项目说明文档

离散数学课程项目

——命题逻辑联接词、真值表、主范式

作 者 姓 名： 陈奕名

学 号： 2351883

指 导 教 师： 李冰

学院、 专业： 软件学院 软件工程

同济大学

Tongji University

# 1 项目分析

## 1.1 项目要求

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）。

2. 求任意一个命题公式的真值表（B），并根据真值表求主范式（C）。

**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）。

# 2 主要功能实现

## 2.1 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。

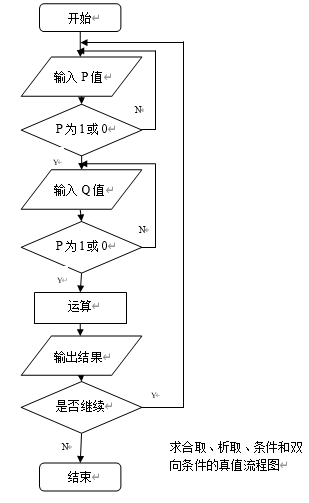
（5）**真值表**:表征逻辑事件输入和输出之间全部可能状态的表格。列出命题公式真假值的表。通常以1表示真，0 表示假。命题公式的取值由组成命题公式的命题变元的取值和命题联结词决定，命题联结词的真值表给出了真假值的算法。 真值表是在逻辑中使用的一类数学表，用来确定一个表达式是否为真或有效。

（6）**主范式**：

**主析取范式**：在含有n个命题变元的简单合取式中,若每个命题变元与其否定不同时存在,而两者之一出现一次且仅出现一次,则称该简单合取式为极小项。由若干个不同的极小项组成的析取式称为主析取范式;与A等价的主析取范式称为A的主析取范式。任意含n个命题变元的非永假命题公式A都存在与其等价的主析取范式,并且是惟一的。

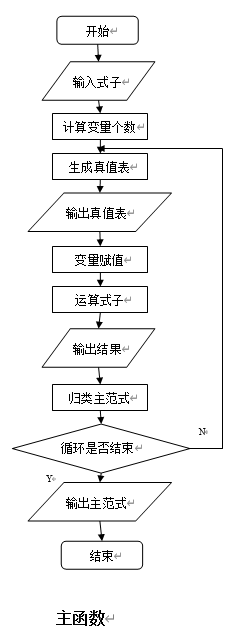
**主合取范式**：在含有n个命题变元的简单析取式中，若每个命题变元与其否定不同时存在，而两者之一出现一次且仅出现一次，称该简单析取式为极大项。由若干个不同的极大项组成的合取式称为主合取范式；与A等价的主合取范式称为A的主合取范式。任意含n个命题变元的非永真命题公式A都存在与其等价的主合取范式，并且是惟一的。

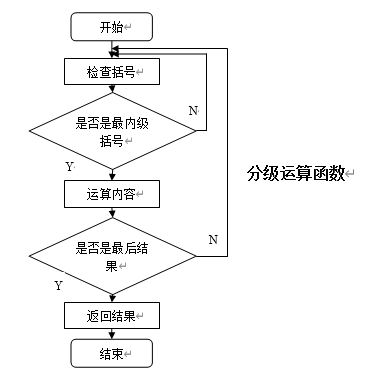
## 2.2 实验过程

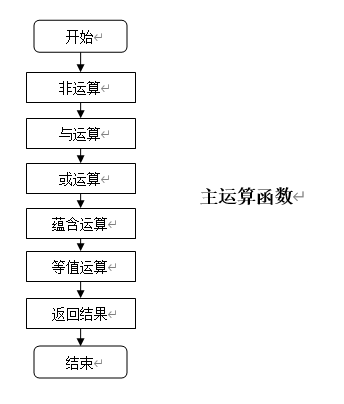
（1）A题部分，首先是对各个输入量的处理，要确定输入的为0或1，否则则为出错，接下来就是运算处理，在C语言中本身支持的有与、或、非这三种，可以用!,&&,||来表示，而在这个实验中，不是与、或、非的可以通过转化变为与、或、非的形式，具体流程图如下：

（2）B，C题部分，首先是输入一个合理的式子，然后从式子中查找出变量的个数，开辟一个二进制函数，用来生成真值表，然后用函数运算，输出结果，并根据结果归类给范式，最后输出范式。

