题目：语法分析程序的设计与实现

实验内容：

编写 LL(1)语法分析程序，实现对算术表达式的语法分析。所分析算术表达式由如下文法产生：

E ->E+T|E-T|T

T ->T*F|T/F|F

F-> id|(E)|num

实验要求：

(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。

(3) 再对输入表达式进行分析的过程中，输出所采用的产生式。

程序代码：

```
#include<iostream>

#include<fstream>

#include<cstdlib>

#include<vector>

#include<stack>

#include<string>

#include"sort.h"

#include<iomanip>
```

```
#include<ctype.h>
#define N 10
using namespace std;
template<typename Bitr>
int search(Bitr s,vector<Bitr> p);
int begin_(vector<vector<int>>produc_,int p){
     int i;
    for( i=0;produc_[i][0]!=p;i++);
    return i;
}
int end_(vector<vector<int>>produc_,int p){
     int i,t(0);
    for( i=0;i<produc_.size();i++)
    {
      if(produc_[i][0]==p)
         t=1;
      else if(t==1)
         return i-1;
    }
    return i-1;
}
template<typename Bitr>
```

```
bool add(Bitr t,vector<Bitr>&a){
    int index=search<Bitr>(t,a);
    if(index<0){
      a.push_back(t);
      return true;
    }
    return false;
}
/******************FIRST 集合构造*************/
void  first(vector<vector<int>>&first_,vector<vector<int>>produc_,int
p,int null_index){
   bool bl_;
   if(!first_[p].empty())
      return;
   int begin,end,index(0);
   begin=begin_(produc_,p);
   end=end_(produc_,p);
   for(int i=begin;i<=end;i++){
     int m=produc_[i][1];
     if(m>=N)
       bl_=add<int>(m,first_[p]);
     else {
```

```
        first(first_,produc_,m,null_index);
       for(int i=0;i<first_[m].size();i++){
          int a=first_[m][i];
          if(a!=null_index)
              bl_=add<int>(a,first_[p]);
        }
       index=search<int>(null_index,first_[m]);
       int m_;

   for( m_=1;(m_<=produc_[i].size())&&(produc_[i][m_]<N)&&(index>=0);
m_++){
           index=search(null_index,first_[m_]);
       }
       if(m_>produc_[i].size())
           bl_=add<int>(null_index,first_[p]);
       else if(produc_[i][m_]>=N)
           bl_=add<int>(produc_[i][m_],first_[p]);
       else if(m_!=1){
         first(first_,produc_,m_,null_index);
         for(int i=0;i<first_[m_].size();i++){
           int a=first_[m_][i];
           if(a!=null_index)
```

```
                bl_=add<int>(a,first_[p]);
            }
        }
    }
  }
}
/******************FOLLOW 集合构造*************/
void
follow(vector<vector<int>>&follow_,vector<vector<int>>produc_,vector<
vector<int>>first_,int size,int null_index){
   int tag=1;
   int index;
   bool bl_;
   index=size+N;
   follow_[0].push_back(index);  //$
   while(tag==1){
     tag=0;
    for(int i=0;i<produc_.size();i++){
      int m=produc_[i][0];
      int t1,j;
      if(!((produc_[i][1]>=N)&&(produc_[i].size()==2))){
         for(j=1;(j+1)<produc_[i].size();j++){
```

```
 t1=produc_[i][j];
int t2=produc_[i][j+1];
if(t1<N){
   if(t2>=N){
     bl_=add<int>(t2,follow_[t1]);
     if(bl_)
        tag=1;
   }
   else {
      for(int k=0;k<first_[t2].size();k++){
         int t3;
          t3=first_[t2][k];
              if(t3!=null_index){
           bl_=add<int>(t3,follow_[t1]);
             if(bl_)
             tag=1;
         }
     }

   }
 }
}
```

```
  t1=produc_[i][j];
if(t1<N){
if(m!=t1){
   for(int k=0;k<follow_[m].size();k++){
            int t3;
             t3=follow_[m][k];

            bl_=add<int>(t3,follow_[t1]);
              if(bl_)
                tag=1;
     }
}
index=search<int>(null_index,first_[t1]);
if(index>=0){
  t1=produc_[i][j-1];
  for(int k=0;k<follow_[m].size();k++){
            int t3;
             t3=follow_[m][k];

            bl_=add<int>(t3,follow_[t1]);
              if(bl_)
                tag=1;
```

```
            }
        }
