

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ <ЛЬВІВСЬКА ПОЛІТЕХНІКА>

Інститут ІКНІ

Кафедра систем штучного інтелекту



ЗВІТ

Лабораторна робота **№1**
З курсу "Дискретна математика"

Виконав:

Гавриляк Тарас

гр. КН-110

Прийняв(ла):

ст. вк. Мельникова Н.І.

Тема:

”Моделювання основних логічних операцій”

Мета роботи:

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

Теоретичні відомості:

1.1. Основні поняття математичної логіки. Логічні операції

Просте висловлювання (атомарна формула, атом) – це розповідне речення, про яке можна сказати, що воно *істинне* (Т 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 , $-$), **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).

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 Т, 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 Т).

1.2. Закони логіки висловлювань

А	В
Закони асоціативності	
$(P \vee Q) \vee R = P \vee (Q \vee R)$	$(P \wedge Q) \wedge R = P \wedge (Q \wedge R)$
Закони комутативності	
$P \vee Q = Q \vee P$	$P \wedge Q = Q \wedge P$
Закони ідемпотентності	
$P \vee P = P$	$P \wedge P = P$
Закони дистрибутивності	
$P \vee (Q \wedge R) = (P \vee Q) \wedge (P \vee R)$	$P \wedge (Q \vee R) = (P \wedge Q) \vee (P \wedge R)$
Закони доповнення	
закон виключення третього: $P \vee (\bar{P}) = T$	закон протиріччя: $P \wedge (\bar{P}) = F$
закон подвійного заперечення $\bar{\bar{P}} = P$	
Закони де Моргана	
$\overline{(P \vee Q)} = \bar{P} \wedge \bar{Q}$	$\overline{(P \wedge Q)} = \bar{P} \vee \bar{Q}$
Закони поглинання	
$(P \vee Q) \wedge P = P$	$(P \wedge Q) \vee P = P$
Співвідношення для сталих (закони тотожності та домінування)	
$P \vee T = T$	$P \wedge T = P$ (тот)
$P \vee F = P$ (тот)	$P \wedge F = F$

1.3. Логіка першого ступеня. Предикати і квантори. Закони логіки першого ступеня

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, \dots, x_n .

Перехід від $P(x)$ до $x P(x)$ або $x P(x)$ називають зв'язуванням предметної змінної x , а саму змінну x – зв'язаною (заквантованою).

Незв'язану змінну називають вільною. У виразах $x P(x)$ або $x P(x)$ предикат належить області дії відповідного квантора. Формулу, що не містить вільних змінних, називають замкненою.

Якщо $D = \{a_1, \dots, a_n\}$ – скінченна предметна область змінної x у предикаті $P(x)$, то можна скористатись логічними еквівалентностями $x P(x) = P(a_1) \wedge \dots \wedge P(a_n)$ та $x P(x) = P(a_1) \vee \dots \vee P(a_n)$.

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*.

Основні закони логіки першого ступеня (логіки предикатів):

1. $\neg(\forall x P(x)) = \exists x(\neg P(x))$, $\forall x P(x) = \neg \exists x(\neg P(x))$.
2. $\neg(\exists x P(x)) = \forall x(\neg P(x))$, $\exists x P(x) = \neg \forall x(\neg P(x))$.
3. $\forall x(P(x) \wedge Q(x)) = \forall x P(x) \wedge \forall x Q(x)$.
4. $\exists x(P(x) \vee Q(x)) = \exists x P(x) \vee \exists x Q(x)$.
5. $\forall x(P(x) \wedge Q) = \forall x P(x) \wedge Q$.
6. $\forall x(P(x) \vee Q) = \forall x P(x) \vee Q$.
7. $\exists x(P(x) \wedge Q) = \exists x P(x) \wedge Q$.
8. $\exists x(P(x) \vee Q) = \exists x P(x) \vee Q$.
9. $\forall x \forall y P(x, y) = \forall y \forall x P(x, y)$.
10. $\exists x \exists y P(x, y) = \exists y \exists x P(x, y)$.
11. $\forall x P(x) = \forall t P(t)$, $\exists x P(x) = \exists t P(t)$.
12. $\forall x P = P$, $\exists x P = P$.

Випереджена нормальна форма – формула, записана у вигляді $Q_1 x_1 Q_2 x_2 \dots Q_n x_n M$, де кожне $Q_i x_i$ ($i = 1, 2, \dots, n$) – це $\forall x_i$ або $\exists x_i$, а формула M не містить кванторів. Вираз $Q_1 x_1 \dots Q_n x_n$ називають префіксом, а M – *матрицею формули*, записаної у випередженій нормальній формі.

