## 实现7个功能

1. 定义域
2. 值域
3. 域
4. 逆关系
5. 右结合
6. 限制
7. 像

## 代码：

#include<bits/stdc++.h>

using namespace std;

class Bin\_fun{//二元关系class

public:

string x,y,s;//s为原子符串，x为第一元素，y为第二元素

Bin\_fun(string newS){

s=newS;

int pos=-1;

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

if(s[i]>='0'&&s[i]<='9'){

pos=i;

break;

}

for(int i=pos;i<s.size();i+=6){

x+=s[i];

y+=s[i+2];

}

}

};

class openFile{//打开文件的class

private:

public:

fstream io;

openFile(string filename){

io.open(filename.c\_str(),ios::in|ios::app);

}

~openFile(){

io.close();

}

};

int myFind(string s){//寻找数字字符并返回位置

for(char i='0';i<='9';i++){

int pos=s.find(i);

if(pos!=s.npos)

return pos;

}

}

void myNoSame(string& s){

int a[300];

memset(a,0,sizeof(a));

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

a[s[i]]++;

}

s.clear();

for(int i=0;i<300;i++)

if(a[i]!=0)

s+=i;

}

string domR(){//定义域

openFile domR("R.txt");

string s;

domR.io>>s;

//cout<<s<<endl;

int pos=myFind(s);//cout<<pos<<":"<<s[pos]<<endl;

string ans("{");//cout<<"invoked"<<endl;

stringstream ss;

while(pos<s.size()){

ss<<s[pos]<<",";

ans+=ss.str();

ss.str("");//清空ss的缓存

pos+=6;

}

ans.replace(ans.size()-1,1,"}");

cout<<"domR="<<ans<<endl;

domR.io<<"domR"<<ans<<endl;

return ans;

}

string ranR(){//值域

openFile ranR("R.txt");

string s;

ranR.io>>s;

//cout<<s<<endl;

int pos=myFind(s)+2;//cout<<pos<<":"<<s[pos]<<endl;

string ans("{");//cout<<"invoked"<<endl;

stringstream ss;

while(pos<s.size()){

ss<<s[pos]<<",";

ans+=ss.str();

ss.str("");//清空ss的缓存

pos+=6;

}

ans.replace(ans.size()-1,1,"}");

cout<<"ranR="<<ans<<endl;

ranR.io<<"ranR"<<ans<<endl;

return ans;

}

string fldR(){//域

openFile fldR("R.txt");

string s;

fldR.io>>s;

string ansTemp;

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

if(s[i]>='0'&&s[i]<='9')

ansTemp+=s[i];

myNoSame(ansTemp);

//cout<<ansTemp<<endl;

string ans("{");

stringstream ss;

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

ss<<ansTemp[i]<<",";

ans+=ss.str();

ss.str("");//清空ss的缓存

}

ans.replace(ans.size()-1,1,"}");

cout<<"fldR="<<ans<<endl;

fldR.io<<"fldR="<<ans<<endl;

return ans;

}

string right\_combine(string a,string b){//a·b

string ans("{}");

Bin\_fun A(a),B(b);

int pos=0;//记录插入位置

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

for(int j=0;j<B.x.size();j++){//把a的第二元素和b的第一元素进行比较

if(A.y[i]==B.x[j]){

stringstream ss;

ss<<"<"<<A.x[i]<<","<<B.y[j]<<">"<<",";

//cout<<ss.str()<<" "<<1+(i\*j+j)\*6<<endl;

ans.insert(1+pos\*6,ss.str());

pos++;

}

}

}//cout<<"ok"<<endl;

ans.erase(ans.size()-2,1);

//cout<<"a·b:"<<ans<<endl;

return ans;

}

void myDelet(string &s,char ch){//删除s中出现的所有ch

int pos=-1;

while(s.find(ch,pos+1)!=s.npos){

pos=s.find(ch);

s.erase(pos,1);

}

//cout<<s<<" "<<ch<<endl;

}

void get\_string\_R\_A(string &R,string &A,string filename){

openFile right(filename.c\_str());//右结合

right.io>>R;//cout<<"R:"<<R<<endl;

myDelet(R,'=');

for(char i='A';i<='Z';i++)

myDelet(R,i);

right.io>>A;

myDelet(A,'=');

for(char i='A';i<='Z';i++)

myDelet(A,i);

cout<<"R:"<<R<<" A:"<<A<<endl;//输出检测R和A的读取情况

}

void solve\_R\_A(){

openFile right("right\_combine.txt");

string R,A;

get\_string\_R\_A(R,A,"right\_combine.txt");

//cout<<"R:"<<R<<" A:"<<A<<endl;//输出检测R和A的读取情况

string ans=right\_combine(R,A); //计算R·A

cout<<"R。A="<<ans<<endl;

right.io<< "\nR。A="<<ans<<endl;

}

string get\_num\_char(string s){//读取出字符串中的数字字符

string ans;

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

if(s[i]>='0'&&s[i]<='9')

ans+=s[i];

return ans;

}

string calc\_limit(string R,string A){

Bin\_fun ob\_R(R);

string aTemp=get\_num\_char(A);

string ans("{}");

int pos=0;//用于 记录位置

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

for(int j=0;j<ob\_R.x.size();j++){

if(ob\_R.x[j]==aTemp[i]){//第一元素与限制集合元素相等

stringstream ss;

ss<<"<"<<aTemp[i]<<","<<ob\_R.y[j]<<">"<<",";

//cout<<ss.str()<<" "<<1+(i\*j+j)\*6<<endl;

ans.insert(1+pos\*6,ss.str());

pos++;

}

}

}

ans.erase(ans.size()-2,1);

//cout<<"R|A="<<ans<<endl;

return ans;

}

void solve\_limit\_R\_A(){//R在A上的限制 R|A

openFile lim("limit.txt");

string R,A;

get\_string\_R\_A(R,A,"limit.txt");

// cout<<"R:"<<R<<" A:"<<A<<endl;//输出检测R和A的读取情况

string ans=calc\_limit(R, A);

cout<<"R|A="<<ans<<endl;

lim.io<<"R|A="<<ans<<endl;

}

string calc\_mirror(string R,string A){

Bin\_fun ob\_R(R);

string aTemp=get\_num\_char(A);

string ans("{}");

int pos=0;//用于 记录位置

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

for(int j=0;j<ob\_R.x.size();j++){

if(ob\_R.x[j]==aTemp[i]){//第一元素与限制集合元素相等

stringstream ss;

ss<<ob\_R.y[j]<<",";

//cout<<ss.str()<<" "<<1+(i\*j+j)\*6<<endl;

ans.insert(1+pos\*2,ss.str());

pos++;

}

}

}

ans.erase(ans.size()-2,1);

//cout<<"R|A="<<ans<<endl;

return ans;

}

void solve\_mirror\_R\_A(){//R在A上的限制 R|A

openFile lim("limit.txt");

string R,A;

get\_string\_R\_A(R,A,"limit.txt");

// cout<<"R:"<<R<<" A:"<<A<<endl;//输出检测R和A的读取情况

string ans=calc\_mirror(R, A);

cout<<"R[A]="<<ans<<endl;

lim.io<<"R|A="<<ans<<endl;

}

int main(){

solve\_R\_A();

solve\_limit\_R\_A();

solve\_mirror\_R\_A();

}

## 运行图：

