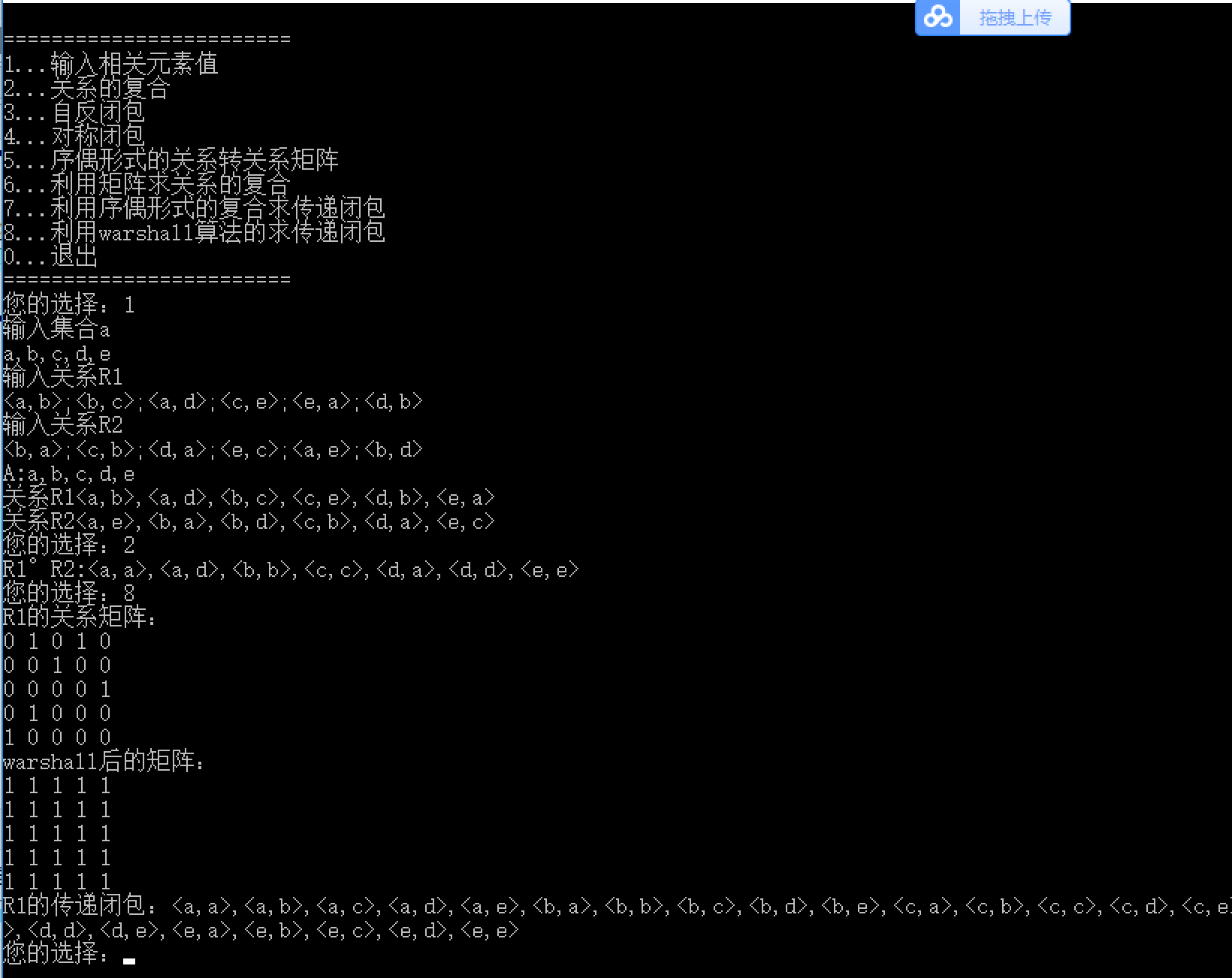
w2判断公式的合法性

1. 运行结果：



1. 代码改进与注释

//

// main.cpp

// 离散6

//

// Created by zsy on 2018/5/22.

// Copyright © 2018年 zsy. All rights reserved.

//

#include<iostream>

#include<set>

#include<cstring>

using namespace std;

void getset(string &a)**{//用来得到这个集合**

set<char>A;

char x;

while((x=getchar())!='\n'){

if(isdigit(x)||isalpha(x)){ **//是数字或者是字母**

pair<set<char>::iterator,bool>r=A.insert(x);

if(r.second)

a+=\*r.first;

}

}

}

void getrela(int r[100][100],string a,int &count){ **//把关系存入**

memset(r,0,sizeof(r)); /**/清零**

int co,ro,sign=0;

char p,q;

string x;

getline(cin,x);

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

if(isdigit(x[i])||isalpha(x[i])){

if(sign%2==0)

p=x[i];

else if(sign%2==1){

q=x[i]; **//把集合分为两类**

for(int j=0;j<a.size();j++){

if(p==a[j])

ro=j;

if(q==a[j])

co=j;

}

r[ro][co]=1;**//记录下来都有哪些元素被包含到**

count++;

}

sign++;

}

}

}

void rR(int M[100][100],int m[100][100],int n,int &count)**{//自反闭包**

memcpy(m,M,100\*100);

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

if(m[i][i]==0){// **m[i][i]不存在就让它出现**

m[i][i]=1;

count++;