函数部分，主要是3个函数，一个为真值表递加函数，通过二进制的加法原理递进产生，一个为分级运算函数，这个函数是通过判断括号，选出最内级括号的内容执行运算函数，这样一级级向外运算，最后得出最终结果，剩下一个为主运算函数，按照运算符号的优先级按顺序进行运算。



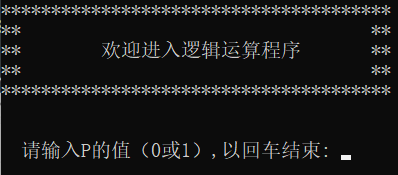




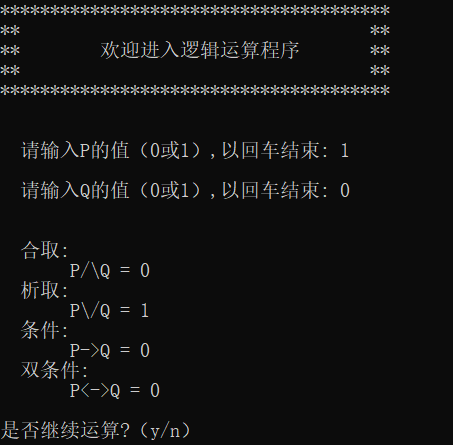
# 3 测试

题目A：

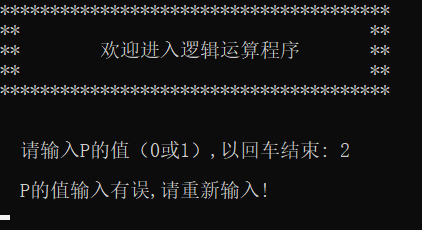
进入界面：



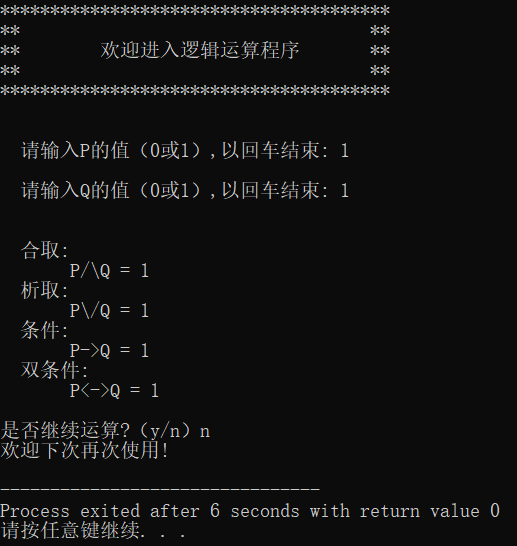
正确运算结果：



错误控制：



退出：

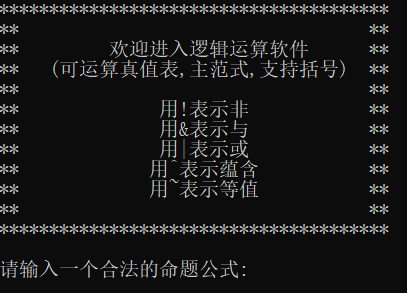


**结果分析：**

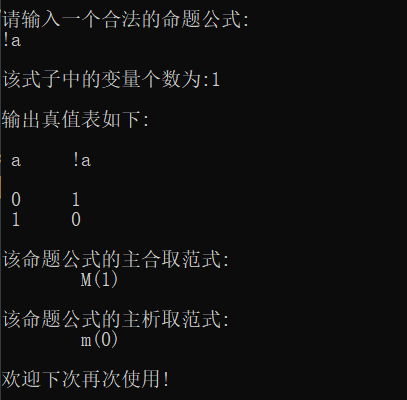
这道题主要是读取数值并进行计算，输出两个命题变元(P和Q)的合取，析取，条件和双向条件的真值结果，同时要注意输入的值要必须0或1，如果不是，则进行错误提示，并进行重新输入。

