НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ОДЕСЬКА ПОЛІТЕХНІКА»

Пригожев Олександр Сергійович

ДИСКРЕТНА МАТЕМАТИКА ДЛЯ РОЗРОБНИКІВ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Навчальний посібник

Рекомендовано в якості навчального посібника для студентів спеціальності 121 – Інженерія програмного забезпечення. (Протокол Вченої ради НУ «Одеська політехніка» № 13 від 29.05.2024)

Одеса, Одеська політехніка - 2024

УДК 510.2

Пригожев О.С. Дискретна математика для розробників програмного забезпечення / Пригожев О.С. . – Одеса, Одеська політехніка, 2024 рік – 160 с

Рецензент: Приходько С.Б., доктор технічних наук, професор, завідувач кафедри програмного забезпечення автоматизованих систем Національного університету кораблебудування імені адмірала Макарова

ЗМІСТ

1 ПОНЯТТЯ МНОЖИНИ

1.1 Основні визначення теорії множин

Поняття множини належить до числа фундаментальних невизначуваних понять математики. Можна сказати, що *множина*— це будь-яка визначена сукупність різних об'єктів. При цьому під об'єктом розуміється будь-яка сутність реального світу, тобто, з точки зору «все, що може бути представлено, названо або сприйнято дослідником». Фактично, кожен об’єкт реального світу може бути у складі деякої множини.

⚠ ***Множина не має суворого визначення. Надалі у книзі використовується поняття множини як будь-якої визначеної сукупності різних об’єктів***

Оскільки ми говоримо про множину, як про сукупність об’єктів, яка має визначення, таке визначення надалі будемо називати характеристичною властивістю множини. *Характеристична властивість* визначає, яким чином були відібрані елементи у множину. Синонімами поняття «характеристична властивість» є характеристичний предикат та породжуюча функція. Поняття характеристичного предикату використовується, як правило, в тому випадку, коли потрібно описати множину через іншу множину, для якої визначений предикат. Термін «породжуюча функція» використовується в тому випадку, коли елементи множини отримані шляхом виконання певної дії. Отже, множина може бути створена шляхом

відбору елементів за деякою умовою, або шляхом виконання будь-якої іншої дії[1], зокрема, і перерахуванням всіх елементів множини у фігурних дужках, без явного формулювання характеристичної властивості.

З шкільного курсу української мови відоме наступне визначення іменника «Іменник - це частина мови, яка означає предмет». Слово «предмет» у даному контексті синонімічно поняттю «об'єкт». Таким чином, іменник в словесному формулюванні характеристичної властивості вказує на назву елемента множини. З правил синтаксису української мови відомо, що існують називні речення. Це тип односкладних речень, в якому головним членом є підмет. Як випливає з цього визначення, це речення називає деякий об'єкт або набір об'єктів, а, отже, за допомогою цього типу речень, можна формулювати характеристичну властивість. Характеристична властивість має на увазі чітку відповідь «Так» або «Ні» на питання чи «Входить елемент в цю множину»?. Тому у формулюванні характеристичної властивості доцільно використати такі конструкції синтаксису, як підрядне речення умови, синонімічний йому дієприслівниковий зворот, інші ввідні синтаксичні конструкції, а також слова, які відіграють у реченні – формулюванні характеристичної властивості роль визначення, яке пов'язане з іменником-підметом.

Наведемо декілька прикладів множин, які створені за допомогою характеристичних властивостей, предикатів та породжуючих функцій.

Приклад 1.1 «Студенти, що сидять в аудиторії». В даному випадку, елементами множини будуть студенти, що виражено підметом цього речення, а умова приналежності - те, що студент перебуває в цій аудиторії виражене дієприслівниковим зворотом. Слід звернути увагу на приховану невизначеність цієї характеристичної властивості, якщо ми розглядаємо конкретний ВНЗ з конкретною назвою у конкретному місті (наприклад, ОНПУ). Ця невизначеність викликана тим, що у характеристичній властивості не вказано, у якій

[1] Більш детально характеристичні предикати розглядаються у темі «Предикати», а «Породжуючі функції» у розділах «Відповідності» та «Операції»

саме аудиторії цього ВНЗ перебувають студенти, що може створювати необхідність завдання у відповідь на характеристичну властивість питання «У якій саме аудиторії сидять студенти?». Однак цього питання може і не виникнути, якщо мова іде про ВНЗ взагалі, тобто як про сутність реального світу. Це явище викликало появу, зокрема, гнучких методологій розробки програмного забезепечення (наприклад, Agile або Scrum), де вимоги до програмного забезпечення уточнюються постійно. Також це стало одним з чинників появи об'єктно-орієнтованих мов програмування. [2] З точки зору математики, це є прикладом поняття «підмножини», яке розглядається нижче

Приклад 1.2 «Цілі числа, більші за 10». В даному випадку сама множина міститиме числа(підмет цього речення), а умова формулюється ввідною конструкцією. Слід відзначити, що ця характеристична властивість є прикладом характеристичного предикату, оскільки в ній використовується інша множина (цілі числа), яка відома зі шкільного курсу математики[2].

Приклад 1.3 Характеристичною властивістю для множини натуральних чисел є визначення натурального числа.

Приклад 1.4 Для множини рішень квадратного рівня є коректним застосування терміну «породжуюча функція» оскільки елементи цієї множини з'являються внаслідок виконання дії (в цьому випадку – рішення квадратного рівняння). Крім того, вони також є допустимими значеннями квадратичної функції, що є правим членом цього рівняння. Те ж саме справедливо і для нерівностей та систем рівнянь та нерівностей[3].

[2] Більш детально про предикати можна дізнатись у відповідному розділі цього посібника
[3] Більш детально про породжуючі функції можна дізнатись у розділі «Відповідності» цього посібника

Множини позначають латинськими буквами A, B, C або словами, які обов'язково повинні починатись з великої літери (наприклад Int або Double). Слід відзначити, що на основі аналізу сучасних літературних наукових джерел, відносно множин сформульований наступне важливе правило для позначення множин:

⚠ ***Різні великі літери позначають множини з різними характеристичними властивостями (характеристичні предикати, породжуючі функції тощо). Однаковими літерами позначаються множини з однаковими характеристичними властивостями (характеристичними предикатами, породжуючими функціями тощо)***

Зауважимо, що в контексті цього правила, одна і та сама літера, у якої присутні різні за значеннями верхній або нижній індекс, буде позначати множини з різними характеристичними властивостями. Тобто літери A_1, A_2 позначають множини з різними характеристичними властивостями, але якщо в тексті використовується лише літера A, без позначення індексу, мається на увазі одна і та сама множина, з однією і тією ж характеристичною властивістю.

Приклад 1.5 Автомобільна фірма відкликає деяку кількість своїх автомобілів, певного класу, що мають конструктивний дефект. Ці автомобілі були випущені в певний період часу(наприклад, з липня по вересень минулого року). Введемо в розгляд множини A та B, для чого визначимо характеристичні властивості цих множин. Для множини A характеристичною властивістю буде той факт, що автомобіль цього виробника має конструктивний дефект, для множини B - те, що автомобіль випущений цим виробником з липня по

вересень минулого року. З урахуванням того, що у цих множинах використовуються різні характеристичні властивості, вони позначаються різними літерами.

Приклад 1.6 Ще одним прикладом використання цього правила є той факт, що у шкільному курсі математики цілі числа та натуральні числа мали різні визначення, отже позначались різними буквами (Z та N відповідно).

Звернемо увагу, що розуміння принципів позначення множин відграє важливу роль в розумінні всього подальшого матеріалу цієї книги.

Елементи множин позначаються малими латинськими літерами a , b, c, або словами з латинських літер, які починаються з малої букви (наприклад int, tree, kol). Слід відзначити, що елементами множини можуть бути й інші множини. Тому, позначення тієї чи іншої множини як елемента іншої множини, або як самостійної множини істотно залежить від контексту застосування позначень.

Потужність множини. Пуста множина. Універсум. Введемо у розгляд поняття потужності множини

⚠ ***Потужність множини - це кількість елементів у множині. Потужність нескінченної множини не визначена та зветься континуумом. Потужність множини позначається як $|A|$***

Приналежність елемента множині позначається за допомогою символу $\in$ (читається як «належить»). Відповідно, якщо елемент не належить множині, це позначається $\notin$ (читається як «не належить»). Уведемо у розгляд поняття пустої множини.

⚠ ***Пуста множина – це множина, яка не має елементів***

Пуста множина позначається символом ∅. Для пустої множини поняття приналежності не має значення.

У реальному світі поняття приналежності є досить розповсюдженим.

Приклад 1.7. Сад можна розглядати як деяку множину дерев, яку позначимо $Tree$. Тоді, той факт, що дерево росте в цьому саду можна позначити так: $tree \in Tree$.

Приклад 1.8 Тип «цілі числа» у мові програмування Сі має наступне визначення: «цілі числа з інтервалу [-2 147 483 648, +2 147 483 647]». Відповідно, використовуючи символ приналежності та відомі математичні поняття, можна записати наступним чином $Int = \{i : i \in Z \text{та} i \in [-2\ 147\ 483648; +2147483647]\}$. *Цей запис можна прочитати наступним чином: «множина Int складається з елементів і таких, що (цей зворот позначено символом «:») і належить множині цілих чисел, та належить проміжку* $[-2\ 147\ 483648; +2147483\ 647]$*». Те, що Int множина, в цьому випадку випливає з її позначення, а також використання фігурних дужок при написанні характеристичної властивості.*

Приклад 1.9 Знак приналежності елемента множині досить часто використовується при рішенні задач щодо вирішення систем та сукупностей нерівностей та квадратичних рівнянь. В обох цих випадках зручно використовувати саме символ приналежності.

Питання приналежності чи не приналежності елемента множині породило цікавий парадокс, вперше описаний англійським математиком Бертраном

Расселом (1872 - 1970). Формулювання цього парадоксу базується на тому, що у множині можуть бути будь-які елементи, включно із множинами. Наприклад, усі розглянуті в прикладах множини не утримують себе в якості елементу. Розглянемо множину К усіх множин, які не утримують себе в якості елементу. Чи містить К саме себе в якості елементу? Якщо так, то, за визначенням К, воно не має бути елементом К — суперечність. Якщо ні - то, за визначенням К, воно має бути елементом К — знову суперечність. Ця нерозв'язна суперечність дістало назву парадоксу Рассела.

Цікава історія формулювання цього парадоксу. Він був сформульований Расселом у вигляді такого міркування: «Якийсь цирульник вивісив оголошення на дверях своєї перукарні: «Голю тих та тільки тих мешканців міста, хто не голиться сам«. Виникає питання - хто голить цирульника? Якщо цирульник голиться сам, то він належить безлічі тих мешканців міста, хто голиться сам. Але в оголошенні стверджується, що наш цирульник ніколи не голить тих, хто входить в цю множину. Отже, наш цирульник не може голити самого себе. Якщо ж цирульника голить хто-небудь інший, то він належить до числа тих, хто не голиться сам. Але в оголошенні сказано, що він голить усіх, хто не голиться сам. Отже, ніхто інший не може голити нашого цирульника. Схоже, що його не може голити ніхто»!

Уникнути цього парадоксу допомагає введення поняття універсальної множини (універсуму), якій належать елементи, що входять в хоча б одно з даних множин. Універсальну множину позначатимемо літерою U. Вибір універсуму залежить, як правило, від вирішуваної задачі і необов'язково включає усі об'єкти реального світу.