}

}

}

void sR(int M[100][100],int m[100][100],int n,int &count){**//对称闭包**

memcpy(m,M,100\*100);

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

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

if(m[i][j]==1&&m[j][i]==0){ **//找到对称的：m[i][j]存在而m[j][i]不存在**

m[j][i]=1;

count++;

}

}

}

}

void tR(int M[100][100],int m[100][100],int n,int &count){**//画关系矩阵**

memcpy(m,M,100\*100);

for(int j=0;j<n;j++){//warshall

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

if(m[i][j]==1){**如果有存在，就输出那个大小的关系矩阵**

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

if(m[j][k]==1&&m[i][k]==0){

m[i][k]=1;

count++;

}

}

}

}

}

}

int matrixmulti(int M[100][100],int m[100][100],int T[100][100],int n){

int count=0;

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

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

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

T[i][j]+=M[i][k]\*m[k][j];

if(T[i][j]>=1){

count++;

break;

}

}

}

}

return count;

}

void matrixplus(int M[100][100],int m[100][100],int n,int &count){

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

for(int j=0;j<n;j++)

if(M[i][j]==0&&m[i][j]==1){

M[i][j]=1;

count++;

}

}

void printrela(int m[100][100],string a,int c){**//传递闭包**

int count1=0;

for(int i=0;i<a.size()&&count1!=c;i++){

for(int j=0;j<a.size();j++){

if(m[i][j]==1){

count1++;

cout<<'<'<<a[i]<<','<<a[j]<<'>'<<(count1==c?'\n':',');

}

}

}

}

void printM(int m[100][100],int n){ //**warshall之后的矩阵**

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

for(int j=0;j<n;j++)

cout<<m[i][j]<<' ';

cout<<endl;

}

}

void tr7(int M[100][100],string a,int count){**//序偶形式的复合求传递闭包**

int A[100][100],B[100][100],C[100][100],ca=count,cb=count,cc=count;

memcpy(B,M,100\*100);

memcpy(C,M,100\*100);

for(int i=0;i<a.size()-1;i++){

ca=cb;

memcpy(A,B,100\*100);

memset(B,0,sizeof(B));

cb=matrixmulti(A,M,B,a.size());

printrela(A,a,ca);

printrela(M,a,count);

printrela(B,a,cb);

matrixplus(C,B,a.size(),cc);

cout<<endl;

}

cout<<"传递关系闭包为:";

printrela(C,a,cc);

cout<<endl;

cout<<"序偶转化为数组:"<<endl;;

printM(C,a.size());

}

int main(){

int r1[100][100],r2[100][100],c1=0,c2=0,choice;

string a;

printf("\n========================\n");

printf("1...输入相关元素值\n");

printf("2...关系的复合\n");

printf("3...自反闭包\n");

printf("4...对称闭包\n");

printf("5...序偶形式的关系转关系矩阵\n");

printf("6...利用矩阵求关系的复合\n");

printf("7...利用序偶形式的复合求传递闭包\n");

printf("8...利用warshall算法的求传递闭包\n");

printf("0...退出\n========================\n");

while(printf("您的选择：")){

scanf("%d",&choice);

fflush(stdin);

if(choice==0) break;

switch(choice){

case 1:{

cout<<"输入集合a"<<endl;

getset(a);

cout<<"输入关系R1"<<endl;

getrela(r1,a,c1);

cout<<"输入关系R2"<<endl;

getrela(r2,a,c2);

cout<<"A:";

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

cout<<a[i]<<(i==a.size()-1?'\n':',');

cout<<"关系R1";

printrela(r1,a,c1);

cout<<"关系R2";

printrela(r2,a,c2);

break;

}

case 2:{

int T[100][100];

int ct=matrixmulti(r1,r2,T,sizeof(a));

cout<<"R1°R2:";

printrela(T,a,ct);

break;

}

case 3:{

int RR[100][100],cR=c1;

cout<<"R1的自反闭包：";

rR(r1,RR,a.size(),cR);

printrela(RR,a,cR);

break;

}

case 4:{

int SR[100][100],cS=c1;

cout<<"R1的对称闭包：";

sR(r1,SR,a.size(),cS);

printrela(SR,a,cS);

break;

}

case 5:{

cout<<"R1的序偶：";

printrela(r1,a,c1);

cout<<"R1的关系矩阵"<<endl;

printM(r1,a.size());

break;

}

case 6:{

int T[100][100];

memset(T,0,sizeof(T));

cout<<"R1的矩阵"<<endl;

printM(r1,a.size());

cout<<"R2的关系矩阵"<<endl;

printM(r2,a.size());

matrixmulti(r1,r2,T,a.size());

cout<<"复合后的矩阵"<<endl;

printM(T,a.size());

break;

}

case 7:{

tr7(r1,a,c1);

break;

}

case 8:{

int TR[100][100],cT=c1;

cout<<"R1的关系矩阵："<<endl;

printM(r1,a.size());

tR(r1,TR,a.size(),cT);

cout<<"warshall后的矩阵："<<endl;

printM(TR,a.size());

cout<<"R1的传递闭包：";

printrela(TR,a,cT);

break;

}

}

}

return 0;

}

/\*

1

a,b,c,d,e

<a,b>;<b,c>;<a,d>;<c,e>;<e,a>;<d,b>

<b,a>;<c,b>;<d,a>;<e,c>;<a,e>;<b,d>

\*/