МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
МИКОЛАЇВСЬКИЙ НАЦІОНАЛЬНИЙ АГРАРНИЙ УНІВЕРСИТЕТ

**Кафедра економічної кібернетики,
комп'ютерних наук та інформаційних технологій**

ДИСКРЕТНА МАТЕМАТИКА

Конспект лекцій для здобувачів першого (бакалаврського) рівня вищої освіти ОПП «Комп'ютерні науки» спеціальності 122 «Комп'ютерні науки» денної форми здобуття вищої освіти

Миколаїв - 2025

УДК 519.1
Д48

Друкується за рішенням науково-методичною комісією факультету менеджменту Миколаївського національного університету від 21.02.2025 року протокол № 6

Укладачі:

О. Ю. Пархоменко – канд. фіз.-мат. наук, доцент,доцент кафедри економічної кібернетики і математичного моделювання, Миколаївський національний аграрний університет;

Рецензенти:

Дармосюк В. М. - канд. фіз.-мат. наук, доцент кафедри вищої та прикладної математики, Миколаївський національний аграрний університет

© Миколаївський національний аграрний університет, 2025

ЗМІСТ

ВСТУП

Дискретна математика є однією з ключових теоретичних основ сучасних комп'ютерних наук. Вона формує фундамент для розуміння логіки програмування, побудови алгоритмів, структурування даних, розробки баз даних, аналізу мереж, а також систем штучного інтелекту. Саме завдяки інструментам і поняттям дискретної математики можлива формалізація інформаційних процесів, створення математичних моделей складних систем, ефективне опрацювання структурованих і неструктурованих даних.

Цей посібник створено для здобувачів першого (бакалаврського) рівня вищої освіти за спеціальністю 122 «Комп'ютерні науки». Матеріал адаптовано відповідно до освітньо-професійної програми та структуровано таким чином, щоб забезпечити поступове формування знань: від базових понять теорії множин до складніших розділів, як-от бінарні відношення, функції, композиції, відношення еквівалентності та порядку.

Особливу увагу приділено прикладному аспекту викладу: майже кожна теоретична концепція супроводжується прикладами з IT-сфери, що сприяє кращому засвоєнню матеріалу студентами спеціальності «Комп'ютерні науки». Наведено практичні інтерпретації множин, логічних операцій, функціональних залежностей, реляційних структур та інших концептів в контексті програмування, баз даних, інформаційної безпеки та аналітики.

Мета посібника – не лише ознайомити студентів із базовими поняттями дискретної математики, а й навчити їх застосовувати ці знання для розв'язання прикладних задач у сфері IT. Сподіваємося, що матеріали цього курсу стануть надійним підґрунтям для подальшого фахового розвитку студентів і успішного засвоєння профільних дисциплін.

1. ТЕОРІЯ МНОЖИН

Поняття множини є одним із базових у математиці. В сучасній науці та інженерії, зокрема в комп'ютерних науках, воно має надзвичайно широке застосування – від структурування даних до опису алгоритмів, формальної логіки та програмних інтерфейсів.

Множина – це сукупність об'єктів, які розглядаються як єдине ціле. Об'єкти, що належать до множини, називають **елементами множини**. Множина вважається визначеною, якщо для кожного об'єкта можна однозначно сказати, чи належить він їй чи ні.

Приклади:

– множина ідентифікаторів користувачів системи доступу (user_id);

– множина IP-адрес, що отримали доступ до сервера протягом останніх 24 годин;

– множина імен змінних у програмі на Python;

– множина дозволених статусів у базі даних ({'active', 'inactive', 'suspended'}).

Поняття множини є результатом абстракції: всі характеристики об'єктів, окрім однієї чи кількох, відкидаються. Наприклад, у множині користувачів, які виконали вхід до системи протягом останньої години, ігноруються такі властивості, як ім'я, роль чи вік, але враховується факт авторизації у певний проміжок часу.

Цікаво, що один і той самий об'єкт може одночасно бути елементом різних множин, залежно від обраної характеристики. Наприклад, IP-адреса 192.168.1.1 може входити до множини усіх локальних адрес, а також до множини адрес, що були заблоковані системою безпеки.

Позначення:

– Множини зазвичай позначаються великими літерами: A, B, C, X, Y, Z.

– Елементи множин – малими: a, b, x, y.

– Належність елемента x до множини A записується як $x \in A$.

– Запис $x \notin A$ означає, що x не належить множині A.

Приклад

– A – множина цілих чисел, менших за 10;

– B – множина ключових слів мови програмування Python;

– C – множина бітів ({0, 1});

– D – множина унікальних кодів помилок, що повертає веб-сервер;

– E – множина ролей користувача в інформаційній системі ({'admin', 'editor', 'viewer'}).

Усі ці приклади ілюструють, що множини можуть складатися як з чисел чи символів, так і з рядків, структур, об'єктів тощо – все залежить від контексту задачі.

1.1. Способи подання множин

У практиці програмування та моделювання часто виникає потреба чітко визначати множини. У математиці існує кілька способів задання множини. Кожен із них має свої переваги, залежно від контексту задачі.

1. Словесний (вербальний) опис

Множина задається описом властивості, яку повинні мати всі її елементи.

Приклад:

Множина A – «усі користувачі, які мають роль адміністратор у системі управління доступом».

2. Перелічення (список елементів)

Множина задається явним переліком своїх елементів у фігурних дужках.

Приклад:

B = {'admin', 'editor', 'viewer'} – множина всіх допустимих ролей користувача.

C = {192, 168, 10, 254} – множина компонентів IP-адреси.

3. Породжувальний (предикатний) спосіб

Множина задається предикатом, тобто логічним висловлюванням, що визначає умову належності елементів.

Запис має вигляд:

$$M = \{x \in X \mid P(x)\}$$

де $P(x)$ – предикат (умова), яка набуває значення істина, якщо x належить множині.

Приклади:

$D = \{n \in N \mid n \bmod 2 = 0\}$ – множина всіх парних натуральних чисел;

$F = \{u \in Users \mid u.status = 'active'\}$ – множина активних користувачів у системі.

4. Рекурсивний (процедурний) спосіб

Множина задається за допомогою правила породження елементів на основі базових. Часто застосовується в теорії обчислень або при роботі з рекурсією в програмуванні.

Приклад:

Множина всіх правильно побудованих дужкових виразів над символами (і):

- Базовий випадок: ε (порожній рядок) належить множині;

- Якщо w належить множині, то також належать і вирази (+ w +), або w_1+w_2, де w_1, w_2 належать множині.

5. Аналітичний спосіб (за допомогою формули)

Множина задається формулою, яка описує залежність або закономірність.

Приклад:

$F = \{x \in R \mid x = 2n + 1, n \in Z\}$ – множина всіх непарних чисел;

$G = \{s \in strings \mid len(s) \leq 8\}$ – всі рядки довжиною не більше 8 символів.

Коротке порівняння:

Спосіб	Коли зручно використовувати
Словесний	При описі множин у специфікаціях або формальних вимогах
Перелічення	Для скінченних, невеликих множин
Предикатний	У базах даних, SQL-запитах, фільтрах
Рекурсивний	Для синтаксичних структур, парсерів, генераторів
Аналітичний	Для математичного опису множин, наприклад при розрахунках

1.2. Порожня множина

У теорії множин та в комп'ютерних науках часто виникають ситуації, коли певна множина не має жодного елемента. Така множина називається порожньою множиною.

Означення.

Порожня множина – це множина, яка не містить жодного елемента.

Позначається символом $\emptyset$ або $\{\}$.

Формально:

$\emptyset = \{x \mid x \neq x\}$ – жоден елемент не задовольняє цю умову.

Приклади в IT:

- результат SQL-запиту, який не знайшов жодного рядка: SELECT * FROM users WHERE status = 'banned' AND role = 'admin';
- множина доступних портів на сервері, якщо всі заблоковані брандмауером;
- множина знайдених ключових слів у пошуку, коли жодне слово не відповідає запиту;
- множина файлів у директорії, якщо вона порожня;
- множина вхідних даних до функції, яка нічого не отримала (inputs = []).

Порожня множина вважається скінченною, хоча й має 0 елементів.

Навіщо потрібна порожня множина?

У багатьох комп'ютерних алгоритмах зручно припускати, що результат операції – це завжди множина. Якщо немає результатів – повертається порожня множина, а не «null» або помилка. Це робить алгоритми лаконічнішими та безпечнішими. Наприклад, у мовах програмування функції пошуку часто повертають [], а не None.

У будь-яку множину A можна вкласти $\emptyset$ як підмножину: $\emptyset \subseteq A$ – завжди істинно.

Порожня множина унікальна – не існує двох різних множин без елементів.

Тобто: якщо A – порожня і B – порожня, то $A = B = \emptyset$.

1.3. Операції над множинами (з прикладами з ІТ)

У теорії множин, як і в програмуванні, над множинами можна виконувати різноманітні операції, які дозволяють будувати нові множини на основі наявних. Основні операції: об'єднання, переріз, різниця, диз'юнктивна сума (симетрична різниця), доповнення.

Розглянемо дві множини A та B. Для графічної ілюстрації будемо використовувати кола Ейлера. Для зображення множини на площині креслять замкнену лінію із заштрихованою внутрішньою областю (найчастіше – це коло, звідси й назва відповідного інструмента, що широко застосовується в теорії множин).

1. Об'єднання

Означення 1. *Об'єднання A і B ($A \cup B$) – множина, що складається з усіх елементів множини A, всіх елементів множини B і не містить ніяких інших елементів* (рис.1), тобто

$$A \cup B = \{x \mid x \in A \vee x \in B\},$$

де символ «$\vee$» позначає логічну операцію диз'юнкції (логічне «або»), тобто, елемент x належить множині $A \cup B$ тоді і тільки тоді, коли або

1. $x \in A$ істинне і $x \in B$ хибне, або
2. $x \in B$ істинне і $x \in A$ хибне, або
3. $x \in A$ істинне і $x \in B$ істинне.

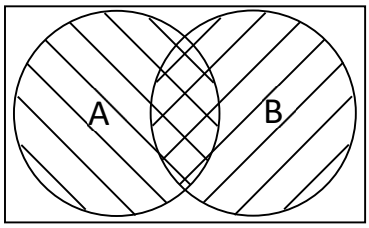


Рис.1

Множина $A \cup B$ містить усі елементи, які належать або A, або B, або обом.

Приклад:

A – користувачі з правом читання файлів;

B – користувачі з правом запису;

$A \cup B$ – усі користувачі, які мають принаймні одне з цих прав.

2. Переріз

Означення 2. *Переріз A і B – множина, що складається з тих і тільки тих елементів, які належать одночасно множині A та множині B* (рис.2), тобто

$$A \cap B = \{x \mid x \in A \wedge x \in B\},$$

де символ «$\wedge$» позначає логічну операцію кон'юнкції (логічне «і»), тобто елемент $x \in A \cap B$ тоді і тільки тоді, коли $x \in A$ і $x \in B$ істинні одночасно.

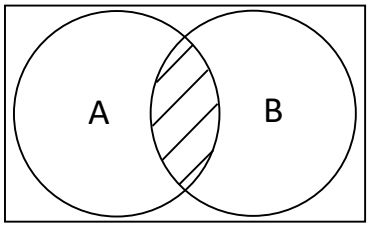


Рис.2

Приклад:

A – зареєстровані учасники онлайн-курсу;

B – користувачі, що склали тестування;

$A \cap B$ – зареєстровані користувачі, які склали тестування.

3. Різниця

Означення 3. *Різниця A і B (відносне доповнення) – множина, що складається з тих і тільки тих елементів, які належать множині A й не належать B* (рис.3), тобто

$$A - B = \{x \mid x \in A \wedge x \notin B\}.$$

Тобто, елемент $x \in A - B$ тоді і тільки тоді, коли $x \in A$ істинне і $x \in B$ хибне одночасно.

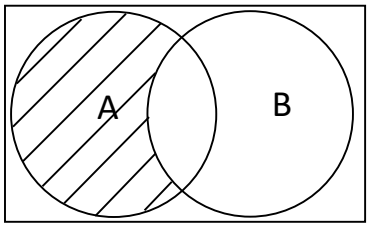


Рис.3

Приклад:

A – всі активні акаунти в системі;

B – акаунти з підозрілою активністю;

$A - B$ або $A \backslash B$ – активні акаунти без ознак підозрілої поведінки.

4. Диз'юнктивна сума (симетрична різниця)

Означення 4. *Диз'юнктивна сума A і B (симетрична різниця) – множина, що складається з усіх елементів A, які не належать множині B, й усіх елементів B, які не належать множині A, та яка не містить ніяких інших елементів* (рис.4), тобто

$$A \oplus B = \{x \mid x \in A \oplus x \in B\}.$$

Тобто, елемент $x \in A \oplus B$ тоді і тільки тоді, коли або

1. $x \in A$ істинне і $x \in B$ хибне одночасно, або
2. $x \in B$ істинне і $x \in A$ хибне одночасно.

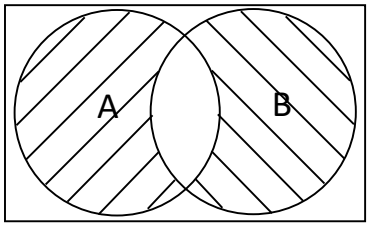


Рис.4

Очевидно, що

$$A \oplus B = (A - B) \cup (B - A).$$

Приклад:

A – IP-адреси, з яких надсилались запити сьогодні;

B – IP-адреси, з яких надсилались запити вчора;

$A \bigoplus B$ або $A\Delta B$ – IP-адреси, які використовувались тільки одного з днів.

У програмах ці операції часто реалізуються через множини (set) у мовах програмування типу Python:

```
admins = {'alice', 'bob'}
editors = {'bob', 'carol'}
print(admins | editors)   # об'єднання
print(admins & editors)   # перетин
print(admins - editors)   # різниця
print(admins ^ editors)   # симетрична різниця
```

1.4. Універсум (основна множина)

Універсум U – це множина, яка містить усі об'єкти, які вважаються допустимими у межах певного контексту. Усі інші множини, що розглядаються у цьому контексті, є підмножинами універсуму.

У комп'ютерних науках роль універсуму виконує набір допустимих значень або «контрольна множина», з якою ми працюємо у задачі. Це дуже схоже на простір визначення у програмуванні, типах даних, базах даних чи формальних мовах.

Приклади з IT:

При фільтрації логів за IP-адресами:

- U – множина всіх IP-адрес, що коли-небудь з'являлися в логах;
- A – IP-адреси, що спробували здійснити логін;
- B – IP-адреси, що отримали відмову.

Тоді $\neg B$ – IP-адреси з універсуму, які не викликали помилку доступу.

У системі керування доступом:

- U – всі користувачі;
- A – користувачі з роллю «адміністратор»;
- B – користувачі з обмеженим доступом;

$U \backslash A$ – всі, хто не є адміністраторами.

У машинному навчанні:

- U – всі зразки у датасеті;
- T – тренувальна вибірка;
- V – валідаційна вибірка;

$U(T \cup V)$ – тестова вибірка (або залишкові зразки).

Будь-яку множину розглядатимемо у зв'язку з універсумом, який на колах Ейлера асоціюватимемо з прямокутником на площині, всередині якого зображатимемо множини, як ми робили, починаючи з рис.1.

Нова операція $U - A = \overline{A}$ – (абсолютне доповнення A) – це множина, що містить усі елементи універсуму, за винятком елементів A (рис.5).

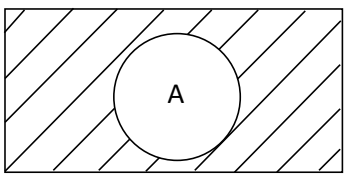


Рис.5

Приклад:

U – усі акаунти системи;

A – акаунти з двофакторною автентифікацією (2FA);

$\bar{A}$ – акаунти без 2FA, що є потенційно вразливими.

