**Міністерство освіти і науки України**

**Національний університет  
“Львівська політехніка”**

**Кафедра систем штучного інтелекту**

**Лабораторна робота № 1**

з дисципліни

«Дискретна математика»

**Виконав**:

студент групи КН-115  
Поставка Маркіян

**Викладач:**

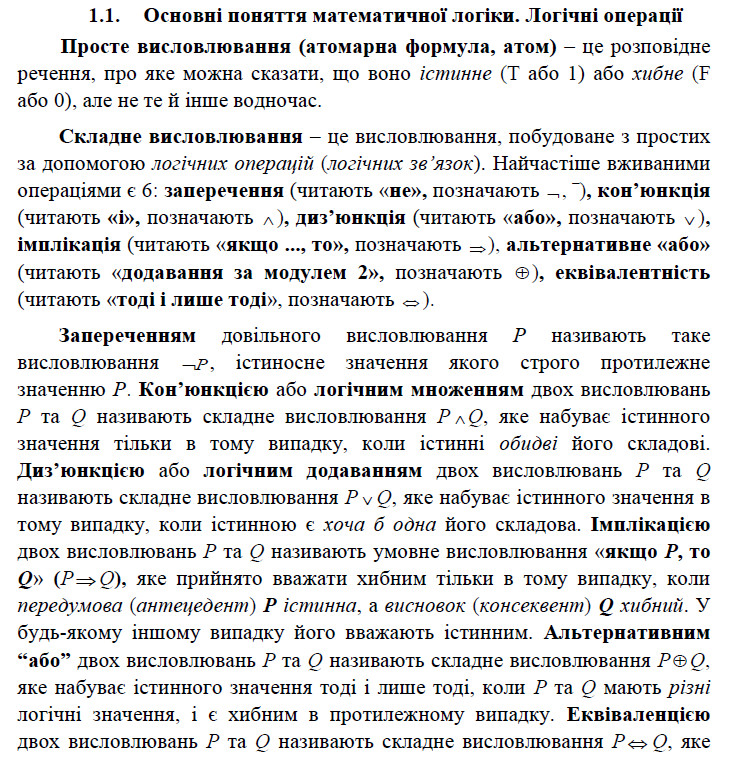
Мельникова Н. І.

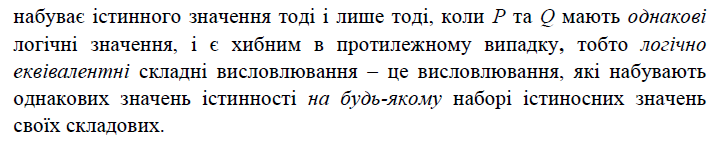
Львів – 2019р.

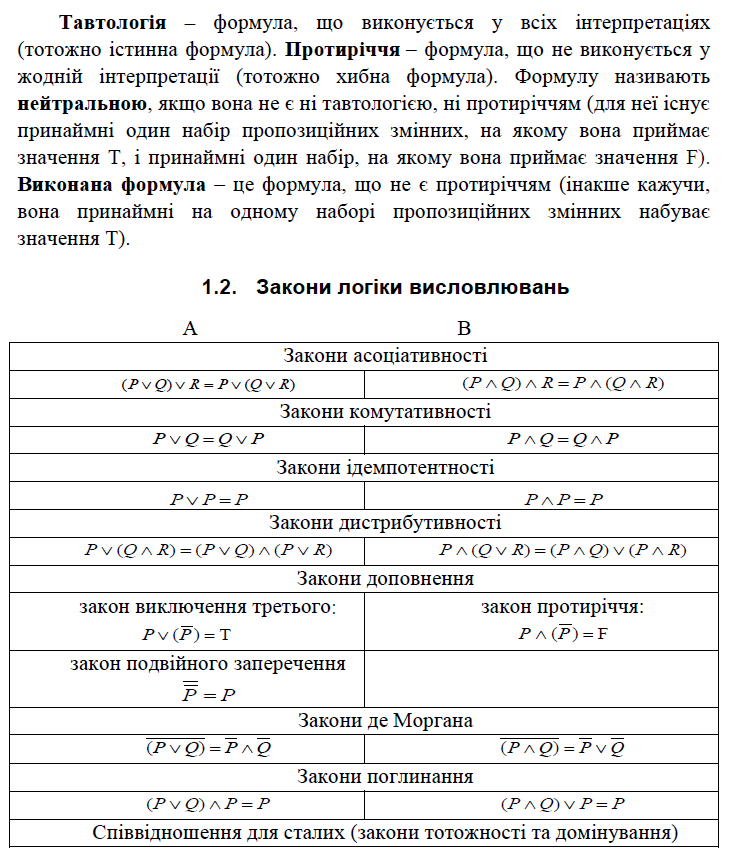
**Тема**: “Моделювання основних логічних операцій”