## B,C题：

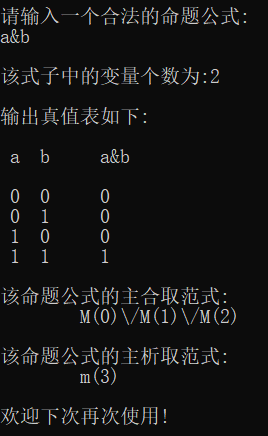
欢迎界面：



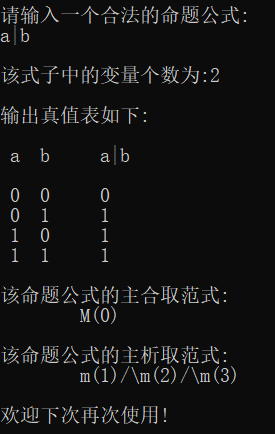
输出非运算公式的真值表，主合取范式和主析取范式：



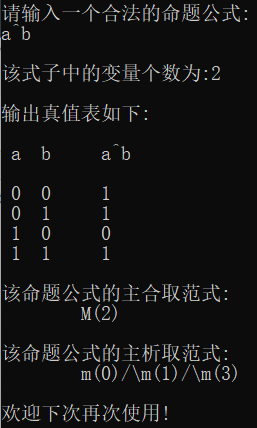
输出与运算公式的真值表，主合取范式和主析取范式：



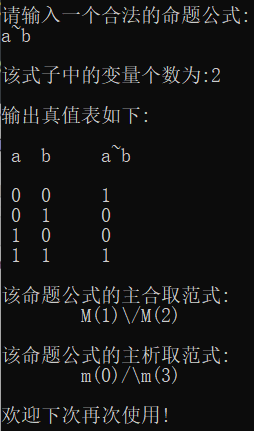
输出或运算公式的真值表，主合取范式和主析取范式：



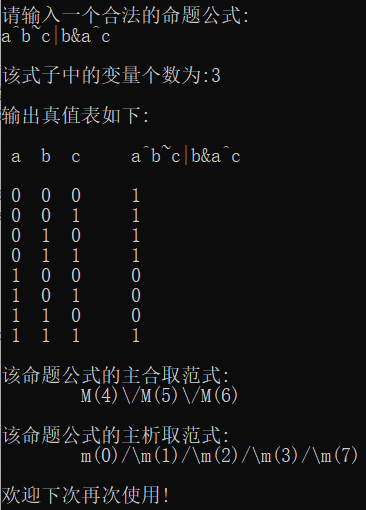
输出蕴含运算公式的真值表，主合取范式和主析取范式：



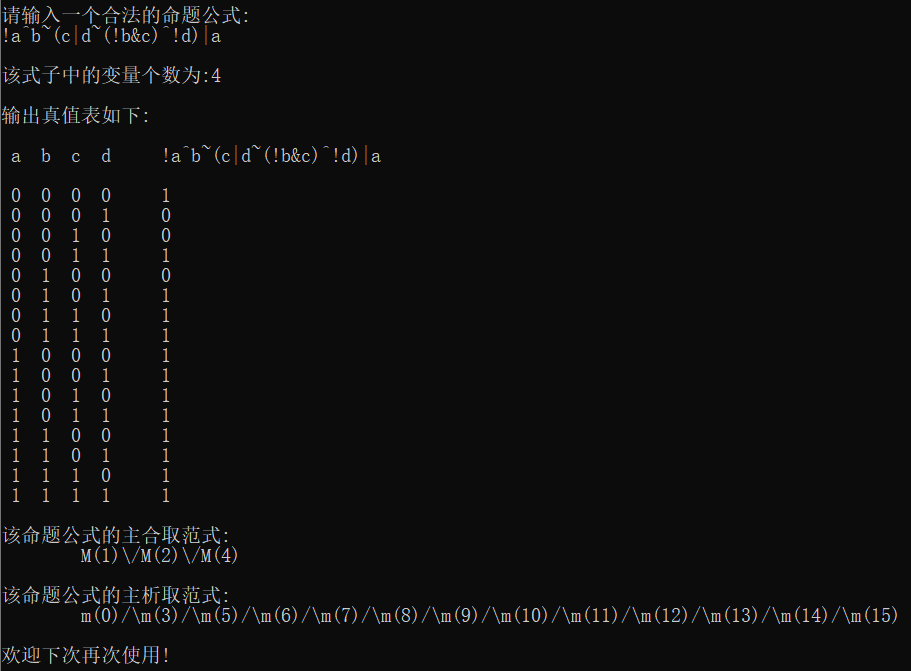
输出等值运算公式的真值表，主合取范式和主析取范式：



输出综合运算公式的真值表，主合取范式和主析取范式：



输出带括号的综合运算运算公式的真值表，主合取范式和主析取范式：



**结果分析：**

B，C题目由于本身关系比较密切，所以将两个做在了一起。这个程序达到了题目要求的各个功能，可以运算与，或，非，蕴含，等值条件组成的表达式，并且支持括号运算。

# 4 心得体会

完成这个程序后，我收获了很多宝贵的经验。首先，在程序设计过程中，我加深了对布尔逻辑运算符的理解。特别是在处理逻辑表达式时，我深入研究了如何根据运算符的优先级和括号的位置正确地解析和计算真值表。通过实现包括非（!）、与（&）、或（|）、蕴含（^）和等值（~）在内的多种逻辑运算符，我不仅提升了对逻辑表达式的掌握，还学会了如何将这些运算符有效地转化为计算机能够理解的操作。