З точки зору програмного забезпечення, поняття універсуму є спорідненим поняттю значень атрибутів розглядаємих об’єктів з предметної області. Більш точно спорідненість понять універсуму та предметної області буде сфомульоване пізніше, під час розгляду поняття декартова добутку множин.

Поняття підмножини та булеану можна суворо сформулювати лише завдяки тому, що раніше було неформально введені поняття «приналежності», «елемент множини» та «універсум».

⚠ ***Множина B є підмножиною множини A, якщо для будь-якого $b \in B$ справедливе, що $b \in A$***

Множина A у такому випадку є надмножиною множини B. Той факт, що множина B є підмножиною множини A позначається як $B \subset A$ або $B \subseteq A$. Зокрема пуста множина є підмножиною будь-якої множини. При цьому, якщо використовується позначення $B \subset A$, це означає, що випадок рівності множин не розглядається (часто говорять, що «B є власною підмножиною A»). Якщо використовується позначення $B \subseteq A$ (часто говорять, що «B є невласною підмножиною A»), це означає, что допускається випадок того, що $B = A$, тобто множини рівні.

⚠ ***Множина B є рівною множині A в тому випадку, коли вони складаються з одних та тих самих елементів***

Звернемо увагу, що це визначення не вимагає рівності характеристичних властивостей множин A та B. Це означає, що рівні множини можуть бути отримані шляхом використання різних характеристичних властивостей.

Приклад 1.10 Автомобільна фірма відкликає деяку кількість своїх автомобілів, певного класу, що мають конструктивний дефект. Ці автомобілі були випущені в певний період часу (наприклад, з липня по вересень минулого року). Введемо в розгляд множини A та B, для чого визначимо характеристичні

властивості цих множин. Для множини A характеристичною властивістю буде той факт, що автомобіль має конструктивний дефект, для множини B - те, що автомобіль випущений з липня по вересень минулого року. Якщо дефект в конструкцію автомобіля був внесений ще на етапі проектування, та був виявлений вже після запуску автомобіля в серію, то все випущені з липня по вересень автомобілі в цьому випадку будуть із дефектом. З погляду визначення рівності множин, в цьому випадку $A = B$. Проте, множини A та B різні, внаслідок відмінності формулювань їх характеристичних властивостей.

Важливим поняттям, яке буде використовуватись далі є поняття булеану.

⚠ ***Булеан множини — множина всіх підмножин деякої множини.***

Булеан множини А позначається як 2^A, $\mathcal{P}(A)$, $\mathcal{B}(A)$.

Приклад 1.11 Булеаном множини $A = \{a, b, c\}$ є множина $2^A =/ \{\emptyset; \{a\}; \{b\}; \{c\}; \{a; b\}; \{b; c\}; \{a; c\}; \{a; b; c\}\}$.

Приклад 1.12 Булеаном множини всіх студентів деякого факультету є всі можливі варіанти складів академічних груп. Зокрема, це означає, що може існувати група без студентів, оскільки пуста множина в цьому випадку є групою без студентів. Однак, з точки зору реально існуючих факультетів, існування групи без студентів неможливе. Тому множина усіх академічних груп студентів є власною підмножиною булеана студентів.

Приклад 1.13 Булеан множини товарів, які поставляються до деякого супермаркету є множиною варіантів поточного стану складу цього супермаркету. Під поняттям «стан складу супермаркету» в даному випадку

розуміється перелік усіх товарів, які зберігаються у супермаркеті у даний момент часу. При цьому пуста множина є фактом того, що склад пустий, тобто в ньому не має товарів. Отже стан складу у відповідний проміжок часу є елементом цього булеану

Графічне завдання множин. Для графічного позначення елементів множин використовуються діаграми Венна та їх різновид - кола Ейлера. Діаграми Венна – це прямокутні діаграми, які графічно позначають взаємозв'язок між множинами, а саме: чи мають множини спільні елементи, чи включаються одна в одну. Факт того, що множина належить деякому універсуму на діаграмах Венна ілюструється шляхом позначення універсуму прямокутником.

Приклад діаграм Венна наведено на Рисунок 1.1

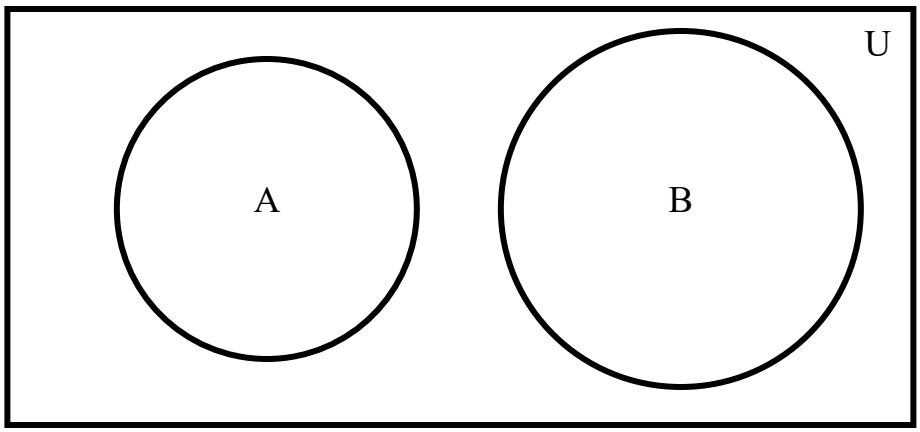


Рисунок 1.1 - Приклад діаграм Венна

На Рисунок 1.1 зображено дві множини, які не мають спільних елементів. Однак вони обидві належать деякій універсальній множині, яка позначена на Рисунок 1.1 прямокутником з буквою U у верхньому правому куті.

У випадку, коли множини мають деякі спільні елементи це позначається на діаграмах Вена, як позначено на Рисунок 1.2

На Рисунок 1.2 перетинання кола у двох точках означає, що множини, які відповідають цим колам, мають спільні елементи. Такий випадок діаграм Венна може виникати, наприклад, у випадку, коли $\mathrm{A} = \{a, b, c\}$ та $\mathrm{B} = \{\mathrm{b}, \mathrm{c}, \mathrm{e}\}$, а універсальна множина U складається з усіх великих і малих літер англійського алфавіту.

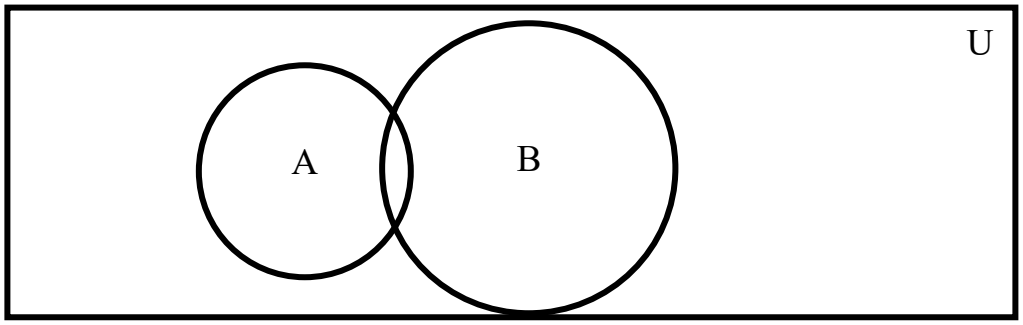


Рисунок 1.2 - Діаграма Вена для множин, які мають спільні елементи

Розглянемо далі приклад діаграм Венна для випадку, коли множина включається одна в одну. Такий випадок наведено на Рисунок 1.3

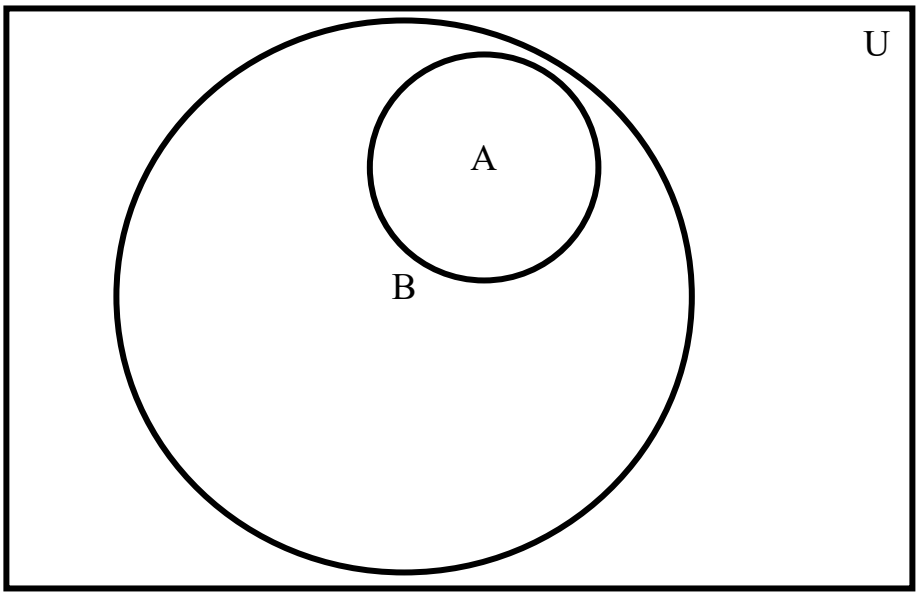


Рисунок 1.3 - Діаграма Вена для випадку $A \subset B$

У випадку, коли множини рівні, це позначається на діаграмах Венна, як показано на Рисунок 1.4

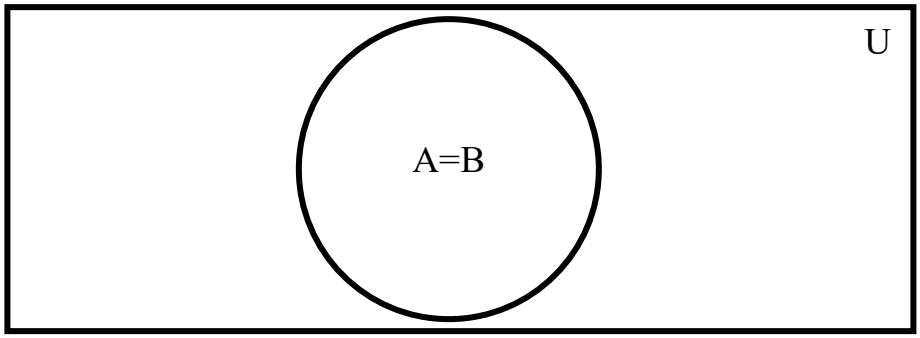


Рисунок 1.4 - Діаграма Венна для випадку рівності множин

Зауважимо, що кожна з діаграм Венна позначає конкретний випадок взаємозв'язку множин. Натомість, формула позначає загальний тип взаємозв'язку. Тому не слід виконувати теоретико-множинні спрощення на діаграмах Венна. Для цього існують аналітичні співвідношення, які будуть розглядатися далі

Використання понять множини, приналежності та включення у розробці програмного забезпечення. Згідно стандартного визначення типу даних[4] – це є множина значень із дозволеними на множині операціями. Таким чином, використання поняття множини дозволяє формально визначити змінну цього типу У Приклад 1.8 було введено тип Int. Відповідно, змінна цього типу може позначатися за допомогою операції приналежності, а саме $i \in Int$. З іншого боку, для множини Int справедливе наступне співвідношення із множиною цілих чисел Z: $Int \subset Z$. Знак включення у цьому виразі означає не лише співвідношення множин між собою, а й той факт, що цілий тип, який визначено у Приклад 1.8 є однією з можливих форм представлення множини цілих чисел у комп'ютерній системі. При цьому використовується символ суворого включення, оскільки множина цілих чисел є нескінченною, а будь-яка програма, яка виконується на комп'ютері, використовує кінцеві множини, у зв'язку із кінцевим розміром структур даних,

[4] За стандартом ISO 2382 «Інформаційні технології – словник»

які використовуються при розробці. Адже навіть динамічні структури даних мають певні обмеження: тип «динамічний рядок» (char*) у операційній системі Windows не може перевищувати двох терабайтів, будь-який інший динамічний тип даних так само обмежений ємністю оперативної пам’яті, дискового простору, простору у хмарному сховищі тощо. Отже надалі варто розрізняти поняття «кінцевої множини із великою кількістю елементів» та поняття «нескінченна множина», тобто множина, кількість елементів у якій невідома.