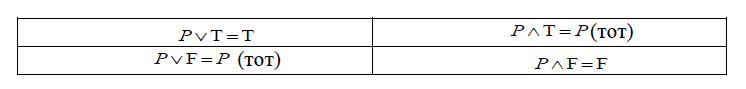
**Мета роботи:** Ознайомитись на практиці із основними поняттями математичної логіки, навчитися будувати складні висловлювання за допомогою логічних операцій та знаходити їхні істинностні значення таблицями істинності, використовувати закони алгебри логіки, освоїти методи доведень.

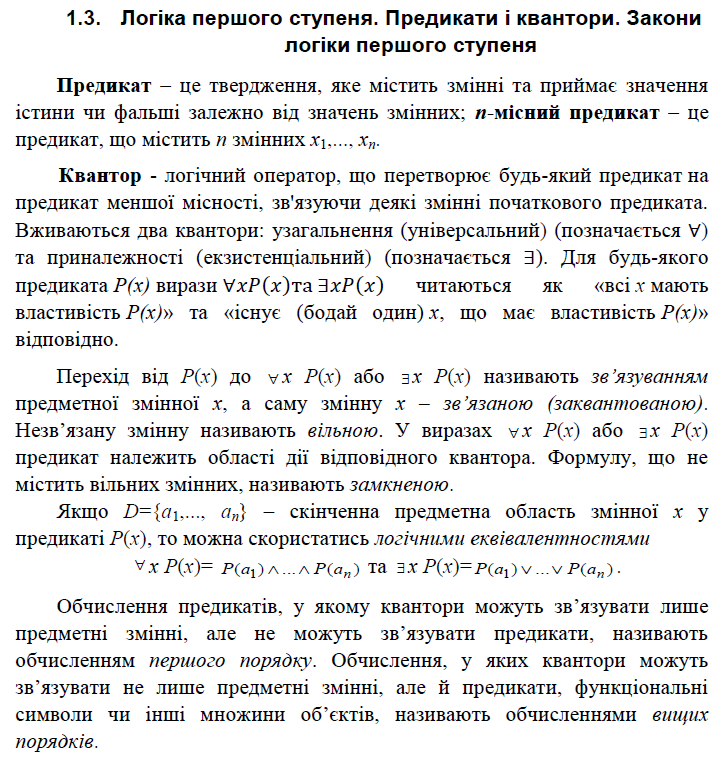
**Теоретичні відомості:**

****

****

****

****

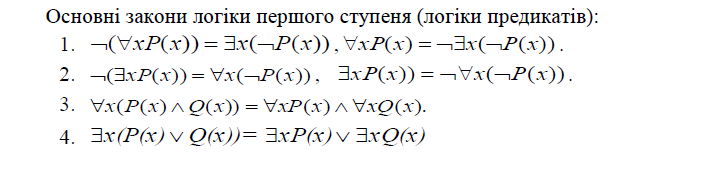
****

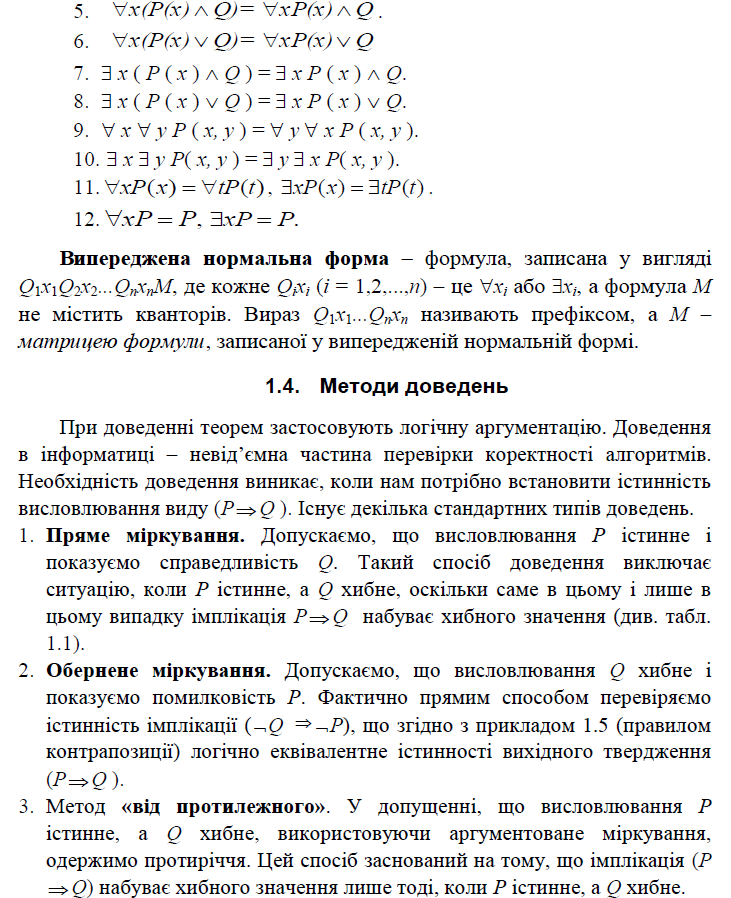
Закони 1-2 дозволяють будувати заперечення формул з кванторами.

Закони 3-4 виражають закони дистрибутивності квантора загальності відносно диз’юнкції.∃ відносно кон’юнкції та квантора існування ∀.

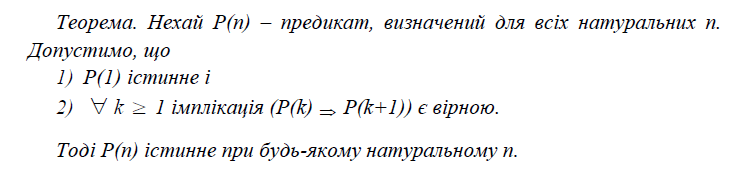
