МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ НАЦІОНАЛЬНОМУ УНІВЕРСИТЕТІ “ЛЬВІВСЬКА ПОЛІТЕХНІКА”

Кафедра систем штучного інтелекту

Лабораторна робота №1

З дисципліни

“Дискретна математика”

Виконав:

Студент групи КН-115

Конопльов Павло

Викладач:

Мельникова Н.І.

Львів – 2019р.

Тема: Моделювання основних логічних операцій

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

Основні поняття математичної логіки. Логічні операції

Просте висловлювання (атомарна формула, атом) – це розповідне речення, про яке можна сказати, що воно *істинне* (T або 1) або *хибне* (F або 0), але не те й інше водночас.

Складне висловлювання – це висловлювання, побудоване з простих за допомогою *логічних операцій* (*логічних зв'язок*). Найчастіше вживаними операціями є 6: **заперечення** (читають «**не»,** позначають $\neg$, $\overline{}$**), кон'юнкція** (читають **«і»,** позначають $\wedge$ **), диз'юнкція** (читають «**або»,** позначають $\vee$ **), імплікація** (читають «**якщо ..., то»,** позначають $\Rightarrow$), **альтернативне «або»** (читають «**додавання за модулем 2»,** позначають $\oplus$ **), еквівалентність** (читають «**тоді і лише тоді**», позначають $\Leftrightarrow$).

Запереченням довільного висловлювання P називають таке висловлювання $\neg P$, істиносне значення якого строго протилежне значенню P. **Кон'юнкцією** або **логічним множенням** двох висловлювань P та Q називають складне висловлювання $P \wedge Q$, яке набуває істинного значення тільки в тому випадку, коли істинні *обидві* його складові. **Диз'юнкцією** або **логічним додаванням** двох висловлювань P та Q називають складне висловлювання $P \vee Q$, яке набуває істинного значення в тому випадку, коли істинною є *хоча б одна* його складова. **Імплікацією** двох висловлювань P та Q називають умовне висловлювання «**якщо *P*, то *Q***» **(**$P \Rightarrow Q$**),** яке прийнято вважати хибним тільки в тому випадку, коли *передумова* (*антецедент*) ***P*** *істинна*, а *висновок* (*консеквент*) ***Q*** *хибний*. У будь-якому іншому випадку його вважають істинним. **Альтернативним "або"** двох висловлювань P та Q називають складне висловлювання $P \oplus Q$, яке набуває істинного значення тоді і лише тоді, коли P та Q мають *різні* логічні значення, і є хибним в протилежному випадку. **Еквіваленцією** двох висловлювань P та Q називають складне висловлювання $P \Leftrightarrow Q$, яке

набуває істинного значення тоді і лише тоді, коли P та Q мають *однакові* логічні значення, і є хибним в протилежному випадку**,** тобто *логічно еквівалентні* складні висловлювання – це висловлювання, які набувають однакових значень істинності *на будь-якому* наборі істиносних значень своїх складових.

Тавтологія – формула, що виконується у всіх інтерпретаціях (тотожно істинна формула). **Протиріччя** – формула, що не виконується у жодній інтерпретації (тотожно хибна формула). Формулу називають **нейтральною**, якщо вона не є ні тавтологією, ні протиріччям (для неї існує принаймні один набір пропозиційних змінних, на якому вона приймає значення T, і принаймні один набір, на якому вона приймає значення F). **Виконана формула** – це формула, що не є протиріччям (інакше кажучи, вона принаймні на одному наборі пропозиційних змінних набуває значення T).

Логіка першого ступеня. Предикати і квантори.

Предикат – це твердження, яке містить змінні та приймає значення істини чи фальші залежно від значень змінних; ***n*-місний предикат** – це предикат, що містить *n* змінних $x_1,..., x_n$.

Квантор - логічний оператор, що перетворює будь-який предикат на предикат меншої місності, зв'язуючи деякі змінні початкового предиката. Вживаються два квантори: узагальнення (універсальний) (позначається) та приналежності (екзистенціальний) (позначається). Для будь-якого предиката *P(x)* вирази читаються як «всі *x* мають властивість *P(x)*» та «існує (бодай один) *x*, що має властивість *P(x)*» відповідно.

Перехід від $P(x)$ до $\forall x\, P(x)$ або $\exists x\, P(x)$ називають *зв'язуванням* предметної змінної *x*, а саму змінну *x* – *зв'язаною (заквантованою)*. Незв'язану змінну називають *вільною*. У виразах $\forall x\, P(x)$ або $\exists x\, P(x)$ предикат належить області дії відповідного квантора. Формулу, що не містить вільних змінних, називають *замкненою*.

Обчислення предикатів, у якому квантори можуть зв'язувати лише предметні змінні, але не можуть зв'язувати предикати, називають обчисленням *першого порядку*. Обчислення, у яких квантори можуть зв'язувати не лише предметні змінні, але й предикати, функціональні символи чи інші множини об'єктів, називають обчисленнями *вищих порядків*.