1.2 Основні операції над множинами та їх властивості

Базові операції над множинами. До базових операцій над множинами відносяться операції об’єднання, перетину та доповнення множин.

⚠ ***Результатом операції об’єднання множин є множина, яка містить елементи, які належать хоча б одній з множин A або B***

Тобто, операція об’єднання має характеристичну властивість, яка проілюстрована наступною формулою

$$C = A \cup B = \{c: c \in A \text{ або } c \in B\} \tag{1.1}$$

Якщо розглядати операцію об’єднання з точки зору діаграм Венна, то результат операції на діаграмі можна позначати штриховкою, як це показано на Рисунок 1.5:

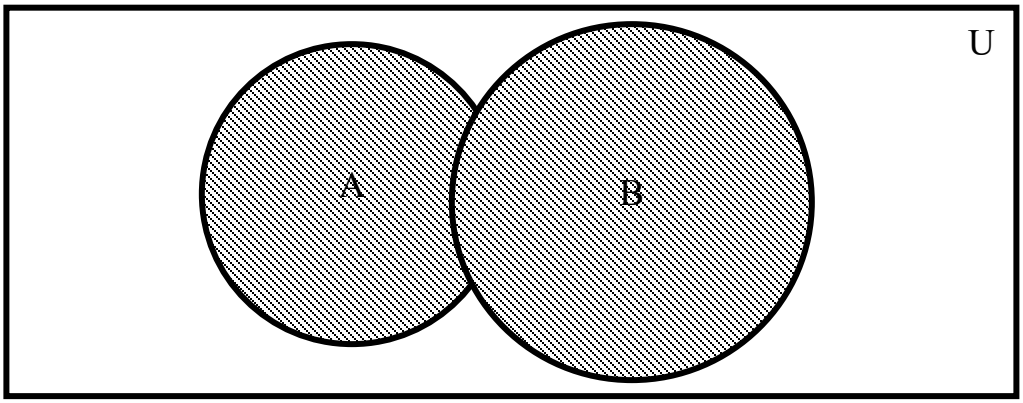


Рисунок 1.5 - Приклад діаграми Венна для операції об'єднання

Слід також зауважити, що для коректного виконання операції об’єднання слід обов’язково задавати універсальну множину. Це не обов’язково має бути формальне завдання, але ця універсальна множина повинна хоча б матися на увазі.

Наведемо деякі приклади операції об’єднання

Приклад 1.14. Нехай задані дві множини $A = a, b, c$ *та* $B = d, e, f$ *то результатом об’єднання буде множина* $A \cup B = a, b, c, d, e, f$.

Приклад 1.15 Результатом об’єднання множин $A = b, c, d$ *та* $B = d, a, k$ *буде множина* $C = A \cup B = a, b, c, d, k$.

Приклад 1.16 Ще одним прикладом виконання теоретико-множинної операції об’єднання, є знаходження рішення сукупності нерівностей. З курсу середньої школи відомо, що рішення сукупності нерівностей – це будь-яке число, яке задовольняє бодай однієї з цих нерівностей. Це визначення відповідає змісту операції об’єднання. Зокрема, рішенням сукупності нерівностей:

$$\left[\begin{array}{l} x - 3 < 0 \\ 2x + 4 \geq 4 \end{array}\right.$$

буде об’єднання двох відкритих інтервалів: $(-\infty; 3] \cup [0; +\infty)$

Приклад 1.17 Якщо розглядати коридор деякого факультету інституту як множину студентів Cor, а аудиторію як іншу множину студентів Aud , то вхід деякої групи студентів також можна позначити через операцію об'єднання, а саме: $Aud = Cor_1 \cup Aud.$ При чому слід зазначати, що $Cor_1 \subseteq Cor$. Також, належність студенту множині Cor_1 означає, що він не тільки знаходиться у коридорі, а й бажає зайти у аудиторію. У даному випадку бажання зайти до аудиторії є додатковою характеристичною властивістю студента, яка допомагає визначити множину Cor_1, одночасно стверджуючи, що елементи цієї множини мають властивість елементів Cor. Саме цей факт позначено символом $Cor_1 \subseteq Cor$

Приклад 1.18 Якщо розглядати сад, як деяку множину дерев $Tree$ (див. Приклад 1.7), то операція об'єднання є еквівалентною посадці у саду нових дерев, де нові дерева так само представляють собою сад, який позначимо $Tree_1$. Тоді посадка дерева математично запишеться таким чином: $Tree = Tree \cup Tree_1$. *Зверніть увагу на дві важливі особливості операції об'єднання. У залежності від того, що розглядається дослідником, як сад: безпосередньо посаджені дерева чи дерева, які посаджені або знаходяться на земельній ділянці саду, операцією об'єднання можна представити також доставку дерев на земельну ділянку (другий випадок розгляду) або доставка буде «схована» у процес посадки (перший випадок розгляду).*

Окремого коментарю заслуговують Приклад 1.15 та Приклад 1.17. У Приклад 1.15 елемент d присутній у множині C один раз, хоча у кожній з множин він є. Це пов'язане з тим, що під час виконання операції об'єднання вважається, що обидві множини, які є аргументами, та множина, яка є результатом належать одному й тому самому універсуму. Належність цього елементу двом підмножинам

одночасно свідчить про наявність одразу двох додаткових характеристичних властивостей відносно характеристичної властивості універсальної множини.

У Приклад 1.17 коридором вважається будь-що, що не є аудиторією, яка розглядається дослідником. У цьому сенсі, на відміну від реального світу, коридором може бути навіть якась інша аудиторія, яка лишилась поза увагою дослідника або ж взагалі вулиця у місті, де розташований ВУЗ. Головною умовою належності до множини Cor_1 є бажання людини зайти у аудиторію. В цьому сенсі час виконання операції об'єднання, яка зазначена у прикладі, може бути значно більшим у порівнянні з її математичним виконанням.

Ми надалі розглядаємо саме математичне виконання будь-якої розглядаємої операції, без аналізу фізичного часу, який необхідно витратити на її виконання. Головною задачею у подальшому є побудова мінімально необхідного математичного опису вирішення задачі, що, у свою чергу, завжди призводить до скорочення часових та ресурсних затрат на вирішення задачі у обчислювальній системі

Зверніть увагу, що у прикладах Приклад 1.17, Приклад 1.18 ілюструється важливий випадок визначення меж розглядаємої предметної області «на місцевості». Таке визначення зветься абстрагуванням та часто виконується за допомогою спеціальних операцій (див. п.)

Визначимо далі формально результат операції перетину множин.

⚠ ***Результатом операції перетину множин A та B є множина, яка складається з елементів, які належать як множині A, так і множині B***

Операція перетину має характеристичну властивість, яка наведена у формулі

$$C = A \cap B = \{c: c \in A \text{ та } c \in B\} \quad (1.2)$$

Відповідно на діаграмах Вена ця операція позначається, як показано на Рисунок 1.6.

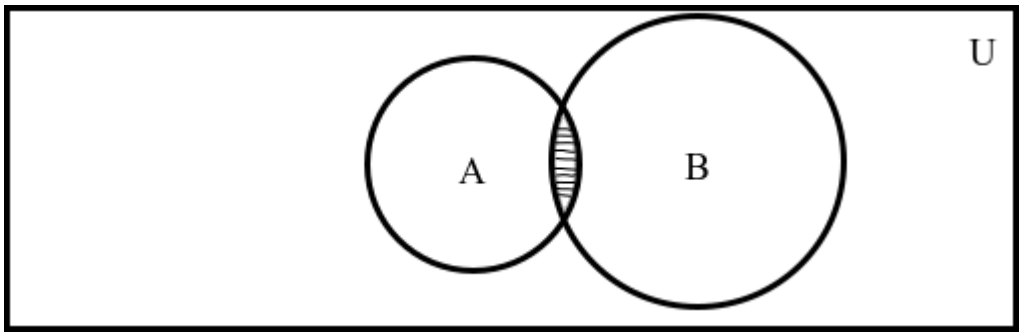


Рисунок 1.6 - Результат операції перетину множин на діаграмі Венна

На Рисунок 1.6 штриховкою зазначено результат операції перетину множин. Розглянемо деякі приклади операції перетину множин

Приклад 1.19 Якщо задані дві множини $A = \{a, b, c\}$ *та* $B = \{b, c, d, e, f\},$ *то результатом їх перетину буде множина* $C = \{b, c\}$.

Приклад 1.20 Результатом перетину множин $A = \{a, b, c\}$ *та* $B = \{d, e, f\}$ *буде пуста множина, оскільки множини* A *та* B *не мають спільних елементів*

Приклад 1.21 Найбільш відомим прикладом з середньої школи прикладом виконання операції перетину множин є рішення будь-якої системи нерівностей. Наприклад, система нерівностей

$$\begin{cases} x + 3 \geq 0 \\ x - 4 < 0 \end{cases}$$

має розв'язок, який записується за допомогою операції перетину множин наступним чином: $(-\infty; 4) \cap [-3; +\infty)$. *Результатом цього перетину є відрізок [-3;4).*

Приклад 1.22 Одним з прикладів використання операції перетину у реальному світі є, наприклад, перерахування викладачем тих осіб, хто має вийти до

дошки. У цьому випадку, множиною A буде множина тих осіб, хто знаходиться у аудиторії, множина B – це множина осіб, яку назвав викладач. Вихід названих осіб до дошки є виконанням операції перетину. Перетин враховує, зокрема, й факт відсутності студента у аудиторії. Якщо деякий студент (позначимо його латинським буквосполученням st), буде відсутній у аудиторії, це можна позначити як $st \notin A$. Якщо, навіть, цього студента покличе викладач (це позначається символом $st \in B$) він не потрапить до результату операції перетину, оскільки він відсутній у множині A. Фактично, відбудеться те, що відбувається й у реальному світі – цей студент не вийде до дошки.

Ситуація, яка описана Приклад 1.22, є значно більш поширеною, аніж здається. Таким саме чином можливо описати, наприклад, визначення присутніх студентів, висадку пасажирів з транспорту, визначення об'єктів або осіб, які підлягають додатковій перевірці, визначення учасників змагань, які проходять до наступного раунду та ін. Отже, операція перетину одночасно моделює багато операцій у реальному світі, що дозволяє казати про універсальність цієї операції з точки зору галузі застосування у рамках моделювання.

