

# 命题逻辑联接词

2153726 罗宇翔

(经济与管理学院, 信息管理与信息系统)

# 目录

<b>1</b>	<b>实验目的</b>	<b>1</b>
<b>2</b>	<b>实验内容</b>	<b>1</b>
<b>3</b>	<b>实验环境</b>	<b>1</b>
3.1	Visual Studio Code . . . . .	1
3.2	g++ . . . . .	1
<b>4</b>	<b>实验原理和方法</b>	<b>1</b>
4.1	合取 . . . . .	1
4.2	析取 . . . . .	2
4.3	条件 . . . . .	2
4.4	双向条件 . . . . .	2
<b>5</b>	<b>实验代码</b>	<b>2</b>
<b>6</b>	<b>实验数据及结果分析</b>	<b>6</b>

---

# 1 实验目的

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。

## 2 实验内容

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)。

## 3 实验环境

### 3.1 Visual Studio Code

Version: 1.89.1

Browser:

Mozilla/5.0 (Macintosh; Intel Mac OS X 10\_15\_7)

AppleWebKit/537.36 (KHTML, like Gecko) Code/1.89.1

Chrome/120.0.6099.291 Electron/28.2.8 Safari/537.36

### 3.2 g++

Apple clang version 14.0.0 (clang-1400.0.29.202)

Target: x86\_64-apple-darwin21.6.0

Thread model: posix

## 4 实验原理和方法

### 4.1 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  $P \wedge Q$ ，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  $P \wedge Q = T$ ，而  $P$ 、 $Q$  只要有一方为  $F$  则  $P \wedge Q = F$ 。 $P \wedge Q$  可用来表示日常用语  $P$  与  $Q$ ，或  $P$  并且  $Q$ 。

---

## 4.2 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  $P \vee Q$ ，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  $P \vee Q = F$ ，而  $P$ 、 $Q$  只要有一为  $T$  则  $P \vee Q = T$ 。 $P \vee Q$  可用来表示日常用语  $P$  或者  $Q$ 。

## 4.3 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  $P \rightarrow Q$ ，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  $P \rightarrow Q = F$ ，其余均为  $T$ 。

## 4.4 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  $P \leftrightarrow Q$ ，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  $P \leftrightarrow Q = T$ ，其余均为  $F$ 。

## 5 实验代码

```
#include <iostream>
#include <string>
using namespace std;
bool conjunction_(int p, int q) {
    return p & q;
}
bool disjunction_(int p, int q) {
    return p | q;
}
bool condition_(int p, int q) {
    return !p | q;
}
bool bicondition_(int p, int q) {
```

---

```
        return (p & q) | (!p & !q);
    }
    void string_repeat(string str, int repeat, bool end = true)
    {
        for (int i = 0; i < repeat; i++)
            cout << str;
        if (end)
            cout << endl;
    }
    void print_welcome(string style = "*") {
        string_repeat(style, 20);
        string_repeat(style, 2, false);
        string_repeat(" ", 4, false);
        cout << "welcome!";
        string_repeat(" ", 4, false);
        string_repeat(style, 2);
        string_repeat(style, 20);
    }
    bool value_validation(string value, string name) {
        if (value != "0" && value != "1") {
            cout << "invalid value for " << name << ", please
retry" << endl;
            return false;
        }
        return true;
    }
    string get_value(string name) {
        cout << "please input " << name << "(0 or 1), end by
enter, q to quit" << endl;
        string tmp;
        cin >> tmp;
        return tmp;
    }
}
```

