

**计算机与信息 学院实验报告**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 实验课程： | 人工智能实验 | | | | |
| 实验编号： | 实验五 | | | | |
| 实验名称： | 人工智能中的逻辑 | | | | |
| 实验人员： | 学号 | 18111207248 | | | |
| 姓名 | 吴钰 | | | |
| 班级 | 18级计算机科学与技术（创新班） | | | |
| 实验日期： | 2020.11.20 | | | | |
| 实验室： | 2060402 | | | | |
|  |  | | | | |
| 实验评价： |  | | | | |
| 实验成绩： | |  | 评价日期： |  |
|  | 指导教师： | |  | | |

# 实验目的

以下两题任选一题

1．编写程序，该程序将任意命题逻辑表达式作为输入，并返回其真值。程序应该允许使用表 5.2 中的任何逻辑连接符。

2．编写程序，使用真值表来确定用命题逻辑表达的论证是否有效。程序应该允许表 5.2中的任何逻辑连接符。

提交材料：（1）实验报告；（2）可执行程序及源代码。将这些材料打包上传到学习通。

# 二、实验要求

## 完成实验后请填写实验报告并上交。要求在实验报告中必须记录实验中遇到的问题及其问题解决方案。

# 三、实验内容

1．编写程序，该程序将任意命题逻辑表达式作为输入，并返回其真值。程序应该允许使用表 5.2 中的任何逻辑连接符。

#include<iostream>

#include<queue>

#include<stack>

/\*

否定 ！

合取 +

析取 －

蕴含 #

等价 $

\*/

using namespace std;

char expression[100];

int deny(int a)

{

if(a==0)

{

a=1;

return a;

}

else

{

a=0;

return a;

}

}

int both(int a,int b)

{

int c;

if(a==1&&b==1)

{

c=1;

return c;

}

else

{

c=0;

return c;

}

}

int xiqu(int a,int b)

{

int c;

if(a==0&&b==0)

{

c=1;

return c;

}

else

{

c=0;

return c;

}

}

int yunhan(int a,int b)

{

int c;

if(a==1&&b==0)

{

c=0;

return c;

}

else

{

c=1;

return c;

}

}

int dengjia(int a,int b)

{

int c;

if((a==1&&b==1)||(a==0&&b==0))

{

c=1;

return c;

}

else

{

c=0;

return c;

}

}

int main()

{

int r1,r2,r3,result,cnt=0,zimu[10],num[10],cnt1=0,name;

cout<<"请输入表达式："<<endl;

cin>>expression;

for(int n1=0;n1<100;n1++)

{

if(expression[n1]=='(')

{

cnt1++;

continue;

}

if(expression[n1]==')')

{

cnt1--;

continue;

}

}

if(cnt1==0)

{

queue<char> q1;//存后缀表达式

stack<char> s1;

stack<int> s2;

for(int i=0;i<100;i++)

{

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

{

s1.push(expression[i]);

continue;

}

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

{

q1.push(s1.top());

s1.pop();

s1.pop();

continue;

}

else if(expression[i]>=97&&expression[i]<=122)

{

q1.push(expression[i]);

continue;

}

else if(expression[i]=='!'||expression[i]=='+'||expression[i]=='-'||expression[i]=='#'||expression[i]=='$')

{

s1.push(expression[i]);

continue;

}

else

{

break;

}

}

while(!q1.empty())

{

if(q1.front()>=97&&q1.front()<=122)

{

int k;

for(k=0;k<10;k++)

{

if(q1.front()==zimu[k])

{

q1.pop();

s2.push(num[k]);

break;

}

}

if(k==10)

{

cout<<"请输入"<<q1.front()<<"的真值："<<endl;

zimu[cnt]=q1.front();

cin>>name;

num[cnt]=name;

s2.push(name);

q1.pop();

cnt++;

}

}

if(q1.front()=='!')

{

q1.pop();

r1=s2.top();

s2.pop();

r3=deny(r1);

s2.push(r3);

}

if(q1.front()=='+')

{

q1.pop();

r2=s2.top();

s2.pop();

r1=s2.top();

s2.pop();

r3=both(r1,r2);

s2.push(r3);

}

if(q1.front()=='-')

{

q1.pop();

r2=s2.top();

s2.pop();

r1=s2.top();

s2.pop();

r3=xiqu(r1,r2);

s2.push(r3);

}

if(q1.front()=='#')

{

q1.pop();

r2=s2.top();

s2.pop();

r1=s2.top();

s2.pop();

r3=yunhan(r1,r2);

s2.push(r3);

