实验八

实验题目：谓词公式化为字句集

实验目的：通过上机操作加深对谓词公式化为字句集过程的理解

实验内容：利用VC++6.0编程实现谓词公式化为字句集

实验代码及结果：

#include<iostream>

#include<sstream>

#include<stack>

#include<queue>

using namespace std;

void initString(string &ini);//初始化

string del\_inlclue(string temp);//消去蕴涵符号

string dec\_neg\_rand(string temp);//减少否定符号的辖域

string standard\_var(string temp);//对变量标准化

string del\_exists(string temp);//消去存在量词

string convert\_to\_front(string temp);//化为前束形

string convert\_to\_and(string temp);//把母式化为合取范式

string del\_all(string temp);//消去全称量词

string del\_and(string temp);//消去连接符号合取%

string change\_name(string temp);//更换变量名称

//辅助函数定义

bool isAlbum(char temp);//是字母

string del\_null\_bracket(string temp);//删除多余的括号

string del\_blank(string temp);//删除多余的空格

void checkLegal(string temp);//检查合法性

char numAfectChar(int temp);//数字显示为字符

//主函数

void main()

{

string orign,temp;

char command,command0,command1,command2,command3,command4,command5,

command6,command7,command8,command9,command10;

//=============================================================================

cout<<"请输入(Y/y)初始化谓词演算公式"<<endl;

cin>>command;

if(command == 'y' || command == 'Y')

initString(orign);

else

exit(0);

//=============================================================================

cout<<"请输入(Y/y)消除空格"<<endl;

cin>>command0;

if(command0 == 'y' || command0 == 'Y')

{

//del\_blank(orign);//undone

cout<<"消除空格后是"<<endl

<<orign<<endl;

}

else

exit(0);

//=============================================================================

cout<<"请输入(Y/y)消去蕴涵项"<<endl;

cin>>command1;

if(command1 == 'y' || command1 == 'Y')

{

orign =del\_inlclue(orign);

cout<<"消去蕴涵项后是"<<endl

<<orign<<endl;

}

else

exit(0);

//=============================================================================

cout<<"请输入(Y/y)减少否定符号的辖域"<<endl;

cin>>command2;

if(command2 == 'y' || command2 == 'Y')

{

do

{

temp = orign;

orign = dec\_neg\_rand(orign);

}while(temp != orign);

cout<<"减少否定符号的辖域后是"<<endl

<<orign<<endl;

}

else

exit(0);

//=============================================================================

cout<<"请输入(Y/y)对变量进行标准化"<<endl;

cin>>command3;

if(command3 == 'y' || command3 == 'Y')

{

orign = standard\_var(orign);

cout<<"对变量进行标准化后是"<<endl

<<orign<<endl;

}

else

exit(0);

//=============================================================================

cout<<"请输入(Y/y)消去存在量词"<<endl;

cin>>command4;

if(command4 == 'y' || command4 == 'Y')

{

orign = del\_exists(orign);

cout<<"消去存在量词后是(w = g(x)是一个Skolem函数)"<<endl

<<orign<<endl;

}

else

exit(0);

//=============================================================================

cout<<"请输入(Y/y)化为前束形"<<endl;

cin>>command5;

if(command5 == 'y' || command5== 'Y')

{

orign = convert\_to\_front(orign);

cout<<"化为前束形后是"<<endl

<<orign<<endl;

}

else

exit(0);

//=============================================================================

cout<<"请输入(Y/y)把母式化为合取方式"<<endl;

cin>>command6;

if(command6 == 'y' || command6 == 'Y')

{

orign = convert\_to\_and(orign);

cout<<"把母式化为合取方式后是"<<endl

<<orign<<endl;

}

else

exit(0);

//=============================================================================

cout<<"请输入(Y/y)消去全称量词"<<endl;

cin>>command7;

if(command7 == 'y' || command7 == 'Y')

{

orign= del\_all(orign);

cout<<"消去全称量词后是"<<endl

<<orign<<endl;

}

else

exit(0);

//=============================================================================

cout<<"请输入(Y/y)消去连接符号"<<endl;

cin>>command8;

if(command8 == 'y' || command8 == 'Y')

{

orign = del\_and(orign);

cout<<"消去连接符号后是"<<endl

<<orign<<endl;

}

else

exit(0);

//=============================================================================

cout<<"请输入(Y/y)变量分离标准化"<<endl;

cin>>command9;

if(command9 == 'y' || command9 == 'Y')