Закони 5-8 дозволяють виносити за межі дії квантора, що зв’язує змінну *х* та формулу, яка не містить *х.*

Закони 9-10 свідчать про комутативність однойменних кванторів. Тобто однойменні квантори можна міняти місцями, а різнойменні – ні.

****

****

****

****

***Означення 7.1. Випереджена нормальна форма*** *–*формула, записана у вигляді *Q1x1Q2x2...QnxnM*, де кожне *Qixi* (*i* = 1,2,...,*n*) –це ∀*xi* або ∃*xi*, а формула *M* не містить кванторів. Вираз *Q1x1...Qnxn*називають префіксом, а *M* – *матрицею формули*, записаної у випередженій нормальній формі**. Фактично, це запис формули з винесеними кванторами за дужки.**

***Приклад 7.1.*** Наведемо приклади формул, записаних у випередженій нормальній формі.

1. ∀*x*∀*y*(*P*(*x,y*∧)*Q*(*y*)).

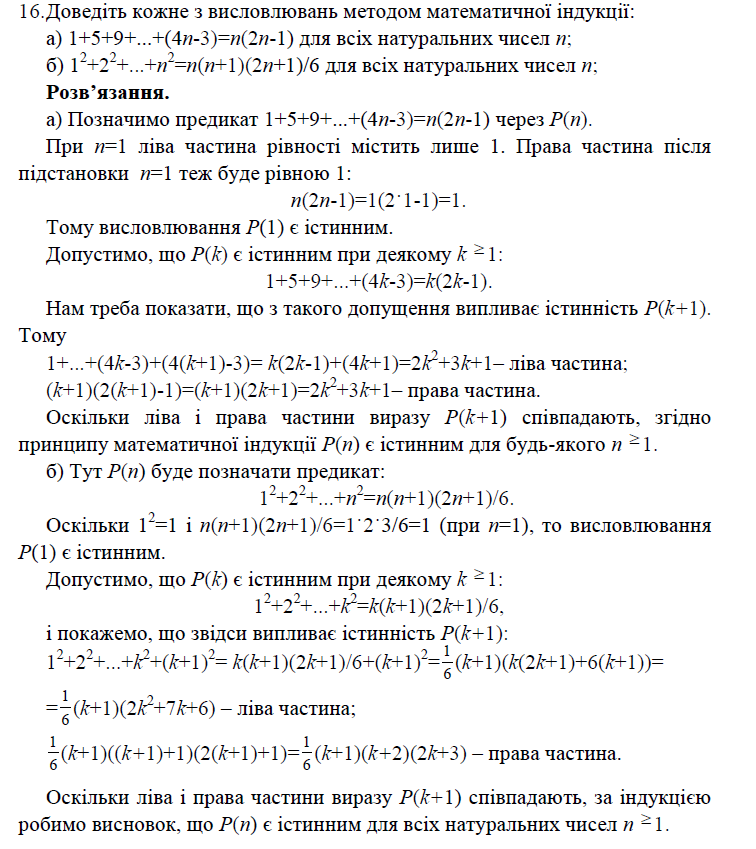
2. ∀*x*∃*y*(*P*(*x*∨)*Q*(*y*)).