Приклад з IT:

all_users = {‘alice’, ‘bob’, ‘carol’, ‘dan’}

blocked_users = {‘carol’, ‘dan’}

safe_users = all_users - blocked_users # => {‘alice’, ‘bob’}

Це поняття має принципове значення для визначення доповнень, всіх можливих результатів, гарантованих перевірок тощо. У системному аналізі, безпеці, теорії баз даних і компіляції завжди слід чітко розуміти, в якому універсумі працює модель.

1.5. Підмножина. Рівність множин

Означення 5. *Множина A називається підмножиною множини B, якщо кожен елемент A є елементом B (рис. 6).*

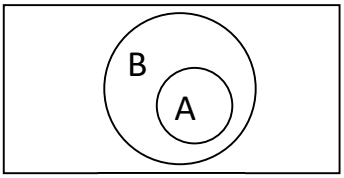


Рис.6

Для позначення цього факту вводиться знак «$\subset$» – символ включення (або «$\subseteq$»); іншими словами, $A \subset B \Leftrightarrow \forall\, x\, (x \in A \Rightarrow x \in B)$.

Якщо необхідно підкреслити, що множина A містить також інші елементи, крім елементів множини B, то використовують символ строгого включення: $B \subset A$. Зв’язок між символами $\subset$ і $\subseteq$ задається виразом $B \subset A \Leftrightarrow B \subseteq A \wedge B \neq A$ Загалом будемо використовувати символ «$\subset$».

Говорять, що множина B є істинною (або власною, від слова «власне») підмножиною A, якщо $B \subset A$ і $B \neq A$ на відміну від неістинних (або невласних) підмножин $\varnothing$ та U будь-якої множини A.

Приклади:

A = {‘admin’, ‘editor’}, B = {‘admin’, ‘editor’, ‘viewer’}

$\Rightarrow A \subset B$ – набір ролей A є строгою підмножиною усіх можливих ролей.

A = {id for id in log if id.source == ‘internal’}

U = {id for id in log}

$\Rightarrow A \subseteq U$ – усі внутрішні запити є підмножиною всіх запитів у журналі подій.

У тестуванні:

T_passed – тести, які пройдені;

T_all – усі передбачені тести;

якщо T_passed $\subset$ T_all, то деякі тести ще не виконані.

Рівність множин

Означення. *Дві множини рівні, якщо вони складаються з одних і тих самих елементів:*

$$A = B \Leftrightarrow \forall\, x\, (x \in A \Leftrightarrow x \in B).$$

Наприклад, $\{a,b,c\} = \{b,c,a\}$.

Справджується таке твердження: $A \subset B \wedge B \subset A$ тоді і тільки тоді, коли $A = B$.

Можна довести таке твердження: включення множин транзитивне, тобто справджується така рівність:

якщо $A \subset B \wedge B \subset C$, то $A \subset C$.

Приклад:

Python-реалізація перевірки рівності множин
a = {‘read’, ‘write’}
b = {‘write’, ‘read’}
print(*a* == *b*) # True
У множинах порядок елементів не має значення, тому {‘*a*’, ‘*b*’} = {‘*b*’, ‘*a*’}.

Порожня множина як підмножина

Порожня множина ∅ є підмножиною будь-якої множини. Це логічно: у ній немає жодного елемента, тому автоматично виконується умова «всі елементи ∅ належать *A*».

Різниця між належністю та включенням

Треба бути уважним, щоб розрізняти елементи множини та підмножини цієї множини. Наприклад, коли пишуть $a \in \{a,b,c\}$ це означає, що елемент a є членом множини, що складається з трьох елементів: a, b і c. Коли ж пишуть $\{a\} \subset \{a,b,c\}$, це означає, що множина, що складається з одного елемента a, є підмножиною множини, яка складається з трьох елементів: a, b, c.

Якщо $a \in A$ – елемент a належить множині A;

$\{a\} \subseteq A$ – одноелементна множина $\{a\}$ є підмножиною A.

Це різні поняття – як у типізації: значення vs. контейнер.

IT-контекст:

Якщо user = ‘bob’, то:

‘bob’ ∈ admins означає, що користувач bob має роль адміністратора;

{bob} ⊆ all_users – множина, що містить bob, є підмножиною всіх користувачів.

1.6. Множина підмножин (степенева множина)

Означення 7. *Множину, елементами якої є всі підмножини множини A, називають булеаном або множиною підмножин (множиною-степенем) множини A і позначають* $P(A)$.

Приклад:

Для триелементної множини $A = \{a, b, c\}$ маємо

$$P(A) = \{\varnothing, \{a\}, \{b\}, \{c\}, \{a, b\}, \{a, c\}, \{b, c\}, \{a, b, c\}\}.$$

У разі кінцевої множини A, що складається з n елементів, множина підмножин $P(A)$ містить 2^n елементів. Слід підкреслити відмінності між відношенням належності ($\in$) і відношенням включення ($\subset$). Як уже зазначалося, множина A може бути своєю підмножиною ($A \subset A$), але вона не може входити до складу своїх елементів ($A \notin A$). Навіть у разі одноелементних підмножин потрібно відрізняти множину $A = \{a\}$ та її єдиний елемент a (дивись приклад). Відношення включення має властивість транзитивності, відношення належності цієї властивості не має. Тобто, із того, що $a \in A \wedge A \in B$ не витікає, що $a \in B$, як здається на перший погляд.

Приклади в IT:

Система прав доступу:

Нехай A = {‘read’, ‘write’, ‘execute’} – множина базових прав.

Тоді $P(A)$ – усі можливі набори доступу, які може мати користувач.

Наприклад:

- ∅ – жодних прав;
- {‘read’} – лише читання;
- {‘read’, ‘execute’} – частковий доступ;
- A – повний доступ.

Вибір ознак у машинному навчанні:

Якщо $F = \{f_1, f_2, f_3\}$ – множина ознак, то $P(A)$ – усі можливі набори ознак, які можна використати для побудови моделей.

Це основа для методу перебору ознак (feature selection).

Фільтри в інтерфейсах:

Якщо доступні фільтри: {size, color, brand}, то $P(A)$ – це всі комбінації фільтрів, які може вибрати користувач.

Python-приклад:

```
from itertools import chain, combinations
def powerset(s):
```

```
    return list(chain.from_iterable(combinations(s, r) for r in range(len(s)+1)))
roles = {‘read’, ‘write’, ‘execute’}
print(powerset(roles))
```

Отже, множина всіх підмножин є фундаментальною конструкцією в IT, що використовується для:

- розробки тестових випадків,
- аналізу складності алгоритмів,
- моделювання конфігурацій систем,
- розв’язання задач у стилі «усі можливі комбінації».

2 АЛГЕБРА МНОЖИН

2.1. Закони алгебри множин (з прикладами з ІТ)

Алгебра множин — це система правил, що описує, як множини поводяться при основних операціях: об'єднання, перетин, доповнення.

Ці закони аналогічні законам булевої алгебри, яка є основою цифрової логіки, роботи комп'ютерів, пошукових систем, компіляторів тощо.

Основні властивості або закони алгебри множин наведені нижче.

1. Комутативні закони

а) $A \cup B = B \cup A$; б) $A \cap B = B \cap A$.

2. Асоціативні закони

а) $A \cup (B \cup C) = (A \cup B) \cup C$; б) $A \cap (B \cap C) = (A \cap B) \cap C$.

3. Дистрибутивні закони

а) $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$; б) $A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$.

Властивості $\varnothing$ та U

4 а) $A \cup \varnothing = A$; 4 б) $A \cap U = A$.

5 а) $A \cup \overline{A} = U$; 5 б) $A \cap \overline{A} = \overline{\varnothing}$.

6 а) $A \cup U = U$; 6 б) $A \cap \varnothing = \varnothing$.

7 а) $\overline{\varnothing} = U$; 7 б) $\overline{U} = \varnothing$.

Закон ідемпотентності (самопоглинання)

8 а) $A \cup A = A$; 8 б) $A \cap A = A$.

Закон поглинання

9 а) $A \cup (A \cap B) = A$; 9 б) $A \cap (A \cup B) = A$.

Теорема де Моргана

10 а) $\overline{A \cup B} = \overline{A} \cap \overline{B}$; 10 б) $\overline{A \cap B} = \overline{A} \cup \overline{B}$.

Властивості доповнення, різниці та рівності

11) $\overline{\overline{A}} = A$;

12) $A - B = A \cap \overline{B}$;

13) $A \oplus B = (A \cap \overline{B}) \cup (\overline{A} \cap B)$;

14) $A \oplus B = B \oplus A$;

15) $(A \oplus B) \oplus C = A \oplus (B \oplus C)$;

16) $A \oplus \varnothing = \varnothing \oplus A = A$.

Можна довести, що:

1) Якщо $A \cup B = U \wedge A \cap B = \varnothing$, то $B = \overline{A}$;

2) $A \subset B$ тоді і тільки тоді $A \cap B = A$ тоді і тільки тоді $A \cup B = B$ тоді і тільки тоді $A \cap \overline{B} = \varnothing$;

3) $A = B \Leftrightarrow (A \cap \overline{B}) \cup (\overline{A} \cap B) = \varnothing$.

Використовуються в логічних умовах, оптимізації запитів, побудові схем.

Python-приклад:

```
admins = {'alice', 'bob'}
blocked = {'bob', 'carol'}
```

safe = admins - blocked # A \ B

Приклад з пошуку в базі даних:
SELECT * FROM users
WHERE NOT (status = 'active' OR role = 'admin');
Це еквівалентно:
WHERE status != 'active' AND role != 'admin'
(аналог закону де Моргана)

2.2 Методи доведення тотожностей алгебри множин

Основний метод доведення тотожності в алгебрі множин ґрунтується на застосуванні означення рівності множин і твердження про рівність множин. Такий зручний інструмент, як кола Ейлера може бути використаний для доведення тотожностей алгебри множин тільки після певної формалізації. Ми її не робитимемо, тому використовуватимемо кола Ейлера як ілюстративний інструмент.

Доведемо, наприклад, тотожність 3а) $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$, використовуючи твердження про рівність множин. Але спочатку проілюструємо її справжність за допомогою кіл Ейлера. Для цього треба представити на колах Ейлера множину з лівої частини тотожності $A \cup (B \cap C)$ і множину з правої частини тотожності $(A \cup B) \cap (A \cup C)$ і упевнитися в їхній рівності.

Для зображення лівої частини тотожності спочатку закреслюємо штрихом одного напрямку $B \cap C$, потім штрихом іншого напрямку закреслюємо $A \cup (B \cap C)$. Результатом є область, що закреслена (рис.7).

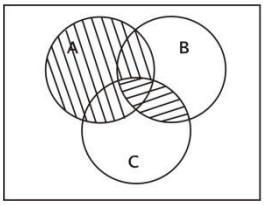


Рис.7

Для зображення правої частини тотожності спочатку закреслимо штрихом одного напрямку область $A \cup B$, штрихом іншого напрямку область $A \cup C$. Результатом $(A \cup B) \cap (A \cup C)$ буде область подвійного закреслення (рис.8).

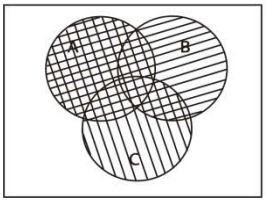


Рис.8

Очевидно, що обидві області співпадають (збігаються).

Приклад.

Спочатку доведемо тотожність 3а) $A \cup (B \cap C) = (A \cup B) \cap \ \cap (A \cup C)$, використовуючи твердження про рівність множин.

Доведемо, що $A \cup (B \cap C) \subset (A \cup B) \cap (A \cup C)$. Для цього візьмемо будь-яке $x \in A \cup (B \cap C)$, тоді за означенням операцій «$\cup$» і «$\cap$», $x \in A \vee$ $\vee (x \in B \wedge x \in C)$. За законом дистрибутивності диз'юнкції відносно кон'юнкції маємо, що $(x \in A \vee x \in B) \wedge (x \in B \vee x \in C)$, тобто $x \in (A \cup B) \wedge x \in (B \cup C)$ або $x \in (A \cup B) \cap (B \cup C)$, що і було потрібно довести.

Доведемо, що $(A \cup B) \cap (B \cup C) \subset A \cup (B \cap C)$. Для цього візьмемо будь-яке $x \in (A \cup B) \cap (B \cup C)$. Звідси $(x \in A \vee x \in B) \wedge (x \in B \vee x \in C)$ або $x \in A \vee$ $\vee (x \in B \wedge x \in C)$, тобто $x \in A \cup (B \cap C)$, що і було потрібно довести.

Згідно з твердженням про рівність множин 3а) $A \cup (B \cap C) = (A \cup B) \cap$ $\cap (A \cup C)$.

Доведемо, що $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$ за означенням рівності множин. Для цього знову візьмемо будь-яке $x \in A \cup (B \cap C)$, тоді за означенням операцій «$\cup$» і «$\cap$», це еквівалентне тому, що $x \in A \vee (x \in B \wedge x \in C)$. Це в свою чергу за законом дистрибутивності диз'юнкції відносно кон'юнкції еквівалентне тому, що $(x \in A \vee x \in B) \wedge (x \in B \vee x \in C)$, тобто $x \in A \cup B \wedge x \in B \cup C$ або $x \in (A \cup B) \cap (B \cup C)$. Таким чином ми довели, що

$$\forall x\,(x \in A \cup (B \cap C)) \Leftrightarrow x \in A \vee (x \in B \wedge x \in C) \Leftrightarrow (x \in A \vee x \in B) \wedge (x \in A \wedge x \in C) \Leftrightarrow$$
$$\Leftrightarrow x \in A \cup B \vee x \in A \cup C \Leftrightarrow x \in (A \cup B) \cap (A \cup C),$$

тобто $\forall x\,(x \in A \cup (B \cap C)) \Leftrightarrow x \in (A \cup B) \cap (A \cup C)$.

За означенням рівності двох множин це означає, що $A \cup (B \cap C) = (A \cup B) \cap$ $\cap (A \cup C)$.

Доведемо ту саму тотожність алгебраїчним способом. Нагадаємо, що при цьому ми маємо використовувати основні властивості (тотожності) алгебри множин причому, якщо ми доводитимемо деяку тотожність, то всі інші вважатимемо доведеними.

Адже, треба довести $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$

Почнемо з правої частини

$$(A \cup B) \cap (A \cup C) \overset{3б)}{=} ((A \cup B) \cap A) \cup ((A \cup B) \cap C) \overset{1б)}{=} (A \cap (A \cup B)) \cup (C \cap (A \cup B)) \overset{3б)}{=}$$
$$\overset{3б)}{=} ((A \cap A) \cup (A \cap B)) \cup ((C \cap A) \cup (C \cap B)) \overset{2а)}{=} (A \cap A) \cup (A \cap B) \cup (C \cap A) \cup (C \cap B) \overset{8б)}{=}$$
$$\overset{8б)}{=} A \cup (A \cap B) \cup (C \cap A) \cup (C \cap B) \overset{9а)1б)}{=} A \cup (A \cap C) \cup (C \cap B) \overset{9а)}{=} A \cup (C \cap B), \text{ і т.д.}$$

ми отримали ліву частину. Тотожність доведено.

Будь-яка теорема алгебри множин виводиться з тотожностей 2а) і 2б),тобто методом алгебраїчних перетворень.

IT-контекст:

SQL-запити: оптимізація умов за допомогою законів:

-- Замість:

WHERE status = 'active' AND (role = 'admin' OR role = 'editor')

-- Можна:

WHERE (status = 'active' AND role = 'admin') OR (status = 'active' AND role = 'editor')

(закон дистрибутивності)

Пошукові запити: запит A AND (B OR C) часто розгортається в пошукових системах як (A AND B) OR (A AND C) — це дає змогу точніше формувати результати.

