实验项目二：语法分析

一、实验题目

根据给定文法编写调试预测分析程序，对任意输入串用预测分析法进行语法分析。

二、实验目的

加深对预测分析法的理解。

三、 设计内容

对LL（1）文法

E→TE´

E´→+TE´|ε

T→FT´

T´→*FT´|ε

F→(E)|i

根据程序所给的预测分析表采用预测分析法实现对输入串进行语法分析。

四、实验原理

1.预测分析算法总控程序预测分析过程：按照栈顶符号X 和当前输入符号a 工作的。对于任何（X，a），总控程序每次都执行下述三种可能的动作之一：

（1）若X＝a＝’#’，则宣布分析成功，停止分析过程；

（2）若X＝a≠’#’，则把X 从栈顶逐出，让a 指向下一输入符号；

（3）若X 是一个非终结符，则查看分析表M。若M 中存放着一条关于X 的产生式，那么，首先把X 逐出栈顶，然后，把产生式的右部符号按反序一一推进栈，同时做这个产生式相应的语义动作（目前程序不做语义分析）。若M[X,a]中存放着一条出错标志，则调用出错诊查程序Error。

2.针对于程序中的void analyse(char Vn[],char Vt[],string M[5][6],string str)函数（预测分析算法总控程序），现给出伪代码如下：

```
void analyse(char Vn[], char Vt[], string M[5][6], string str)
{
    初始化变量定义：    int i, j, p, q, h, flag = 1;
    初始化字符定义：char a, X;
    初始化字型型栈的定义：stack<char> st;
    '#' 和Vn[0]依次入栈。
    j = 0;  //j指向输入串的指针
    h = 1; //h代表输出分析步骤，第1……N步
    a = str[j];
    cout << "步骤          " << "分析栈          " << "剩余输入串
    " << "所用产生式" << endl;
    while (flag == 1)
    {
        cout << h << "          ";//显示步骤
        h++;
        调用showstack( )显示分析栈中内容
        cout << "          ";
        for (i = j; i < str.size(); i++)
            cout << str[i];   //显示剩余字符串
        取栈顶元素放入到X中
        如果X是终结符
            如果X与剩余输入串的第一个元素相比较相等时
                如果 (X != '#')
                {
                    cout << "          " << X << "匹配" << endl;
                    栈顶元素出栈;
                    读入输入串的下一个字符并赋值给a;
                }
```

```
                否则
                {
                    cout << "            " << "接受！" << endl << endl;
                    flag置为0;
                }
            否则
            {
                调用出错信息函数;
                break;
            }
        否则
        {
            将非终结符转换为行下标并赋值给p;
            将剩余串a中的终结符转换为列下标并赋值给q;
            string S1("NULL"), S2("null");
            如果M[p][q]为NULL
            {
                调用出错信息函数;
                break;  //对应项为空，则出错
            }
            否则
            {
                string str0 = M[p][q];
                cout << "            " << X << "-->" << str0 << endl;    //
显示相应的产生式
                st.pop();
                if (str0 != "$")    //$代表“空”字符
                    for (i = str0.size() - 1; i >= 0; i--)
                        产生式右端逆序进栈到st
            }
        }
    }
}
```

五、示例代码

```
#include <iostream>
#include <stdio.h>
#include <string>
#include <stack>
using namespace std;

char Vn[] = { 'E','e','T','t','F' }; //定义文法的非终结符，小写字母e表示E’
```

```
char Vt[] = { 'i','+','*','(',')','#' };//定义文法的终结符
int LENVt = sizeof(Vt);

void showstack(stack<char> st) //从栈底开始显示栈中的内容
{
    int i, j;
    char ch[100];
    j = st.size();
    for (i = 0; i < j; i++)
    {
        ch[i] = st.top();   //依次获取栈顶元素并赋值给ch数组
        st.pop();   //栈顶元素出栈
    }
    for (i = j - 1; i >= 0; i--)
    {
        cout << ch[i]; //逆序打印ch数组里的的元素
        st.push(ch[i]);//元素逆序入栈
    }
}

int find(char c, char array[], int n)//在array数组中查找字符c
{
    int i;
    int flag = 0;
    for (i = 0; i < n; i++)
    {
        if (c == array[i])
            flag = 1;
    }
    return flag;
}

int location(char c, char array[])//在array数组中查找字符c,若查找成功返回字
//符c的下标
{
    int i;
    for (i = 0; c != array[i]; i++);
    return i;
}

void error()//打印程序出错信息
{
    cout << "                error!" << endl;
}
```

```
void analyse(char Vn[], char Vt[], string M[5][6], string str)
{
    int i, j, p, q, h, flag = 1;
    char a, X;
    stack<char> st;
    请填写语句  //'#' 入栈。

    请填写语句；//Vn[0]入栈

    j = 0;    //j指向输入串的指针
    h = 1;
    a = str[j];
    cout << "步骤              " << "分析栈                " << "剩余输入串
    " << "所用产生式" << endl;
    while (flag == 1)
    {
        cout << h << "              ";//显示步骤
        h++;
        showstack(st);          //显示分析栈中内容
        cout << "                ";
        for (i = j; i < str.size(); i++)
            cout << str[i];     //显示剩余字符串
        X = st.top();
        if (find(X, Vt, LENVt) == 1)
            if (X == a)   //分析栈的栈顶元素和剩余输入串的第一个元素相
比较
                if (X != '#')
                {
                    cout << "                    " << X << "匹配" << endl;
                    st.pop();
                    a = str[++j];       //读入输入串的下一个字符
                }
                else
                {
                    cout << "                " << "接受！" << endl << endl;
                    flag = 0;
                }
            else
            {
                error();
                break;
            }
        else
```

```
        {
            请填写语句        //实现下标的转化（非终结符转换为行下标）
            请填写语句            //实现下标的转化（终结符转换为列下标）
            string S1("NULL"), S2("null");
            if (M[p][q] == S1 || M[p][q] == S2)     //查找二维数组中的产生式
            {
                error();
                break;   //对应项为空，则出错
            }
            else
            {
                string str0 = M[p][q];
                cout << "                " << X << "-->" << str0 << endl;   //显示相应的产生式
                st.pop();
                if (str0 != "$")    //$代表“空”字符
                    for (i = str0.size() - 1; i >= 0; i--)
                        请填写语句        //产生式右端逆序进栈
            }
        }
    }
}

int main()
{
    string M[5][6] = { "Te","NULL","NULL","Te","NULL","NULL",
                       "NULL","+Te","NULL","NULL","$","$",
                       "Ft","NULL","NULL","Ft","NULL","NULL",
                       "NULL","$","*Ft","NULL","$","$",
                       "i","NULL","NULL","(E)","NULL","NULL"
    };//预测分析表
    string str;
    int errflag, i;
    cout << "文法：E->E+T|T     T->T*F|F     F->(E)|i" << endl;
    cout << "请输入分析串（以#结束）： " << endl;
    do
    {
        errflag = 0;
        cin >> str;
        for (i = 0; i < str.size(); i++)
```

```
                if (!find(str[i], Vt, LENVt))
                {
                    cout << "输入串中包含非终结符" << str[i] << "（输入错误）！
" << endl;
                    errflag = 1;
                }
        } while (errflag == 1);
        analyse(Vn, Vt, M, str);
        return 0;
}
```

六．撰写电子版实验报告

实验报告命名方式：班级-学号-姓名-语法分析。

编译原理语法分析器
实验报告

学院

班级：

姓名：

学号：

实验二：语法分析

一. 实验题目

二. 实验目的

三. 实验内容

四. 实验结果

五. 实验心得