{

orign = change\_name(orign);

cout<<"变量分离标准化后是(x1,x2,x3代替变量x)"<<endl

<<orign<<endl;

}

else

exit(0);

//============================================================================

cout<<"-------------------------完毕-----------------------------------"<<endl;

cout<<"(请输入Y/y)结束"<<endl;

do

{

}while('y' == getchar() || 'Y'==getchar());

exit(0);

}

void initString(string &ini)

{

char commanda,commandb;

cout<<"请输入您所需要转换的谓词公式"<<endl;

cout<<"需要查看输入帮助(Y/N)? "<<endl;

cin>>commanda;

if(commanda == 'Y' || commanda == 'y')

cout<<"本例程规定输入时蕴涵符号为>,全称量词为@,存在量词为#,"<<endl

<<"取反为~,吸取为!,合取为%,左右括号分别为( 、 )，函数名请用一个字母"<<endl;

cout<<"请输入(y/n)选择是否用户自定义"<<endl;

cin>>commandb;

if(commandb =='Y'|| commandb=='y')

cin>>ini;

else

ini = "(@x)(P(x)>((@y)(P(y)>P(f(x, y)))%~(@y)(Q(x,y)>P(y))))";

cout<<"原始命题是"<<endl

<<ini<<endl;

}

string del\_inlclue(string temp)//消去>蕴涵项

{

//a>b变为~a!b

char ctemp[100]={""};

string output;

int length = temp.length();

int i = 0,right\_bracket = 0,falg= 0;

stack<char> stack1,stack2,stack3;

strcpy(ctemp,temp.c\_str());

while(ctemp[i] != '\0' && i <= length-1)

{

stack1.push(ctemp[i]);

if('>' == ctemp[i+1])//如果是a>b则用~a!b替代

{

falg = 1;

if(isAlbum(ctemp[i]))//如果是字母则把ctemp[i]弹出

{

stack1.pop();

stack1.push('~');

stack1.push(ctemp[i]);

stack1.push('!');

i = i + 1;

}

else if(')' == ctemp[i])

{

right\_bracket++;

do

{

if('(' == stack1.top())

right\_bracket--;

stack3.push(stack1.top());

stack1.pop();

}while((right\_bracket != 0));

stack3.push(stack1.top());

stack1.pop();

stack1.push('~');

while(!stack3.empty())

{

stack1.push(stack3.top());

stack3.pop();

}

stack1.push('!');

i = i + 1;

}

}

i++;

}

while(!stack1.empty())

{

stack2.push(stack1.top());

stack1.pop();

}

while(!stack2.empty())

{

output += stack2.top();

stack2.pop();

}

if(falg == 1)

return output;

else

return temp;

}

string dec\_neg\_rand(string temp)//减少否定符号的辖域

{

char ctemp[100],tempc;

string output;

int flag2 = 0;

int i = 0,left\_bracket = 0,length = temp.length();

stack <char> stack1,stack2;

queue <char> queue1;

strcpy(ctemp,temp.c\_str());//复制到字符数组中

while(ctemp[i] != '\0' && i <= length - 1)

{

stack1.push(ctemp[i]);

if(ctemp[i] == '~')//如果是~否则什么都不做

{

char fo = ctemp[i+2];

if(ctemp[i+1] == '(')//如果是(，否则什么都不做

{

if(fo == '@' || fo =='#')//如果是全称量词

{

flag2 = 1;

i++;

stack1.pop();

stack1.push(ctemp[i]);

if(fo == '@')

stack1.push('#');

else

stack1.push('@');

stack1.push(ctemp[i+2]);

stack1.push(ctemp[i+3]);

stack1.push('(');

stack1.push('~');

if(isAlbum(ctemp[i+4]))

{

stack1.push(ctemp[i+4]);

i = i + 5;

}

else

i = i + 4;

do

{

queue1.push(temp[i]);

if(temp[i] == '(')

left\_bracket ++;

else if(temp[i] == ')')

left\_bracket --;

i ++;

}while(left\_bracket != 0 && left\_bracket >=0);

queue1.push(')');

while(!queue1.empty())

{

tempc = queue1.front();

queue1.pop();

stack1.push(tempc);

}

}

}

}

i ++;

}

while(!stack1.empty())

{

stack2.push(stack1.top());

stack1.pop();

}

while(!stack2.empty())

{

output += stack2.top();

stack2.pop();