Випереджена нормальна форма – формула, записана у вигляді

$Q_1x_1Q_2x_2...Q_nx_nM$, де кожне Q_ix_i ($i = 1,2,...,n$) – це $\forall x_i$ або $\exists x_i$, а формула M не містить кванторів. Вираз $Q_1x_1...Q_nx_n$ називають префіксом, а M – *матрицею формули*, записаної у випередженій нормальній формі.

Методи доведень

Пряме міркування. Допускаємо, що висловлювання P істинне і показуємо справедливість Q. Такий спосіб доведення виключає ситуацію, коли P істинне, а Q хибне, оскільки саме в цьому і лише в цьому випадку імплікація $P \Rightarrow Q$ набуває хибного значення.

Обернене міркування. Допускаємо, що висловлювання Q хибне і показуємо помилковість P. Фактично прямим способом перевіряємо істинність імплікації ($\neg Q \Rightarrow \neg P$).

Метод «від протилежного». У допущенні, що висловлювання P істинне, а Q хибне, використовуючи аргументоване міркування, одержимо протиріччя. Цей спосіб заснований на тому, що імплікація ($P \Rightarrow Q$) набуває хибного значення лише тоді, коли P істинне, а Q хибне.

1. ***Принцип математичної індукції*** – це така **теорема**:

Теорема. Нехай P(n) – предикат, визначений для всіх натуральних n. Допустимо, що

1) *P(1) істинне і*
2) *$\forall$ $k \geq 1$ імплікація $(P(k) \Rightarrow P(k+1))$ є вірною.*

Тоді P(n) істинне при будь-якому натуральному n.

Варіант №11

1. Формалізувати речення. Якщо Василь не прийде на іспит, то він не зможе отримати позитивну оцінку.

   Розв’язання.

   X-Василь

   (P)-прийти на іспит

   (Q)-отримати позитивну оцінку

$$P(x) \Rightarrow Q$$

2. Побудувати таблицю істинності для висловлювань:

$$\mathbf{(x \vee \overline{y}) \Rightarrow ((y \wedge \overline{z}) \Rightarrow (x \vee y))}$$

x	y	z	$\overline{y}$	$\overline{z}$	$(x \vee \overline{y})$	$(y \wedge \overline{z})$	$(x \vee y)$	$(y \wedge \overline{z}) \Rightarrow (x \vee y)$	$(x \vee \overline{y}) \Rightarrow ((y \wedge \overline{z}) \Rightarrow (x \vee y))$
T	T	T	F	F	T	F	T	T	T
T	T	F	F	T	T	T	T	T	T
T	F	T	T	F	T	F	T	T	T
T	F	F	T	T	T	F	T	T	T
F	T	T	F	F	F	F	T	T	T
F	T	F	F	T	F	T	T	T	T
F	F	T	T	F	T	F	F	T	T
F	F	F	T	T	T	F	F	T	T

3. Побудовою таблиць істинності вияснити, чи висловлювання є тавтологією або протиріччям:
   Висловлювання є тавтологією, оскільки є всі істині значення.

4. За означенням без побудови таблиць істинності та виконання еквівалентних перетворень перевірити, чи є тавтологією висловлювання:

$$\mathbf{((p \Rightarrow q) \wedge (q \Rightarrow r)) \Rightarrow (p \Rightarrow r)}$$

   1. $((p \Rightarrow q) \wedge (q \Rightarrow r))$ – T; $(p \Rightarrow r)$ – F;
   2. $((T \Rightarrow q) \wedge (q \Rightarrow F))$ – T; p – T; r – F;
   3. T – $T \Rightarrow q$ $\qquad$ q = T

4. T – q $\Rightarrow$ F

Оскільки q не дорівнює T, то висловлювання не є тавтологією.

5. Довести, що формули еквівалентні:

$$(p \wedge q) \rightarrow (p \wedge r) \text{ та } (p \wedge r) \leftrightarrow (q \wedge r)$$

Побудуємо таблицю істинності для двох висловлювань:

p	q	r	$(p \wedge q)$	$(p \wedge r)$	$(q \wedge r)$	$(p \wedge q) \rightarrow (p \wedge r)$	$(p \wedge r) \leftrightarrow (q \wedge r)$
T	T	T	T	T	T	T	T
T	T	F	T	F	F	F	T
T	F	T	F	T	F	T	F
T	F	F	F	F	F	T	T
F	T	T	F	F	T	T	F
F	F	T	F	F	F	T	T
F	F	F	F	F	F	T	T

Оскільки значення для кожного виразу є різними то вони не є еквівалентними.