Також слід зауважити, що цей приклад ілюструє поетапність виконання операцій. Викладач лише формує множину студентів, шляхом перерахування елементів, а безпосередньо перетин цих множин є вихід цих студентів до дошки. Зауважте, також, що має місце цікавий лінгвістичний збіг: слова «операція» та «перетин» є іменником, породженим від дієслова, а слово «вийти» є безпосередньо дієсловом. В обох випадках, перш за все, мається на увазі якась дія.

Останньою операцією, яка є стандартною для множин, є операція доповнення множин, яка визначається наступним чином

⚠ ***Результатом операції доповнення множин є всі елементи, які не належать множині, яка є аргументом цієї операції***

Зауважимо, що операція доповнення обов'язково вимагає зазначення універсуму, який розглядається під час виконання цієї операції. Тому цю операцію часто іменують «доповнення до універсуму». На діаграмі Венна ця операція позначається так, як показано на Рисунок 1.7

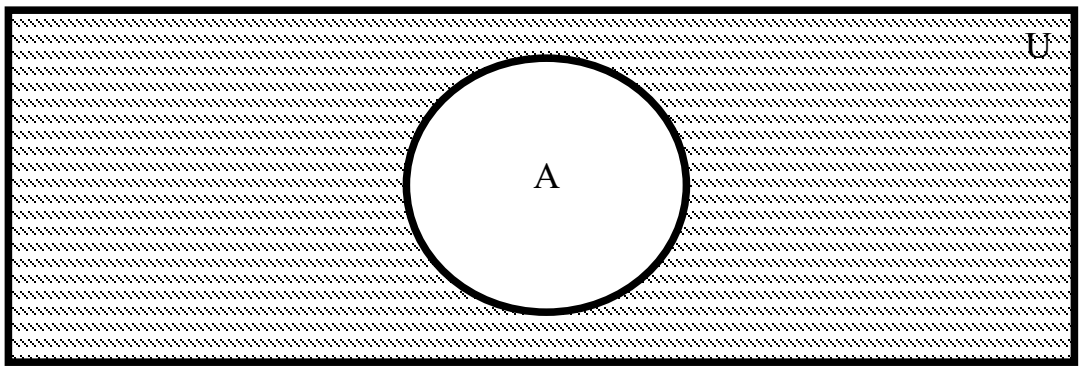


Рисунок 1.7- Діаграма Венна для операції доповнення

На Рисунок 1.7 заштрихована область є результатом операції доповнення для множини A.

Розглянемо деякі приклади операції доповнення

Приклад 1.23. Нехай задана універсальна множина U, яка включає в себе великі та малі літери англійського алфавіту. Якщо задана множина A усіх малих літер англійського алфавіту, її доповненням $\bar{A}$ буде множина усіх великих літер цього алфавіту.

Приклад 1.24. Якщо розглядати студентів, які сидять у аудиторії (див. Приклад 1.17) то доповненням цих студентів можуть бути як студенти, які знаходяться у коридорі, так і будь-які люди, які знаходяться поза межами

аудиторії, але належать області, яка розглядається (універсуму). Якщо ж дивитись на студентів, які знаходяться в аудиторії (див. Приклад 1.22), як на універсум U, тоді, якщо позначити студентів, які викликані до дошки як множину A, тоді всі інші студенти у аудиторії будуть утворювати доповнення до цієї множини $\bar{A}$

З наведеного прикладу випливає похідна операція від операцій перетину та доповнення, яка зветься різницею множин. Ситуація, яка наведена у Приклад 1.22 використовуючи поняття доповнення та різниці, можна формально проілюструвати виразом: $A = U \backslash \bar{A}$. У цьому випадку множина A та її доповнення трактується так само, як у Приклад 1.24.

Формальне визначення операції різниці виглядає наступним чином

⚠ ***Результатом операції різниці множин A та B є множина C, яка складається з елементів, які належать множині A та не належать множині B***

Характеристична властивість операції різниці наведена у формулі:

$$C = A \backslash B = \{c: c \in A \text{ та } c \notin B\} \quad (1.3)$$

Якщо брати до уваги характеристичну властивість перетину множин, яка наведена у формулі $C=A \cap B = \{c: c \in A \text{ та } c \in B\}$ (1.2) та характеристичну властивість доповнення і застосувати її до визначення операції різниці, то можна тотожність, яка дозволяє представити операцію різниці у вигляді основних теоретико-множинних операцій наступним чином: .

$$A \backslash B = A \cap \bar{B} \quad (1.4)$$

Ще однією окремою задачею є виділення елементів, які належать одній і тільки одній з двох множин. Ця операція має назву «симметрична різниця» та визначається наступним чином:

⚠ ***Симетричною різницею двох множин A та B називається множина C, яка містить елементи, що належать або множині A, або множині B, але не їм обом одночасно.***

Формально ця характеристична властивість записується у вигляді наступної формули:

$$\mathrm{C} = \mathrm{A} \div B = \{\mathrm{c}: 1)\ \mathrm{c} \in A \text{ та } c \notin B \text{ або } 2)\ c \notin A \text{ та } \mathrm{c} \in B\} \qquad (1.5)$$

Враховуючи раніше перелічені характеристичні властивості перетину, об'єднання та доповнення, а також визначення симетричної різниці, можна записати наступну тотожність:

$$\mathrm{A} \div B = (A \backslash B) \cup (B \backslash A) \qquad (1.6)$$

Властивості операцій над множинами. Як і у будь-яких математичних операцій, відомих зі шкільного курсу, у операцій над множинами є властивості комутативності (перестановка аргументів операції), асоціативності (зміни порядку обчислення у випадку послідовного виконання однієї та тієї ж самої операції) та дистрибутивності. Розглянемо ці властивості більш детально

Операція комутативності являє собою перестановку аргументів під час виконання операції. Вона властива операціям об'єднання, перетину та симетричної різниці множин, тобто мають місце наступні тотожності:

$$\mathrm{A} \cup B = \mathrm{B} \cup A \tag{1.7}$$

$$\mathrm{A} \cap B = \mathrm{B} \cap A \tag{1.8}$$

$$\mathrm{A} \div B = \mathrm{B} \div A \tag{1.9}$$

Властивість асоціативності є у операцій перетину, об'єднання та симетричної різниці множин. Формально ця властивість записується наступним чином:

$$\mathrm{A} \cap B \cap C = \mathrm{A} \cap (B \cap C) = (A \cap B) \cap C \tag{1.10}$$

$$\mathrm{A} \cup B \cup C = \mathrm{A} \cup (B \cup C) = (A \cup B) \cup C \tag{1.11}$$

$$\mathrm{A} \div B \div C = \mathrm{A} \div (B \div C) = (A \div B) \div C \tag{1.12}$$

Властивість дистрибутивності є між операціями перетину та об'єднання, об'єднання та перетину, а також між операцією перетину по відношенню до симетричної різниці

$$\mathrm{A} \cap (B \cup C) = (A \cap B) \cup (A \cap C) \tag{1.13}$$

$$\mathrm{A} \cup (B \cap C) = (A \cup B) \cap (A \cup C) \tag{1.14}$$

$$\mathrm{A} \cap (B \div C) = (A \cap B) \div (A \cap C) \tag{1.15}$$

Окрім відомих з математики властивостей, операції над множинами мають додаткові властивості, які ми розглянемо нижче.

Операції перетину та об'єднання мають властивість ідемпотентності, яка наведена у наступних двох формулах:

$$\mathrm{A} \cap A = \mathrm{A} \tag{1.16}$$

$$A \cup A = A \tag{1.17}$$

Властивість ідемпотентності можна сформулювати наступним чином

⚠ ***Перетин або об'єднання множин з однаковими характеристичними властивостями є тією ж самою множиною***

Приклад 1.25 Якщо розглядати студентів, які сидять в аудиторії (див. Приклад 1.22), то властивість ідемпотентності є прикладом того, що викладач каже «всім студентам вийти до дошки». У цьому випадку, група фактично переміститься у іншу частину аудиторії (зауважимо, що місцезнаходження студентів в аудиторії наразі немає значення для нас). Отже, це все одно буде множина студентів, які є в аудиторії.

Іншою специфічною властивістю для множин є властивість інвалютивності. Властивість інвалютивності формулюється згідно наступної формули:

$$\bar{\bar{A}} = A \tag{1.18}$$

З точки зору визначення операції доповнення множин, це означає «заперечення не приналежності», звідки, логічно, випливає відношення приналежності множини.

У курсі середньої школи особливу роль по відношенню до арифметичної операції множення відігравали цифри «1» та «0»[5]. Така їх особлива роль призвела до того, що існують поняття «нуля» та «одиниці» по відношенню до будь-якої

[5] Вибір саме операції множення обумвлений тим, що у теорії формальних алгебр, яка є базою для теорії множин, бінарна операція має назву множення, хоча часто не є множенням у, скажімо так, арифметичному сенсі цього слова. Більш детально про теорію формальних алгебр – у другій частині цієї книги

арифметичної операції. Це означає, що такий елемент веде себе так само, як числа «0» або «1» у звичайних математичних операціях.

Для теорії множин такими двома елементами є універсум (U) та пуста множина (∅). По відношенню до операції об'єднання універсум еквівалентний нулю, а пуста множина еквівалетна одиниці. Тобто мають місце наступні дві властивості:

$$U \cup A = U \tag{1.19}$$

$$\emptyset \cup A = A \tag{1.20}$$

У формулі $U \cup A = U$ (1.19) універсум відіграє таку саму роль, як і нуль в операції множення. Тому назва цієї властивості – *властивість нуля об'єднання*. Відповідно, оскільки пуста множина відіграє відносно об'єднання таку ж саму роль, як і одиниця у операції множення, властивість $\emptyset \cup A = A$ (1.20) називається *властивість одиниці об'єднання.*

Розглянемо тепер операцію перетину. По відношенню до неї, для універсуму та пустої множини справедливі наступні властивості

$$U \cap A = A \tag{1.21}$$

$$\emptyset \cap A = \emptyset \tag{1.22}$$

У цих формулах ситуація цілковито протилежна до попередньо розглянутих властивостей одиниці та нуля об'єднання. А саме, універсум веде себе тотожньо до одиниці у арифметичної операції множення. Тому властивість$U \cap A = A$ (1.21) отримала назву *властивість одиниці перетину*. З аналогічних причин властивість $\emptyset \cap A = \emptyset$ (1.22) отримала назву *властивість нуля перетину* .

Для операцій доповнення, симетричної різниці та різниці так само існують специфічні властивості відносно пустої множини та універсума. Для операції доповнення справедливі наступні твердження:

$$\overline{U} = \emptyset \tag{1.23}$$

$$\overline{\emptyset} = \mathrm{U} \tag{1.24}$$

Справедливість цих рівностей випливає з того, що до універсуму входять всі розглядаємі нами елементи. Відповідно, доповненням до всіх елементів буде пуста множина, а доповненням до пустої множини будуть всі елементи, тобто універсум.

Для симетричної різниці справедливі наступні твердження:

$$\mathrm{A} \div \emptyset = \emptyset \div A = \mathrm{A} \tag{1.25}$$

$$\mathrm{A} \div U = \mathrm{U} \div A = \bar{A} \tag{1.26}$$

Справедливість цих тверджень доведемо аналітично. Справедливість лівої та центральної частин формули A÷∅=∅÷ $A = \mathrm{A}$ (1.25) випливає з комутативності симетричної різниці (див. формулу A÷B=B÷ A (1.9)). Доведемо далі аналітично справедливість правої частини формули. У цьому випадку зручніше приводити ліву або центральну частину до правої. Слід зазначити, що аналітичне доведення будь якої формули слід проводити з урахуванням того, що будь яка формула з теорії множин є символічною заміною лівої частини на праву, або навпаки.