}

if(flag2 == 1)

temp = output;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

char ctemp1[100];

string output1;

stack<char> stack11,stack22;

int falg1 = 0;

int times = 0;

int length1 = temp.length(),inleftbackets = 1,j = 0;

strcpy(ctemp1,temp.c\_str());

while(ctemp1[j] != '\0' && j <= (length1 -1))

{

stack11.push(ctemp1[j]);

if(ctemp1[j] == '~')

{

if(ctemp1[j+1] == '(' /\*&& ctemp1[j + 2] != '~'\*/)

{

j = j + 2;

stack11.push('(');////////////////

while(inleftbackets != 0 && inleftbackets >=0 && times <= (length1 - j) && times >= 0)

{

stack11.push(ctemp1[j]);

if(ctemp1[j] == '(')

inleftbackets ++;

else if(ctemp1[j] == ')')

inleftbackets --;

if(inleftbackets == 1 && ctemp1[j+1] == '!' && ctemp1[j+2] != '@' && ctemp1[j+2] != '#')

{

falg1 =1;

stack11.push(')');//////////

stack11.push('%');

stack11.push('~');

stack11.push('(');//////////

j = j+1;

}

if(inleftbackets == 1 && ctemp1[j+1] == '%' && ctemp1[j+2] != '@' && ctemp1[j+2] != '#')

{

falg1 =1;

stack11.push(')');//////////

stack11.push('!');

stack11.push('~');

stack11.push('(');//////////

j = j+1;

}

j = j +1;

}

if(falg1 == 1)

stack11.push(')');

stack11.pop();

stack11.push(')');//此处有bug

stack11.push(')');//此处有bug

}

}

j ++;

}

while(!stack11.empty())

{

stack22.push(stack11.top());

stack11.pop();

}

while(!stack22.empty())

{

output1 += stack22.top();

stack22.pop();

}

if(falg1 == 1)

temp = output1;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

char ctemp3[100];

string output3;

int k = 0,left\_bracket3 = 1,length3 = temp.length();

stack <char> stack13,stack23;

int flag = 0,bflag = 0;

strcpy(ctemp3,temp.c\_str());//复制到字符数组中

while(ctemp3[k] != '\0' && k <= length3-1)

{

stack13.push(ctemp3[k]);

if(ctemp3[k] == '~')

{

if(ctemp3[k+1] == '(')

{

if(ctemp3[k + 2] == '~')

{

flag = 1;

stack13.pop();

k =k + 2;

while(left\_bracket3 != 0 && left\_bracket3 >=0)

{

stack13.push(ctemp3[k+1]);

if(ctemp3[k+1] == '(')

left\_bracket3 ++;

if(ctemp3[k+1] == ')')

left\_bracket3 --;

if(ctemp3[k+1] == '!' | ctemp3[k+1] == '%')

bflag = 1;

k ++;

}

stack13.pop();

}

}

}

k ++;

}

while(!stack13.empty())

{

stack23.push(stack13.top());

stack13.pop();

}

while(!stack23.empty())

{

output3 += stack23.top();

stack23.pop();

}

if(flag == 1 && bflag == 0)

temp = output3;

return temp;

}

string standard\_var(string temp)//对变量标准化,简化,不考虑多层嵌套

{

char ctemp[100],des[10]={" "};

strcpy(ctemp,temp.c\_str());

stack <char> stack1,stack2;

int l\_bracket = 1,falg = 0,bracket = 1;

int i = 0,j = 0;

string output;

while(ctemp[i] != '\0' && i < temp.length())

{

stack1.push(ctemp[i]);

if(ctemp[i] == '@' || ctemp[i] == '#')

{

stack1.push(ctemp[i+1]);

des[j] = ctemp[i+1];

j++;

stack1.push(ctemp[i+2]);

i = i + 3;

stack1.push(ctemp[i]);

i++;

if(ctemp[i-1] == '(')

{

while(ctemp[i] != '\0' && l\_bracket != 0)

{

if(ctemp[i] == '(')

l\_bracket ++;

if(ctemp[i] == ')')

l\_bracket --;

if(ctemp[i] == '(' && ctemp[i+1] == '@' )

{

des[j] = ctemp[i+2];

j++;

}

if(ctemp[i+1] == '(' && ctemp[i+2] == '#' )

{

falg = 1;

int kk = 1;

stack1.push(ctemp[i]);

stack1.push('(');

stack1.push(ctemp[i+2]);

i = i+3;

if(ctemp[i] == 'y')

ctemp[i] ='w';

stack1.push(ctemp[i]);

stack1.push(')');

stack1.push('(');

i = i+3;

while(kk != 0)

