}**

**else if(strcmp(AnalyzeChart[state][flag1].c\_str (),"s8")==0)//移近**

**{**

**ufstack.push('\*');**

**ufstack.push('8');**

**cout<<"shift \* "<<endl;**

**//ip++;**

**inputid.pop();**

**}**

**else if(strcmp(AnalyzeChart[state][flag1].c\_str (),"s9")==0)//移近**

**{**

**ufstack.push('/');**

**ufstack.push('9');**

**cout<<"shift / "<<endl;**

**//ip++;**

**inputid.pop();**

**}**

**else if(strcmp(AnalyzeChart[state][flag1].c\_str (),"r1")==0)//reduce**

**{**

**for(int i=0;i<6;i++)**

**ufstack.pop();**

**cout<<"reduce E->E+T"<<endl;**

**float ntop=digit.top();**

**//digit.pop();**

**digit.pop();**

**cout<<ntop<<"+"<<digit.top()<<"=";**

**ntop+=digit.top();**

**digit.pop();**

**digit.push(ntop);**

**cout<<digit.top()<<endl;**

**int st=ufstack.top()-'0';**

**ufstack.push('E');**

**char ch=\*AnalyzeChart[st][6].c\_str ();**

**if(strcmp(AnalyzeChart[st][6].c\_str (),"10")==0)**

**{**

**ufstack.push(\*AnalyzeChart[st][6].c\_str ());**

**ufstack.push(\*(AnalyzeChart[st][6].c\_str ()+1));**

**}**

**else**

**ufstack.push(ch);**

**}**

**else if(strcmp(AnalyzeChart[state][flag1].c\_str (),"r7")==0)//reduce**

**{**

**for(int i=0;i<6;i++)**

**ufstack.pop();**

**cout<<"reduce E->E-T"<<endl;**

**float ntop=digit.top();**

**//digit.pop();**

**digit.pop();**

**cout<<digit.top()<<"-"<<ntop<<"=";**

**ntop=digit.top()-ntop;**

**digit.pop();**

**digit.push(ntop);**

**cout<<digit.top()<<endl;**

**int st=ufstack.top()-'0';**

**ufstack.push('E');**

**char ch=\*AnalyzeChart[st][6].c\_str ();**

**if(strcmp(AnalyzeChart[st][6].c\_str (),"10")==0)**

**{**

**ufstack.push(\*AnalyzeChart[st][6].c\_str ());**

**ufstack.push(\*(AnalyzeChart[st][6].c\_str ()+1));**

**}**

**else**

**ufstack.push(ch);**

**}**

**else if(strcmp(AnalyzeChart[state][flag1].c\_str (),"r2")==0)//reduce**

**{**

**ufstack.pop();**

**ufstack.pop();**