Застосовуючи до лівої частини виразу A÷∅=∅÷ $A = \mathrm{A}$ (1.25) A ÷ ∅ визначення симетричної різниці (див формулу A÷B=$A\backslash B$ ∪$(B\backslash A)$ (1.6))

отримаємо наступний вираз: $(A\backslash\emptyset) \cup (\emptyset\backslash A)$. Надалі, для виразів у дужках застосуємо визначення звичайної різниці (див. формулу $A\backslash B = A \cap \bar{B}$ (1.4)). Отримаємо наступний вираз: $\left(A \cap \overline{\emptyset}\right) \cup (\emptyset \cap \bar{A})$. Застосувавши до виразу у других дужках властивість $\emptyset \cap A = \emptyset$ (1.22), отримаємо наступний вираз: $\left(A \cap \overline{\emptyset}\right) \cup$. Враховуючи, що результатом будь-якого теоретико-множинного виразу є множина, будемо вважати вираз у дужках окремою множиною та застосуємо до отриманого виразу рівняння $\emptyset \cup A = A$ (1.20), отримаємо спрощений вираз: $A \cap \overline{\emptyset}$. Застосовуючи формулу $\emptyset$=U (1.24) отримаємо наступну рівність: $A \cap U$. Після застосування рівності $U \cap A = A$ (1.21), з урахуванням властивості $A \cap B$=$B \cap A$ (1.8), отримаємо результат: A, який свідчить про коректність рівності $A \div \emptyset = \emptyset \div A = A$ (1.25).

Справедливість лівої та центральної частин формули $A \div U = U \div A = \bar{A}$ (1.26) випливає з формули $A \div B = B \div A$ (1.9). Для повного доведення твердження, будемо трансформувати вираз $A \div U$. Застосуємо до нього визначення симетричної різниці (див формулу $A \div B = A\backslash B \cup (B\backslash A)$ (1.6)) отримаємо наступний вираз: $(A\backslash U) \cup (U\backslash A)$. Надалі, для виразів у дужках застосуємо визначення звичайної різниці (див. формулу $A\backslash B = A \cap \bar{B}$ (1.4)). Отримаємо наступний вираз: $(A \cap \bar{U}) \cup (U \cap \bar{A})$. Застосувавши до виразу у других дужках властивість $U \cap A = A$ (1.21), отримаємо наступний вираз: $(A \cap \bar{U}) \cup \bar{A}$. Застосувавши до виразу у дужках властивість $U = \emptyset$ (1.23), отримаємо: $(A \cap \emptyset) \cup \bar{A}$. Застосувавши до виразу у дужках рівність $\emptyset \cap A = \emptyset$ (1.22), з урахуванням властивості $A \cap B$=$B \cap A$ (1.8), отримаємо: $\emptyset \cup \bar{A}$. Звідки, застосувавши рівність $\emptyset \cup A = A$ (1.20), отримаємо $\bar{A}$, що дозволяє пересвідчитися у коректності вказаної рівності.

Крім перелічених властивостей для симетричної різниці введемо у розгляд ще дві властивості:

$$A \div A = \emptyset \tag{1.27}$$

$$A \div \bar{A} = U \tag{1.28}$$

Справедливість цих рівностей пропонується довести читачу самостійно

При доведенні будь-яких теоретико-множинних рівностей важливо користуватися дужками та розуміти, що будь-яка рівність у дужках є множиною, яка не є рівною всім іншим членам виразу і у цій множині відмінна від інших членів виразу характеристична властивість. Тому будь-яка зміна порядку дій повинна грунтуватися виключно на властивостях теоретико-множинного виразу. Зазвичай, в усіх випадках теоретико-множинних виразів порядок дій зазначається дужками, особливо у випадках застосування теоретико-множинних операцій різних типів

Для різниці існують наступні співвідношення

$$A \setminus U = \emptyset \tag{1.29}$$

$$A \setminus \emptyset = A \tag{1.30}$$

$$A \setminus A = \emptyset \tag{1.31}$$

$$A \setminus \bar{A} = A \tag{1.32}$$

Справедливість цих тотожностей легко довести за допомогою властивостей $A \setminus B = A \cap \bar{B}$ (1.4), $A \cap A = A$ (1.16), $U \cup A = U$ (1.19), $\emptyset \cap A = \emptyset$ (1.22)

Окрім перелічених, у операцій об’єднання, перетину та доповнення, існують ще декілька специфічних властивостей. До них, зокрема, відноситься властивість поглинання, яка ілюструється наступними виразами.

$$A \cup (A \cap B) = A \tag{1.33}$$

$$A \cap (A \cup B) = A \tag{1.34}$$

Для того, щоб зрозуміти суть цих властивостей, слід розуміти, що головною «одиницею часу» у обчислювальній математиці, обчислювальній техніці та програмуванні є будь-яка операція. Час виконання якої-небудь послідовності дій вимірюється кількістю операцій.

У випадку, коли виконується дія над множиною, результатом дії завжди буде те, що буде бачити спостерігач за цією предметною областю після виконання усіх дій, які позначені у цьому виразі. Пояснимо цю ситуацію більш детально на прикладі.

Приклад 1.26 Уведемо у розгляд універсальну множину U, яка визначається наступною характеристичною властивістю: люди, які є в аудиторії. Надалі будемо розглядати множину $A = \{a: a -$ викладач та студенти в аудиторії$\}$ *та* $B = \{b: b -$ студенти в аудиторії$\}$. *Зауважимо, що з характеристичних властивостей цих множин випливає справедливість наступного твердження* $B \subseteq A$. *Зрозуміло, що всі ці множини належать універсуму, який був визначений раніше. Отже, формула* $\mathrm{A} \cup (A \cap B) = \mathrm{A}$ *(1.33) визначає ситуацію, коли спочатку викладач виходить на перерву, а потім повертається назад. Зауважимо також, що ліва частина цієї тотожності вимагає, щоб до аудиторії повернувся той самий викладач, який виходив звідти на початку виконання дії. Права частина цієї рівності стверджує, що будь-хто, хто загляне в аудиторію вже після завершення перерви, побачить саме множину A. Тобто усі попередні події для нього будуть «невидимі» або «поглинуті». Звідти і назва властивості – властивість поглинання. Формула* $\mathrm{A} \cap (A \cup B) = \mathrm{A}$ *(1.34) ілюструє дещо іншу ситуацію. Вираз у дужках ілюструє факт того, що до аудиторії зайшли студенти, а потім вийшли з неї. Зауважимо, що множина A ілюструє саме тих викладача та студентів, які були в аудиторії на початку обчислення*

теоретико-множинного виразу. Тобто ті студенти, які належать множині B будуть так само «поглинуті» для спостерігача, який зайде вже після виконання усіх дій

Отже, як бачимо з наведеного прикладу, під час виконання теоретико-множинних дій потрібно завжди розуміти принцип співвідношення позначення множини та характеристичної властивості цієї множини, який наведено на стор. 7 цього посібника.

Ще однією важливою властивістю дій над множинами є властивість Де-Моргана, яка ілюструється наступними формулами:

$$\overline{A \cap B} = \bar{A} \cup \bar{B} \tag{1.35}$$

$$\overline{A \cup B} = \bar{A} \cap \bar{B} \tag{1.36}$$

Операція доповнення у лівій частині також відіграє роль дужок, тобто спочатку виконується операція перетину (об'єднання), а вже потім операція доповнення.

Пояснення справедливості цих властивостей так само, як і властивості поглинання, може бути проведено на прикладі

Приклад 1.27 У якості універсальної множини візьмемо множину студентів, яка склала іспит з деякої дисципліни. Будемо розглядати дві множини $A = \{a: a -$ множина студенток групи, яка склала іспит$\}$ *та* $B = \{b: b -$ множина студентів, які склали іспит з оцінкою добре$\}$. *Перетин цих множин (див. ліву частину формули* $A \cap B = \bar{A} \cup \bar{B}$ *(1.35)) є множина студенток, яка склала іспит з оцінкою добре. Відповідно, доповнення до цієї множини буде множина студентів, яка склала іспит з оцінками «задовільно» та «відмінно», а*

також множина хлопців, яка отримала оцінку «добре». Права частина формули $A \cap B = \bar{A} \cup \bar{B}$ *(1.35) відповідно є множиною усіх хлопців у групі, а також множина студентів, яка склала іспит з оцінками «задовільно» та «відмінно». Виходячи з характеристичної властивості об'єднання, приходимо до такого ж самого результату, який є у лівій частині рівності.*

Приклад 1.28 У якості універсальної множини візьмемо множину людей, яка присутня у аудиторії. Будемо також вважати, що в аудиторії представлено дві академічних групи студентів. Позначимо це як множину A*. Ця множина буде складатися зі студентів цих академічних груп. Якщо в аудиторію зайде третя академічна група (позначимо її як множину* B *студентів), то відбудеться операція об'єднання академічних груп. Відповідно, доповнення вже цих трьох груп буде множиною викладачів та інших співробітників, які знаходяться у аудиторії, де знаходиться ця академічна група. Ця послідовність дій відображає ліву частину формули* $A \cup B = \bar{A} \cap \bar{B}$ *(1.36) Такий самий результат можна отримати, якщо спочатку обрати всіх в аудиторії, крім цих двох академічних груп (доповнення множини* A*). Буде отримано множину викладачів, яка знаходиться в цій аудиторії. Потім обрати всіх, хто не є студентами третьої групи. В цьому випадку до відбору потраплять як викладачі та співробітники, так і студенти другої групи. Відповідно перетин цих множин дає той самий результат, що й ліва частина розглянутої формули.*

Спрощення виразів та доказ тотожностей у алгебрі множин. Розглянуті нами властивості допомагають виконувати спрощення теоретико множинних виразів. Надамо деякі рекомендації щодо спрощення теоретико-множинних виразів та приклади такого спрощення.

Оскільки результат спрощення виразу у алгебрі множин є множиною, то рекомендовано кожний математичний вираз, який створено за допомогою теоретико-множинних операцій брати у дужки або враховувати пріорітети теоретико множинних операцій: першою виконується операція доповненння, а далі операції перетину та об'єднання.

Кожне спрощення теоретико множинного виразу є символьною заміною виразів теорії множин за допомогою властивостей теорії множин. Розглянемо декілька прикладів спрощення таких виразів.

Приклад 1.29 Спростити вираз $(\overline{A \cap B}) \cup B$. *Перш за все, для того, щоб отримати можливість працювати під час спрощення з множинами* A *та* B *незалежно одне від одного, скористуємося властивістю Де-Моргана у вигляді формули* $\overline{A \cap B} = \bar{A} \cup \bar{B}$ *(1.35). Отримаємо наступний вираз:* $(\bar{A} \cup \bar{B}) \cup B$. *Далі бачимо, що всі бінарні операції у виразі однакові та є операціями об'єднання множин. Тому, надалі скористуємося властивістю асоціативності у вигляді формули* $A \cup B \cup C = A \cup (B \cup C) = (A \cup B) \cup C$ *(1.11), змінивши положення дужок у виразі наступним чином:* $A \cup (\bar{B} \cup B)$. *Далі неважко побачити, що в силу визначення операції доповнення, вираз у дужках дорівнює універсальній множині* U. *Тоді отримаємо наступний вираз:* $A \cup U$. *Цей вираз є лівою частиною виразу* $U \cup A = U$ *(1.19), отже маємо фінальну відповідь:* U. *Це означає, що справедлива рівність:* $(\overline{A \cap B}) \cup B = U$.