其次，这个项目让我更加了解了递归算法和数组的使用。在解析表达式时，我需要反复处理括号和运算符之间的关系，递归地分解复杂的表达式，直到最终得到结果。这种思维方式和算法设计让我意识到递归不仅是一种强大的编程工具，而且能大大简化代码的复杂性。

在调试和优化代码的过程中，我也学到了如何提高程序的效率。通过仔细设计数据结构和逻辑流程，我确保了每次计算都能正确且高效地执行。此外，我还处理了输入和输出的格式问题，确保程序能够清晰地显示真值表和主范式的结果。

总的来说，这个项目不仅锻炼了我的编程能力，也让我更深刻地理解了逻辑运算和计算机科学的基本原理，为未来的学习和实践打下了坚实的基础。

# 6 源代码

## 6.1 A题

#include <iostream>

#include <windows.h>

#include <limits>

#include <cstdlib>

using namespace std;

int main() {

int a[4];

int i, j;

char s;

bool continueOperation = true;

while (continueOperation) {

tt:cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

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

cout << "\*\* 欢迎进入逻辑运算程序 \*\*" << endl;

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

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

cout << "\n 请输入P的值（0或1）,以回车结束: ";

while (!(cin >> i) || (i != 0 && i != 1)) {

cout << "\n P的值输入有误,请重新输入!" << endl;

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

Sleep(2000);

system("cls");

goto tt;

}

cout << "\n 请输入Q的值（0或1）,以回车结束: ";

while (!(cin >> j) || (j != 0 && j != 1)) {

cout << "\n Q的值输入有误,请重新输入!" << endl;

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

Sleep(2000);

system("cls");

goto tt;

}

a[0] = i && j; // AND

a[1] = i || j; // OR

a[2] = (!i) || j; // Implication

a[3] = ((!i) || j) && ((!j) || i); // Equivalence

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;

do {

cout << "\n是否继续运算?（y/n）";

cin >> s;

if (s == 'y') {

system("cls");

} else if (s == 'n') {

cout << "欢迎下次再次使用!" << endl;

continueOperation = false;

} else {

cout << "输入错误,请重新输入!" << endl;

}

} while (s != 'y' && s != 'n');

}

return 0;

}

## 6.2 BC题

#include <iostream>

#include <cstdlib>

#include <cstring>

#include <cmath>

#include <conio.h>

using namespace std;

#define N 50

void panduan(int b[N], int f); // 赋值函数

int tkh(char sz[N], char ccu[N], int icu[N], int h0); // 分级运算函数

int fkh(char sz[N], char ccu[N], int icu[N], int h0); // 主运算函数

int main()

{

int i1, i2, d = 1, icu[N], kh = 0, jg, j = 0, h0;

int bj = 0, hq[N], h = 0, x = 0, xq[N];

char sz[N], ccu[N], sz0[N];

hq[0] = -1;

xq[0] = -1;

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

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

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

cout << "\*\* (可运算真值表,主范式,支持括号) \*\*" << endl;

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

cout << "\*\* 用!表示非 \*\*" << endl;

cout << "\*\* 用&表示与 \*\*" << endl;

cout << "\*\* 用|表示或 \*\*" << endl;

cout << "\*\* 用^表示蕴含 \*\*" << endl;

cout << "\*\* 用~表示等值 \*\*" << endl;

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

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

cout << endl;

cout << "请输入一个合法的命题公式:" << endl;

cin.getline(sz, N);

strcpy(sz0, sz);

for (unsigned i1 = 0; i1 < strlen(sz); i1++)

{

if (sz[i1] == ')' || sz[i1] == '(')

kh++;

if (sz[i1] >= 'a' && sz[i1] <= 'z' || sz[i1] >= 'A' && sz[i1] <= 'Z')