3. ∀*x*∀*y*∃*z*(*Q*(*x,y*∧)*R*(*z*)).

4. ∀*x*∀*y*∀*z*∃*u*(*P*(*x,z*∨)*P*(*y,z*∨)*Q*(*x,y,u*)). ▲

Для того, щоб перевести формулу у випереджену нормальну форму, необхідно виконати наступні перетворення:

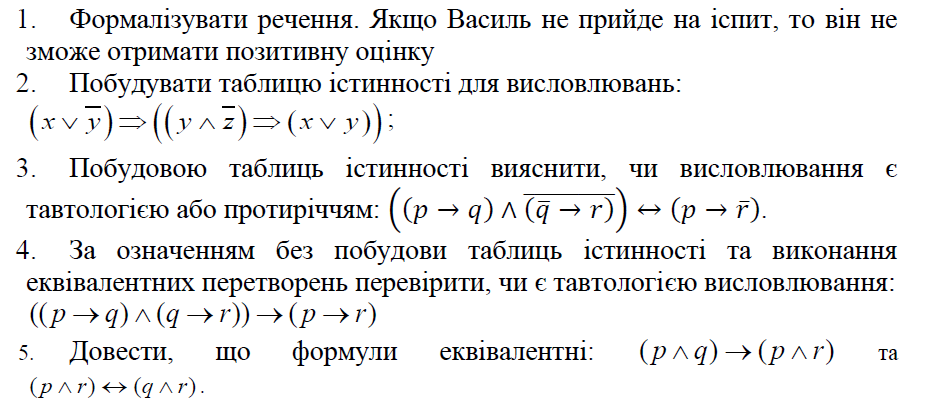
1. Використати правила усунення імплікації (*P→Q*=https://studfiles.net/html/2706/1080/html_JKlIkqZ1Iv._plg/img-m4EE4s.png∨*Q*) та еквівалентності (*P~Q*=(*P→Q*(∧)*Q→P*) ).
2. Застосувати закон подвійного заперечення (https://studfiles.net/html/2706/1080/html_JKlIkqZ1Iv._plg/img-6JSC3Q.png) та закони де Моргана (https://studfiles.net/html/2706/1080/html_JKlIkqZ1Iv._plg/img-bLe_Xm.png).
3. Застосувати закони: ¬(∀*xP*(*x*))=https://studfiles.net/html/2706/1080/html_JKlIkqZ1Iv._plg/img-VFqRoN.png  та ¬( ∃*xP*(*x*))=https://studfiles.net/html/2706/1080/html_JKlIkqZ1Iv._plg/img-xvjY0d.png.
4. Застосувати закони логіки першого ступеня 3-8.
5. Винести квантори у префікс, для чого скористатись законами логіки першого ступеня 3-8.

****

**Додаток 1:**

Варіант 11.

Постановка задачі:



Рішення:

1.

P – прийти на іспит.

B – отримати позитивну оцінку.

X – Василь.

2.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| x | y | z | x∨¬y | y∧¬z | x∨y | y∧¬z→(x∨y) | x∨¬y→(y∧¬z→(x∨y)) |
| 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 |
| 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 0 | 1 | 1 | 1 | 1 |
| 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 1 | 0 | 1 | 1 | 1 |
| 1 | 0 | 1 | 1 | 0 | 1 | 1 | 1 |
| 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 |

3.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| p | q | r | p→q | ¬q→r | ¬(¬q→r) | (p→q)∧¬(¬q→r) | p→¬r | (p→q)∧¬(¬q→)r≡(p→¬r) |
| 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 |
| 0 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 1 |
| 1 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 0 |
| 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 |

Висловлювання не є тавтологією і не є протиріччям. Воно нормальної форми.