2.3. Узагальнення операцій над множинами, розбиття множини, декартів добуток множин

Із властивостей комутативності й асоціативності операцій об'єднання випливає, що об'єднання кількох множин можна виконати, послідовно об'єднуючи їх, причому порядок входження множин не впливає на результат, наприклад $A \cup B \cup C = (A \cup B) \cup C = A \cup (B \cup C) = (B \cup C) \cup A$. Отже, об'єднання сукупності множин можна подати співвідношенням

$$A_1 \cup A_2 \cup \ldots \cup A_n = \bigcup_{i=1}^{n} A_i .$$

Аналогічно на n множин узагальнюється операція перерізу:

$$A_1 \cap A_2 \cap \ldots \cap A_n = \bigcap_{i=1}^{n} A_i .$$

Використовуючи узагальнення операцій об'єднання та перерізу на n множин, можна узагальнити також інші співвідношення, наприклад закон де Моргана, який в узагальненому вигляді має вигляд

$$\overline{\bigcup_{i=1}^{n} A_i} = \bigcap_{i=1}^{n} \overline{A}_i \text{ і } \overline{\bigcap_{i=1}^{n} A_i} = \bigcup_{i=1}^{n} \overline{A}_i .$$

Означення. *Сукупність множин* $A_1, A_2, \ldots, A_n$ *називається розбиттям множини A, якщо об'єднання всіх цих множин співпадає з множиною A, тобто*

$$1. \bigcup_{i=1}^{n} A_i = A$$

переріз будь-яких двох різних множин A_i *і* A_j *є порожньою множиною, тобто*

$$2. \ A_i \cap A_j = \varnothing \ \forall i \neq j$$

Приклад:

A_1, A_2, A_3 — користувачі трьох систем автентифікації.

$\bigcup A_i$ — всі користувачі, які мають доступ до принаймні однієї з систем.
$\bigcap A_i$ — ті, хто має доступ до всіх трьох.

Групи користувачів у системі:
U — усі користувачі;
A_1, A_2, A_3 — окремі ролі: адмін, редактор, гість;
Кожен користувач має лише одну роль ⇒ це розбиття.

Декартів добуток множин

Означення. *Прямим (або декартовим) добутком множин A і B називається множина всіх упорядкованих пар елементів (a,b), з яких перший належить множині A, а другий – множині B (позначається $A\times B$):*

$$A\times B=\{(x,y)\mid x\in A\wedge y\in B\}$$

Порядок входження пар може бути будь-яким, але розташування елементів у кожній парі визначається порядком множин, що перемножуються. Тому $A\times B\neq B\times A$, тобто прямий добуток властивості комутативності не має.

Приклади з IT:

Реляційна база даних:
Users × Roles — всі можливі призначення ролей користувачам;
Наприклад в базах даних:
SELECT * FROM Users CROSS JOIN Roles;
Комбінації параметрів у тестуванні:
Browsers × Devices — тести всіх комбінацій браузера і пристрою.
У просторі станів:
A = {logged_in, guest}, B = {dark_theme, light_theme}
⇒ $A \times B$ — всі варіанти режимів інтерфейсу.

В загальному вигляді для двох множин A і B виконується $A\times B\neq B\times A$.

Операція прямого добутку множин узагальнюється на будь-яку їх кількість і записується у вигляді

$$\prod_{i=1}^{n} A_i = A_1\times A_2\times\ldots\times A_n$$

причому елементом прямого добутку n множин є впорядкована послідовність із n елементів $(a_1,a_2,\ldots,a_n)$, яка називається ще кортежем або вектором завдовжки n, а також впорядкованою n-кою.

Властивості асоціативності для прямого добутку також не виконуються, але виконується властивість дистрибутивності відносно об'єднання, перерізу і відносного доповнення (різниці):

$$(A_1\cup A_2)\times B=(A_1\times B)\cup(A_2\times B);$$

$$(A_1 \cap A_2) \times B = (A_1 \times B) \cap (A_2 \times B);$$
$$(A_1 - A_2) \times B = (A_1 \times B) - (A_2 \times B).$$

Якщо як співмножник декартового добутку n-множин використовується одна множина A, то це записується так:

$$A \times A \times A \times \ldots \times A = A^n.$$

Операція декартового добутку відрізняється від операцій, введених раніше, тим, що елементи добутку множин суттєво відрізняються від елементів співмножників і є об'єктами іншої природи. Наприклад, якщо R – множина дійсних чисел, то декартовий добуток $R \times R$ – множина всіх точок площини.

Python-приклад:

Редагувати

```
from itertools import product
users = ['alice', 'bob']
roles = ['admin', 'editor']
print(list(product(users, roles)))
```

Усі ці операції активно використовуються:

- у системах доступу (RBAC),
- у базах даних (SQL JOINs),
- у множинній фільтрації,
- у складних пошукових системах,
- у моделюванні конфігурацій та станів.

2.4. Кола Ейлера як інструмент графічного зображення в теорії множин

Кола Ейлера (або діаграми Венна) — це наочний спосіб зображення множин та операцій над ними. Множини представляються замкненими областями (найчастіше — колами), а універсум — прямокутником.

Навіщо це потрібно в IT:

- для наочного аналізу умов в складних логічних виразах (як у фільтрах, запитах SQL, системах доступу);
- у візуальному програмуванні, наприклад при створенні блок-схем або обробників подій;
- у data science, щоб показати взаємоперетин категорій або фільтрів;
- у аналізі тестових сценаріїв, щоб не упустити жодну комбінацію.

Розглянемо інструмент ілюстрації в теорії множин, кола Ейлера, для графічного зображення всіх операцій перерізу з трьома множинами A, B, C (рис.9).

1) $A \cap B \cap C$ 2) $A \cap B \cap \overline{C}$ 3) $A \cap \overline{B} \cap C$ 4) $\overline{A} \cap B \cap C$
5) $\overline{A} \cap B \cap \overline{C}$ 6) $\overline{A} \cap \overline{B} \cap C$ 7) $A \cap \overline{B} \cap \overline{C}$ 8) $\overline{A} \cap \overline{B} \cap \overline{C}$

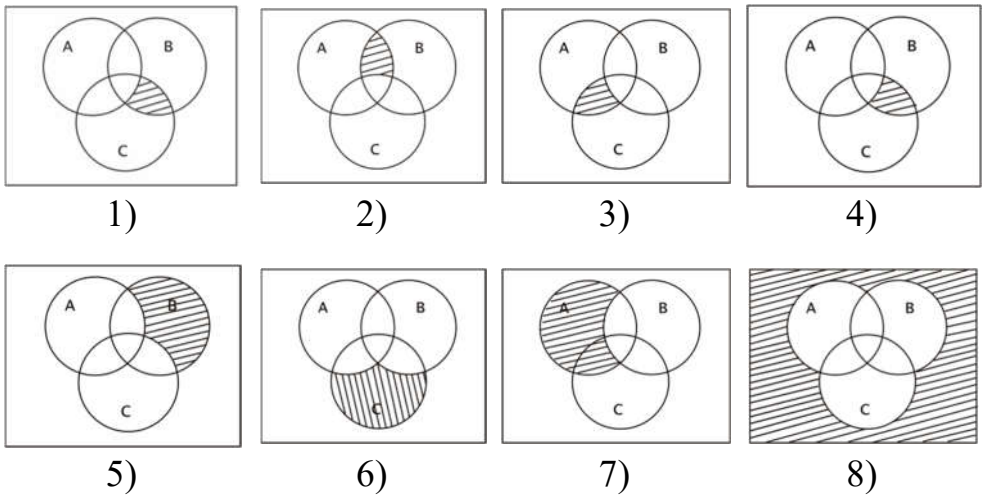


Рис.9.

Тут зображені всі варіанти розбиття площини універсуму U, якщо в операціях задіяні три множини. Ці варіанти можна використовувати для графічного зображення будь яких операцій з трьома множинами. Їхній алгебраїчний вираз можна використовувати для алгебраїчного запису результату операції з множинами. Наведемо приклади.

Приклад. На колах Ейлера зображено результат операцій з множинами. Виходячи з зображення навести результат в алгебраїчному вигляді.

1. $A \cap \overline{B} \cup C$

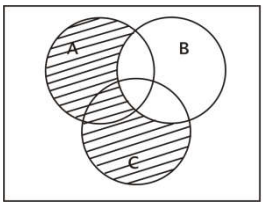


Рис.10.

$$A \cap \overline{B} \cup C = (A \cap \overline{B} \cap \overline{C}) \cup (A \cap \overline{B} \cap C).$$

Ми використали поняття розбиття універсуму. Але для алгебраїчного вигляду можна знайти і інші варіанти. Наприклад,

$$A \cap \overline{B} \cup C = (A \cup C) - B = (A \cup C \cup B) - B = (A \cup C) \cup \overline{B} \text{ і т.д.}$$

2. $(A \cap B) \cup (A \cap C) \cup (B \cap C)$

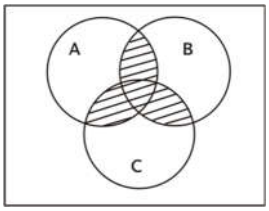


Рис.11.

$$(A \cap B) \cup (A \cap C) \cup (B \cap C) = (A \cap B \cap \overline{C}) \cup (A \cap \overline{B} \cap C) \cup (\overline{A} \cap B \cap C) \cup$$
$$(A \cap B \cap C) = C \cap (A \cup B) \cup (A \cap B) = B \cap (A \cup C) \cup (A \cap C) \text{ і т.д.}$$

3. $(A \cap B) \cup (A \cap C)$

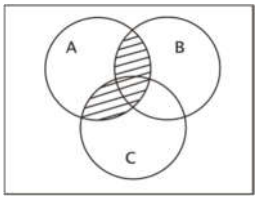


Рис.12.

$$(A \cap B) \cup (A \cap C) = (A \cap B \cap \overline{C}) \cup (A \cap \overline{B} \cap C) \cup (A \cap B \cap C) =$$

$$= A \cap (B \cup C) = (A \cap B) \oplus (A \cap C) \cup (A \cap B \cap C) \text{ і т.д.}$$

Як ви вже мабуть побачили, що аналогічно формулам алгебри логіки формули алгебри множин теж мають кілька різних варіантів. Ці варіанти відрізняються операціями і кількістю множин, що входять у формулу. Причому вираз, в який входять об'єднання перерізів всіх множин не тільки не є самим коротким, а навпаки є найдовшим.

Контрольні запитання

1. Що таке множина? Наведіть приклади множин, які виникають в IT-системах.
2. Що називають елементом множини? Як перевірити, чи належить об’єкт множині?
3. Назвіть основні способи подання множин. У яких випадках зручно використовувати кожен із них?
4. Як задається множина за допомогою предиката? Наведіть приклад із мов програмування.
5. Що таке скінченна і нескінченна множини? Наведіть приклади з комп’ютерного середовища.
6. Що таке порожня множина? Як її інтерпретувати у програмуванні?
7. Чому порожня множина вважається підмножиною будь-якої множини?
8. Наведіть приклади ситуацій у програмуванні, де результатом є порожня множина.
9. Поясніть парадокс Рассела. Який висновок щодо безпеки визначень множин?
10. Які основні операції визначені над множинами?
11. Наведіть приклади використання операцій над множинами у SQL-запитах або фільтрах.
12. Що таке універсум? Як його задають у прикладній задачі?
13. Наведіть приклади універсуму в задачах з баз даних, авторизації, обробки логів.
14. Як виглядає операція абсолютного доповнення множини? Наведіть IT-приклад.
15. Що означає відношення підмножини? Чим воно відрізняється від належності?
16. Як відрізнити запис $a \in A$ від $\{a\} \subseteq A$? Наведіть приклад.
17. Як перевірити рівність двох множин? Які методи для цього використовують?
18. Що таке істинна (строга) підмножина?
19. Що таке степенева множина (множина підмножин)? Наведіть приклад у сфері доступу або прав.
20. Скільки підмножин має множина з n елементів? Як це використовується в алгоритмах перебору?
21. Що таке алгебра множин? Які операції входять до її основи?
22. Сформулюйте комутативні, асоціативні та дистрибутивні закони алгебри множин.
23. У чому полягає відмінність між законом поглинання і законом ідемпотентності?

24. Наведіть приклад використання законів де Моргана у логічних умовах програми або SQL.
25. Як алгебра множин співвідноситься з булевою алгеброю?
26. Як довести тотожність множин: через означення рівності чи алгебраїчно?
27. У яких випадках зручно використовувати кола Ейлера для візуалізації?
28. Поясніть, як виглядає симетрична різниця множин у термінах користувачів системи.
29. Як виглядає дистрибутивність у фільтрації множин користувачів?
30. Який практичний сенс має використання узагальнених операцій над множинами?
31. Що таке розбиття множини? Наведіть приклади з реляційної бази даних.
32. Які умови має виконувати сукупність множин, щоб бути розбиттям?
33. Що таке декартів добуток множин? Які приклади існують у програмному тестуванні?
34. Чому декартів добуток не є комутативним? У чому практична різниця?
35. Як виглядає декартовий добуток у вигляді таблиці БД або JSON-масиву пар?

3. БІНАРНІ ВІДНОШЕННЯ

Бінарні відношення — це фундаментальна концепція в дискретній математиці, яка має широке застосування у комп'ютерних науках, зокрема в таких галузях, як реляційні бази даних, структури даних, побудова графів, алгоритми, моделювання систем, логіка предикатів тощо.

3.1. Основні поняття та властивості відношень

Відношення — це правило або властивість, яка встановлює зв'язок між об'єктами. Якщо цей зв'язок описує пари об'єктів, говорять про бінарне (двомісне) відношення.

Означення. *Бінарним відношенням A, що діє з множини X у множину Y, називається деяка підмножина декартова добутку множин X і Y $X \times Y$, тобто $A \subset X \times Y$.*

Бінарне відношення повністю визначається множиною всіх пар елементів (x, y), для яких воно справджується. Тому будь-яке бінарне відношення можна розглядати як множину впорядкованих пар (x, y).

Приклади бінарних відношень:

1. входити до складу деякого колективу;
2. відношення належності;
3. включення множин;
4. працювати в одній фірмі;
5. нерівність чисел;
6. бути братом;
7. ділитися на яке-небудь натуральне число.

Для наведених відношень запишемо відповідні їм співвідношення:

1. Розов – староста групи 1ЕК;
2. ·Коцар і Степаненко вчаться в одній групі;
3. · $A \subset B$;
4. Резнік і Тимошенко працюють в одній фірмі;
5. $5 > 2$;
6. Богдан брат Віктора;
7. 6 ділиться на 3.

Приклади з ІТ:

Відношення	Опис у прикладній сфері
(user,role)∈R(user, role) ∈ R	Користувач має певну роль у системі
(ip,service)∈R(ip, service) ∈ R	IP-адреса зверталась до певного сервісу
(файл,користувач)∈R(файл, користувач) ∈ R	Користувач має доступ до файлу
(таблиця,поле)∈R(таблиця, поле) ∈ R	Поле належить до таблиці бази даних
(процес,ресурс)∈R(процес, ресурс) ∈ R	Процес використовує ресурс

Бінарне відношення встановлює відповідність елементів множини X елементам множини Y. Зрозуміло, що співвідношення xAy можна записати у вигляді $(x, y) \in A$, де $A \subset X \times Y$. Наприклад, Чуйко займає посаду старшого менеджера еквівалентно (Чуйко, старший менеджер) $\in$ „займати посаду”, $5 > 2$ еквівалентно $(5,2) \in$ «>» (але не можна записати $(5,2) \in$ «<»). Елемент x називають першою координатою, а елемент y – другою координатою впорядкованої пари (x, y). Множина перших координат називається областю визначення (лівою областю) відношення A і позначається $D_0(A)$.