{

for (i2 = 0; i2 < j; i2++)

if (ccu[i2] == sz[i1])

d = 0;

if (d == 1)

{

ccu[j] = sz[i1];

j++;

}

d = 1;

}

}

cout << endl;

cout << "该式子中的变量个数为:" << j << endl;

h0 = j;

cout << endl;

cout << "输出真值表如下:" << endl << endl;

for (i1 = 0; i1 < h0; i1++)

cout << " " << ccu[i1] << " ";

cout << " " << sz << endl;

cout << endl;

for (i1 = 0; i1 < j; i1++)

icu[i1] = 0;

for (i2 = 0; i2 < j; i2++)

cout << " " << icu[i2] << " ";

jg = tkh(sz, ccu, icu, h0);

if (jg == 0)

hq[h++] = bj;

else

xq[x++] = bj;

cout << " " << jg << endl;

strcpy(sz, sz0);

for (i1 = 0; i1 < (int)pow(2.0, j) - 1; i1++)

{

++bj;

panduan(icu, j - 1);

jg = tkh(sz, ccu, icu, h0);

if (jg == 0)

hq[h++] = bj;

else

xq[x++] = bj;

strcpy(sz, sz0);

for (i2 = 0; i2 < j; i2++)

cout << " " << icu[i2] << " ";

cout << " " << jg << endl;

}

if (hq[0] == -1)

cout << endl << "该命题公式不存在主合取范式." << endl;

else

{

cout << endl << "该命题公式的主合取范式:" << endl << "\t";

for (i1 = 0; i1 < h; i1++)

{

if (i1 > 0)

cout << "\\/";

cout << "M(" << hq[i1] << ")";

}

}

if (xq[0] == -1)

cout << endl << "该命题公式不存在主析取范式." << endl;

else

{

cout << endl << endl << "该命题公式的主析取范式:" << endl << "\t";

for (i1 = 0; i1 < x; i1++)

{

if (i1 > 0)

cout << "/\\";

cout << "m(" << xq[i1] << ")";

}

}

cout << endl << endl << "欢迎下次再次使用!" << endl;

getch();

return 0;

}

void panduan(int b[N], int f)

{

int i;

i = f;

if (b[f] == 0)

b[f] = 1;

else

{

b[f] = 0;

panduan(b, --i);

}

}

int tkh(char sz[N], char ccu[N], int icu[N], int h0)

{

int i, j, h, s, kh = 0, wz[N], a;

char xs1[N], ckh[N];

s = strlen(sz);

for (i = 0; i < s; i++)

if (sz[i] == '(' || sz[i] == ')')

{

wz[kh] = i;

ckh[kh] = sz[i];

kh++;

}

if (kh == 0)

return fkh(sz, ccu, icu, h0);

else

{

for (i = 0; i < kh; i++)

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

break;

for (j = wz[i - 1] + 1, h = 0; j < wz[i]; j++, h++)

xs1[h] = sz[j];

xs1[h] = '\0';

a = fkh(xs1, ccu, icu, h0);

if (a == 1)

sz[wz[i - 1]] = 1;

else

sz[wz[i - 1]] = -2;

for (j = wz[i - 1] + 1; j < s + wz[i - 1] - wz[i]; j++)

sz[j] = sz[j + wz[i] - wz[i - 1]];

sz[j] = '\0';

return tkh(sz, ccu, icu, h0);

}

}

int fkh(char sz[N], char ccu[N], int icu[N], int h0)

{

int i, h = 0, j = 0, j1 = 0, j2 = 0, j3 = 0, j4 = 0, j5 = 0, i1, i2, p1 = -1, p2 = -1, s;

char dt[N];

s = strlen(sz);

if (s == 1)

if (sz[0] == -2)

return 0;

else

return 1;

else

{

for (i = 0; i < s - j; i++)

if (sz[i] == '!')