4.

Виконуємо завдання за допомогою методу відшукання контр прикладу.

Припускаємо, що формула не є тавтологією.

Тоді

Підставляємо значення p і r у висловлювання

Спробуємо підставити q=T або q=F і бачимо що при жодному із значень q вираз не буде правдою.

З цього робимо висновок, що дане висловлювання не буде протиріччям при будь-яких значеннях q.

Отже, воно є тавтологією, що і потрібно було довести.

5.

Для доведення складемо таблицю істинності.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| p | q | r | p∧q | p∧r | p∧q→p∧r | q∧r | p∧r≡q∧r | p∧q→p∧r≡(p∧r≡q∧r) |
| 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 |
| 0 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 1 |
| 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 |
| 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 0 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

З таблиці ми бачимо, що дане висловлювання не є тавтологією. Воно нормальної форми.

**Додаток 2:**

Постановка задачі:

Написати на будь-якій відомій студентові мові програмування програму для реалізації програмного визначення значень таблиці істинності логічних висловлювань при різних інтерпретаціях для наступної формули:



Вимоги до програми:

Програма має передбачати такі можливості:

1. Автоматичне знаходження істинносних значень (із записом таблиці істинності ) складного висловлювання для всіх інтерпретацій простих висловлювань, які входять в нього, для відповідного завдання.
2. Введення вхідних даних вручну.
3. Перевірку на некоректне введення даних.

Код програми:

#include "stdafx.h"

#include <iostream>

#include <conio.h>

using namespace std;

int input\_var(int a, int code) // Метод для перевірки на коректність вхідних даних(

{

int ck = 0; // змінна для перевірки

while (ck != 2)

{

a = 0;

cout << "Enter " << (char)code << endl; // Виведення назви змінної типу x, y, z через інкрементацію ascii коду в виклику функції

cin >> a;

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

{

ck = 2;

}

else

{

cout << endl;

cout << "Again ";

}

}

return a;

}

int and(int a, int b)

{

if (a\*b == 0) return 0;

if (a\*b == 1) return 1;

}

int or (int a, int b)

{

if (a + b == 0) return 0;

if (a + b>0) return 1;

}

int not(int a)

{

if (a == 0) return 1;

else if (a == 1) return 0;

}

int impl(int a, int b)

{

if (b == 0) return 0;

else return 1;

}

void auto\_(int rez) // Метод для виводу всієї таблиці істинності

{

int x = 0; int y = 0; int z = 0;

cout << "====================================TABLE=====================================" << endl;

cout << "| X | Y | Z |(X or nY)|(Y and nZ)| (X or Y)| 2->3 | 1->[2->3] |" << endl;

cout << "==============================================================================" << endl;

x = 0;

while (x < 2)

{

y = 0;

while (y < 2)

{

z = 0;

while (z < 2)

{

int hp = impl(and (y, not(z)), or (x, y)); // Побічна допоміжна змінна

rez = impl(or (x, not(y)), hp);

cout << "| " << x << " | " << y << " | " << z << " | "<< or(x, not(y)) << " | " << and (y, not(z)) << " | "<<or(x, y) << " | " << hp << " | " << rez << " |" << endl;

z++;

}

y++;

}

x++;

}

cout << "==============================================================================" << endl;

}

void s\_manual(int rez) // Метод для виведення значень тыльки при ваших вхыдних даних

{

int code = 88;

int x = 0; int y = 0; int z = 0;

x = input\_var(x, code++);

y = input\_var(y, code++);

z = input\_var(z, code++);

cout << "====================================TABLE=====================================" << endl;

cout << "| X | Y | Z |(X or nY)|(Y and nZ)| (X or Y)| 2->3 | 1->[2->3] |" << endl;

cout << "==============================================================================" << endl;

int hp = impl(and (y, not(z)), or (x, y)); // Побічна допоміжна змінна

rez = impl(or (x, not(y)), hp);

cout << "| " << x << " | " << y << " | " << z << " | "<< or(x, not(y)) << " | " << and (y, not(z)) << " | "<<or(x, y) << " | " << hp << " | " << rez << " |" << endl;

cout << "==============================================================================" << endl;