Множина других координат називається областю значень (правою областю) відношення A і позначається $D_3(A)$. Якщо $x \in X$, $y \in Y$, тоді $D_0(A) \subset X$, $D_3(A) \subset Y$. У таких випадках кажуть, що A є відношенням від X до Y, що воно ставить у відповідність елементам множини X елементи множини Y, тому таке відношення (див. рис.1) називають також відповідністю та позначають $X \to Y$. Якщо $Y = X$, то будь-яке відношення $A\colon\ X \to Y$ є підмножиною $X \times X$ і називається відношенням в X. Тобто в цьому випадку відношення A ставить у відповідність елементу множини X елемент множини X (рис.2).

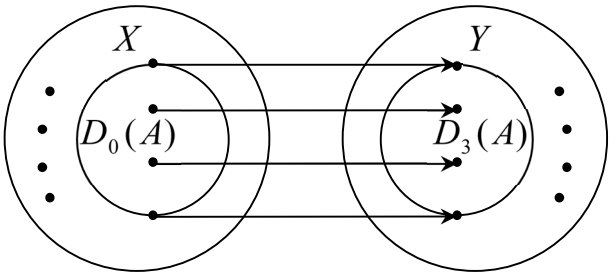


Рис.1

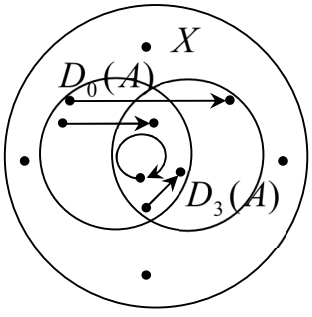


Рис.2

Якщо $D_0(A) = X$, то кажуть, що відношення A задано на X.

Приклад. Надано дві множини $X = \{2, 3\}$, $Y = \{3, 4, 5, 6\}$.

Нехай A – відношення «бути дільником» від X до Y. Тоді $A = \{(2, 4), (2, 6), (3, 3), (3, 6)\}$. Відношення B – « = » від X до Y: $B = \{(3, 3)\}$; відношення C

– «>» від X до Y: $C = \varnothing$. $D_0(A) = \{2, 3\} = X$, $D_3(A) = \{3, 4, 6\} \subset Y$, $D_0(B) = \{3\}$, $D_3(B) = \{3\}$, $D_0(C) = \varnothing$, $D_3(C) = \varnothing$.

Відокремлюють такі випадки відношень в X:

1. **Повне** (універсальне) відношення $P = X \times X$, яке справджується для будь-якої пари (x_i, x_j) елементів з X, тобто $x_i \; P \; x_j \; \forall \; x_i, x_j \in X$. Наприклад, P – відношення «вчитися в одній групі» у множині X, де X – множина студентів групи 1ЕК.

2. Тотожне (діагональне) відношення E, що виконується тільки між елементом і ним самим. Наприклад, рівність на множині дійсних чисел.

3. Порожнє відношення, якому не задовольняє жодна пара елементів з X, тобто $x_i \; \overline{A} \; x_j \; \forall \; x_i, x_j \in X$. Наприклад, A – відношення «бути братом» у множині X, де X – множина жінок.

Повне і порожнє відношення можна визначити і для випадку відношень від X до Y. Повним відношенням буде $D = X \times Y$, яке справджується для будь якої пари (x, y), де $x \in X$, $y \in Y$.

Порожнім відношенням є відношення, якому не задовольняє жодна пара елементів (x, y), де $x \in X$, $y \in Y$.

Означення. *Розглянемо відношення $A \subset X \times Y$. Нехай елемент $x \in X$. Перерізом відношення A за елементом x називається множина елементів y з Y, для яких пара $(x, y) \in A$.*

Множину всіх перерізів відношення A називають фактором-множиною множини Y за відношенням A і позначають Y / A. Вона повністю визначає відношення A.

Приклад. $X = \{x_1, x_2, x_3, x_4, x_5\}$, $Y = \{y_1, y_2, y_3, y_4\}$. Відношення $A = \{(x_1, y_1), (x_2, y_1), (x_2, y_2), (x_2, y_4), (x_3, y_3), (x_3, y_4), (x_4, y_4), (x_5, y_2), (x_5, y_3), (x_5, y_4)\}$.

Випишемо перерізи за всіма елементами множини X у такому вигляді:

Таблиця 3.1

x_1	x_2	x_3	x_4	x_5
$\{y_1\}$	$\{y_1, y_2, y_4\}$	$\{y_3, y_4\}$	y_4	$\{y_2, y_3, y_4\}$

Об'єднання множин другого рядка утворять фактор-множину Y / A.

Типові приклади з комп'ютерної практики:

Бінарне відношення в реляційній БД — будь-який запис у таблиці типу user_id | permission_id — це пара елементів (a,b) із множин Users × Permissions.

Граф доступу — множина пар (user, resource) відображає, хто до чого має доступ.

Логіка контролю помилок — пари (код помилки, компонент) вказують на взаємозв'язок.

3.2. Подання бінарних відношень за допомогою матриці та графа.

З означення бінарного відношення зрозуміло, що воно, як будь-яка множина, може бути подане вербальним способом, переліком, предикатним або аналітичним способом. Але є способи подання бінарних відношень, властиві тільки для них. З попереднього зрозуміло, що відношення може бути подане за допомогою фактора-множини. Розглянемо ще два способи подання скінченого бінарного відношення: за допомогою матриці й графа.

Матричний спосіб ґрунтується на поданні відношення $A \subset X \times Y$ відповідною йому прямокутною таблицею (матрицею), що складається з нулів та одиниць, де стовпці – перші координати, а рядки – другі, причому на перетині i -го стовпця і j -го рядка буде 1, якщо виконується співвідношення $x_i \; A \, x_j$, або 0 – якщо воно не виконується.

Приклад.

Для наведеного у попередньому прикладі відношення матриця має такий вигляд:

Таблиця 3.2

	x_1	x_2	x_3	x_4	x_5
y_1	1	1	0	0	0
y_2	0	1	0	0	1
y_3	0	0	1	0	1
y_4	0	1	1	1	1

Розглянемо матриці для окремих випадків відношень в X. Матриця повного відношення – це квадратна матриця, що складається лише з одиниць; матриця тотожного (діагонального) відношення – це квадратна матриця, яка складається з нулів та одиниць на головній діагоналі; матриця порожнього відношення – це квадратна матриця, що складається лише з нулів.

Приклад (з ІТ):

Нехай: $A = \{u1,u2,u3\}$ — користувачі, $B = \{r1,r2\}$ — ролі. Відношення доступу $R \subseteq A \times B$

$$R = \{(u1,r1),(u2,r2),(u3,r2)\}$$

Тоді матриця виглядає так:

	$r1$	$r2$
$u1$	1	0
$u2$	0	1
$u3$	0	1

У програмуванні це часто реалізується як бітова матриця (наприклад, у представленні суміжності в графах) або логічна таблиця (DataFrame, 2D array, adjacency matrix).

Бінарне відношення можна також зображати за допомогою орієнтованого графа. Його вершини відповідають елементам множин X та Y, а дуга, спрямована від вершини x_i до y_j, означає, що співвідношення $x_i \; A \, y_j$ виконується.

Приклад.

Граф відношення, наведеного у прикладі 3.5, має вигляд, показаний на рис.3.

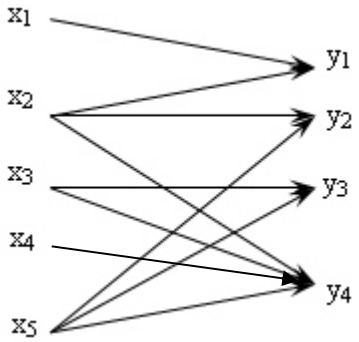


Рис.3

Граф бінарного відношення від X до Y – це дводольний граф. Дводольний граф – це такий граф, множина вершин якого розбита на дві підмножини (долі) таким чином, що ребра з'єднують тільки вершини різних долей (рис.3). Відношення в X зображується графом із вершинами, що відповідають елементам цієї множини. Якщо $x_i \; A \, x_i$ й $x_i \; A \, x_j$, то вершини зв'язуються двома протилежно спрямованими дугами, які умовно можна замінювати однією неспрямованою дугою (ребром). Співвідношенню $x_i \; A \, x_j$ відповідає петля.

<u>Приклад 7</u>. Бінарне відношення

$A = \{(x_1, x_2), (x_1, x_3), (x_1, x_4), (x_1, x_5), (x_2, x_3), (x_2, x_4), (x_3, x_4), (x_4, x_4), (x_4, x_5)\}$

подається графом, зображеним на рис.3.4.

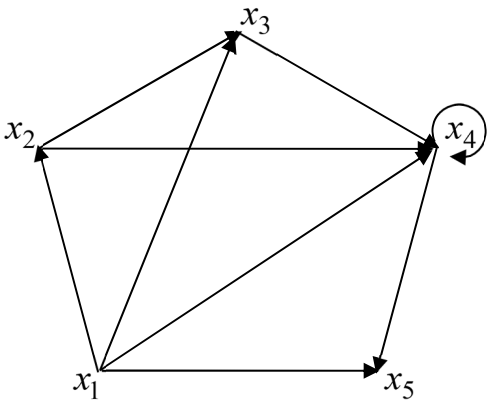


Рис.4

Граф повного відношення $P = X \times X$, де $X = \{x_1, x_2, x_3, x_4\}$, показано на рис.5; граф діагонального відношення – на рис.6, а граф порожнього відношення – на рис.7.

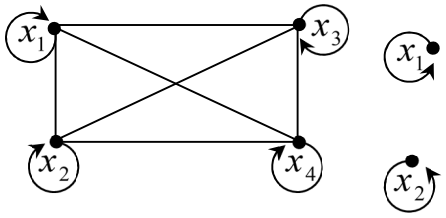


Рис.5

Рис.6

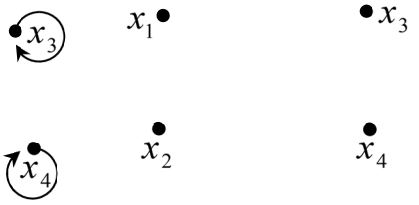


Рис.7

Зазначимо, що оскільки відношення — це множина, над ним можуть виконуватися всі теоретико-множинні операції, тобто виконуються операції об’єднання, перерізу, різниці, диз’юнктивної суми.

1. Об’єднання відношень A і B:

$$A \cup B = \{(x, y) \mid (x, y) \in A \vee (x, y) \in B\}.$$

2. Переріз відношень A і B:

$$A \cap B = \{(x, y) \mid (x, y) \in A \wedge (x, y) \in B\}.$$

3. Різниця відношень A і B:

$$A - B = \{(x, y) \mid (x, y) \in A \wedge (x, y) \notin B\}.$$

4. Диз’юнктивна сума відношень A і B:

$$A \oplus B = \{(x, y) \mid (x, y) \in A \oplus (x, y) \in B\}.$$

5. Доповнення відношення A:

$$\overline{A} = \{(x, y) \mid (x, y) \notin A\}.$$

Крім того, виділяються специфічні для відношень операції: обернення (симетризація) і композиція.

Приклад. Нехай відношення «бути матір’ю дочки» визначене на деякій множині жінок $Q_1 = \{$(Оксана, Наталя), (Олена, Надія), (Олена, Ольга),

(Катерина, Галина)}. Відношення Q_2 «бути матір'ю сина» задано парами Q2 = {(Оксана, Олег), (Олена, Андрій), (Катерина, Сергій)}.

Відношення $R_1 = Q_1 \cup Q_2$ є відношенням «бути матір'ю», пари цього відношення визначають всіх матерів, які мають дочок або синів.

Відношення $R_2 = Q_1 \cap Q_2 = \varnothing$, тобто не є змістовним.

Відношення $R_3 = Q_1 - Q_2 = Q_1$, $R_4 = Q_2 - Q_1 = Q_2$.

Відношення $R_5 = Q_1 \oplus Q_2 = Q_1 \cup Q_2$.

Означення. *Відношення, симетричне (обернене) деякому відношенню* $A \subset X \times Y$, *є підмножиною множини* $Y \times X$, *утвореною тими парами* $(y, x) \in X \times Y$, для яких $(x, y) \in A$. *Відношення симетричне до* A *позначається* A^{-1}.

Із попереднього зрозуміло, що перехід від A до A^{-1} здійснюється взаємною перестановкою координат кожної впорядкованої пари. Наприклад, відношення A – «x дільник y» має обернене до нього A^{-1} – «y кратне x». Для наведеного вище у прикладі 3 відношення A обернене відношення $A^{-1} = \{(4, 2), (6, 2), (3, 3), (6, 3)\}$.

Відношення, обернене відношенню Q_1 – це відношення $Q_1^{-1} = \{$(Наталя, Оксана), (Надія, Олена), (Ольга, Олена), (Галина, Катерина)$\}$ – «бути дочкою».

Відношення, обернене відношенню Q_2 – це відношення «бути сином», яке складається з пар $Q_2^{-1} = \{$(Олег, Оксана), (Андрій, Олена), (Сергій, Катерина)$\}$.

Слід зазначити, що при переході від A до A^{-1} область визначення стає областю значень і навпаки. Матриця A^{-1} – це транспонована матриця відношення A. Граф оберненого відношення A^{-1} утворюється із графа відношення A заміною всіх дуг на протилежні.

Приклад (IT):

$R = \{(user, role)\}$

$R^{-1} = \{(role, user)\}$ — роль пов'язана з користувачем.

3.3. Композиція відношень.

Нехай надано три множини X, Y, Z та два відношення $A \subset X \times Y$, $B \subset Y \times Z$.

Означення 4. *Композицією відношень A і B є відношення C, що складається з усіх тих пар $(y,x) \in X \times Y$, для яких існує таке $y \in Y$, що $(x,y) \in A$ й $(y,z) \in B$. Будемо позначати композицію відношень символом* o*, а саме*

$$B \text{o} A = C = \{(x,z) \in X \times Z : \exists\, y \in Y((x,y) \in A \wedge (y,z) \in B)\}.$$

Схематично утворення композиції наведено на рис.8.

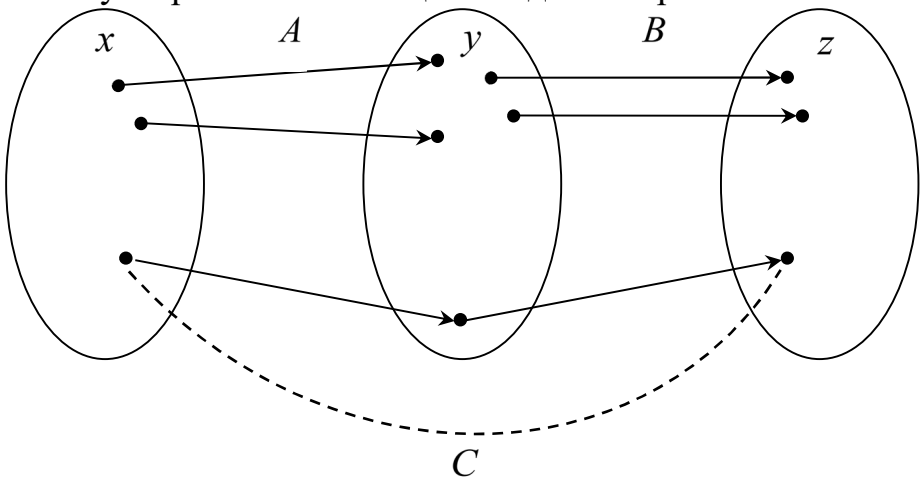


Рис.8

Можна легко показати, що переріз відношення C за x збігається з перерізом відношення B за підмножиною $A(x) \subset Y$, тобто $C(x) = B(A(x))$.