---

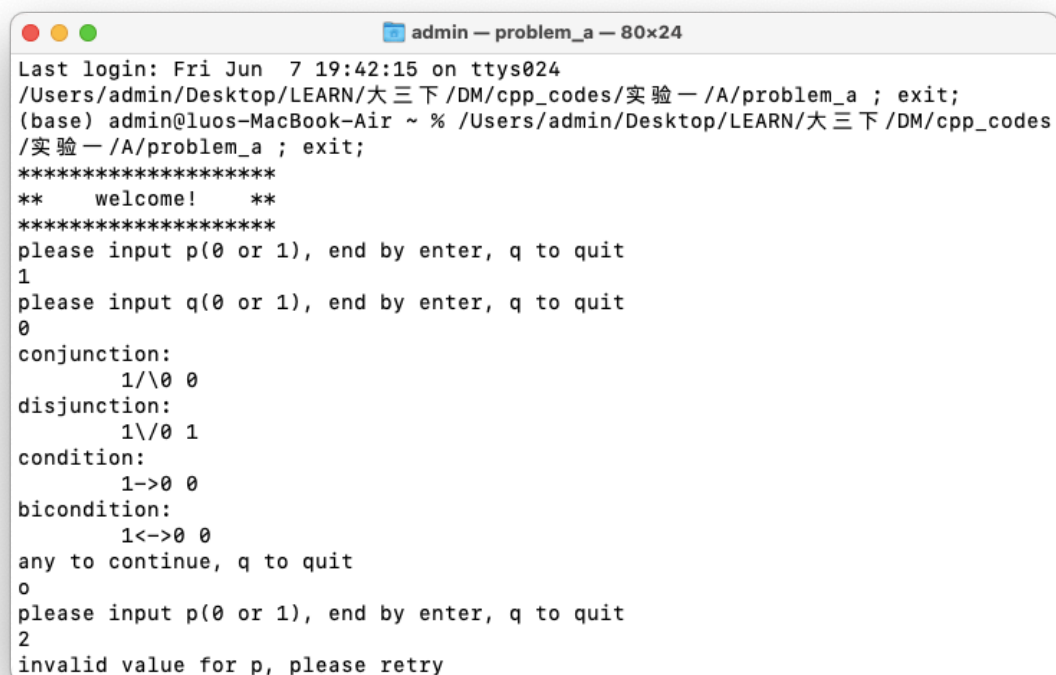
```
}
```

```
int main() {
    print_welcome();
    while (true) {
        string p, q;
        do {
            p = get_value("p");
            if (p == "q") {
                cout << "quit" << endl;
                return 0;
            }
        } while (!value_validation(p, "p"));
        do {
            q = get_value("q");
            if (q == "q") {
                cout << "quit" << endl;
                return 0;
            }
        } while (!value_validation(q, "q"));
        int intP, intQ = 0;
        if (p != "0")
            intP = 1;
        if (q != "0")
            intQ = 1;
        cout << "conjunction:\n\t" << intP << "/\" << intQ
        << " " << conjunction_(intP, intQ) << endl;
        cout << "disjunction:\n\t" << intP << "\\\" << intQ
        << " " << disjunction_(intP, intQ) << endl;
        cout << "condition:\n\t" << intP << "->" << intQ <<
        " " << condition_(intP, intQ) << endl;
```

---

```
        cout << "bicondition:\n\t" << intP << "<->" << intQ
<< " " << bicondition_(intP, intQ) << endl;
        cout << "any to continue, q to quit" << endl;
        string continueConfiguration;
        cin >> continueConfiguration;
        if (continueConfiguration == "q") {
            cout << "quit" << endl;
            return 0;
        }
    }
    return 0;
}
```

## 6 实验数据及结果分析

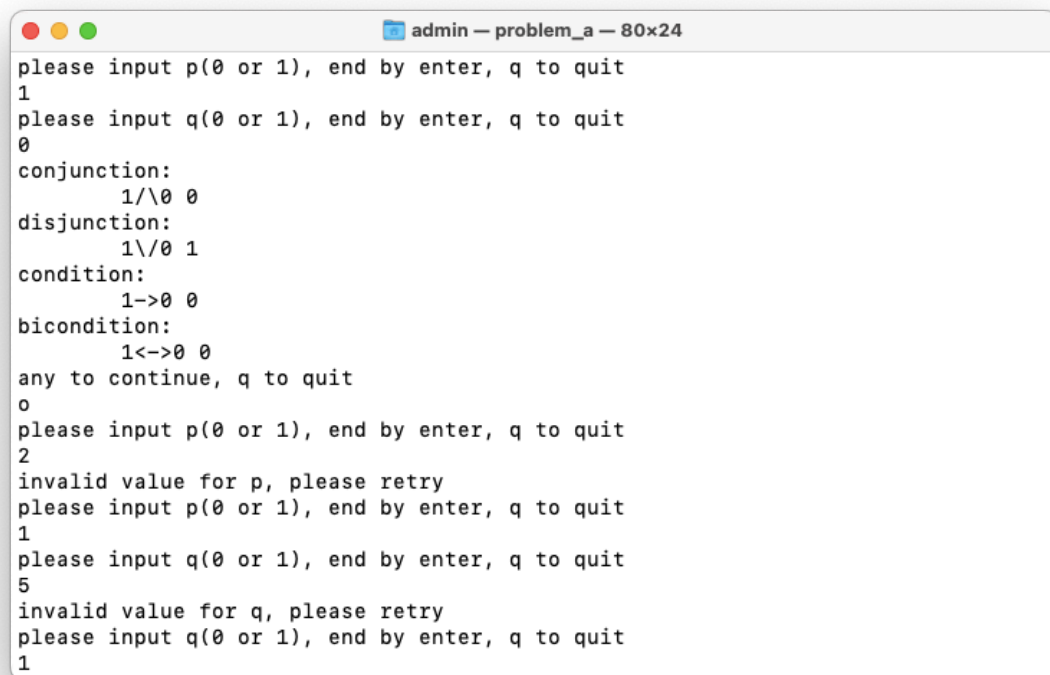


```
admin — problem_a — 80x24
Last login: Fri Jun  7 19:42:15 on ttys024
/Users/admin/Desktop/LEARN/大三下/DM/cpp_codes/实验一/A/problem_a ; exit;
(base) admin@luos-MacBook-Air ~ % /Users/admin/Desktop/LEARN/大三下/DM/cpp_codes
/实验一/A/problem_a ; exit;
*****
**  welcome!  **
*****
please input p(0 or 1), end by enter, q to quit
1
please input q(0 or 1), end by enter, q to quit
0
conjunction:
    1/\0 0
disjunction:
    1\/\0 1
condition:
    1->0 0
bicondition:
    1<->0 0
any to continue, q to quit
0
please input p(0 or 1), end by enter, q to quit
2
invalid value for p, please retry
```