}

int main()

{

system("MODE CON: COLS=100 LINES=30"); // Настройки консольки: довжина і висота

int a[5]; // Array for input

int n = 0;

int rez = 0; //

int ck = 0; // Menu start

while (ck != 3)

{

ck = 0;

system("@cls||clear"); // Очистка екрану

cout << "1 to auto, 2 to semi manual, 3 to exit" << endl;

cin >> ck;

if (ck == 1) auto\_(rez); // Виклик авто - методу

else if (ck == 2) s\_manual(rez); // Виклик ручного методу

getch();

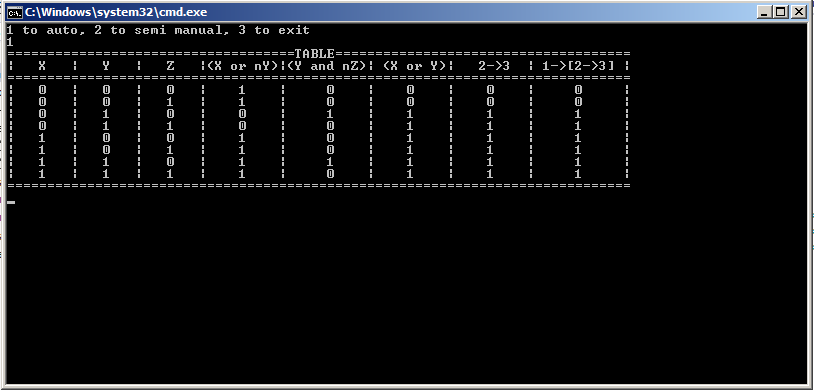
}

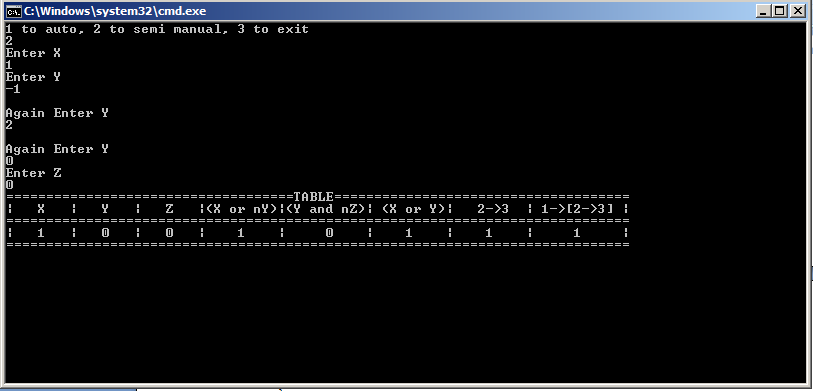
\_getch(); // Затримка екрану

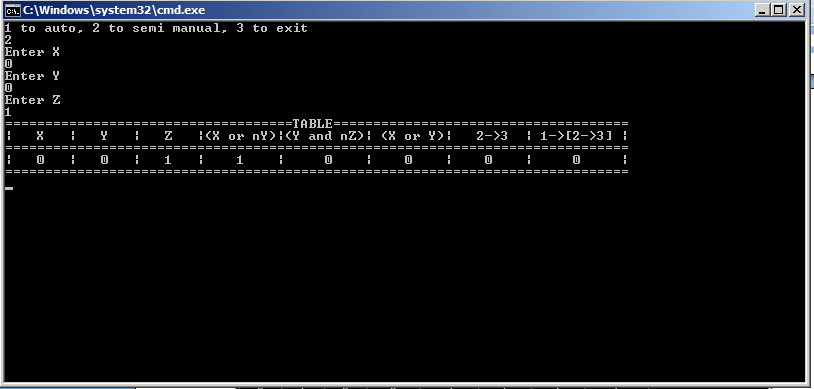
return 0;

}

Результат роботи програми:







**Висновок:** Я ознайомився на практиці із основними поняттями математичної логіки, навчився будувати складні висловлювання за допомогою логічних операцій та знаходити їхні істинностні значення таблицями істинності, використовувати закони алгебри логіки, освоїв методи доведень.