Приклад. Розглянемо композицію відношення A з прикладу 4, а саме $A = \{(x_1, y_1), (x_2, y_1), (x_2, y_2), (x_2, y_4), (x_3, y_3), (x_3, y_4), (x_4, y_4), (x_5, y_2), (x_5, y_3), (x_5, y_4)\}$ з відношенням $\text{B} = \{(\text{y}_1, \text{z}_2), (\text{y}_2, \text{z}_1), (\text{y}_2, \text{z}_2), (\text{y}_3, \text{z}_3), (\text{y}_4, \text{z}_3)\}$.

Це

$C = A \text{o} B =$

$= \{(x_1, z_2), (x_1, z_3), (x_2, z_2), (x_2, z_3), (x_3, z_1), (x_3, z_2), (x_3, z_3), (x_4, z_5), (x_5, z_1), (x_5, z_2),$

$(x_5, z_3)\}$. Візьмемо переріз відношення C за x_3: $C(x_3) = \{z_1, z_2, z_3\}$. З іншого боку, маємо $B(A(x_3)) = B(\{y_1, y_2, y_3\}) = \{z_2\} \cup \{z_1, z_2\} \cup \{z_3\} = \{z_1, z_2, z_3\}$.

Приклад. Розглянемо композицію відношення Q_1, з прикладу 8 яке діє з множини $X_1 = \{$Оксана, Олена, Катерина$\}$ в множину $X_2 = \{$(Наталя, Надія, Ольга, Галина)$\}$ з відношенням «бути матір'ю дочки» $Q_3 = \{$(Наталя, Зоряна), (Наталя, Сніжана), (Надія, Юлія), (Ольга, Богдана), (Ольга, Світлана), (Галина, Олена)$\}$, яке діє з множини X_2 в множину $X_3 = \{$(Зоряна, Сніжана, Юлія, Богдана, Світлана, Алла)$\}$. Це відношення $C_1 = Q_3 \text{o} Q_1 = \{$(Оксана, Зоряна), (Оксана, Сніжана), (Олена, Юлія), (Олена, Богдана), (Олена, Світлана), (Катерина, Алла)$\}$, яке є відношенням «бути бабусею онучки» і діє з множини X_1 в множину X_3.

Приклад. Розглянемо композицію відношення Q_1 з відношенням «бути матір'ю сина» $Q_4 = \{$(Надія, Олексій), (Галина, Богдан)$\}$, яке діє з множини X_2 в множину $X_4 = \{$Олексій, Богдан$\}$. Це відношення $C_2 = Q_4 \text{o} Q_1 = \{$(Олена, Олексій), (Катерина, Богдан)$\}$, яке є відношенням «бути бабусею онука». Візьмемо перерізи відношення C_1 за всіма елементами множини X_1:

C_1 (Оксана) = {Зоряна, Сніжана} – перелічені онучки Оксани;

C_2 (Олена) = {Юлія, Богдана} – онучки Олени;

C_3 (Катерина) = {Алла} – онучка Катерини.

Фактор-множина X_3 / C_1 – це множина {Зоряна, Сніжана, Юлія, Богдана, Алла} – множина онучок всіх жінок з множини X_1.

Візьмемо перерізи відношення C_2 за всіма елементами множини X_1:

C_2 (Оксана) = {Олексій};

C_2 (Олена) = {Богдан};

C_2 (Катерина) = $\varnothing$.

Фактор-множина X_3 / C_2 – це множина {Олексій, Богдан} – множина онуків всіх жінок з множини X_1.

Щодо властивостей композиції можна зазначити таке:

1. $B \text{o} A \neq A \text{o} B$, тобто не виконується закон комутативності;
2. $D\text{o}(B\text{o}A) = (D\text{o}B)\text{o}\text{A} = \text{DoBoA}$, тобто виконується закон асоціативності;
3. $(B\text{o}A)^{-1} = A^{-1}\text{o}\, B^{-1}$.

Перші дві властивості очевидні, третю проілюструємо на наведеному прикладі відношень Q_1 і Q_3.

Приклад. Відношення $(Q_1 \text{o} Q_3)^{-1} = C_1^{-1} = \{$(Зоряна, Оксана), (Сніжана, Оксана), (Юлія, Олена), (Богдана, Олена), (Світлана, Олена), (Алла, Катерина)$\}$, яке є відношенням «бути онучкою бабусі» діє з множини X_3 в множину X_1.

Відношення Q_1^{-1} ми вже знаходили. Це $Q_1^{-1} = \{$(Наталя, Оксана), (Надія, Олена), (Ольга, Олена), (Галина, Катерина)$\}$ і воно діє з множини X_2 в множину X_1.

Знайдемо обернене відношення до Q_3.

$Q_3^{-1} = \{$(Зоряна, Наталя), (Сніжана, Наталя), (Юлія, Надія), (Богдана, Ольга), (Світлана, Ольга), (Алла, Галина)$\}$ і це відношення діє з множини X_3 в множину X_2.

Знайдемо композицію відношень $Q_3^{-1} \text{o} Q_1^{-1} = \{$(Зоряна, Оксана), (Сніжана, Оксана), (Юлія, Олена), (Богдана, Олена), (Світлана, Олена), (Алла, Катерина)$\}$. Як ви бачите, відношення $(Q_1 \text{o} Q_3)^{-1}$ і відношення $Q_3^{-1} \text{o} Q_1^{-1}$ збігаються.

Приклад (IT):

R = {(user,group)}

S = {(group,permission)}

⇒ R∘S={(user,permission)} — результат **транзитивного успадкування прав** через групу.

3.4. Подання композиції відношень матрицями та графами.

Твердження. *Матриця композиції відношень* $C = B \circ A$ *утворюється як добуток матриць відношень* B *й* A *з подальшою заміною відмінних від нуля елементів одиницями.*

Справді, елемент c_{ik} матриці композиції знайдемо як суму добутків відповідних елементів матриць B та A (відповідно до правила множення матриць):

$$c_{ik} = b_{i1}a_{1k} + b_{i2}a_{2k} + \ldots + b_{in}a_{nk} = \sum_{j=1}^{n} b_{ij}a_{jk} .$$

Очевидно, така сума відмінна від нуля тоді й тільки тоді, коли хоча б один доданок відмінний від нуля, тобто дорівнює одиниці:

$$b_{ij}a_{jk} = 1 \Leftrightarrow b_{ij} = 1 \wedge a_{ij} = 1 \Leftrightarrow y_j Bz_i \wedge Ay_j \Leftrightarrow x_k B \circ Az_i .$$

Якщо у виразі c_{ik} не один, а кілька одиничних доданків, то кожен з них відповідає одному й тому самому співвідношенню $x_k \circ BAz_i$, через що їх сума має бути замінена одиницею.

Приклад. Проілюструємо за допомогою графів рівність

$$(Q_3 \circ Q_1)^{-1} = Q_3^{-1} \circ Q_1^{-1} .$$

Для цього спочатку побудуємо графи відношень Q_1 і Q_3 (рис.13). Виходячи з цих графів, побудуємо граф композиції $C_1 = Q_3 \circ Q_1$ (рис.14). Побудуємо граф оберненого відношення $C_1^{-1} = (Q_3 \circ Q_1)^{-1}$ (рис.15). Потім побудуємо графи відношень Q_1^{-1} і Q_3^{-1} (рис.16). Нарешті побудуємо граф композиції $Q_3^{-1} \circ Q_1^{-1}$ (рис.14).

Очевидно, що графи на рис. 15 і 17 збігаються.

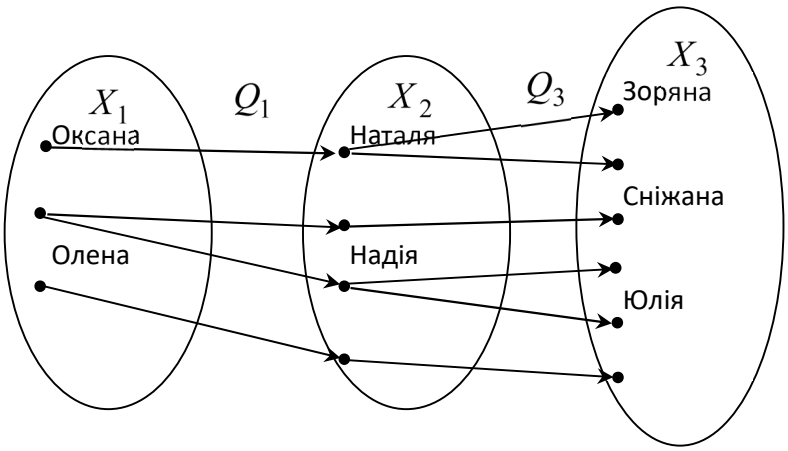


Рис.13

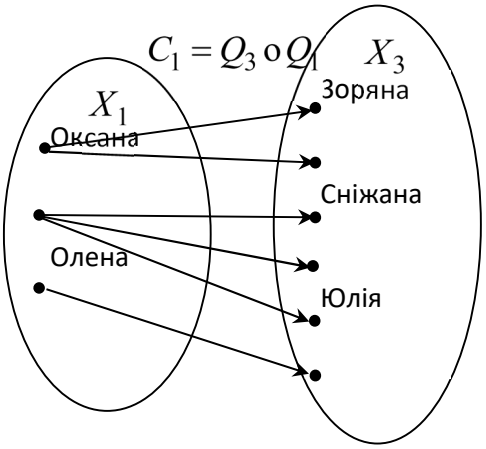


Рис.14

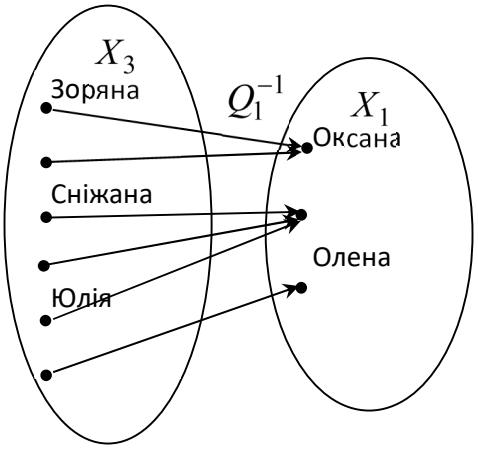


Рис.15

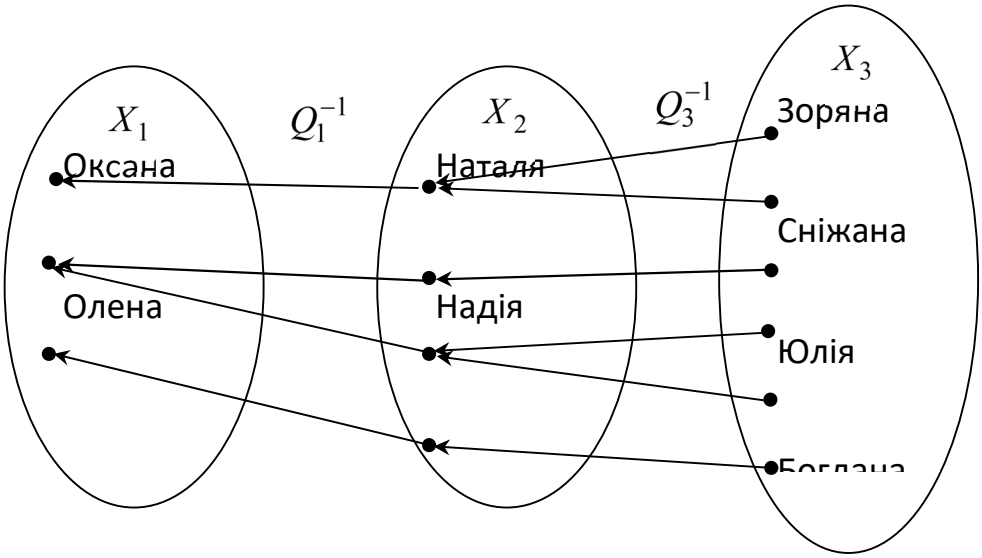


Рис.16

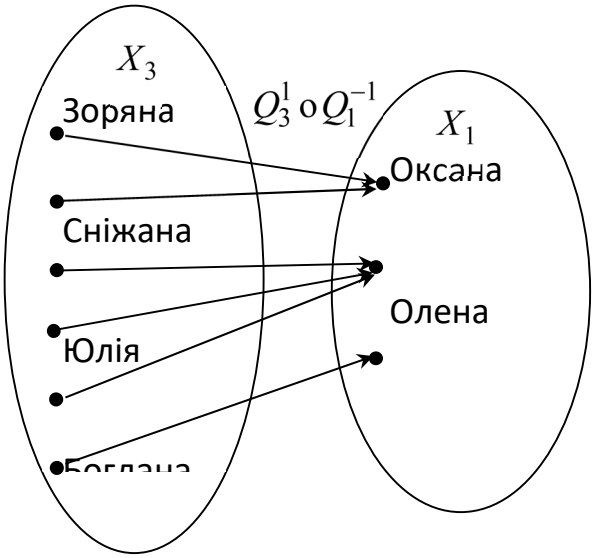


Рис.17

Приклад. Знайдемо матрицю композиції відношень A та B із прикладу 14:

$$A = \begin{array}{c|ccccc} & x_1 & x_2 & x_3 & x_4 & x_5 \\ \hline x_1 & 0 & 0 & 1 & 0 & 0 \\ x_2 & 0 & 0 & 0 & 0 & 0 \\ x_3 & 0 & 1 & 0 & 0 & 0 \\ x_4 & 0 & 0 & 1 & 0 & 1 \\ x_5 & 0 & 0 & 0 & 0 & 0 \end{array}; \qquad B = \begin{array}{c|ccccc} & x_1 & x_2 & x_3 & x_4 & x_5 \\ \hline x_1 & 0 & 0 & 0 & 0 & 0 \\ x_2 & 1 & 0 & 0 & 0 & 0 \\ x_3 & 1 & 0 & 0 & 0 & 0 \\ x_4 & 0 & 0 & 0 & 0 & 0 \\ x_5 & 0 & 0 & 1 & 0 & 0 \end{array};$$

$$B \times A = \begin{array}{c|ccccc} & x_1 & x_2 & x_3 & x_4 & x_5 \\ \hline x_1 & 1 & 0 & 0 & 0 & 0 \\ x_2 & 0 & 0 & 0 & 0 & 0 \\ x_3 & 1 & 0 & 0 & 0 & 0 \\ x_4 & 1 & 0 & 1 & 0 & 0 \\ x_5 & 0 & 0 & 0 & 0 & 0 \end{array}.$$

3.5. Властивості відношень.

Нехай A – бінарне відношення у множині X ($A \subset X \times X$).

1) Відношення A є *рефлексивним*, якщо $E \subset A$, тобто, іншими словами, воно завжди виконується між елементом і ним самим ($\forall x \in X\ (x\,A\,x)$). Як приклад такого відношення можна навести відношення нестрогої нерівності ($\leq, \geq$) на N, Z й R, відношення «жити в одному будинку», яке визначено на деякої множині X людей.

2) Відношення A є *антирефлексивним*, якщо $A \cap E = \varnothing$, тобто якщо співвідношення $x_i \, A \, x_j$ виконується, то $x_i \neq x_j$. Це, наприклад, відношення строгої нерівності на N, Z та R, відношення «бути старшим», «бути підлеглим» у множині людей.

3) Відношення A є *симетричним*, якщо $A = A^{-1} A$ = *A-1*, тобто при виконанні співвідношення $x_i \, A \, x_j$ виконується співвідношення $x_j \, A \, x_i$. Як приклад такого відношення можна навести відстань між двома точками на площині, відношення «бути братом» у множині чоловіків.