}

if(q1.front()=='$')

{

q1.pop();

r2=s2.top();

s2.pop();

r1=s2.top();

s2.pop();

r3=dengjia(r1,r2);

s2.push(r3);

}

}

result=s2.top();

s2.pop();

cout<<"该表达式结果为："<<result<<endl;

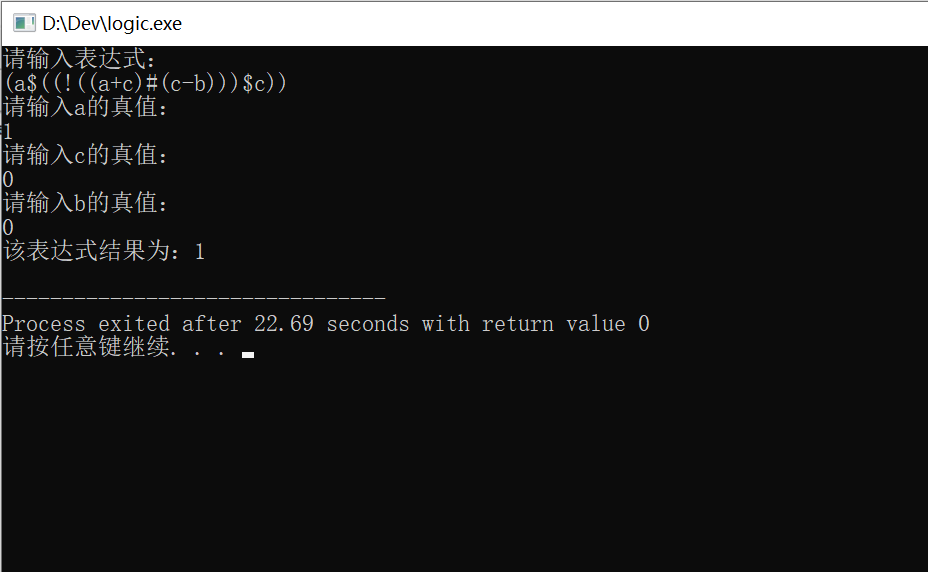
}

else

cout<<"括号输入错误!"<<endl;

return 0;

}



# 四、实验小结

实验收获：我们将A、B、C称为逻辑函数的输入，Y称为该逻辑函数的输出。三变量的逻辑函数共有八组输入，每一组输入的结果可能使得输出为0或者1。 真值表就为我们很直观的展示了每一组输入对应的输出情况：如当ABC输入同时为000时输出1，当ABC输入001时同时为Y为0等。

        在没有约束项的真值表（逻辑函数）中，每一种输入情况都可能出现。

      意思是在没有特定说明下，ABC的输入可以为000,001,010,011,100,101,110,111中的任意一组。即输入可以为真值表中的任何一组输入。

        假设在有一个N组取值的真值表（我们的真值表有8组取值），某一时刻，电路或是逻辑表达式只能有一组特定输入，又因为我们的输入有N种情况，那么为了写出逻辑函数的表达式，我们的表达式中将会有N项，每一项正好代表着一种输入情况出现与否，当某一种输入情况出现时，那么我们代表着这组输入的项就是真，即1，其他项为0。

        在上面真值表的例子中，输入取值可能出现八种情况，所以我们要用八项来分别表示每一种输入情况是否出现。这些项的含义就是当一种输入情况出现时，我们代表着这种输入情况的项取值就为真。

        我们可以用暂时代表当输入为000，001,010,011……111时表达式中的项，这些符号仅仅是代表了我们的每一项，如何计算项是什么，下面再详述，这里仅是简单的代替。

        根据上面所述构成项的规则——每一项代表着一种输入是否出现，所以我们可以通过查看真值表知道：当输入为000时，因为代表了输入为000的出现，所以为真，为1，其他项均为0；

        同样的道理，由于代表了输入001的出现与否，所以当输入为001时，即001这种输入情况出现了，此时就为真，即1。其他项各自代表了自己的输入情况，而同一时刻下只能出现一种输入，又因为现在的输入为001，所以其他项皆为0。

        通过上面的解释，我们已经知道当一种输入情况产生时，与之对应的项是真，也就是1，而其他每一项都为0，而真值表中又明确的告诉我们：每一组输入取值所对应的输出是什么。

   那么我们让每一种输出取值和对应的输入的项去做与运算，这样，当有某一种输入出现时，因为代表着该输入的项为1，所以这一项的输出会保持原样（），而其他的项都为0，所以其他的输出也是无效的。