**cout<<"reduce E->T"<<endl;**

**int st=ufstack.top()-'0';**

**ufstack.push('E');**

**char ch=\*AnalyzeChart[st][6].c\_str ();**

**if(strcmp(AnalyzeChart[st][6].c\_str (),"10")==0)**

**{**

**ufstack.push(\*AnalyzeChart[st][6].c\_str ());**

**ufstack.push(\*(AnalyzeChart[st][6].c\_str ()+1));**

**}**

**else**

**ufstack.push(ch);**

**}**

**else if(strcmp(AnalyzeChart[state][flag1].c\_str (),"r3")==0)//reduce**

**{**

**for(int i=0;i<6;i++)**

**ufstack.pop();**

**cout<<"reduce T->T\*F"<<endl;**

**float ntop=digit.top();**

**digit.pop();**

**// digit.pop();**

**cout<<ntop<<"\*"<<digit.top()<<"=";**

**ntop=ntop\*digit.top();**

**digit.pop();**

**digit.push(ntop);**

**cout<<ntop<<endl;**

**int st=ufstack.top()-'0';**

**ufstack.push('T');**

**char ch=\*AnalyzeChart[st][7].c\_str ();**

**if(strcmp(AnalyzeChart[st][7].c\_str (),"11")==0)**

**{**

**ufstack.push(\*AnalyzeChart[st][7].c\_str ());**

**ufstack.push(\*(AnalyzeChart[st][7].c\_str ()+1));**

**}**

**else if(strcmp(AnalyzeChart[st][7].c\_str (),"12")==0)**

**{**

**ufstack.push(\*AnalyzeChart[st][7].c\_str ());**

**ufstack.push(\*(AnalyzeChart[st][7].c\_str ()+1));**

**}**

**else**

**ufstack.push(ch);**

**}**

**else if(strcmp(AnalyzeChart[state][flag1].c\_str (),"r8")==0)//reduce**

**{**

**for(int i=0;i<6;i++)**

**ufstack.pop();**

**cout<<"reduce T->T/F"<<endl;**

**float ntop=digit.top();**

**digit.pop();**

**// digit.pop();**

**cout<<digit.top()<<"/"<<ntop<<"=";**

**if(ntop!=0)**

**ntop=digit.top()/ntop;**

**else**

**{**

**cout<<"error,除数不能是零"<<endl;**

**ntop=0;**

**}**

**cout<<ntop<<endl;**

**digit.pop();**

**digit.push(ntop);**

**//cout<<ntop<<"\_\_\_\_\_\_\_\_\_"<<digit.top()<<endl;**

**int st=ufstack.top()-'0';**

**ufstack.push('T');**

**char ch=\*AnalyzeChart[st][7].c\_str ();**

**if(strcmp(AnalyzeChart[st][7].c\_str (),"11")==0)**