Як бачимо, під час розгляду процесу спрощення, всі раніше наведені тотожності для перетворення теоретико-множинних виразів використовувались як своєрідні шаблони для трансформації виразів. Такий самий підхід використовувався і під час трансформації математичних виразів у курсі середньої школи. Це

пов’язане з тим, що будь-яка математична рівність є, насамперед, графічною рівністю [3]. Це означає, що кожен з цих виразів задає одну й ту саму функцію.

Також, під час доведення цього прикладу було коротко доведено справедливість наступної властивості теорії множин:

$$\mathrm{A} \cup \bar{A} = \mathrm{U} \tag{1.37}$$

Так само легко переконатися у справедливості наступної властивості:

$$\mathrm{A} \cap \bar{A} = \emptyset \tag{1.38}$$

Зауважимо, що дві останні формули можуть бути отримані одна з одної за дуже цікавим принципом, а саме: знак об’єднання змінюється на знак перетину, знак перетину змінюється на знак об’єднання, універсум змінюється на пусту множину а пуста множина на універсум. Цей принцип має назву принцип двоїстості[6] та є справедливим для будь-якого виразу з теорії множин (у цьому читачеві пропонується переконатися самостійно). Зауважимо також, що заміна пустої множини на універсум та універсуму на пусту множину діє у принципу двоїстості лише для цих двох понять. Для будь яких довільних множин заміна множини на її доповнення не є коректним.

Розглянемо більш складні приклади спрощення теоретико-множинних виразів:

Приклад 1.30 Спростимо вираз $(C \backslash D) \div C$. *Скористаємося визначенням симетричної різниці та врахуємо, що вираз, який стоїть у дужках, є множиною.*

[6] Більш детально про цей принцип та його формальне математичне визначення – далі у книзі

Отримаємо наступний вираз: $\left((\mathrm{C}\backslash D)\backslash C\right) \cup \left(C\backslash(C\backslash D)\right)$*. Тепер окремо спростимо кожний з виразів у дужках.*

До виразу $(\mathrm{C}\backslash D)\backslash C$ *застосуємо визначення різниці та запишемо вираз наступним чином:* $(\mathrm{C} \cap \bar{D}) \cap \bar{C}$*. Надалі, використаємо властивість асоціативності:*$C \cap (\bar{D} \cap \bar{C})$*. У дужках після цього використаємо властивість комутативності:* $C \cap (\bar{C} \cap \bar{D})$*. Після цього знов використаємо властивість асоціативності:* $(\mathrm{C} \cap \bar{\mathrm{C}}) \cap \bar{D}$*. Застосуємо до виразу у дужках властивість* $\mathrm{A} \cap \bar{A} = \emptyset$ *(1.38). Отримаємо наступний результат:* $\emptyset \cap \bar{D}$*. Оскільки* $\bar{D}$ *є такою самою множиною, як і будь-яка інша множина, то отриманий вираз відповідає лівій частині рівності* $\emptyset \cap A = \emptyset$ *(1.22). Отже кінцевим результатом цього виразу буде пуста множина* $\emptyset$*.*

Відносно виразу $C\backslash(C\backslash D)$ *так само, застосуємо спочатку визначення звичайної різниці. Отримаємо наступний вираз:* $C \cap \left(\overline{\mathrm{C} \cap \bar{D}}\right)$*. Далі, до дужок, над якими виконується операція доповнення застосуємо властивість Де-Моргана* $\overline{A \cap B} = \bar{A} \cup \bar{B}$ *(1.35):* $C \cap \left(\bar{\mathrm{C}} \cup \bar{\bar{D}}\right)$*. Після цього до множини* $\bar{\bar{D}}$ *застосуємо властивість інвалютивності:* $C \cap (\bar{\mathrm{C}} \cup D)$*. До отриманого виразу застосуємо властивість дистрибутивності перетину відносно об'єднання* $A \cap B \cup C = (A \cap B) \cup (A \cap C)$ *(1.13). Отримаємо:* $(C \cap \bar{C}) \cup (C \cap D)$*. Застосувавши знов у перших дужках властивість* $\mathrm{A} \cap \bar{A} = \emptyset$ *(1.38) отримаємо наступний результат:* $\emptyset \cup (C \cap D)$*. Застосувавши до отриманого результату властивість* $\emptyset \cup A = \mathrm{A}$ *(1.20) остаточно будемо мати:* $C \cap D$*.*

Таким чином, початковий вираз запишеться у вигляді $\emptyset \cup (C \cap D)$*. Знов врахувавши властивість* $\emptyset \cup A =$ A *(1.20) остаточно будемо мати відповідь:* $C \cap D$*. Отже, має місце наступна рівність:* $(C\backslash D) \div C = \mathrm{C} \cap D$

Приклад 1.30 додатково ілюструє ще одну особливість теоретико-множинних спрощень. А саме – послідовне застосування декількох властивостей для отримання коректної лівої або правої частини відповідного рівняння. Під коректним застосуванням відповідного рівняння мається на увазі відповідність якоїсь частини виразу відповідній частині рівняння властивості. При чому, у рівнянні властивості можуть бути використані інші літери щодо позначення множин

Розглянемо далі приклад доказу теоретико-множинних тотожностей. На відміну від чисельних тотожностей, цей тип тотожностей має відмінність, а саме: цей тип тотожностей не має інструменту для переносу членів через знак рівності для спрощення виразів. В усьому іншому доказ цього типу тотожностей нічим не відрізняється від доказу звичайних математичних тотожностей. Можна приводити ліву частину до правої, можна праву до лівої або синхронно обидві частини рівності до отримання вірної рівності.

Наприклад, тотожність $(C \backslash D) \div C = \complement \cap D$ є вірною. Це легко довести, спрощуючи ліву частину рівності (див. Приклад 1.30).

1.3 Застосування теорії множин та операцій над множинами у програмній інженерії

Опис значень типів даних у мовах програмування. Раніше у цьому розділі вже було сказано, що множини позначаються великими латинськими літерами. Але, у сучасних літературних джерелах досить часто можна зустріти позначення множин словами, що починаються з великої літери. Якщо застосовувати цей підхід до мов програмування, то кожний тип даних можна представити за допомогою множини.

Приклад 1.31. Розглянемо формальний опис цілого типу даних. У сучасних середовищах програмування на мовах C/C++ цей тип знаходиться у інтервалі значень [-2 147 483 648, +2 147 483 647] та містить суто цілі числа. Цей факт можна записати у вигляді множини із наступною характеристичною властивістю: $Int = \{i : i \in \mathbb{Z} \text{ та } i \in [-2\,147\,483\,648, +2\,147\,483\,647]\}$. *Надалі змінна цілого типу може бути описана математично наступним чином* $i \in Int$.

Порівняємо між собою запис з Приклад 1.31 та з мови програмування Сі та Java

Таблиця 1.1 - Порівняльна таблиця варіантів описів типів змінних

	Ім'я змінної	Наявність знаку приналежності	Тип змінної (як множина значень[7])
Математичний опис	Перед типом, починається з літери	Зазначається явно	Зазначається явно
Мова програмування Сі	Після типу, починається з літери	Немає, але передбачається	
Мова програмування Java			

Аналізуючи цю таблицю можна зробити висновок про те, що визначення змінної у типізованих мовах програмування лише несуттєво відрізняються від математичного опису.

Якщо ж казати про мови із динамічною типізацією, то єдиним типом там буде перелік усіх можливих значень змінної. Відповідно, множина значень буде

[7] Суворо кажучи, відповідно до стандартного визначення (див. ISO 2382:2015), тип даних визначається не зовсім, як множина значень. Але для даного випадку це несуттєво

містити усі можливі значення змінної. Окрім цього, у процесу динамічної типізації є власні особливості математичного опису, які будуть розглянуті пізніше.

Особливості використання операції включення. Якщо розглянути множину Int, яка наведена у Приклад 1.31, то можна побачити, що для неї справедливе співвідношення $\mathrm{Int} \subset Z$. З математичної точки зору це є операція включення. Але, якщо розглядати це з точки зору програмного забезпечення, зокрема того факту, що множина Int містить значення цілого типу, можна також сказати, що відношення включення є математичним описом дії під назвою «розробка» по відношенню до програмного забезпечення. У цьому випадку – розробка (а точніше – вибір) типів даних для програмного забезпечення.

Зверніть увагу, що відношення між множинами у цьому випадку позначає якусь дію. Це є справедливим для будь-яких множин. Зокрема, якщо розглядати множину студентів, які знаходяться у аудиторії, то будь-яка підмножина цієї множини так само буде знаходитись у цій аудиторії. Зрозуміло, що множина та підмножина поєднуються співвідношенням «включення». У той же час, у реальному світі та природній мові це позначається дієсловом «знаходитись», тобто виконанням дії.

Таке співпадіння поняття включення та дій, які виконуються у реальному світі не є випадковим. У подальшому, у цій книзі, буде показано співвідношення поняття дії, суворого математичного поняття «відношення», а також суворого математичного поняття операції.

Особливості використання поняття універсуму та булеану. Раніше (див. п. 1.1) було розглянуто поняття універсальної множини, або коротше – універсуму та зазначено, що ця множина містить усі елементи, які можуть бути у множинах, якими оперує дослідник.

Звернемо увагу, що далеко не всі математичні дії мають назву саме «операції». Наприклад, сінус, косінус, логарифм практично всюди (як у математичній літературі, так і у програмуванні) іменуються функціями. Така різниця пов'язана із тим, що у випадку операції аргументи та результат належать одній і тій самій множині (у розумінні кількості елементів, їх властивостей та змісту). І навпаки, якщо ми кажемо про функції, то їх аргументи та результат належать різним множинам. Найпростішим прикладом цього випадку є функція сінусу радіанної міри кута. Множина її аргументів – раціональні числа, множина її значень – інтервал $[-1; 1]$. Протилежно – аргументи та результат операції «додавання» належать одній та тій самій множині

Зауважимо, що операція «ділення» у математиці практично ніколи не розглядається як самостійна операція. Замість цього використовується поняття раціонального числа та множення на раціональне число. Це, зокрема, пов'язане і з тим, що відома зі школи арифметична дія «ділення» практично не використовується у математичній літературі у вигляді саме операції.

Так само, теоретико-множинні дії «об'єднання», «перетин» та інші є саме операціями. Оскільки їх аргументи та результат належать одній та тій самій множині – булеану. У якості вихідної множини для його побудови використовується інша множина – універсум. Таким чином, будь-який результат теоретико-множинної операції буде належати булеану цієї множини.

Особливості використання теоретико-множинних операцій. Так само, як і використання операції включення, використання теоретико множинних операцій під час моделювання предметних областей вимагає розуміння принципів їх функціонування. Розглянемо приклад моделювання за допомогою теоретико-множинних операцій.

