离散数学

项目说明文档

**命题逻辑联接词、真值表、主范式A**



同济大学

Tongji University

姓名： 林觉凯

学号： 2253744

指导老师： 李冰

学院专业： 软件学院 软件工程

# 一.实验目的：

本实验课程训练学生掌握命题逻辑中的联接词、真值表、主范式等，进一步能用它们来解决实际问题。通过实验提高学生编写实验报告、总结实验结果的能力；使学生具备程序设计的思想，能够独立完成简单的算法设计和分析。

# 二.实验内容：

1. 从键盘输入两个命题变元P和Q的真值，求它们的合取、析取、条件和双向条件的真值。(A)。

**详细说明：**

## 1. 逻辑联接词的运算

本实验要求利用C/C＋＋语言，实现二元合取、析取、条件和双向条件表达式的计算。充分利用联接词和逻辑运算符之间的相似性来实现程序功能。

## 2. 求任意一个命题公式的真值表

本实验要求利用C/C＋＋语言，实现任意输入公式的真值表计算。一般将公式中的命题变元放在真值表的左边，将公式的结果放在真值表的右边。命题变元可用数值变量表示，合式公式的表示及求真值表转化为逻辑运算结果；可用一维数表示合式公式中所出现的n个命题变元，同时它也是一个二进制加法器的模拟器，每当在这个模拟器中产生一个二进制数时，就相当于给各个命题变元产生了一组真值指派。算法逻辑如下：

(1)将二进制加法模拟器赋初值0。

(2)计算模拟器中所对应的一组真值指派下合式公式的真值。

(3)输出真值表中对应于模拟器所给出的一组真值指派及这组真值指派所对应的一行真值。

(4)产生下一个二进制数值，若该数值等于2n-1，则结束，否则转(2)。

# 三.实验环境；

采用C或C＋＋编程语言，MS Visual Studio 2019实验环境实现。

# 四. 实验原理和实现过程(算法描述)；

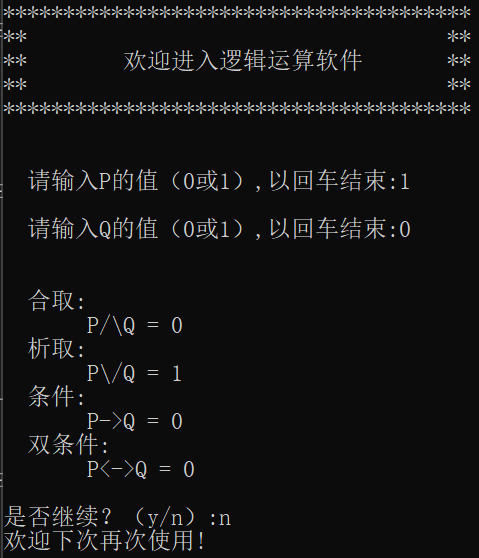
## 1.实验原理

(1)**合取**：二元命题联结词。将两个命题P、Q联结起来，构成一个新的命题P∧Q, 读作P、Q的合取, 也可读作P与Q。这个新命题的真值与构成它的命题P、Q的真值间的关系为只有当两个命题变项P = T, Q = T时方可P∧Q =T, 而P、Q只要有一方为F则P∧Q = F。P∧Q可用来表示日常用语P与Q, 或P并且Q。

(2)**析取**：二元命题联结词。将两个命题P、Q联结起来，构成一个新的命题P∨Q, 读作P、Q的析取, 也可读作P或Q。这个新命题的真值与构成它的命题P、Q的真值间的关系为只有当两个命题变项P = F, Q = F时方可P∨Q =F, 而P、Q只要有一为T则P∨Q = T。P∨Q可用来表示日常用语P或者Q。

(3)**条件**：二元命题联结词。将两个命题P、Q联结起来，构成一个新的命题P→Q, 读作P条件Q, 也可读作如果P，那么Q。这个新命题的真值与构成它的命题P、Q的真值间的关系为只有当两个命题变项P = T, Q = F时方可P→Q =F, 其余均为T。

(4)**双向条件**：二元命题联结词。将两个命题P、Q联结起来，构成一个新的命题P←→Q, 读作P双条件于Q。这个新命题的真值与构成它的命题P、Q的真值间的关系为当两个命题变项P = T, Q =T时方可P←→Q =T, 其余均为F。

**五．项目示例**

1. **项目实现**

按照项目要求首先我们需要一个欢迎界面，该函数的作用是打印欢迎界面。

//打印欢迎界面函数

void Print\_Welcome()

{

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << "\*\* \*\*" << endl;

cout << "\*\* 欢迎进入逻辑运算软件 \*\*" << endl;

cout << "\*\* \*\*" << endl;

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl << endl;

}

输入函数我做了代码的健壮性判断：输入的不是0或1、输入的数据类型错误，它的作用是保证输入P或Q的真值只能是0或者1

//获得选项所对应的选项值函数（进行了错误处理，完善代码的健壮性）

int Get\_Int(char key)