       }
      }
     }
   }
}
/****************预测分析表构造*************/
void
table(vector<vector<int>>&table_,vector<vector<int>>produc_,vector<ve
ctor<int>>first_,vector<vector<int>>&follow_,int null_index){
   for(int i=0;i<produc_.size();i++){
        int m=produc_[i][0];
       int n=produc_[i][1];
       if(n==null_index)
          for(int j=0;j<follow_[m].size();j++){
             int t=follow_[m][j];
             table_[m][t-N-1]=i;
          }
       else if(n>=N)
          table_[m][n-N-1]=i;
       else
```

```
            for(int j=0;j<first_[m].size();j++){
               int t=first_[m][j];
               if(t!=null_index)
                 table_[m][t-N-1]=i;
            }
    }
}
/*非递归预测分析算法*/
void
pred_anal(vector<vector<int>>table_,vector<vector<int>>produc_,vector
<string>termi_,vector<string>un_ter_,string s){
     vector<int>temp;
    stack<int> st;
    string s1;
    int t,index,ip(0),x,tg(0);
     char c;

    cout<<setiosflags(ios::left)<<setw(20) <<"栈顶"<<setw(20)<<"输入
"<<setw(20)<<"输出"<<endl;  //<<setiosflags(ios::left)
    s1="$";
    t=search<string>(s1,termi_)+N+1;
    st.push(t);
```

```
st.push(0);
s1.clear();
for(int i=0;i<s.size();i++){
   c=s[i];
   if(islower(c)){
      if(tg==0)
         tg=1;
      s1.push_back(c);
   }
   else if(tg==1){
     index=search<string>(s1,termi_)+N+1;
    temp.push_back(index);
    s1.clear();
    tg=0;
    s1.push_back(c);
    index=search<string>(s1,termi_)+N+1;
    temp.push_back(index);
    s1.clear();
   }
   else{
    s1.push_back(c);
     index=search<string>(s1,termi_)+N+1;
```

```
     temp.push_back(index);
     s1.clear();
    }
}
if(!s1.empty()){
    index=search<string>(s1,termi_)+N+1;
    temp.push_back(index);
    s1.clear();
}
temp.push_back(t);
do{
    x=st.top();
    if(x<N)
      cout<<setw(20)<<un_ter_[x];
    else
       cout<<setw(20)<<termi_[x-N-1];
    for(int i=ip;i<temp.size();i++)
       cout<<termi_[temp[i]-N-1];
    cout<<setw(20)<<' ';
    if(x>=N||x==t){
       if(x==(temp[ip])||x==t){
          st.pop();
```

```
        ip++;
    }
    else {
      cout<<"error"<<endl;
      break;
    }

}
else if(table_[x][temp[ip]-N-1]!=-1){
    st.pop();
    int t=table_[x][temp[ip]-N-1];
    if(produc_[t][1]!=N)
      for(int i=produc_[t].size()-1;i>=1;i--)
         st.push(produc_[t][i]);
    for(int i=0;i<produc_[t].size();i++){
        int m=produc_[t][i];
        if(i==0)
           cout<<un_ter_[m]<<"->";
        else if(m<N)
           cout<<un_ter_[m];
        else if(m==N)
           cout<<"null";
```

```
            else
                cout<<termi_[m-N-1];
        }



    }
    else{
        cout<<"error"<<endl;
        break;
    }
    cout<<endl;

 }while(x!=t);

}
int main(){
  fstream infile;
  infile.open("1.txt",ios::in|ios::binary);
  if(!infile){
  cout<<"cannot open the file"<<endl;
  exit(1);
  }
```

```
 char ch,ch_;
vector<string> termi;
vector<string>un_ter;
vector<vector<int>>produc;
vector<vector<int>>FIRST;
vector<vector<int>>FOLLOW;
vector<vector<int>>pre_table;
 char m[50];
 bool bl_;
 vector<int>n;
 int
flag(1),i(0),index(0),temp,temp1,l_loop=0,t=0,tag=0,null_index=N;
 string s;

 while(infile.peek()!=EOF){
    infile.read((char*)&ch,sizeof(char));
   if(isupper(ch)!=0){
       if(tag==1){
           m[i]='\0';
           index=search<string>(m,termi);
           if(index<0){
              termi.push_back(m);
```

```
            index=termi.size()-1;
        }
        i=0;
        tag=0;
        index+=(N+1);
        n.push_back(index);
    }
if(flag==1){
    ch_=ch;
    m[i++]=ch;
    m[i]='\0';