Приклад 1.32. Необхідно промоделювати дію «Посадка пасажирів в автобус». Якщо ми будемо розглядати автобус, то він є складним об'єктом, який містить багато різних атрибутів: номер двигуна, прізвище водія, державний реєстраційний номер, кількість пасажирів, яке може перевести автобус тощо. У випадку посадки пасажирів в автобус нас буде цікавити тільки кількість пасажирів, які можуть сісти до автобуса або вже знаходяться у ньому, та їх перелік. Саме це ми будемо використовувати у якості універсальної множини $U = \{p_1; p_2; p_3; \dots; p_n\}$ *. Пасажиромісткість салону у цьому випадку отримується як потужність множини* U*. Отже, множина* U *у цьому випадку є кінцевою множиною та має місце рівність* $n = |U|$*, де* n *– максимально можливий індекс елементу множини* U*. Відповідно, далі будемо розглядати булеан цієї множини. Це дозволяє використовувати теоретико-множинні операції для опису посадки пасажирів в автобус, а саме – теоретико-множинну операцію об'єднання. Першим аргументом буде слугувати салон автобуса, який щойно під'їхав до зупинки, а другим аргументом – множина пасажирів, яка стоїть на зупинці. Обидві ці множини належать до розглядаємого нами універсуму.*

Слід зазначити, що у зазначеному прикладі, завдяки використанню саме теорії множин, є можливість позначити пустий салон автобуса за допомогою пустої множини. І навпаки, неможливість виконання чергової посадки пасажирів, якщо маршрутка вщент заповнена, досягається за рахунок того, що індекси для пасажирів, які стоять на зупинці будуть вищими, ніж потужність множини U, тобто результат буде неприпустимим.

Зважте на останній зворот тексту «результат буде неприпустимим». Так само, згідно правил пасажирських перевезень, є неприпустимим перевантаження автобусу водієм. У випадку порушення можливо отримати неприпустимий результат: аварія, несправність транспортного засобу тощо

Так само, розглянутий приклад допускає посадку будь-якої кількості пасажирів у маршрутку, включно із ситуацією, що із зупинки не сяде жодного пасажиру.

Ще однією важливою особливістю є той факт, що ця модель не розрізняє два різних автобуса з однаковою кількістю пасажирів. Для цього потрібно більша кількість інформація. Шляхи формального опису такої інформації описані у другому розділі цієї книги.

У розглянутому прикладі так само можна змоделювати дію «вийти з автобуса». Це можна зробити за допомогою однієї з двох теоретико-множинних операцій: перетину або різниці. Різниця при цьому буде лише у другому аргументі: у випадку застосування операції перетину, другий аргумент буде зазначати пасажирів, які мають лишитися у салоні автобуса, а у випадку застосування операції різниці, другим аргументом буде виступати множина пасажирів, яка повинна покинути автобус.

Особливості використання теоретико-множинних операцій. Іншим прикладом використання теорії множин може бути побудова умов для вирішення подальшого напряму виконання дій у програмі. Розглянемо це на прикладі.

Приклад 1.33. Необхідно вирішити за допомогою комп'ютера та мови програмування, яка обирається довільно наступну задачу: визначити приналежність точок відповідній області на координатній прямій. Область 1 має рівняння $x > 3$ *область 2 має рівняння* $3x < 1$*. Відображення обох областей на координатному відрізку відображено на рис.*

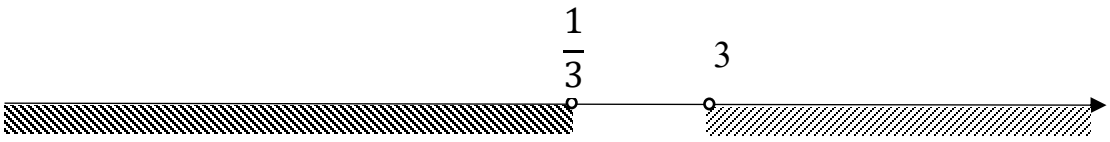


Рисунок 1.8 - Області на координатній прямій

Для поданих нерівностей можливо два випадки їх композиції у єдину задачу: система нерівностей та сукупність нерівностей. У випадку системи нерівностей необхідно, щоб значення x та y задовольняли обом нерівностям. Система нерівностей записується за допомогою фігурних дужок, які з'єднують обидві нерівності. Для нерівностей у цьому прикладі запис буде мати наступний вигляд:

$$\begin{cases} x > 3 \\ x < \frac{1}{3} \end{cases}$$

Введемо у розгляд множину $A = \{x : x > 3\}$ та $B = \left\{x : x < \frac{1}{3}\right\}$. Рішенням будь-якої системи нерівностей є перетин множин $A \cap B$, який у даному випадку є пустою множиною (що підтверджується Рисунок 1.8, де заштриховані області не перетинаються). Отже, дана система не має рішень. Одночасно, характеристична властивість цього перетину має вид $A \cap B = \left\{x : x > 3 \text{ та } x < \frac{1}{3}\right\}$. Цей логічний вираз буде завжди хибним, але в умовному операторі будь-якої мови програмування слід використовувати логічний сполучник «and», який буде позначатися відповідним ключовим словом або символом, який передбачено у документації до мови програмування. Але, в силу постійної логічної

хибності виразу, істинна частина такого умовного оператора ніколи не буде виконана.

Іншим випадком є сукупність цих же нерівностей. В цьому випадку, запис задається наступним чином:

$$\begin{bmatrix} x > 3 \\ x < \frac{1}{3} \end{bmatrix}$$

У випадку сукупності нерівностей, їх рішенням буде об'єднання множин $A \cup B = \left\{x: x > 3 \text{ або } x < \frac{1}{3}\right\}$ *Зрозуміло, що у цій множині будуть міститися елементи. Для виділення цих елементів слід з'єднати в умовному операторі умови* $x > 3$ *та* $x < \frac{1}{3}$ *логічним сполучником «or», використавши його позначення відповідно до документації до мови програмування.*

Приклад 1.33 ілюструє декілька математичних аспектів та аспектів мов програмування. З точки зору математики, він узагальнює співвідношення понять «система нерівностей», «перетин множин», «геометричний перетин», «сукупність нерівностей», «об'єднання множин», «геометричне об'єднання».

Рішенням системи нерівностей буде перетин множин, характеристичними властивостями яких є нерівності члени. З цього випливає що, щоб визначити програмно приналежність точок рішенню системи нерівностей слід використовувати логічний сполучник «та», а також перевіряти результат на пусту множину, для того, щоб умовний оператор мав сенс.

Рішенням сукупності нерівностей буде об'єднання множин, характеристичними властивостями яких будуть нерівності члени. Щоб програмно визначити приналежність точки рішенню сукупності нерівностей, слід застосовувати поєднання цих нерівностей логічним сполучником «або», а також перевіряти

результат на універсальну множину, оскільки у цьому випадку умовний оператор не буде мати сенсу[8].

1.4 Завдання до розділу 1

Завдання 1.1. Навести булеан для наступних множин

а) {1,2,3}

б) {a,b}

в) {b,c,t}

Завдання 1.2. Визначте серед наведених понять, що буде позначатися як множина, а що – як булеан цієї множини

а)склад товарів та товар

б) салон пасажирського автобусу та пасажир

в) зупинка пасажирського транспорту та пасажир

Завдання 1.3. Вкажіть справделиві тотожності

а) $\{a,b\} \subseteq \{a,b,c,d\}$

б) $\{a,b\} \subseteq \{b,c,d\}$

в) $\mathbb{Z} \subseteq \{1,2\}$

г) $\{1,2,3\} \subseteq \mathbb{Z}$

Завдання 1.4. Споростити вираз

а) $\big(A \cap (\bar{A} \cup B)\big) \backslash C$

б) $\big((B \div C) \backslash C\big) \cup B$

в) $A \backslash B \backslash A$

г) $A \cup (\overline{B \backslash A})$

Завдання 1.5. Перевірте справедливість тотожностей

[8] У розділі «Мінімізація булевих функцій» цей принцип отримає подальший розвиток

а) $A \cap B = A(B \backslash A)$

б) $(A \cup B) \backslash A = A \cup (B \div C)$

в) $A \cup B = A \cup (B \cap A)$

г) $A \cap B = A \backslash (\bar{A} \cap \bar{B})$

Завдання 1.6. Наведіть приклади підмножин для наступних множин

а) $\{a, b, c, d, e\}$

б) $\{1,3,4,8\}$

в) $\{c, d, e\}$

г) $\{1,2,3,4,5\}$

2 ДЕКАРТОВИЙ ДОБУТОК МНОЖИН. ВІДПОВІДНОСТІ ТА ВІДНОШЕННЯ

2.1 Декартовий добуток множин

Поняття декартова добутку множин. Апарат теорії множин є дуже ефективним з точки зору засобів моделювання. Зокрема тому, що множина є, фактично, еквівалентом поняття «класу», а елемент множини еквівалентом поняття об'єкту. Але не менш важливим є знання про структуру об'єктів. Головною проблемою в цьому випадку є те, що структура вимагає впорядкованості. Практично всюди при описі структур використовуються слова, які, так чи інакше містять опис структури. Наприклад, при описі програм

Для вирішення цієї проблеми у математиці було введено поняття декартова добутку множин.

⚠ ***Декартовим добутком множин $A_1 \times A_2 \times \ldots \times A_n$ є множина B яка складається з усіх векторів $(a_1, a_2, a_3, \ldots, a_n)$ для яких обов'язково виконується умова $a_1 \in A_1$, $a_2 \in A_2$, …, $a_n \in A_n$***

Приклад 2.1. Найпростішим прикладом декартова добутку є відома декартова система координат. У залежності від того, скільки осей координат буде використано та які числа будуть використані у якості значень, справедливі, наприклад, такі записи: $Z \times Z$ (двовимірна система координат, на осях знаходяться цілі числа), $Q \times Q \times Q$ (тривимірна система координат, на осях дійсні числа), $R \times R \times R$ (тривимірна система координат, на осях раціональні числа) тощо.

Синонімами поняття вектору з наведеного твердження є кортеж або впорядкована множина.

Поняття декартова добутку, таким чином, є узагальненням відомих з курсу середньої школи координат (вимірів) на будь який тип об'єктів, включно із об'єктами реального світу.

Поняття вектору, як такого, у курсі середньої школи асоціюється із направленим відрізком, який має чітко визначений початок і кінець. Фактично, геометричний вектор є такою самою впорядкованою множиною геометричних точок на прямій, площині або у просторі.

У літературі можна зустріти спеціальне позначення для двоелементного вектору $(a; b)$ – впорядкована пара, а також для двоелементної множини $\{a, b\}$ – невпорядкована пара. Використання цих назв обумовлено особливістю викладання та жодним чином не обмежує властивості цих об'єктів як двоелементного вектору або двоелементної множини

Приклад 2.2 Виконаємо дію зі знаходження декартового добутку множин $A = \{a, b, c\}$ *та* $B = \{e, f\}$*. Результат декартова добутку має наступний вигляд:* $C = A \times B = \{(a, e); (a, f); (b, e); (b, f); (c, e); (c, f)\}$*. Зверніть увагу, що декартів добуток* $B \times A$ *буде виглядати по іншому:* $D = B \times A = \{(e; a); (e; b); (e; c); (f; a); (f; b); (f; c)\}$*. Вочевидь, ці дві множини складаються з різних елементів, отже не є рівними. Це ілюструє, що декартів добуток загалом не є комутативним.*

Поняття рівності двох векторів мають додаткові обмеження, у порівнянні з множинами. Якщо для рівності множин достатньо, щоб вони складалися з одних та тих самих елементів, то для рівності векторів необхідно, щоб, крім цього, ці елементи слідували в одному та тому самому порядку.