4) Відношення A є *асиметричним*, якщо $A \cap A^{-1} = \varnothing$, тобто із двох співвідношень $x_i \, A \, x_j$ й $x_j \, A \, x_i$ щонайменше одне не виконується. Як приклад такого відношення можна навести відношення «бути батьком», «бути підлеглим» у множині людей, відношення строгого включення в множині всіх підмножин деякого універсуму. Очевидно, якщо відношення асиметричне, то воно й антирефлексивне.

5) Відношення A є *антисиметричним*, якщо $A \cap A^{-1} \subset E$, тобто обидва співвідношення $x_i \, A \, x_j$ та $x_j \, A \, x_i$ одночасно виконуються тоді й тільки тоді, коли $x_i = x_j$. Як приклад можна навести нестрогу нерівність $(\leq, \geq)$ на N, Z та R.

6) Відношення A є *транзитивним*, якщо $A \circ A \subset A$, тобто з виконання співвідношень $x_i \, A \, x_j$ й $x_j \, A \, x_k$ випливає виконання співвідношення $x_i \, A \, x_k$. Як приклад можна навести відношення «бути дільником» у Z, «бути старшим» у множині людей.

Матриця рефлексивного відношення характеризується тим, що всі елементи її головної діагоналі – одиниці (рис.18), а матриця антирефлексивного відношення – тим, що зазначені елементи є нулями (рис.19).

$$\begin{bmatrix} 1 & & & & \\ & 1 & & & \\ & & 1 & & \\ & & & 1 & \\ & & & & 1 \end{bmatrix}$$

Рис.18

$$\begin{bmatrix} 0 & & & & \\ & 0 & & & \\ & & 0 & & \\ & & & 0 & \\ & & & & 0 \end{bmatrix}$$

Рис.19

Зауважимо, що симетричність відношення спричинює також симетричність матриці. Матриця асиметричного відношення характеризується тим, що всі елементи її головної діагоналі – нулі й немає жодної пари одиниць на місцях, симетричних відносно головної діагоналі (рис.20). Матриця антисиметричного відношення має ті самі властивості, що й асиметричного, за винятком вимоги рівності нулю елементів головної діагоналі (рис.21).

$$\begin{bmatrix} 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 \end{bmatrix}$$

Рис.20

$$\begin{bmatrix} 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 \end{bmatrix}$$

Рис.21

Матриця транзитивного відношення характеризується тим, що коли $a_{ji} = 1$ й $a_{kj} = 1$, то $a_{ki} = 1$, причому наявність одиничних елементів на головній діагоналі не порушує транзитивності матриці.

Граф рефлексивного відношення характеризується тим, що петлі є у всіх вершинах (рис.21), граф антирефлексивного відношення не має жодної петлі (рис.22).

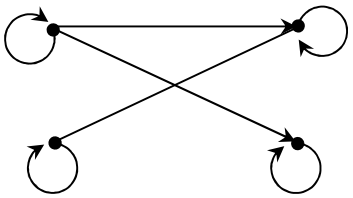


Рис.22

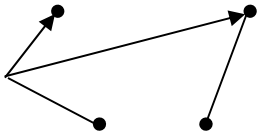


Рис.23

Для симетричного відношення вершини графа можуть бути пов'язані тільки парами протилежно спрямованих дуг (тобто ребрами). У графа асиметричного відношення петлі відсутні, а вершини можуть бути пов'язані тільки однією спрямованою дугою (рис.24).

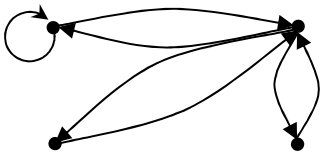


Рис.24

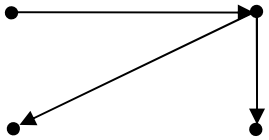


Рис.25

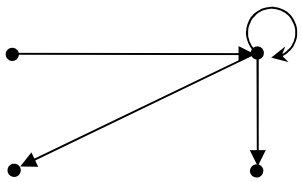


Рис.26

У разі антисиметричного відношення можуть бути петлі, але зв'язок між вершинами, якщо він є, також відображається тільки однією спрямованою дугою (рис.25).

Граф транзитивного відношення характеризується тим, що коли через деяку сукупність вершин графа проходить шлях, то існують дуги, які з'єднують будь-яку пару вершин з цієї сукупності в напрямку шляху. Як правило, на графі транзитивного відношення зображують тільки цей шлях, а зумовлені транзитивністю дуги опускають. Такий граф називають графом редукції (або кістяковим графом). Наприклад, граф редукції на п'яти вершинах має такий вигляд (рис.15):

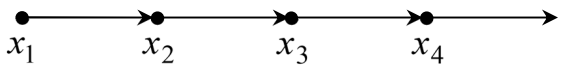


Рис.27

Бінарні (двомісні) відношення, що розглядаються, є окремим випадком n-місних відношень при $n = 2$.

Приклади з ІТ:

Властивість	Формальне означення	Приклад з ІТ-середовища
Рефлексивність	∀x∈A, (x,x)∈R∀x ∈ A,\ (x, x) ∈ R	Користувач бачить свій профіль
Антирефлексивність	∀x∈A, (x,x)∉R∀x ∈ A,\ (x, x) ∉ R	Користувач не ставить лайк сам собі
Симетричність	∀x,y, (x,y)∈R⇒(y,x)∈R∀x, y,\ (x, y) ∈ R ⇒ (y, x) ∈ R	Система дружби — якщо А друг В, то В друг А
Асиметричність	∀x,y, (x,y)∈R⇒(y,x)∉R∀x, y,\ (x, y) ∈ R ⇒ (y, x) ∉ R	Відношення «більше» або «має вищі права»
Антисиметричність	∀x≠y, (x,y)∈R∧(y,x)∈R⇒x=y∀x ≠ y,\ (x, y) ∈ R ∧ (y, x) ∈ R ⇒ x = y	Якщо користувач А має рівні права з В, то вони однакові
Транзитивність	∀x,y,z, (x,y)∈R∧(y,z)∈R⇒(x,z)∈R∀x, y, z,\ (x, y) ∈ R ∧ (y, z) ∈ R ⇒ (x, z) ∈ R	Якщо А має доступ до групи, яка має доступ до ресурсу, то А має доступ до ресурсу

3.6. Багатомісні відношення. Зв'язок відношень з реляційними базами даних.

Багатомісним, а саме n-місним відношенням A називається будь-яка підмножина декартового добутку n множин $X_1 \times X_2 \times \ldots \times X_n$, тобто n-місним є відношення $A = \{(x_1, x_2, \ldots, x_n) \mid (x_1, x_2, \ldots, x_n) \in x_1 \times x_2 \times \ldots \times x_n\}$, де $(x_1, x_2, \ldots, x_n)$ називається n-вимірним вектором, кортежем або просто впорядкованою n-кою.

Можна також визначити n-місне відношення за індукцією. За означенням бінарне відношення – це множина впорядкованих пар (x_1, x_2), де $x_1 \in X_1$, $x_2 \in X_2$. Впорядкована трійка (x_1, x_2, x_3) може розглядатися як упорядкована пара $((x_1, x_2), x_3)$, де перша координата (x_1, x_2) є впорядкованою парою, причому $(x_1, x_2) \in X_1 \times X_2$, і т. д. І нарешті, n-вимірний вектор $(x_1, x_2, \ldots, x_n)$ може розглядатися як упорядкована пара $((x_1, x_2, \ldots, x_{n-1}), x_n)$.

Приклад. Як приклади тримісного (тернарного) відношення можна навести всі арифметичні операції над числами (в них виділяється перший операнд, другий операнд і результат операції). Так, арифметична операція додавання D, що задана, наприклад, в множині дійсних чисел R, це тримісне відношення

$D \subset R \times R \times R$. $D = \{(d_1, d_2, d_3)\}$, де d_1 – перший доданок, d_2 – другий доданок, d_3 – сума.

Ще один приклад тримісного відношення можна навести, використовуючи вже відомі з прикладу 2 множину прізвищ співробітників філії деякої фірми $X = \{$Стеценко, Чуйко, Козак, множину $Y = \{$старший менеджер, менеджер$\}$, а також множину $Z = \{1000, 800\}$, яка є множиною посадових окладів. Тоді тримісне відношення $A \subset X \times Y \times Z$, $A = \{$(Стеценко, старший менеджер, 1000), (Чуйко, менеджер, 1000), (Козак, менеджер, 1000),$\}$ є відношенням, яке надає кожному співробітнику посаду і відповідний посадовий оклад.

Як ще один приклад тримісного (тернарного) відношення можна навести відношення, яке задає зв'язок між батьками і дітьми. Таке відношення є множиною трійок $\{(a, b, c)\}$, де a – ім'я матері, b – ім'я батька, c – ім'я дитини.

Пропорція $\frac{x}{y} = \frac{z}{u}$ ілюструє чотиримісне відношення P, де $P = \{(x, y, z, u)\}$.

Багатомісні відношення (у такому числі і бінарні) мають безпосередній зв'язок з реляційними базами даних, що базуються на основі реляційної алгебри, яка є алгеброю відношень (relation-відношення). Реляційні бази даних ще називають табличними, тому що багатомісні відношення дуже зручно представити у табличному вигляді. Наприклад, відношення A з прикладу 17 можна задати таблицею

Таблиця 3.3

№ п/п	Прізвище	Посада	Посадовий оклад
1	Стеценко І.О.	старший менеджер	1000
2	Чуйко Г.П.	менеджер	800
3	Козак С.Б.	менеджер	800

У чому значення багатомісних відношень в ІТ:

Таблиця 3.4

Застосування	***n*-місне відношення**
Таблиці в SQL	кожен запис — кортеж
Формати JSON/CSV	кожен об'єкт/рядок — n-ка
Об'єкти у програмуванні	поля об'єкта = координати в добутку
Структури даних (DataFrame)	аналогічно: атрибути = стовпці = множини
Вхідні дані для ML-моделі	вектори ознак — n-ки
Повнотекстовий індекс	(слово, документ, позиція) — тримісне відношення
Мережеві події	(IP, порт, мітка часу)

2. Функціональне відношення.

Відношення $f \subset X \times Y$ називається функціональним, якщо його елементи (впорядковані пари) мають різні перші координати: $\forall x \in D_0(f)\ \exists! y((x,y) \in f)$. Іншими словами, кожному $x \in X : (x,y) \in f$ відповідає один і тільки один елемент $y \in Y$. Очевидно, для функціонального відношення A кожний переріз за будь-яким $x \in X$ містить не більш як один елемент. Якщо $x \notin D_0(f)$, то переріз за x – порожній.

Якщо $D_0(f) = X$, то функціональне відношення f називається всюди визначеним. Матриця функціонального відношення містить у кожному стовпці не більш як один одиничний елемент, а його граф характеризується тим, що з кожної вершини може виходити тільки одна дуга (враховуючи й петлі).

Приклад. Наведений раніше приклад відношення «бути матір'ю сина» $Q_2 = \{$(Оксана, Олег), (Олена, Андрій), (Катерина, Сергій)$\}$ є також прикладом функціонального відношення. Для множин $X = \{1,2,3,4\}$ й $Y = \{1,4,9,16,25\}$ відношення $B = \{(1,1)(2,4)(3,9)(4,16)\}$ також є функціональним. Це відношення можна визначити як відношення, яке задає для кожного елемента множини X його квадрат.

Контрольні запитання

1. Що таке бінарне відношення? Як воно задається формально?
2. Що означає, що відношення є підмножиною декартового добутку?
3. Наведіть приклади бінарних відношень у сфері ІТ.
4. Що таке область визначення та область значень бінарного відношення?
5. Як зміниться область визначення, якщо відношення задане не на всій множині?
6. Що таке унарне відношення? Наведіть приклади.
7. Що означає, що відношення задане на множині?
8. Якими способами можна подати бінарне відношення?
9. Як виглядає матриця бінарного відношення? Наведіть приклад.
10. Як виглядає граф бінарного відношення? Що означають дуги?
11. Який вигляд має матриця повного, порожнього та діагонального відношення?
12. Що таке дводольний граф? У якому випадку відношення зображується дводольним графом?
13. Які операції можна виконувати над бінарними відношеннями як над множинами?
14. Що таке об’єднання двох бінарних відношень? Наведіть приклад.
15. Що таке перетин відношень? Де в ІТ це зустрічається?
16. Яка різниця між відношенням і його доповненням?
17. Що таке симетричне (обернене) відношення? Як його побудувати?
18. Як змінюється граф при переході до оберненого відношення?
19. Що таке композиція бінарних відношень? Наведіть формальне означення.
20. Як побудувати композицію відношень у прикладі «користувач — група — ресурс»?
21. Чи є композиція комутативною? Чому?
22. Які властивості має композиція (асоціативність, дистрибутивність)?
23. Як обчислити композицію відношень за їхніми матрицями?
24. Як інтерпретується композиція відношень у реляційних базах даних?
25. Що означає, що відношення є рефлексивним?
26. Що таке симетричне, асиметричне та антисиметричне відношення?
27. Що таке транзитивність? Як її розпізнати в графі?
28. Наведіть приклади транзитивного відношення з практики ІТ.
29. Як виглядає граф рефлексивного або симетричного відношення?
30. Що таке багатомісне (n-місне) відношення?
31. Як n-місне відношення пов’язане з таблицями у реляційних базах?
32. Наведіть приклад тримісного відношення у вигляді таблиці.
33. Як відображається багатомісне відношення в JSON, CSV чи SQL?
34. Який зв’язок між кортежем і об’єктом в ООП?

4. ВІДОБРАЖЕННЯ (ФУНКЦІЇ)

Відображення, або функція, — це особливий тип бінарного відношення, який має ключове значення в програмуванні, базах даних, теорії обчислень, побудові алгоритмів, інтерпретації типів тощо. Для спеціальності «Комп'ютерні науки» це одна з базових абстракцій.

Будь-яке функціональне відношення можна інтерпретувати як функцію. У цьому випадку перший елемент упорядкованої пари $(x, y) \in A$ розглядається як аргумент або змінна, а другий — як значення функції (образ). Це можна позначати у вигляді як $y = f(x)$ чи xfy або $(x, y) \in f$ — усі ці записи є рівнозначними.

Важливо розрізняти: сама функція — це множина впорядкованих пар, а значення функції в конкретній точці — це другий елемент відповідної пари.

Слід враховувати, що обернене (симетричне) до функціонального відношення не обов'язково буде функцією. У наведених вище прикладах обидва відношення були функціональними, як і їх обернення. Але так буває не завжди. Наприклад, якщо розглянути відношення, де кожна людина пов'язується зі своєю матір'ю, то симетричне до нього відношення (тобто — від дитини до матері) є функцією, оскільки кожна дитина має одну матір. Натомість саме початкове відношення не є функцією, бо одна особа (мати) може мати кількох дітей, тобто відповідати кільком елементам.

Ще один приклад — відношення «бути матір'ю дочки»:

{(Оксана,Наталя),(Олена,Надія),(Олена,Ольга),(Катерина,Галина)}

— не є функцією, бо одна особа (Олена) має дві дочки. А обернене до нього:

{(Наталя,Оксана),(Надія,Олена),(Ольга,Олена),(Галина,Катерина)}

— вже є функцією: кожна дочка має одну матір.

Якщо функціональне відношення визначене на всіх елементах множини A, тобто для кожного $a \in A$ існує єдиний відповідний $b \in B$, — таке відношення називається відображенням множини A в множину B і позначається $f: A \to B$.

Інакше кажучи, відображення — це функція, повністю визначена на своїй області визначення A і така, що кожному елементу A ставить у відповідність точно один елемент із B.

Приклади з ІТ:

Відображення	***A***	***B***	***f*(*a*)**
Індексування масиву	індекси	значення	*f*(2)=’error’
Функція у Python	параметри	повернене значення	def f(x): return x**2
Таблиця з ключами	user_id	email	user_id → email
Хеш-функція	ключ	хеш	hash(password)
Структура словника (dict/map)	ключ	значення	users['bob'] = 'admin'

4.1. Типи відображень.