{

int Ret\_Int = 0;

while(1)

{

cout << endl << " 请输入" << key << "的值（0或1）,以回车结束:";

cin >> Ret\_Int;

//代码的健壮性判断：输入的不是0或1、输入的数据类型错误

if (Ret\_Int != 0 && Ret\_Int != 1 || cin.good() == 0)

{

cout << endl << " " << key << "的值输入有误,请重新输入!" << endl;

cin.clear();

cin.ignore(1024, '\n');

continue;

}

break;

}

return Ret\_Int;

}

按照项目要求，在每次使用的最后要给用户选择：是否循环。本函数的功能是判断是否继续使用运算软件函数同时进行了错误处理，完善代码的健壮性。

//判断是否继续使用运算软件函数（进行了错误处理，完善代码的健壮性）

bool Is\_Continue()

{

bool choice;

char keydown;

while (1)

{

cout << endl << "是否继续？（y/n）:";

cin >> keydown;

//只能输入y或者n的代码健壮性检验

if (cin.good() == 0 || keydown != 'y' && keydown != 'n')

{

cout << "输入错误,请重新输入!" << endl;

cin.clear();

cin.ignore(1024, '\n');

continue;

}

if (keydown == 'y')

choice = true;

else

{

cout<<"欢迎下次再次使用!"<<endl;

choice = false;

}

return choice;

}

}

主函数是完成程序的主题流程，我们用一个数组a来存储最终的结果，接着输出每一个数组内的内容。

//主函数，完成程序的主题架构

int main()

{

Print\_Welcome();

//用一个数组a来存储最终的结果

int a[4];

int P, Q;

while (1)

{

P = Get\_Int('P');

Q = Get\_Int('Q');

a[0] = P && Q; //合取

a[1] = P || Q; //析取

a[2] = (!P) || Q; //条件

a[3] = ((!P) || Q) && ((!Q) || P); //双条件

cout << "\n\n 合取:\n P/\\Q = " << a[0] << endl;

cout << " 析取:\n P\\/Q = " << a[1] << endl;

cout << " 条件:\n P->Q = " << a[2] << endl;

cout << " 双条件:\n P<->Q = " << a[3] << endl;

if (!Is\_Continue()) //判断是否继续

break;

}

return 0;

}

1. **项目收获**

此次项目的过程（第一题的A题）比较简单，主要是要将整个程序的过程范围几个部分：输入部分，运算部分和输出部分。与离散数学的基础知识相结合，我对分区块解决问题有了更好的使用，同时对基础知识有了更好的巩固。

附件：源程序

#include <iostream>

using namespace std;

//打印欢迎界面函数

void Print\_Welcome()

{

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << "\*\* \*\*" << endl;

cout << "\*\* 欢迎进入逻辑运算软件 \*\*" << endl;

cout << "\*\* \*\*" << endl;

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl << endl;

}

//获得选项所对应的选项值函数（进行了错误处理，完善代码的健壮性）

int Get\_Int(char key)

{

int Ret\_Int = 0;

while(1)

{

cout << endl << " 请输入" << key << "的值（0或1）,以回车结束:";

cin >> Ret\_Int;

//代码的健壮性判断：输入的不是0或1、输入的数据类型错误

if (Ret\_Int != 0 && Ret\_Int != 1 || cin.good() == 0)

{

cout << endl << " " << key << "的值输入有误,请重新输入!" << endl;

cin.clear();

cin.ignore(1024, '\n');

continue;

}

break;

}

return Ret\_Int;

}

//判断是否继续使用运算软件函数（进行了错误处理，完善代码的健壮性）

bool Is\_Continue()

{

bool choice;

char keydown;

while (1)

{

cout << endl << "是否继续？（y/n）:";

cin >> keydown;

//只能输入y或者n的代码健壮性检验

if (cin.good() == 0 || keydown != 'y' && keydown != 'n')

{

cout << "输入错误,请重新输入!" << endl;

cin.clear();

cin.ignore(1024, '\n');

continue;

}

if (keydown == 'y')

choice = true;

else

{

cout<<"欢迎下次再次使用!"<<endl;

choice = false;

}

return choice;

}

}

//主函数，完成程序的主题架构

int main()

{

Print\_Welcome();

//用一个数组a来存储最终的结果

int a[4];

int P, Q;

while (1)

{

P = Get\_Int('P');

Q = Get\_Int('Q');

a[0] = P && Q; //合取

a[1] = P || Q; //析取

a[2] = (!P) || Q; //条件

a[3] = ((!P) || Q) && ((!Q) || P); //双条件

cout << "\n\n 合取:\n P/\\Q = " << a[0] << endl;

cout << " 析取:\n P\\/Q = " << a[1] << endl;

cout << " 条件:\n P->Q = " << a[2] << endl;

cout << " 双条件:\n P<->Q = " << a[3] << endl;

if (!Is\_Continue()) //判断是否继续

break;

}

return 0;

}