**{**

**ufstack.push(\*AnalyzeChart[st][7].c\_str ());**

**ufstack.push(\*(AnalyzeChart[st][7].c\_str ()+1));**

**}**

**else if(strcmp(AnalyzeChart[st][7].c\_str (),"12")==0)**

**{**

**ufstack.push(\*AnalyzeChart[st][7].c\_str ());**

**ufstack.push(\*(AnalyzeChart[st][7].c\_str ()+1));**

**}**

**else**

**ufstack.push(ch);**

**}**

**else if(strcmp(AnalyzeChart[state][flag1].c\_str (),"r4")==0)//reduce**

**{**

**ufstack.pop();**

**ufstack.pop();**

**cout<<"reduce T->F"<<endl;**

**int st=ufstack.top()-'0';**

**ufstack.push('T');**

**char ch=\*AnalyzeChart[st][7].c\_str ();**

**if(strcmp(AnalyzeChart[st][7].c\_str (),"11")==0)**

**{**

**ufstack.push(\*AnalyzeChart[st][7].c\_str ());**

**ufstack.push(\*(AnalyzeChart[st][7].c\_str ()+1));**

**}**

**else if(strcmp(AnalyzeChart[st][7].c\_str (),"12")==0)**

**{**

**ufstack.push(\*AnalyzeChart[st][7].c\_str ());**

**ufstack.push(\*(AnalyzeChart[st][7].c\_str ()+1));**

**}**

**else**

**ufstack.push(ch);**

**}**

**else if(strcmp(AnalyzeChart[state][flag1].c\_str (),"r5")==0)//reduce**

**{**

**for(int i=0;i<6;i++)**

**ufstack.pop();**

**cout<<"reduce F->(E)"<<endl;**

**int st=ufstack.top()-'0';**

**ufstack.push('F');**

**char ch=\*AnalyzeChart[st][8].c\_str ();**

**if(strcmp(AnalyzeChart[st][8].c\_str (),"13")==0)**

**{**

**ufstack.push(\*AnalyzeChart[st][8].c\_str ());**

**ufstack.push(\*(AnalyzeChart[st][8].c\_str ()+1));**

**}**

**else if(strcmp(AnalyzeChart[st][8].c\_str (),"14")==0)**

**{**

**ufstack.push(\*AnalyzeChart[st][8].c\_str ());**

**ufstack.push(\*(AnalyzeChart[st][8].c\_str ()+1));**

**}**

**else**

**ufstack.push(ch);**

**}**

**else if(strcmp(AnalyzeChart[state][flag1].c\_str (),"r6")==0)//reduce**

**{**

**for(int i=0;i<2;i++)**

**ufstack.pop();**

**cout<<"reduce F->id"<<endl;**

**int st=ufstack.top()-'0';**

**ufstack.push('F');**