При відображенні X в Y кожний елемент x з X має один і тільки один образ $(\forall x \in X \; \exists! y \in Y (y = f(x)))$.

Однак зовсім не обов'язково, щоб кожний елемент з Y був образом деякого елемента з X. Графічно цю ситуацію можна зобразити так, як, наприклад, показано на рис.3.28.

Якщо ж будь-який елемент з Y є образом принаймні одного елемента з X, то кажуть, що множина X відображується на Y (явище сюр’єкції, або накриття). Цю ситуацію зображено на рис.3.29.

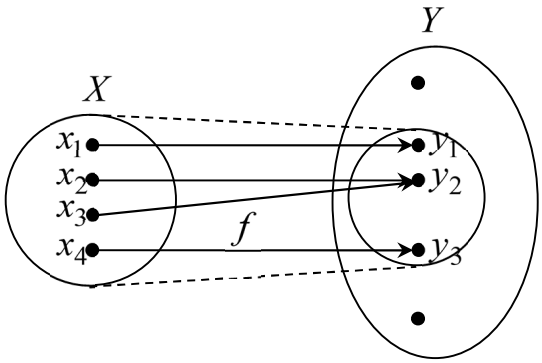


Рис.3.28

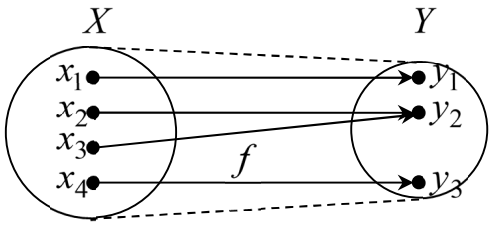


Рис.3.29

Якщо для будь-яких двох різних елементів x_1 й x_2 з X їх образи $y_1 = f(x_1)$, $y_2 = f(x_2)$ також різні, то відображення f називається ін'єкцією, або взаємно однозначним відображенням. Цю ситуацію показано на рис.3.30.

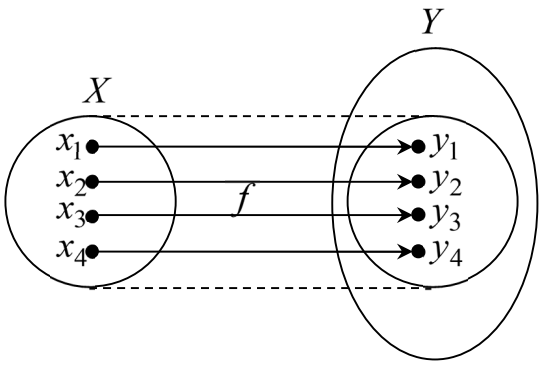


Рис.3.30

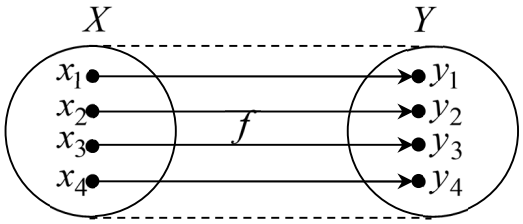


Рис.3.31

Відображення, яке є одночасно сюр'єктивним та ін'єктивним, називається бієкцією (накладанням). У цьому випадку кажуть, що між елементами X й Y існує взаємно однозначна відповідність. При цьому (рис.3.31) виникає питання, чи є обернене відношення f також взаємно однозначним.

Будь-яке відображення f із X в Y є елементом множини $P(X \times Y)$.

Якщо $f-$ взаємно однозначне відображення, а $X = Y$, то $f : X \to Y$ називається відображенням множини X на себе, причому $f \circ f^{-1} = f^{-1} \circ f = E$. Приклад наведений на рис.3.32.

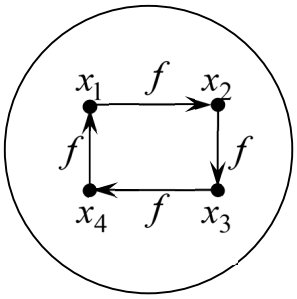


Рис.3.32

Образи і прообрази. Загалом при відображенні $f : X \to Y$ елемент із Y може бути образом не одного, а кількох елементів із X . Так, для розглянутого у прикладі на рис.3.29 відношення елемент y_2 є образом для елементів x_2, x_3.

Сукупність усіх елементів, образом яких є заданий елемент y, називається повним прообразом елемента y і позначається . У наведеному прикладі $f^{-1}(y) = \{x_2, x_3\}$.

Сукупність елементів $f(x)$, які є образами всіх елементів множини X, називається образом цієї множини та позначається $f(X)$.

Нехай $R \subset Y$. Сукупність усіх елементів із X, образи яких належать R, називається повним прообразом множини R і позначається $(f^{-1}(R))$.

4.2. Композиція відображень.

Якщо $f: X \to Y$, $q: Y \to Z$, то їх композиція $q \circ f: X \to Z$, причому $(q \circ f)(x) = q(f(x))$. Наприклад, якщо $f = \sin$, $q = \ln$, то

$$(q \circ f)(x) = (\ln \circ \sin)(x) = \ln(\sin(x)) = \ln \sin x.$$

Теорема. *Функція f є взаємно однозначним функціональним відношенням тоді й тільки тоді, коли f^{-1} – взаємно однозначне функціональне відношення.*

Теорема. *Композиція двох функціональних відношень є функціональним відношенням.*

Приймемо ці теореми без доведення. Наведемо приклади для ілюстрації цих теорем.

Приклад. Дійсно, якщо $f = \{(x_1, y_1), (x_2, y_2), (x_3, y_3)\}$, що діє з множини X у множину Y, є функціональним відношенням, то й обернене (симетричне) $f^{-1} = \{(y_1, x_1), (y_2, x_2), (y_3, x_3)\}$, що діє з множини Y у множину X, є функціональним.

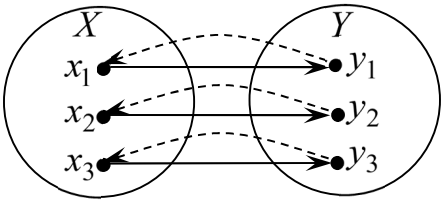


Рис.3.33

На рис.3.33 функціональне відношення f показано суцільною лінією, а обернене f^{-1} – переривчастою.

Розглянемо ще одне функціональне відношення $g = \{(y_1, z_2), (y_2, z_1), (y_3, z_3)\}$, що діє з множини Y у множину Z. Візьмемо композицію відношень f і g:

$g \circ f = \{(x_1, z_2), (x_2, z_1), (x_3, z_3)\}$, що теж є функціональним. Графічне зображення наведено на рис.3.34.

$g \circ f$

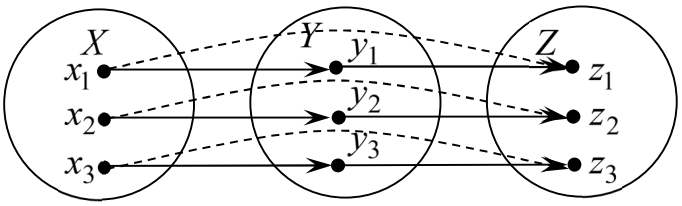


Рис.3.34

Композицію до f наведено переривчастою лінією.

Приклад.

Користувачі → Групи → Доступ до ресурсів

Маємо три множини:

A — користувачі (User)

B — групи користувачів (Group)

C — ресурси, до яких може бути наданий доступ (Resource)

Нехай:

$f: A \to B$ — відображення користувачів у групи (тобто кожен користувач є членом певної групи);

$g: B \to C$ — відображення груп у дозволені ресурси (тобто кожна група має доступ до визначених ресурсів).

Тоді композиція:

$g \circ f: A \to C$ — визначає відображення користувачів у ресурси, доступні через їхню групу.

Практичне застосування в IT:

- в ACL (access control lists): розрахунок ефективних прав доступу користувача через успадкування;
- в ORM-системах: автоматична побудова JOIN через зовнішні ключі;
- у безпеці: розрахунок рівня допуску до об'єктів через групову політику.
- в RBAC-моделях (role-based access control): User → Role → Permissions

4.3. Суперпозиція функцій.

Означення 3.5. *Функція, що утворюється з функцій* $f_1, f_2, \ldots, f_n$ *деякою підстановкою їх одна в одну і перейменуванням аргументів, називається суперпозицією* $f_1, f_2, \ldots, f_n$.

Для багатомісних функцій $f: A^m \to B$, $g: B^n \to C$ можливими є різні варіанти підстановки f у g, які дають функції різних типів. Наприклад, при $m = 3$, $n = 4$ функція $h_1 = g(x_1, f(y_1, y_2, y_3), x_3, x_4)$ має шість аргументів і діє з $B \times A^3 \times B^2 \to C$, а функція $h_2 = g(f(y_1, y_2, y_3), f(z_1, z_2, z_3), x_3, x_4)$ має вісім аргументів та діє з $A^6 \times B^2 \to C$. Особливо цікавим є випадок, коли задано множину функцій типу $f_1: A^{m_1} \to A, A^{m_2} \to A, \ldots, A^{m_n} \to A$. У цьому разі може бути виконане будь-яке перейменування аргументів, наприклад перейменування x_3 в x_2, що породжує з функції $f(x_1, x_2, x_3, x_4)$ функцію трьох аргументів $f(x_1, x_2, x_2, x_4)$.

Приклад. У функції $f_1(x_1, x_2, x_3) = x_1 + 2x_2 + 7x_3$ перейменування x_3 в x_2 приводить до функції $f_1(x_1, x_2, x_3) = x_1 + 2x_2 + 7x_3 = f_2(x_1, x_2) = x_1 + 9x_2$. Перейменування x_1 та x_3 в x_2 приводить до одномісної функції $f_3(x_2) = 10x_2$.

Приклад. Англо-український словник установлює відповідність між множиною англійських та українських слів. Ця відповідність не є

функціональною (оскільки одному англійському слову, як правило, ставляться у відповідність кілька українських слів); крім того, вона практично ніколи не є повністю визначеною: завжди можна знайти англійське слово, що міститься в цьому словнику.

4.4. Відношення еквівалентності

Ми вже розглянули властивості бінарних відношень у загальному вигляді. Серед них виокремлюються особливі типи, які мають комплекс характерних властивостей. Такі відношення поділяють на певні категорії — наприклад, функціональні відношення, з якими ми вже знайомі.

Зосередимо увагу на відношеннях еквівалентності. Це формалізоване представлення таких інтуїтивних понять, як подібність, не відрізнити, взаємозамінність, рівноцінність.

Означення. Бінарне відношення $\sim$, задане на множині A, називається відношенням еквівалентності, якщо воно задовольняє такі три умови:

Рефлексивність: кожен елемент є еквівалентним самому собі:

$\forall a \in A,\ a \sim a$

Симетричність: якщо $a \sim b$, то й $b \sim a$: $\forall a,b \in A,\ a \sim b \Rightarrow b \sim a$

Транзитивність: якщо $a \sim b$ і $b \sim c$, то $a \sim c$: $\forall a,b,c \in A,\ a \sim b \wedge b \sim c \Rightarrow a \sim c$

Відношення еквівалентності дозволяє класифікувати елементи множини — об’єднувати їх у класи, всередині яких усі елементи вважаються "однаковими" згідно з певним критерієм.

Приклад.

Деякі типові приклади відношень еквівалентності:

- «мешкають в одному будинку» на множині мешканців міста;
- «навчаються в одній академічній групі» для студентів факультету;
- «подібність» для трикутників на площині;
- «бути паралельними» для прямих.

Класи еквівалентності

Для будь-якого елемента $a \in A$, класом еквівалентності називається множина всіх елементів, що є з ним еквівалентними:

$[a] = \{x \in A \mid x \sim a\}$

Твердження. Будь-який елемент належить до свого класу еквівалентності:

$a \in [a]$

Твердження. Якщо два елементи еквівалентні, то вони належать одному й тому ж класу: $a \sim b \Rightarrow [a] = [b]$

Теорема. *Якщо на множині X задано відношення еквівалентності, то воно задає розбиття множини і це розбиття – єдине.*

Цю теорему ми приймемо без строгого доведення, але проаналізуємо його хід.

Нехай на множині X задано відношення еквівалентності. Розглянемо сукупність усіх класів еквівалентності $\{[x]_\sim\}$, які можуть бути утворені. Можна довести, що:

1. $\bigcup_{x\in X}[x]_\sim = X$.
2. $\forall x, y \in X : [x]_\sim \neq [y]_\sim ([x]_\sim \cap [y]_\sim = \varnothing)$.

Перший пункт означає, що об'єднання всіх класів еквівалентності дорівнює всій множині X. Дійсно, будь-який елемент x множини X є одночасно елементом множини X і елементом деякого класу еквівалентності (наприклад, за твердженням 3.2 класу $[x]_\sim$).

Другий пункт означає, що переріз будь-яких двох різних класів еквівалентності порожній, тобто різні класи еквівалентності не перерізаються.

Обидва пункти означають, що класи еквівалентності утворюють розбиття множини X. Можна довести єдиність такого розбиття.

Слід зауважити, що в множині X може бути задане не єдине відношення еквівалентності, але кожне відношення задає певне розбиття множини X і за цим відношенням еквівалентності це розбиття єдине.

3.3.2. Матриця і граф відношення еквівалентності.

Нехай відношення еквівалентності задано в множині X. Елементи, що належать одному класу еквівалентності, попарно еквівалентні між собою. Отже, стовпці матриці відношення еквівалентності для елементів одного класу еквівалентності однакові та містять одиниці у всіх рядках, які відповідають цим елементам. Оскільки класи еквівалентності не перерізаються, у стовпцях, які відповідають елементам різних класів, не буде одиниць в одних і тих самих рядках.

При побудові матриці відношення розташуємо елементи множини так, щоб ті елементи, які належать одному класу еквівалентності, були поруч. Тоді одиничні елементи матриці відношення еквівалентності утворять непересічні квадрати, діагоналі яких розташовуються на головній діагоналі матриці.

Приклад 3.29. Для відношення еквівалентності, заданого класами еквівалентності

$$X_1 = \{x_1, x_2, x_4\};\ X_2 = \{x_3, x_8\};\ X_3 = \{x_5, x_6, x_7, x_9, x_{10}\},$$

матриця матиме такий вигляд (табл.3.4):

Таблиця 3.4

	x_1	x_2	x_4	x_3	x_8	x_5	x_6	x_7	x_9	x_{10}
x_1	1	1	1	0	0	0	0	0	0	0
x_2	1	1	1	0	0	0	0	0	0	0
x_4	1	1	1	0	0	0	0	0	0	0
x_3	0	0	0	1	1	0	0	0	0	0
x_8	0	0	0	1	1	0	0	0	0	0
x_5	0	0	0	0	0	1	1	1	1	1
x_6	0	0	0	0	0	1	1	1	1	1
x_7	0	0	0	0	0	1	1	1	1	1
x_9	0	0	0	0	0	1	1	1	1	1
x_{10}	0	0	0	0	0	1	1	1	1	1

Граф відношення еквівалентності також має характерний вигляд. Це граф, кожна компонента з'єднання якого, що відповідає класу еквівалентності, є повним графом із петлями на кожній вершині.

Для цього прикладу граф має вигляд, зображений на рис.3.37.