Figure 1: 正常输入

正常输入下，正确计算命题的合取、析取、条件和双向条件的真值。

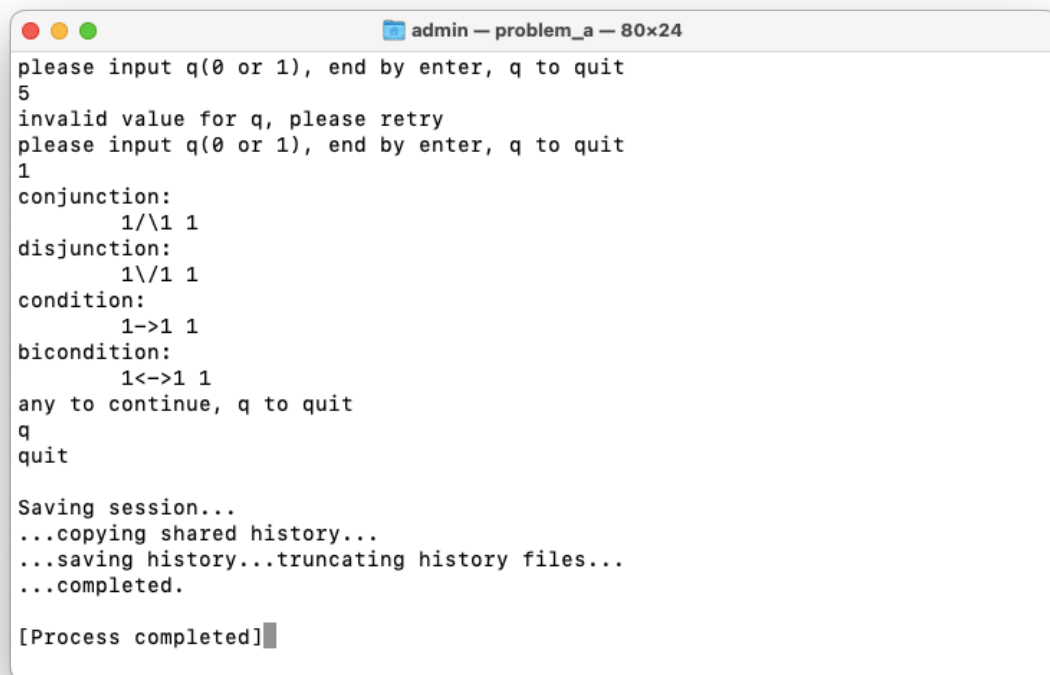




```
admin — problem_a — 80x24
please input p(0 or 1), end by enter, q to quit
1
please input q(0 or 1), end by enter, q to quit
0
conjunction:
    1/\0 0
disjunction:
    1/\0 1
condition:
    1->0 0
bicondition:
    1<->0 0
any to continue, q to quit
0
please input p(0 or 1), end by enter, q to quit
2
invalid value for p, please retry
please input p(0 or 1), end by enter, q to quit
1
please input q(0 or 1), end by enter, q to quit
5
invalid value for q, please retry
please input q(0 or 1), end by enter, q to quit
1
```

Figure 2: 错误输入

错误输入下，拒绝计算并提示错误信息。



```
admin — problem_a — 80x24
please input q(0 or 1), end by enter, q to quit
5
invalid value for q, please retry
please input q(0 or 1), end by enter, q to quit
1
conjunction:
    1/\1 1
disjunction:
    1\1 1
condition:
    1->1 1
bicondition:
    1<->1 1
any to continue, q to quit
q
quit

Saving session...
...copying shared history...
...saving history...truncating history files...
...completed.

[Process completed]
```

Figure 3: 退出

输入 **q** 正常退出程序。