head.h文件

#ifndef HEAD

#define HEAD

#include<iostream>

#include<cstdio>

#include<stack>

#include<queue>

#include<cstring>

#include<string>

#include<cstdlib>

#include<fstream>

#define N 200

using namespace std;

int Degree(char c); // 求表达式的优先级

bool Calc(char c); // 逻辑表达式数的转换

string MidToPost(string c); // 求表达式对应的后缀表达式

char GetValuePost(string c);

bool conditionalS(bool a,bool b);

bool Equal(bool a,bool b);

void Count(string);

#endif

主函数文件

#include"head.h"

int main() {

//char [N];

FILE \*fp;

fp=fopen("logic\_run.txt","r");

char ch[100];

char ans;

int i = 0;

int count=0;

cout<<"离散课程逻辑表达式求值，可在文件中输入任何逻辑表达式均可打印出其真值表"<<endl;//print the introduce

cout<<"q|p相当于qvp"<<endl;

while (~fscanf(fp,"%s",ch)){//read the file put on ch

string str;

for(i=0;i<strlen(ch);i++){

str+=ch[i];

}

Count(str);

}

return 0;

}

Subfunction文件

#include"head.h"

//处理后缀表达式

int Degree(char c) {

if (c == '(') return 6;

if (c == '!') return 5;

if (c == '^') return 4; // 相当于q^p

if (c == '|') return 3; // 相当于qVp，优先级最低

if (c == '>') return 2;

if (c == '=') return 1;

else

return 0;

}

string MidToPost(string c) { //求表达式对应的后缀表达式

stack<char> s; //字符串中去手动去空格

string q="";

int n = c.size() ;

for (int i = 0; i<n; i++) {

if (c[i] != ' ') { // 除去空格

// 如果遇到运算数，直接加入到队列中，用队列来放后缀表达式

if (c[i] == 'F' || c[i] == 'T') {

q += c[i];

}

else if (c[i] == '!'&&!s.empty() && s.top() == '!') {

s.pop(); // 如果遇到！而且栈顶也是!那么直接抵消出栈

}

else if (!s.size()) {

s.push(c[i]);// 如果栈为空，遇到运算符直接入栈

}

else if (c[i] == ')') {

// 如果是右括号，则弹出对应左括号前的所有的运算符 ，加入到队列中

while (s.top() != '(') {

q += s.top();

s.pop();

}

s.pop(); // 弹出左括号

continue;

}

else if (Degree(s.top()) == 6 || (Degree(c[i]) > Degree(s.top()))) {

s.push(c[i]); // 如果栈顶是左括号，或者当前优先级高，都入栈

}

else if (Degree(s.top()) != 6 && Degree(c[i]) <= Degree(s.top())) {

q += s.top();

s.pop(); // 如果遇到运算符没有栈顶运算符级别高，出栈

while (!s.empty() && Degree(s.top()) != 6 && Degree(c[i]) <= Degree(s.top())) {

q+=s.top(); // 从栈中弹出比当前优先级高的运算符

s.pop();

}

s.push(c[i]); //将当前运算符加入到队列

}

}

}

while (!s.empty()) {

q += s.top(); // 最后将栈里面所有元素弹出加入到队列

s.pop();

}

//cout<<"\n"<<q<<endl;

return q;

}

getValuePost文件

#include"head.h"

//此部分用于运算后缀表达式

bool Calc(char c) {//将'F'定义为false 'T'定义为true

if (c == 'F') return false; //

else return true; //

}

bool conditionalS(bool a,bool b){//a-->b的运算函数

if(!a||b)return true;

else return false;

}

bool Equal(bool a,bool b){ //a <--> b的运算函数

if(a==b)return true;

else return false;

}

char GetValuePost(string q) {

//后缀表达式求值

bool r = true;

char x, y, ans;

stack<char> s;

int n = q.size();//cout<<n<<endl;

for (int i = 0; i < n; i++) {

if (q[i] == 'T' || q[i] == 'F') {

s.push(q[i]);

}

else {

x = s.top();

s.pop();

if(q[i]=='!') {//!q

if (Calc(x) == true)

s.push('F');

else

s.push('T');

}

else{

y = s.top();

s.pop();

switch(q[i]){

case '^':r = Calc(x) && Calc(y);break;//'q^p'定义为q&&p

case '|':r = Calc(x) || Calc(y);break;//qvp即q|p定义为q||p

case '>':r = conditionalS(Calc(x),Calc(y));break;//运用函数运算

case '=':r = Equal(Calc(x),Calc(y));break;//

default:break;

}

if (r == 1){

s.push('T');

}

else{

s.push('F');

}

}

}

ans = s.top();

}

return ans;

}

Count文件

#include"head.h"

//计算运算数的个数并打印表格

void Count(string s){

int ch[26],value[30],Count=0;//value用于赋值0 or 1

memset(ch,0,sizeof(ch));

memset(value,5,sizeof(value));//初始化 赋值数组为5用于debug

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

switch(s[i]){

case '|':case'^':case'=':case'>':case'!':break;

default:ch[s[i]-'a']++;

}

//打印 运算数部分

for(int i=0;i<26;i++){

if(ch[i]!=0){

Count++;

printf("%c\t",i+'a');

}

}

cout<<s<<endl; //输出运算式

//给运算数赋值部分

int n=1<<Count;

string stemp;//由于代替s做运算

while(n){

stemp=s;//stemp还原

int ntemp=n;

for(int i=0;i<Count;i++){

value[i]=ntemp%2;

ntemp/=2;

}

int j=0;//初始化下标

for(int i=0;i<26;i++){

if(ch[i]!=0){

if(value[j++]==0)

ch[i]='F';

else ch[i]='T';

printf("%c\t",ch[i]);//打印真值表初始值

}

}

for(int i=0;i<stemp.size() ;i++){

if(stemp[i]>='a'&&stemp[i]<='z')//运用桶排序桶的号即为字符，不为零既存在的思想

stemp[i]=ch[stemp[i]-'a'];//用T和F替换命题真值

}

string post;

post = MidToPost(stemp);

printf("%c\n",GetValuePost(post));

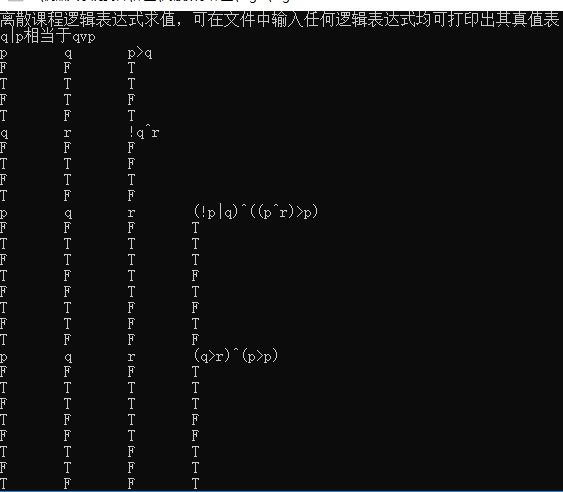
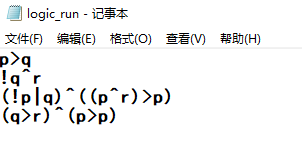
n--;

}

return ;

}

运行图

txt文件