**char ch=\*AnalyzeChart[st][8].c\_str ();**

**if(strcmp(AnalyzeChart[st][8].c\_str (),"13")==0)**

**{**

**ufstack.push(\*AnalyzeChart[st][8].c\_str ());**

**ufstack.push(\*(AnalyzeChart[st][8].c\_str ()+1));**

**}**

**else if(strcmp(AnalyzeChart[st][8].c\_str (),"14")==0)**

**{**

**ufstack.push(\*AnalyzeChart[st][8].c\_str ());**

**ufstack.push(\*(AnalyzeChart[st][8].c\_str ()+1));**

**}**

**else**

**ufstack.push(ch);**

**}**

**else if(strcmp(AnalyzeChart[state][flag1].c\_str (),"acc")==0)**

**{**

**cout<<"Analyze complate! "<<endl;**

**inputid.pop();**

**break;**

**}**

**else if(strcmp(AnalyzeChart[state][flag1].c\_str (),"e2")==0)**

**{**

**cout<<"不配对的右括号！ "<<endl;**

**inputid.pop();**

**}**

**else if(strcmp(AnalyzeChart[state][flag1].c\_str (),"e3")==0)**

**{**

**cout<<"缺少运算符 "<<endl;**

**inputid.pop();**

**}**

**else if(strcmp(AnalyzeChart[state][flag1].c\_str (),"e1")==0)**

**{**

**cout<<"缺少运算对象 "<<endl;**

**//ufstack.push('i');**

**//ufstack.push('5');**

**inputid.pop();**

**}**

**else if(strcmp(AnalyzeChart[state][flag1].c\_str (),"e4")==0)**

**{**

**cout<<"缺少右括号 "<<endl;**

**ufstack.push(')');**

**ufstack.push('11');**

**}**

**else if(strcmp(AnalyzeChart[state][flag1].c\_str (),"e5")==0)**

**{**

**cout<<"算符重复 "<<endl;**

**inputid.pop();**

**}**

**else //M[X,a]=一个产生式**

**{**

**cout<<"Strange error happen !"<<endl;**

**break;**

**}**

**}**

**h++;**

**}**

**cout<<"最终结果是"<<digit.top()<<endl;**

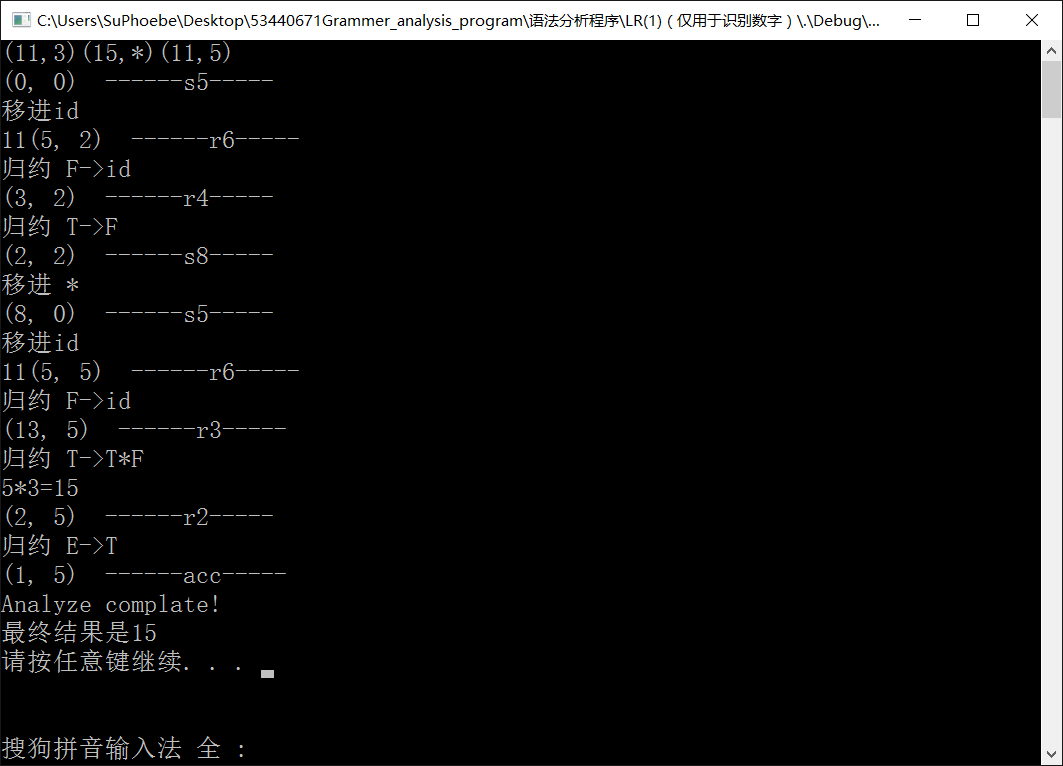
**system("pause");**

**return 0;**

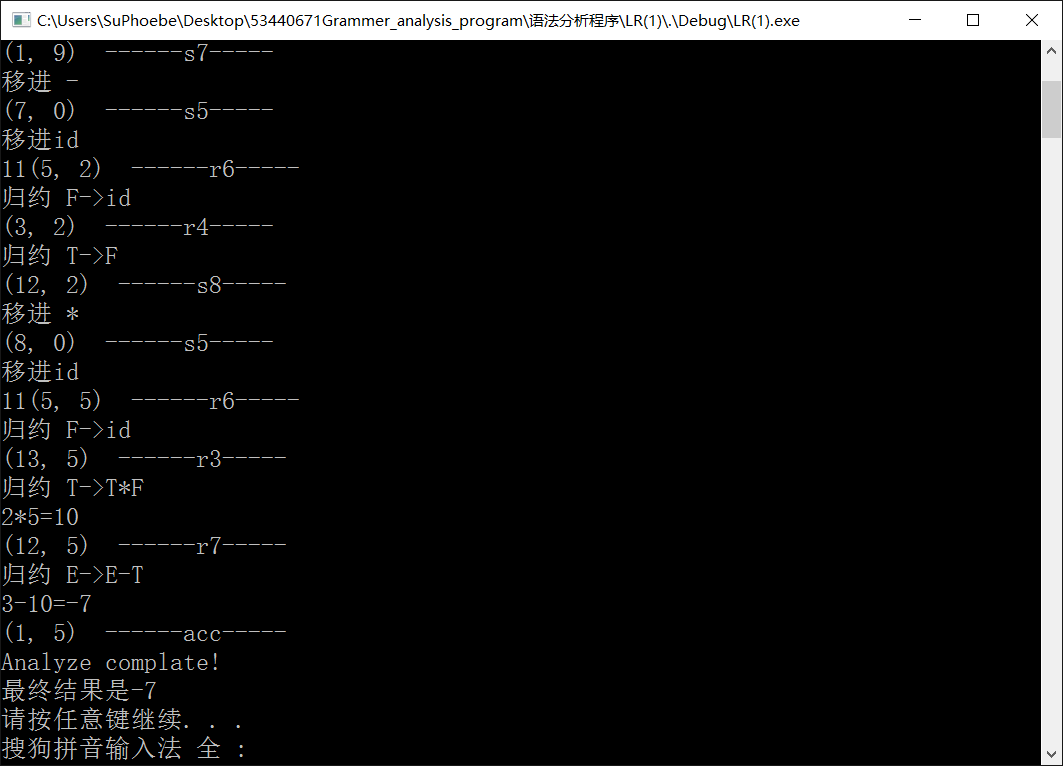
**}**

## 运行结果截图：

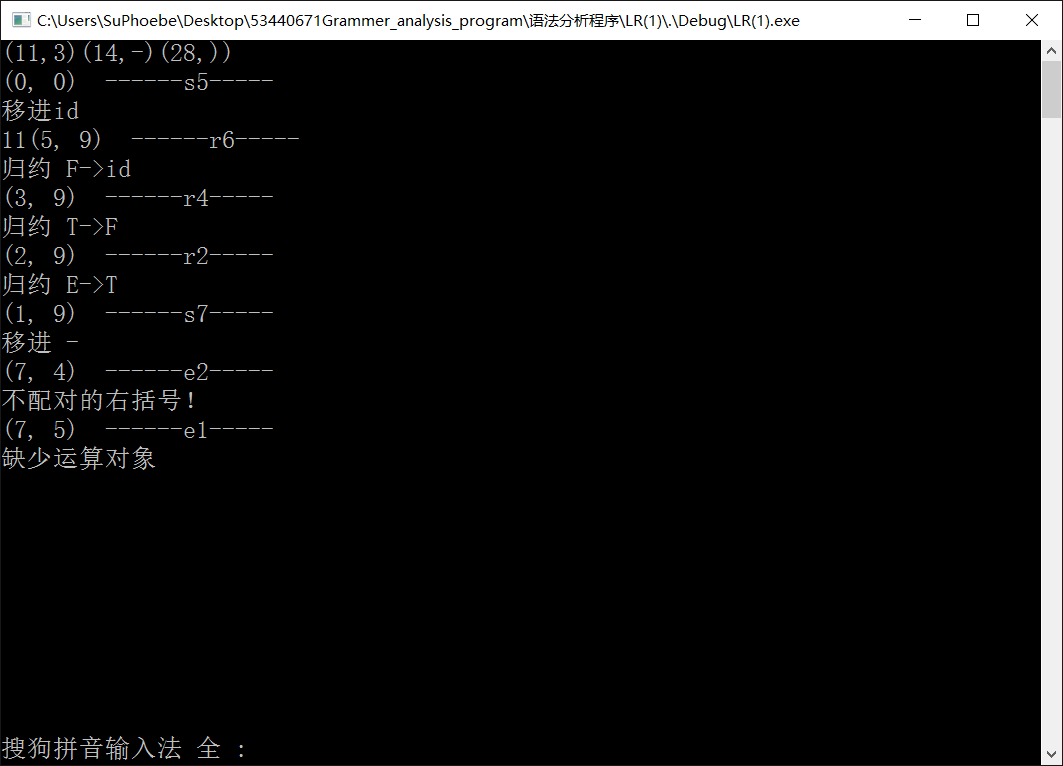
1. **输入符号串为 5\*3**



1. **输入符号串为3-5\*(2)**



1. **错误处理示例，输入符号串为3-)**



# 实验总结

上周天写完了代码和实验总结，周四学委通知我们更改需求了。文法改成了Pascal的文法，而我因为没有按照全部文法的分析，所以就没法改了。

只写了一份求FIRST集合和FELLOW集合的代码是适用于所有文法的。

代码部分全手撸，程序截图也是自己完成，其他文法分析图片等借用网上资源。