{

for (i1 = 0; i1 < h0; i1++)

if (sz[i + 1] == ccu[i1])

p1 = icu[i1];

if (sz[i + 1] == -2)

p1 = 0;

if (p1 == -1)

p1 = sz[i + 1];

dt[j + 2] = !p1;

sz[i] = j + 2;

j++;

p1 = 0;

for (i1 = i + 1; i1 < s - j; i1++)

sz[i1] = sz[i1 + 1];

}

p1 = -1;

j1 = j;

for (i = 0; i < s - j1 - 2 \* j2; i++)

if (sz[i] == '&')

{

for (i1 = 0; i1 < h0; i1++)

{

if (sz[i - 1] == ccu[i1])

p1 = icu[i1];

if (sz[i + 1] == ccu[i1])

p2 = icu[i1];

}

for (i2 = 2; i2 < j + 2; i2++)

{

if (sz[i - 1] == i2)

p1 = dt[i2];

if (sz[i + 1] == i2)

p2 = dt[i2];

}

if (sz[i - 1] == -2)

p1 = 0;

if (sz[i + 1] == -2)

p2 = 0;

if (p1 == -1)

p1 = sz[i - 1];

if (p2 == -1)

p2 = sz[i + 1];

dt[j + 2] = p1 && p2;

sz[i - 1] = j + 2;

j++;

j2++;

p1 = -1;

p2 = -1;

for (i1 = i; i1 < s - j1 - 2 \* j2; i1++)

sz[i1] = sz[i1 + 2];

i--;

}

for (i = 0; i < s - j1 - 2 \* j2 - 2 \* j3; i++)

if (sz[i] == '|')

{

for (i1 = 0; i1 < h0; i1++)

{

if (sz[i - 1] == ccu[i1])

p1 = icu[i1];

if (sz[i + 1] == ccu[i1])

p2 = icu[i1];

}

for (i2 = 2; i2 < j + 2; i2++)

{

if (sz[i - 1] == i2)

p1 = dt[i2];

if (sz[i + 1] == i2)

p2 = dt[i2];

}

if (sz[i - 1] == -2)

p1 = 0;

if (sz[i + 1] == -2)

p2 = 0;

if (p1 == -1)

p1 = sz[i - 1];

if (p2 == -1)

p2 = sz[i + 1];

dt[j + 2] = p1 || p2;

sz[i - 1] = j + 2;

j++;

j3++;

p1 = -1;

p2 = -1;

for (i1 = i; i1 < s - j1 - 2 \* j2 - 2 \* j3; i1++)

sz[i1] = sz[i1 + 2];

i--;

}

for (i = 0; i < s - j1 - 2 \* j2 - 2 \* j3 - 2 \* j4; i++)

if (sz[i] == '^')

{

for (i1 = 0; i1 < h0; i1++)

{

if (sz[i - 1] == ccu[i1])

p1 = icu[i1];

if (sz[i + 1] == ccu[i1])

p2 = icu[i1];

}

for (i2 = 2; i2 < j + 2; i2++)

{

if (sz[i - 1] == i2)

p1 = dt[i2];

if (sz[i + 1] == i2)

p2 = dt[i2];

}

if (sz[i - 1] == -2)

p1 = 0;

if (sz[i + 1] == -2)

p2 = 0;

if (p1 == -1)

p1 = sz[i - 1];

if (p2 == -1)

p2 = sz[i + 1];

dt[j + 2] = !p1 || p2;

sz[i - 1] = j + 2;

j++;

j4++;

p1 = -1;

p2 = -1;

for (i1 = i; i1 < s - j1 - 2 \* j2 - 2 \* j3 - 2 \* j4; i1++)

sz[i1] = sz[i1 + 2];

i--;

}

for (i = 0; i < s - j1 - 2 \* j2 - 2 \* j3 - 2 \* j4 - 2 \* j5; i++)

if (sz[i] == '~')

{

for (i1 = 0; i1 < h0; i1++)

{

if (sz[i - 1] == ccu[i1])

p1 = icu[i1];

if (sz[i + 1] == ccu[i1])

p2 = icu[i1];

}

for (i2 = 2; i2 < j + 2; i2++)

{

if (sz[i - 1] == i2)

p1 = dt[i2];

if (sz[i + 1] == i2)

p2 = dt[i2];

}

if (sz[i - 1] == -2)

p1 = 0;

if (sz[i + 1] == -2)

p2 = 0;

if (p1 == -1)

p1 = sz[i - 1];

if (p2 == -1)

p2 = sz[i + 1];

dt[j + 2] = (!p1 || p2) && (!p2 || p1);

sz[i - 1] = j + 2;

j++;

j5++;

p1 = -1;

p2 = -1;

for (i1 = i; i1 < s - j1 - 2 \* j2 - 2 \* j3 - 2 \* j4 - 2 \* j5; i1++)

sz[i1] = sz[i1 + 2];

i--;

}

return dt[j + 1];

}

}