    index=search<string>(m,un_ter);
    if(index<0){
        un_ter.push_back(m);
        index=un_ter.size()-1;
    }
    i=0;
    temp=index;
}
else if(flag==2){   // new producer
    flag=0;
    if(ch==ch_){
```

```
        l_loop=1;
        if(t==0){
          t=1;
         m[i++]=ch;
         m[i++]='\'';
         m[i]='\0';
         un_ter.push_back(m);
         temp1=un_ter.size ()-1;
         i=0;
        n.push_back(temp1);
        n.push_back(null_index);
        produc.push_back(n);
        n.clear();
     }
    n.push_back(temp1);
}
    else{
         m[i++]=ch;
         m[i]='\0';
           index=search<string>(m,un_ter);
          if(index<0){
           un_ter.push_back(m);
```

```
            index=un_ter.size()-1;
          }
           i=0;
           n.push_back(temp);
           n.push_back(index);
        }
        flag=0;
    }
  else{
        m[i++]=ch;
        m[i]='\0';
        index=search<string>(m,un_ter);
        if(index<0){
          un_ter.push_back(m);
          index=un_ter.size()-1;
        }
        i=0;
        n.push_back(index);
  }
}
else if(ch=='\n'||ch=='|'){
    if(tag==1){
```

```
        m[i]='\0';
        index=search<string>(m,termi);
        if(index<0){
            termi.push_back(m);
            index=termi.size()-1;
        }
        i=0;
        tag=0;
        index+=(N+1);
        n.push_back(index);
}
if(l_loop==1){
 n.push_back(temp1);
 produc.push_back(n);
 n.clear();
}
else if(l_loop==0){
 produc.push_back(n);
 n.clear();
}
if(ch=='\n'){
   flag=1;
```

```
        t=0;
        l_loop=0;
    }
    else
        flag=2;
  }
else if(ch==' ')
      flag=2;
else if(ch!='\r'){
    m[i++]=ch;
    tag=1;
    if(flag==2)
        n.push_back(temp);
    flag=0;
  }

}
insert_sort(produc,produc.size());
s="$";
termi.push_back(s);
 for(i=0;i<produc.size();i++){
    cout<<"产生式("<<i<<')'<<un_ter[produc[i][0]]<<"->";
```

```
    for(int j=1;j<produc[i].size();j++){
        int m=produc[i][j];
        if(m<N)
            cout<<un_ter[m];
        else if(m==N)
            cout<<"null";
        else
            cout<<termi[m-N-1];
    }
    cout<<endl;
}
cout<<"FIRST 集"<<endl;
FIRST.resize(un_ter.size());
for(i=0;i<un_ter.size();i++){
    cout<<un_ter[i]<<":";
    first(FIRST,produc,i,null_index);
    for(int j=0;j<FIRST[i].size();j++){
        int m=FIRST[i][j];
        if(m==null_index)
            cout<<"null"<<'\t';
        else
          cout<<termi[m-N-1]<<'\t';
```

```
    }
    cout<<endl;
}
 FOLLOW.resize(un_ter.size());
follow(FOLLOW,produc,FIRST,termi.size(),null_index);
cout<<"FOLLOW 集"<<endl;
for(i=0;i<un_ter.size();i++){
    cout<<un_ter[i]<<":";
    for(int j=0;j<FOLLOW[i].size();j++)
        cout<<termi[FOLLOW[i][j]-N-1]<<'\t';
    cout<<endl;
}
pre_table.resize(un_ter.size());
for(int row=0;row<un_ter.size();row++)
    for(int col=0;col<termi.size();col++)
        pre_table[row].push_back(-1);
table(pre_table,produc,FIRST,FOLLOW,null_index);
cout<<"预测分析表："<<endl;
cout<<"    ";
for(int col=0;col<termi.size();col++)
    cout<<termi[col]<<'\t';
cout<<endl;
```

```
    for(int row=0;row<un_ter.size();row++){
        cout<<un_ter[row]<<' ';
        for(int col=0;col<termi.size();col++)
            cout<<pre_table[row][col]<<'\t';
        cout<<endl;
    }
 cout<<"enter expression"<<endl;
 cin>>s;
 pred_anal(pre_table,produc,termi,un_ter,s);

  infile.close();
  system("pause");
  return 0;
}
template<typename Bitr>
 int search(Bitr s,vector<Bitr> p){
    int i=0;
   if(p.empty())
      return -1;
   for(i=0;i<p.size();i++)
      if(s==p[i])
         return i;
```

```
    return -1;
  }
```