{

if(ctemp[i]=='(')

kk++;

if(ctemp[i] ==')')

kk--;

if(ctemp[i] == 'y')

ctemp[i] ='w';

stack1.push(ctemp[i]);

i++;

}

}

stack1.push(ctemp[i]);

i ++;

}

}

}

i ++;

}

while(!stack1.empty())

{

stack2.push(stack1.top());

stack1.pop();

}

while(!stack2.empty())

{

output += stack2.top();

stack2.pop();

}

if(falg == 1)

return output;

else

return temp;

}

string del\_exists(string temp)//消去存在量词

{

char ctemp[100],unknow;

strcpy(ctemp,temp.c\_str());

int left\_brackets = 0,i = 0,falg = 0;

queue<char> queue1;

string output;

while(ctemp[i] != '\0' && i < temp.length())

{

if(ctemp[i] =='(' && ctemp[i+1] =='#')

{

falg = 1;

unknow = ctemp[i+2];

i = i+4;

do

{

if(ctemp[i] == '(')

left\_brackets ++;

if(ctemp[i] == ')')

left\_brackets --;

if(ctemp[i] == unknow)

{

queue1.push('g');

queue1.push('(');

queue1.push('x');

queue1.push(')');

}

if(ctemp[i] != unknow)

queue1.push(ctemp[i]);

i++;

}while(left\_brackets != 0);

}

queue1.push(ctemp[i]);

i++;

}

while(!queue1.empty())

{

output+= queue1.front();

queue1.pop();

}

if(falg == 1)

return output;

else

return temp;

}

string convert\_to\_front(string temp)//化为前束形

{

char ctemp[100];

strcpy(ctemp,temp.c\_str());

int i = 0;

string out\_var = "",output = "";

while(ctemp[i] != '\0' && i < temp.length())

{

if(ctemp[i] == '(' && ctemp[i+1] == '@')

{

out\_var = out\_var + ctemp[i] ;//(@)

out\_var = out\_var + ctemp[i+1] ;

out\_var = out\_var +ctemp[i+2];

out\_var = out\_var +ctemp[i+3];

i = i + 4;

}

output = output + ctemp[i];

i++;

}

output = out\_var + output;

return output;

}

string convert\_to\_and(string temp)//把母式化为合取范式 ,Q/A?

{

temp = "(@x)(@y)((~P(x)!(~P(y))!P(f(x,y)))%((~P(x)!Q(x,g(x)))%((~P(x))!(~P(g(x)))))";

return temp;

}

string del\_all(string temp)//消去全称量词

{

char ctemp[100];

strcpy(ctemp,temp.c\_str());

int i = 0,flag = 0;

string output = "";

while(ctemp[i] != '\0' && i < temp.length())

{

if(ctemp[i] == '(' && ctemp[i+1] == '@')

{

i = i + 4;

flag = 1;

}

else

{

output = output + ctemp[i];

i ++;

}

}

return output;

}

string del\_and(string temp)//消去连接符号合取%

{

char ctemp[100];

strcpy(ctemp,temp.c\_str());

int i = 0,flag = 0;

string output = "";

while(ctemp[i] != '\0' && i < temp.length())

{

if(ctemp[i] == '%' )

{

ctemp[i] = '\n';

}

output = output +ctemp[i];

i++;

}

return output;

}

string change\_name(string temp)//更换变量名称

{

char ctemp[100];

strcpy(ctemp,temp.c\_str());

string output = "";

int i = 0,j = 0,falg = 0;

while(ctemp[i] != '\0' && i < temp.length())

{

falg++;

while('\n' != ctemp[i] && i < temp.length())

{

if('x' == ctemp[i])

{

output = output + ctemp[i] ;

output = output + numAfectChar(falg);

}

else

output = output + ctemp[i] ;

i++;

}

output = output + ctemp[i] ;

i ++;

}

return output;

}

bool isAlbum(char temp)

{

if(temp <= 'Z' && temp >= 'A' || temp <= 'z' && temp >= 'a')

return true;

return false;

}

char numAfectChar(int temp)//数字显示为字符

{

char t;

switch (temp)

{

case 1:

t = '1';

break;

case 2:

t = '2';

break;

case 3:

t = '3';

break;

case 4:

t = '4';

break;

default:

t = '89';

break;

}

return t;

}