Програма:

```
    if (x == 1 && ny == 1)
    {
        xony = 1;
        printf("\t      %d", xony);
    }
    else
    {
        xony = 0;
        printf("\t      %d", xony);
    }

    if (y == 0 && nz == 0)
    {
        yanz = 0;
        printf("\t\t      %d", yanz);
    }
    else if (y == 1 || nz == 1)
    {
        yanz = 1;
        printf("\t\t      %d", yanz);
    }

    if (x == 1 && y == 1)
    {
        xory = 1;
        printf("\t\t   %d", xory);
    }
    else
    {
        xory = 0;
        printf("\t\t   %d", xory);
    }

    if ((yanz == 0) && (xory == 1))
    {
        if1 = 1;
        printf("\t\t\t\t%d", if1);
    }
    else
    {
        if1 = 0;
        printf("\t\t\t\t%d", if1);
```

ConsoleApplication4 (Глобальная область)

```cpp
#include <stdio.h>
#include <stdlib.h>
#include <iostream>

using namespace std;

int main()
{
    int x, y, z, ny = 0, nz = 0, xony = 0, yanz = 0, xory = 0, if1 = 0, if2 = 0;

    do
    {
        printf("Enter x, y, z(0 or 1): ");
        scanf_s("%d %d %d", &x, &y, &z);
    }

    while ((x >= 2) || (y >= 2) || (z >= 2));

    printf("X   Y   Z   not y   not z   (x or not y)    (y and not z)   (x or y)    (if(y and not z) that (x or y))     if((x or not y) that (if(y and not z) that (x or y)))");
    printf("\n%d    %d  %d", x, y, z);

    if (y == 0)
    {
        ny = 1;
        printf("\t  %d", ny);
    }
    else
    {
        ny = 0;
        printf("\t  %d", ny);
    }

    if (z == 0)
    {
        nz = 1;
        printf("\t  %d", nz);
    }
    else
    {
        nz = 0;
        printf("\t  %d", nz);
    }

```

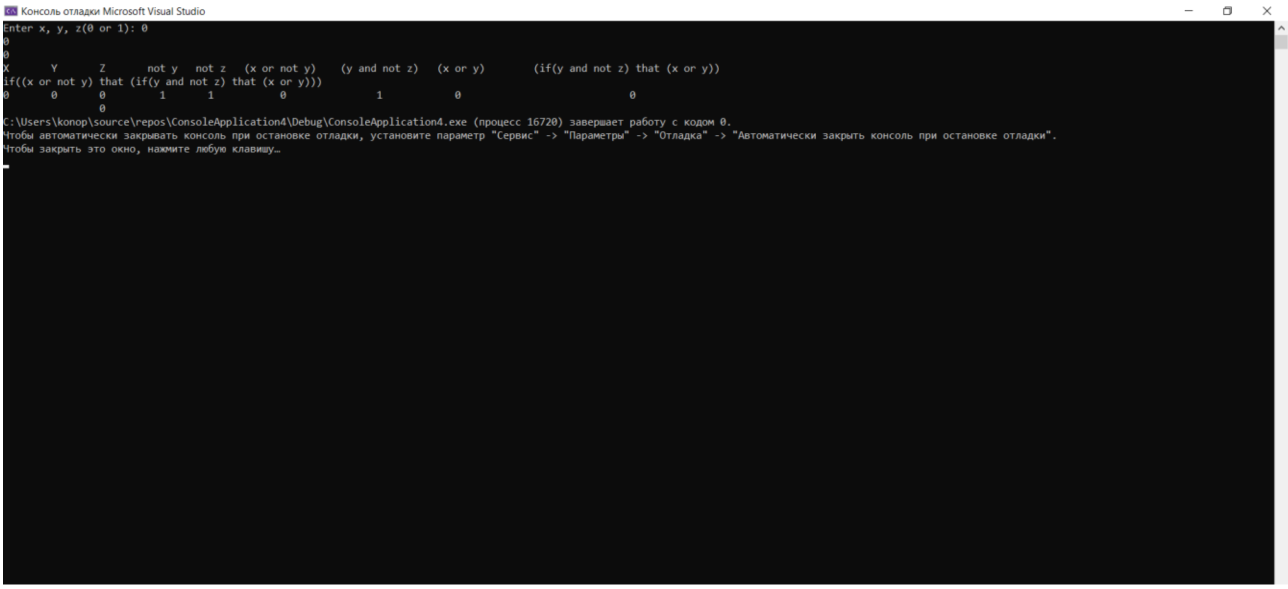
100 % Проблемы не найдены.

ConsoleApplication4 (Глобальная область)

```c
    {
        yanz = 1;
        printf("\t\t      %d", yanz);
    }

    if (x == 1 && y == 1)
    {
        xory = 1;
        printf("\t\t   %d", xory);
    }
    else
    {
        xory = 0;
        printf("\t\t   %d", xory);
    }

    if ((yanz == 0) && (xory == 1))
    {
        if1 = 1;
        printf("\t\t\t\t%d", if1);
    }
    else
    {
        if1 = 0;
        printf("\t\t\t\t%d", if1);
    }

    if ((xony == 0) && (if1 == 1))
    {
        if2 = 1;
        printf("\t\t\t\t\t\t%d", if2);
    }
    else
    {
        if2 = 0;
        printf("\t\t\t\t\t\t%d", if2);
    }

    return 0;

}
```

100 % Проблемы не найдены.

Результат виконання програми:



Висновок: під час лабораторної роботи я навчився будувати складні висловлювання за допомогою логічних операцій, знаходити їхні істинностні значення таблицями істинності, використовувати закони логіки, та методи доведень.