程序设计说明:

<1>先将文法产生式放到文本文件中，用空格代替“->”，程序从文件中读取表达式。在读取过程中，完成的动作：(1)从表达式中识别出非终结符，并保存在非终结符数组中；(2) 从表达式中识别出终结符，并保存在终结符数组中；(3)对产生式进行消除左递归，空用数字 N 表示；(4)将产生式保存到产生式数组中（非终结符用它们所在数组下标表示，终结符用它们所在数组下标+N+1 表示，即对终结符进行 N+1 偏移，以区别非终结符）

产生式表：

0	E -> T E’
1	E’ -> null
2	E’ -> + T E’
3	E’ -> - T E’
4	T -> F T’
5	T’ -> null
6	T’ -> * F T’
7	T’ -> / F T’
8	F -> id
9	F -> (E)
10	F -> num

<2>调用 first 算法为每个非终结符生成 first 集

FIRST 集

E: id (num

E’: null + -

T: id (num

T': null * /

F: id (num

<3>调用 follow 算法为每个终结符生成 follow 集

FOLLOW 集

E: $)

E': $)

T: + - $)

T': + - $)

F: * / + - $)

<4>调用 pre_table 算法构造预测分析表

预测分析表：

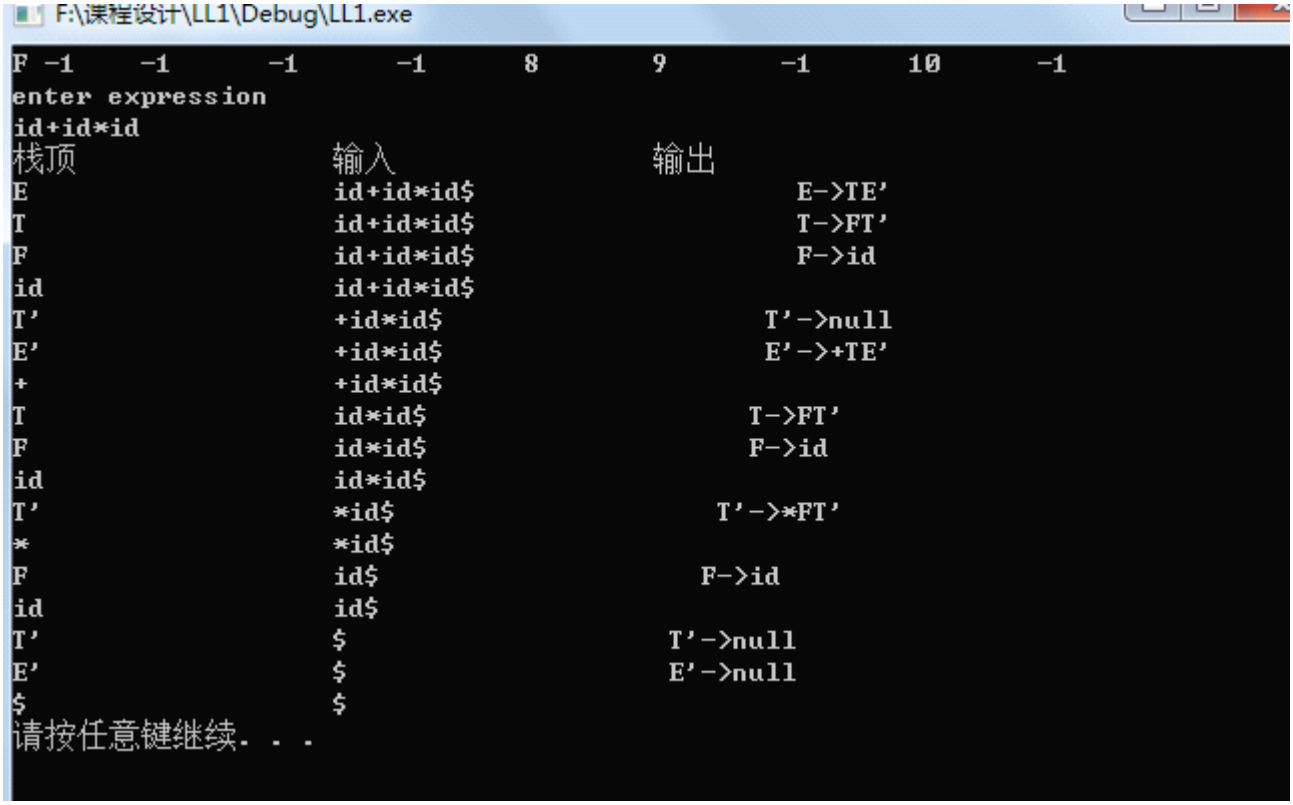
	+	-	*	/	id	(	)	num	$
E	-1	-1	-1	-1	0	0	-1	0	-1
E'	2	3	-1	-1	-1	-1	1	-1	1
T	-1	-1	-1	-1	4	4	-1	4	-1
T'	5	5	6	7	-1	-1	5	-1	5
F	-1	-1	-1	-1	8	9	-1	10	-1

<5>调用 pre_anal 算法对输入表达式进行分析

输入表达式 id+id*id,执行结果：

栈顶	输入	输出
E	id+id*id$	E->TE’
T	id+id*id$	T->FT’
F	id+id*id$	F->id
id	id+id*id$	
T’	+id*id$	T’->null
E’	+id*id$	E’->+TE’
+	+id*id$	
T	id*id$	T->FT’
F	id*id$	F->id
id	id*id$	
T’	*id$	T’->*FT’
*	*id$	
F	id$	F->id
id	id$	
T’	$	T’->null
E’	$	E’->null
$	$	

请按任意键继续. . .



实验总结：

本次实验主要算法书中都已给出，主要的是如何对表达式进行处理。该程序中将终结符，非终结符分别转化为所对应数组下标，从而转化为对数字之间的操作。产生式也转化为数字串。