1.4. Методи доведень

При доведенні теорем застосовують логічну аргументацію. Доведення в інформатиці – невід'ємна частина перевірки коректності алгоритмів. Необхідність доведення виникає, коли нам потрібно встановити істинність висловлювання виду $(P \Rightarrow Q)$. Існує декілька стандартних типів доведень.

1. **Пряме міркування.** Допускаємо, що висловлювання P істинне і показуємо справедливність 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 (див. табл. 1.1).

2. **Обернене міркування.** Допускаємо, що висловлювання Q хибне і показуємо помилковість P . Фактично прямим способом перевіряємо істинність імплікації $(\neg Q \rightarrow \neg P)$, що згідно з прикладом 1.5 (правилом контрапозиції) логічно еквівалентне істинності вихідного твердження $(P \Rightarrow Q)$.

3. **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.

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

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

1) $P(1)$ істинне і

2) $k \geq 1$ імплікація $(P(k) \Rightarrow P(k+1))$ є вірною.
Тоді $P(n)$ істинне при будь-якому натуральному n .

Контрольні запитання до практичних занять та лабораторної роботи з теми № 1

1. Що називають простим і складним висловлюванням? Наведіть приклади простих та складних висловлювань.
2. За допомогою яких логічних операцій будують складні висловлювання?
3. Яка формула є тавтологією?
4. Яка формула є протиріччям?
5. Які є закони логіки висловлювань?
6. Якими способами перевіряють еквівалентність тверджень?
7. Назвіть головну ідею метода відшукування контрприкладу.
8. Що називають предикатом і кванторами? Наведіть приклади предикатів і кванторів.
9. Які змінні називають зв'язаними, а які вільними? Наведіть приклади формул з такими змінними.
10. Які закони логіки першого ступеня ви знаєте?
11. Яку формулу називають випередженою нормальною формою?
12. Перелічіть кроки, за допомогою яких переводять формулу у випереджену нормальну форму.
13. Які методи доведень ви знаєте?
14. У чому полягає принцип математичної індукції?

Завдання до практичних занять та лабораторної роботи з теми № 1

1. Отримати індивідуальний варіант завдань.
2. Розв'язати індивідуальне завдання згідно варіанту з Додатку № 1.
3. Написати на будь-якій відомій студентів мові програмування програму для реалізації вказаних логічних операцій над заданими висловлюваннями відповідно до варіанту з Додатку №2.
4. Оформити звіт про виконану роботу
Звіт повинен включати:
 - 4.1. титульний аркуш,
 - 4.2. тема,
 - 4.3. мета,
 - 4.4. теоретичні відомості,
 - 4.5. завдання варіанту з додатку 1,
 - 4.6. розв'язок,
 - 4.7. завдання варіанту з додатку 2,
 - 4.8. скрін коду програми з коментарями,
 - 4.9. скрін результату виконання програми,
 - 4.10. висновок про виконання лабораторної та практичної роботи.
5. Продемонструвати викладачеві результати, відповісти на запитання стосовно виконання роботи. Лабораторна робота вважається зарахованою, якщо програма протестована разом з викладачем та

отримано вірний результат під час аудиторних занять.

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:
 - задати кількість простих висловлювань;
 - задати логічні операції, які пов'язують прості висловлювання.
3. Некоректне введення даних.

Варіант № 6

1. Формалізувати речення.

Якщо завтра буде холодно, я одягну тепле пальто, якщо рукав буде полагоджений; завтра буде холодно, а рукав не буде полагоджений, отже, я не одягну тепле пальто.

Нехай

p – буде холодно;

q – одягну тепле пальто;

b – рукав полагоджений.

Тоді формалізовані речення матимуть вигляд:

$$p \Rightarrow (q \Rightarrow b) \wedge p \Rightarrow ((\neg b) \Rightarrow (\neg q)).$$

2. Побудувати таблицю істинності для висловлювань:

$$(x \Rightarrow (y \Rightarrow z)) \Rightarrow ((x \wedge y) \Rightarrow z);$$

x	y	z	$y \Rightarrow z$	$x \Rightarrow (y \Rightarrow z)$	$x \wedge y$	$x \wedge y \Rightarrow z$	$(x \Rightarrow (y \Rightarrow z)) \Rightarrow ((x \wedge y) \Rightarrow z)$
0	0	0	1	1	0	1	1
0	0	1	1	1	0	1	1
0	1	0	0	1	0	1	1
0	1	1	1	1	0	1	1
1	0	0	1	1	0	1	1
1	0	1	1	1	0	1	1
1	1	0	0	0	1	0	1
1	1	1	1	1	1	1	1