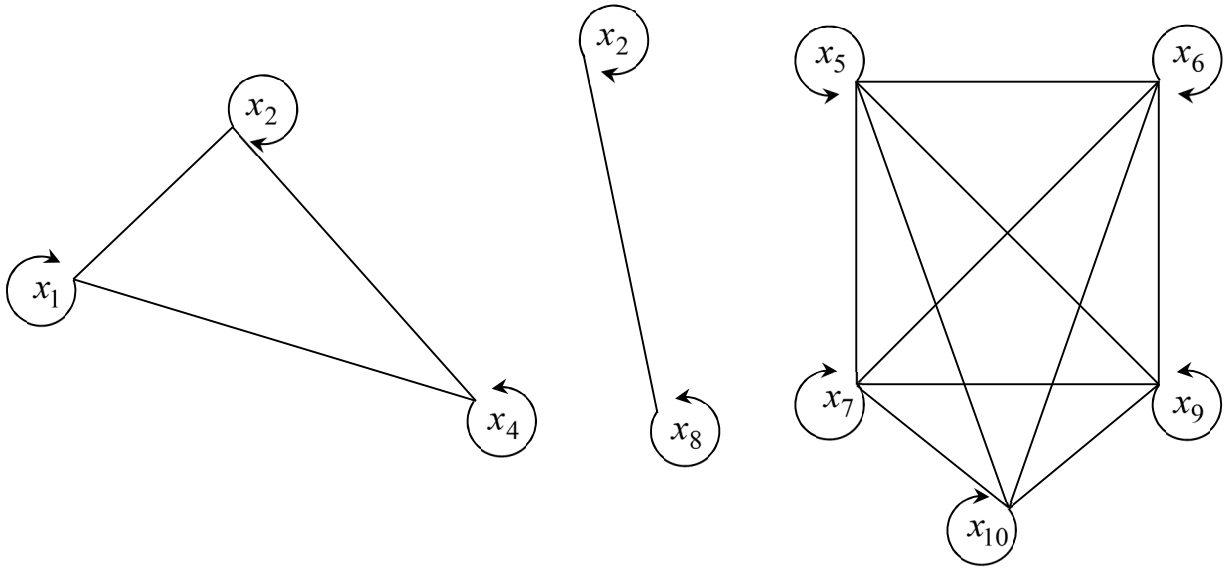


Рис.3.37

4.5. Відношення порядку.

3.4.1. Загальні властивості.

Розглянемо ще один окремий випадок бінарних відношень, який має велике значення в застосуваннях теорії множин. Це відношення порядку.

Означення. *Бінарне відношення* σ *в множині* X *називається відношенням нестрогого порядку, якщо воно*:

1) *рефлексивне* $(\forall x \in X(x \sigma x))$;

2) *антисиметричне* $(\forall x, y \in X(x \sigma y \wedge y \sigma x \Rightarrow x = y))$;

3) *транзитивне* $(\forall x, y, z \in X(x \sigma y \wedge y \sigma z \Rightarrow x \sigma z))$.

Часто відношення σ позначають «≤», оскільки нестрога нерівність є прикладом відношення нестрогого порядку в множині цілих Z або дійсних R, що найчастіше використовується (як і «≥»), але ототожнювати їх усе ж не варто. Як приклад відношення нестрогого порядку в множині людей можна назвати відношення «бути не старшим» або «бути не молодшим».

Означення. *Бінарне відношення p в множині X називається відношенням строгого порядку, якщо воно*:

1) *асиметричне* $(\forall y \in X(x p y \Rightarrow \overline{y p x}))$;

2) *транзитивне* $(\forall x, y, z \in X(x p y \wedge y p z \Rightarrow x p z))$.

Приклад. Як приклад відношення строгого порядку можна навести відношення

1) «>» або «<» у множинах N, Z чи R;

2) відношення «бути молодшим» або «бути старшим» у множині людей;

3) бути прямим «предком» в множині людей.

Якщо виконується співвідношення $x p y$ $(x \sigma y)$, то кажуть, що елемент x передує y, а y іде за x (y передує x, а x іде за y).

Множина, в якій визначено відношення порядку (строгого або нестрогого), називається упорядкованою, і кажуть, що порядок уведено цим відношенням.

Якщо виконується співвідношення $x A y$ або $y A x$, де A – відношення строгого або нестрогого порядку, то кажуть, що елементи a і y зрівнянні за відношенням порядку A.

Означення. *Множина M називається абсолютно (лінійно) впорядкованою, якщо для будь-яких двох її елементів x та y виконується $x \sigma y$ або $y \sigma x$ ($x p y$ або $y p x$) тобто, якщо будь-які два елементи множини M зрівнянні за деяким відношенням порядку.*

Приклад. Розглянемо множину дійсних чисел R і відношенням порядку « ≤» (або «≥», «<», «>»). Зрозуміло, що два будь-яких дійсних числа зрівнянні (виконується $a \le b$ або $b \le a$). Тому множина R з відношенням нестрогого порядку «≤» («≥») є абсолютно впорядкованою.

Може виявитися, що для деяких пар (x, y) жодне зі співвідношень $x \sigma y$ або $y \sigma x$ ($x p y$ або $y p x$) не виконується. Такі елементи x й y називаються *незрівнянними*. У цьому випадку кажуть, що множина є частково впорядкованою.

3.4.5. Матриці відношень порядку.

Матриця відношення нестрогого порядку. Оскільки відношення нестрогого порядку є рефлексивним, головна діагональ матриці цього відношення містить одиниці. Через те що воно є асиметричним, жоден одиничний елемент не має симетричного собі відносно головної діагоналі. Оскільки це відношення є транзитивним, наявність одиниць на перетині i-го стовпця та j-го рядка й одиниці на перетині j-го стовпця і k-го рядка спричинює наявність одиниці на перетині i-го стовпця та k-го рядка.

Граф відношення нестрогого порядку. Внаслідок виконання властивостей рефлексивності, асиметричності та транзитивності граф відношення нестрогого порядку характеризується тим, що в кожній його вершині існує петля, жодна пара вершин не пов'язана дугами протилежного напрямку, а всі його вершини будь-якого шляху попарно пов'язані між собою дугами в напрямку цього шляху. Як звичайно, граф транзитивного відношення будемо зображати графом редукції.

Приклад. Для відношення нестрогого порядку $A =$«бути дільником» на множині $X = \{1,2,3,4,6,7,12,14,21,28,42,84\}$ й $X = \{1,2,3,4\}$ матриці мають такий вигляд:

$A \subset X \times Y =$

	1	2	3	4	6	7	12	14	21	28	42	84
1	1	0	0	0	0	0	0	0	0	0	0	0
2	1	1	0	0	0	0	0	0	0	0	0	0
3	1	0	1	0	0	0	0	0	0	0	0	0
4	1	1	0	1	0	0	0	0	0	0	0	0
6	1	1	1	0	1	0	0	0	0	0	0	0
7	1	0	0	0	0	1	0	0	0	0	0	0
12	1	1	1	1	1	0	1	0	0	0	0	0
14	1	1	0	0	0	1	0	1	0	0	0	0
21	1	0	1	0	0	1	0	0	1	0	0	0
28	1	1	0	1	0	1	0	1	0	1	0	0
42	1	1	1	0	1	1	0	1	1	0	1	0
84	1	1	1	1	1	1	1	1	1	1	1	1

;

$A \subset X \times Y =$

	1	2	3	4
1	1	0	0	0
2	1	1	0	0
3	1	0	1	0
4	1	1	0	1

.

Приклад. Як приклад для побудови графа відношення нестрогого порядку розглянемо відношення з прикладу 3.37. На рис.3.40 зображено його граф редукції з петлями.

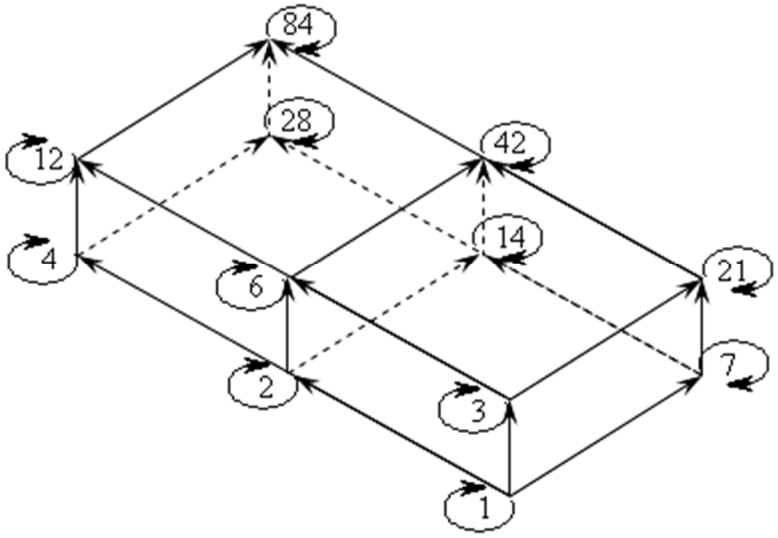


Рис.3.40

Матриця відношення строгого порядку. Матриця відношення строгого порядку будується аналогічно матриці нестрогого порядку і різниться тільки наявністю нулів на головній діагоналі (внаслідок властивості антирефлексивності).

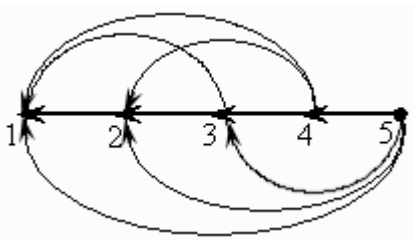
Приклад. Для відношення строгого порядку «>» на множині $X = \{1, 2, 3, 4, 5\}$ матриця має вигляд:

	1	2	3	4	5
1	0	1	1	1	1
2	0	0	1	1	1
3	0	0	0	1	1
4	0	0	0	0	1
5	0	0	0	0	0

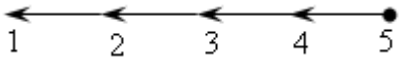
.

Граф відношення строгого порядку. Граф відношення строгого порядку будується аналогічно графу нестрогого порядку і різнитися тільки відсутністю петель.

Приклад. Для побудови графа строгого порядку скористаємося матрицею з прикладу 3.38. Граф відношення строгого порядку «>» на множині $X = \{1, 2, 3, 4, 5\}$ має вигляд, показаний на рис.3.41,а. Його граф редукції зображено на рис.3.41,б.



а)



б)

Рис.3.41

Контрольні запитання

1. Що називається відображенням?
2. У чому полягає різниця між функціональним відношенням і функцією?
3. Що таке область визначення і область значень функції?
4. Які є способи задання функції?
5. Як виглядає граф функції у вигляді множини впорядкованих пар?
6. Що означає запис $f(x) = y$ з погляду теорії множин?
7. Яке відображення називається ін'єктивним?
8. Що означає сюр'єктивність функції?
9. У яких випадках відображення є бієктивним?
10. Наведіть приклади ін'єкції, сюр'єкції та бієкції з програмування.
11. Чи є відображення email → user_id бієкцією? Обґрунтуйте.
12. Яким чином у Python реалізується ін'єкція? А бієкція?
13. Що таке образ елемента та множини при відображенні?
14. Що називається прообразом елемента та множини?
15. Чи може один елемент мати кілька прообразів?
16. Наведіть приклад, де образ кількох елементів збігається.
17. Що таке композиція відображень?
18. Як формується композиція $g \circ f$? Наведіть приклад.
19. Які властивості має композиція (асоціативність, існування оберненої тощо)?
20. Як реалізується композиція функцій у Python або SQL?
21. Що таке суперпозиція функцій?
22. У чому різниця між композицією та суперпозицією?
23. Як виглядає граф функції?
24. Які властивості має граф функції?
25. Що спільного між графом функції та графом функціонального відношення?
26. Як виглядає матриця функції? Які її особливості?
27. Як функції використовуються у структурах типу dict, map, JSON?
28. Що таке хеш-функція, і чому вона є функцією?
29. Як реалізовано відображення у базах даних (таблиці, ключі)?
30. Як пов'язано відображення з типами в ООП?
31. Який сенс має ін'єкція у випадку створення унікальних ідентифікаторів?
32. Чому відображення важливе для розв'язання задач ML (X → y)?
33. Яке відображення називається відображенням на себе?
34. Яка функція є оберненою до даної? У яких випадках вона існує?
35. Чи завжди композиція двох функцій є функцією?

Список рекомендованої літератури

1. Biggs N. Discrete Mathematics. Oxford Science Publications, 1990.
2. Matson A. F. Discrete Mathematics with applications. John Wiley and Sons Inc., 1993.
3. Андрійчук В. І., Комарницький М. Я., Іщук Ю. Б. Вступ до дискретної математики. Київ : Центр навчальної літератури, 2004.
4. Балога С. І. Дискретна математика : навчальний посібник. Ужгород : ПП «АУТОДОР-Шарк», 2021. 124 с.
5. Балога С. І., Король І. Ю. Методичні вказівки і завдання до лабораторних робіт з курсу «Дискретна математика» для студентів 1-го курсу інженерно-технічного факультету спеціальності «Комп'ютерні системи та мережі». Ужгород : УжНУ «Говерла», 2012. 56 с.
6. Бардачов Ю. М. та ін. Дискретна математика : підручник. Київ : Вища школа, 2002. 287 с.
7. Бардачов Ю. М., Соколова Н. А., Ходаков В. Є. Дискретна математика. Київ : Вища школа, 2002.
8. Бондаренко М. Ф., Білоус Н. В., Раткас А. Г. Комп'ютерна дискретна математика : підручник. Харків : Компанія СМІТ, 2004. 480 с.
9. Борисенко О. А. Дискретна математика : підручник. Суми : Університетська книга, 2007. 255 с.
10. Гвоздьова Є. В., Гірник М. О. Дискретна математика : навчальний посібник для студентів напрямів підготовки «Комп'ютерні науки» та «Економічна кібернетика». Львів : Вид-во Львівської комерційної академії, 2015. 123 с.
11. Дармосюк В. М., Пархоменко О. Ю., Васильєва Л. Я. Комбінаторика : практикум з розв'язання задач : посібник для самостійної та дистанційної роботи студентів. Миколаїв : Миколаївський національний університет ім. В. О. Сухомлинського, 2022. 78 с.
12. Дискретна математика : конспект лекцій для здобувачів першого (бакалаврського) рівня вищої освіти ОПП «Комп'ютерні науки» спеціальності 122 «Комп'ютерні науки» денної форми здобуття вищої

освіти / уклад. О. В. Шебаніна, С. І. Тищенко, В. О. Крайній, О. Ю. Пархоменко, І. І. Хилько. Миколаїв : МНАУ, 2023. 162 с. URL: https://dspace.mnau.edu.ua/jspui/handle/123456789/14555

13. Журавчак Л. М. Дискретна математика для програмістів : навчальний посібник. Львів : Львівська політехніка, 2019. 420 с.

14. Коваленко Л. Б., Станішевський С. О. Дискретна математика : навчальний посібник для студентів економічних, менеджерських та електротехнічних спеціальностей вищих навчальних закладів. Харків : ХНАМГ, 2006. 192 с.

15. Кривий С. Л. Дискретна математика. Київ : Букрек, 2017. 568 с.

16. Матвієнко М. П. Дискретна математика : підручник. 2-ге вид., перероб. і допов. Київ : Ліра-К, 2019. 324 с.

17. Оленко А. Я., Ядренко М. Й. Дискретна математика. Київ : Видавничий центр Київського університету, 1997.

18. Харченко В. М. Практикум з дискретної математики / В. М. Харченко. Ніжин : НДУ ім. М. Гоголя, 2022. 148 с. URL: https://surl.li/dfrxwl

19. Ядренко М. Й. Дискретна математика. Київ, 2003.

Навчальне видання

ДИСКРЕТНА МАТЕМАТИКА

Конспект лекцій

Укладач:

Пархоменко Олександр Юрійович

Формат 60х84 1/16. Ум. друк. арк. 10,06
Тираж 50 прим. Зам. № __

Надруковано у видавничому відділі
Миколаївського національного аграрного університету
54020, м. Миколаїв, вул. Георгія Гонгадзе, 9

Свідоцтво суб'єкта видавничої справи ДК № 4490 від 20.02.2013 р.