3. Побудовою таблиць істинності вияснити чи висловлювання є тавтологіями або суперечностями:

$$((p \wedge q) \Rightarrow (q \Leftrightarrow r)) \Rightarrow (\neg(p \vee r))$$

p	q	r	$p \wedge q$	$q \Leftrightarrow r$	$(p \wedge q) \Rightarrow (q \Leftrightarrow r)$	$p \vee r$	$\neg(p \vee r)$	Загальний вигляд
0	0	0	0	1	1	0	1	1
0	0	1	0	0	1	1	0	0
0	1	0	0	0	1	0	1	1
0	1	1	0	1	1	1	0	0
1	0	0	0	1	1	1	0	0
1	0	1	0	0	1	1	0	0
1	1	0	1	0	0	1	0	1
1	1	1	1	1	1	1	0	0

Висловлювання не є тавтологією і також не є суперечністю.

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

$$((p \rightarrow q) \wedge (q \rightarrow p)) \rightarrow p;$$

Висловлювання не є тавтологією, якщо хоча б в одному з випадків воно буде хибним.

Знайдімо значення змінних при яких вираз не буде істинним.

$$((p \rightarrow q) \wedge (q \rightarrow p)) = T, \text{ а } p = F$$

$q \rightarrow q$ завжди набуває істинного значення (T).

Так, як $p = F$, то $p \rightarrow q$ завжди набуватиме істинного значення.

Так, як $((p \rightarrow q) \wedge (q \rightarrow p))$ завжди набуватиме істинного значення, при $p = F$

Увесь даний вираз буде не істинним (F).

Отже вираз не є тавтологією.

5. Довести, що формули еквівалентні:

$$p \rightarrow (q \wedge r) \text{ та } p \vee (q \oplus r).$$

$$1. p \rightarrow (q \wedge r) = (\neg p) \vee (q \wedge r) = ((\neg p) \vee q) \wedge ((\neg p) \vee r)$$

$$\begin{aligned}
2. & p \vee (q \oplus r) = p \vee (\neg (q \Leftrightarrow r)) = p \vee (\neg ((q \rightarrow r) \wedge (r \rightarrow q))) = \\
& = p \vee ((\neg (q \rightarrow r) \vee (\neg (r \rightarrow q)))) = p \vee (\neg ((\neg q) \vee r) \vee (\neg ((\neg r) \vee q))) = \\
& = p \vee (q \wedge (\neg (r)) \vee (r \wedge (\neg (q))))
\end{aligned}$$

Формули не є еквівалентними.

Програма до завдання №2

```

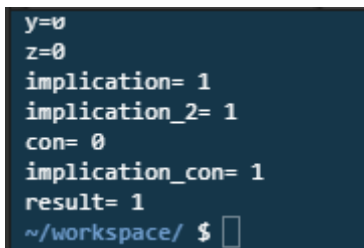
1.#include <stdio.h>
2.
3.int main() {
4.
5.    int x, y, z;
6.
7.    do {
8.        printf("x=");
9.        scanf("%d", &x);
10.        if(( x < 0 ) || ( x > 1 ))
11.            printf("Try again\n");
12.    } while( (x < 0) || ( x > 1 ) );
13.
14.
15.    do {
16.        printf("y=");
17.        scanf("%d", &y);
18.        if(( y < 0 ) || ( y > 1 ))
19.            printf("Try again");
20.    } while( (y < 0) || ( y > 1 ) );
21.
22.    do {
23.        printf("z=");
24.        scanf("%d", &z);
25.        if(( z < 0 ) || ( z > 1 ))
26.            printf("Try again");
27.    } while((z < 0) || (z > 1 ));
28.
29.    int implication = !y | z;
30.    printf("implication= %d\n", implication );
31.
32.    int implication_2 = !x | implication;
33.    printf("implication_2= %d\n", implication_2);
34.
35.    int con = x & y;
36.    printf("con= %d\n", con);

```



```
37.  
38. int implication_con = !con | z;  
39.     printf("implication_con= %d\n", implication_con);  
40.  
41. int result = !implication_2 | implication_con;  
42.     printf("result= %d\n", result);  
43.  
44. return 0;  
45.}
```

Результат виконання програми:



```
y=0  
z=0  
implication= 1  
implication_2= 1  
con= 0  
implication_con= 1  
result= 1  
~/workspace/ $
```

При різних значення змінних x , y , z програма обраховує всі послідовні складні висловлювання і в залежності від їхніх результатів обраховує результат початкового (усього) виразу.

Висновок:

Під час виконання лабораторної роботи я ознайомився з основними поняттями математичної логіки, навчився будувати складні висловлювання за допомогою логічних операцій, навчився знаходити їхні значення істинності за допомогою таблиць істинності, використовуючи закони алгебри логіки та освоїв їхні методи доведення.