Приклад 2.3 Кортежі $(a; b)$ *та* $(b; a)$ *не є рівними, бо хоча вони і складаються з одних та тих самих елементів порядок їх слідування різний.*

Приклад 2.4 Множини $\{a; b\}$ *та* $\{b; a\}$ *є рівними, бо вони складаються з одних та тих самих елементів. Порядок слідування елементів у даному випадку неважливий.*

Приклад 2.5 Кортежі $(a; b)$ *та* $(a; c)$ *не є рівними, бо вони складаються з різних елементів. Щодо порядку слідування елементів можна сказати наступне: елементи за першими позиціями рівні, за другими позиціями – не рівні.*

Приклад 2.6 Кортежі $(a; b)$ *та* $(a; b)$ *є рівними, бо вони складаються з одних та мають однаковий порядок слідування .*

Зауважимо, що кортежі $(a; b; c)$ та $(a; b)$ взагалі не можна порівнювати, тому що вони мають різну кількість елементів (різну довжину)

⚠ ***Кількість елементів кортежу – довжина кортежу***

Властивості декартова добутку. Як вже було зазначено раніше, декартів добуток множин не є комутативним:

$$\mathrm{A} \times B \neq B \times A \tag{2.1}$$

Декартів добуток є дистрибутивним відносно операцій перетину та об'єднання.

$$\mathrm{A} \times (B \cap C) = (A \times B) \cap (A \times C) \tag{2.2}$$

$$\mathrm{A} \times (B \cap C) = (A \times B) \cap (A \times C) \tag{2.3}$$

Найпростіший спосіб пересвідчитися у справедливості цих формул – згадати, що будь-який теоретико-множинний вираз у дужках є, насамперед, множиною, так само, як і члени виразу. Отже, не має значення, яким саме чином виконувати дії, зазначені у формулах $A \times (B \cap C) = (A \times B) \cap (A \times C)$ (2.2) та $A \times (B \cap C) = (A \times B) \cap (A \times C)$ (2.3). Результат буде один та той самий.

Найбільш дискусійним є питання стосовно того, чи є декартів добуток асоціативним. У ранніх працях з дискретної математики стверджувалося, що декартів добуток є асоціативним. З точки зору класичної математики він так само є асоціативним. Але застосування декартова добутку не обмежується лише класичною математикою. Декартів добуток використовується у неявному випадку досить широко у інформаційних технологіях, програмуванні та методах штучного інтелекту. Досвід використання поняття декартова добутку у цих галузях свідчить про те, що декартів добуток, у загальному випадку використання, не є асоціативним [1,2].

$$A \times B \times C \neq (A \times B) \times C \neq A \times (B \times C) \tag{2.4}$$

По відношенню до порожньої множини, декартів добуток має наступну властивість:

$$A \times \emptyset = \emptyset \times A = \emptyset \tag{2.5}$$

Більш детально про сфери та особливості застосування декартова добутку у наступних розділах цієї книги.

З курсу математики середньої школи відомо поняття проекції точки на координатні осі та проекція вектору на координатні осі. В обох випадках це є, фактично, відповідна координата. Узагальнимо це відоме поняття проекції шляхом введення у розгляд узагальненого поняття проекції кортежу (вектору, n-ки)

⚠ ***і-я проекція кортежу – його і-ий компонент.***

Значення і при цьому пробігає значення від 1 до довжини кортежу.

Застосування операції декартова добутку у розробці програмного забезпечення. Одним з найпоширеніших застосувань декартова добутку є моделювання структури об'єктів реального світу. Розглянемо це на прикладі

Приклад 2.7. Раніше нами був розглянутий приклад моделювання пасажирського автобуса за допомогою теорії множин (див. Приклад 1.32). У цьому прикладі ми змогли промоделювати лише салон автобусу як булеан множини пасажирів. Але будь-який пасажирський автобус, окрім пасажирського салону має початкову та кінцеву зупинки маршруту, державний реєстраційний номер та водія, за яким закріплений цей автобус. Уведемо у розгляд необхідні множини.

Нехай множина $St = St_1, St_2, \dots, St_n$ *– множина зупинок автобусу. Уведемо у розгляд множину кінцевих зупинок автобусу* St_e *та застосуємо до неї відповідні обмеження:* $St_e \subset St$, $|St_e| = 2$. *Також введемо у розгляд множину водіїв автобусів* D. *Надалі ми (поки що) не будемо деталізувати інформацію про водія, обмежуючись лише позначенням елементу множини (подібно до того, як це було зроблено у Приклад 1.32 по відношенню до пасажирів):* $D = \{d_1, d_2, \dots, d_n\}$ $D = d : d -$ водійавтобусу. *Так само, не будемо, поки що, деталізувати державний реєстраційний номер транспортного засобу. Для позначення множини державних реєстраційних номерів будемо використовувати поняття* $Num = \{num : num -$ державний реєстраційний номер$\}$. *Множину пасажирів будемо позначати буквою* P. *У загальному вигляді маршрутний автобус може бути представлений наступним декатртовим добутком.:*

$$\text{Bus} = St_e \times D \times \text{Num} \times \mathcal{P}(P)$$

де St_e – множина кінцевих зупинок автобусу, D – водій автобусу, Num – державний реєстраційний номер автобусу, $\mathcal{P}(P)$ – булеан пасажирів.

З іншого боку, зупинка має назву та множину пасажирів, які стоять на цій зупинці. У цьому випадку зупинка буде описуватись наступним чином:

$$\text{Station} = \text{St} \times \mathcal{P}(P)$$

де St – множина зупинок автобусу, $\mathcal{P}(P)$ – булеан множини пасажирів, які стоять на зупинці.

Отже, посадка пасажирів в автобус на деякій зупинці буде мати наступний вигляд:

$$\text{пр}_4 Bus = \text{пр}_4 Bus \cup \text{пр}_2 Station$$

У цьому випадку мається на увазі, що пасажирами на зупинці вважаються всі люди, які бажають сісти саме в цій автобус.

Звернемо увагу, що наразі в нас не має жодних засобів, які б дозволяли, наприклад, змінити кінцеву зупинку автобусу або, скажімо, водія. Ці можливості можуть забезпечити такі математичні об’єкти, як відповідності, відношення та предикати.

2.2 Відповідності

Визначення відповідностей. Досить часто у задачі виникає ситуація, що не всі елементи декартова добутку є допустимими для цієї задачі, або дійсно мають розглядатися під час її вирішення. Розглянемо цю ситуацію на прикладах

Найпростішим прикладом є графік будь-якої функції, яка вивчалася під час курсу середньої школи. Зокрема, якщо розглядати рівняння прямої на площині і врахувати, що у геометрії є аксіома приналежності або неприналежності точок деякій прямій, то ця аксіома відіграє роль характеристичної властивості підмножини декартова добутку $Q \times Q$.

Інший варіант, коли використовується лише частина елементів декартова добутку, описана у прикладі

Приклад 2.8 Уведемо у розгляд множину громодян деякої країни $Cit = \{cit: \text{громадянин країни}\}$ *та множину посвідчень особи цієї ж країни* $Pass = \{pas: \text{посвідчення особи}\}$. *Уведемо у розгляд множину* C, *яка визначається аналітично як декартів добуток цих множин* $C = Cit \times Pass$. *Зрозуміло, що, оскільки в реальному світі громадянин може мати лише єдине посвідчення, яке підтверджує його громадянство, то ми повинні розглядати не всю множину* C, *а лише деяку її підмножину* $C_0 \subseteq C$

Ці та інші подібні ситуації виникають скрізь і під час вирішення будь-яких задач, зокрема й у інформаційних технологіях. Саме тому підмножина декартова добутку отримала спеціальну назву – відповідність

⚠ ***Відповідність – підмножина декартова добутку.***

Відповідності позначають наступним чином: $f(A, B, G)$. Існують два «скорочених» виду позначень відповідностей: $f: A \to B$ та $G \subseteq A \times B$. Така різноманітність позначень пов’язана із тим, що відповідності використовуються у математиці не тільки для позначення структури об’єктів, а й для позначення різних типів перетворень елементів множин. Коли мається на увазі саме перетворення

елементів множин використовується перше позначення (f: A → B). Коли мається на увазі деяка підмножина елементів, структура яких описується за допомогою декартова добутку, використовується друга нотація.

Приклад 2.9. З курсу середньої школи відоме позначення будь якої функції через символ $f(x)$. Це позначення є складним, оскільки символ f позначає відповідність на множині дійсних чисел, а символ (x) позначає елемент множини, яка стоїть перед символом стрілки. У цьому випадку $f: Q \to Q$,, а, відповідно, $x \in Q$. Загалом позначення $f: Q \to Q$ у контексті теорії функцій, має назву типу функції

Відповідність може задаватися довільно, та до неї можуть входити будь-яка кількість будь-яких кортежей з декартова добутку. Єдиним обмеженнями є відповідність умовам задачі, для вирішення якої будується ця відповідність.

Розглянемо відповідність G ⊆ $A \times B$. Ця відповідність є найпростішим випадком відповідності – підмножиною декартова добутку двох множин. Розглянемо далі основні поняття, пов'язані із відповідностями, які вводяться на базі найпростішого випадку

Визначимо поняття області відправлення відповідності

⚠ ***Множина A у відповідності $G \subseteq A \times B$ ($f: A \to B$) є областю відправлення відповідності***

Наприклад, областю відправлення відповідності з Приклад 2.8 є множина громадян країни Cit, а у випадку Приклад 2.9 областю відправлення є множина дійсних чисел Q.

Зазначимо, що визначення область відправлення відповідності істотно залежить від порядку слідування елементів у декартовому добутку. Область відправлення – це перша множина зліва, яка входить у декартовий добуток. Якщо у декартовому добутку використовуються дужки, областю відправлення буде весь вираз у дужках. З точки зору позначення відповідності зі стрілкою, областю відправлення буде весь вираз, який знаходиться ліворуч від стрілки.

Приклад 2.10. У відповідності $G \subseteq (A \times B) \times C$ *областю відправлення відповідності буде множина* $A \times B$

Приклад 2.11. У відповідності $f: (A \times B) \times C \to D$ *областю відправлення буде множина* $(A \times B) \times C$

⚠ ***Областю прибуття відповідності*** $\boldsymbol{G \subseteq A \times B}$ ***(*** $\boldsymbol{f: A \to B}$***) є множина*** $\boldsymbol{B}$

Областю прибуття відповідності з Приклад 2.8 є множина паспортів Pass, а у випадку Приклад 2.9 областю прибуття є множина дійсних чисел Q.

⚠ ***Графіком відповідності*** $\boldsymbol{f(A, B, G)}$ ***(*** $\boldsymbol{G \subseteq A \times B}$***) є множина*** $\boldsymbol{G}$ ***пар відповідності***

Найпростішим прикладом графіку відповідності є будь-який графік залежності у курсі математики або фізики середньої школи.

Введемо далі у розгляд поняття $\text{пр}_i G$

⚠ $\text{пр}_i \boldsymbol{G}$ ***є множиною*** $\boldsymbol{i}$***-их проекцій відповідності*** $\boldsymbol{G}$***.***

Слід розрізняти першу проекцію конкретного елементу декартова добутку та множину перших проекцій відповідності. На письмі це розрізняється видом літери, яка зазначається після знаку проекції. Якщо літера велика – це множина перших проекцій. Якщо літера маленька – це перша проекція. Але це не виключає того, що будь-якою проекцією може бути множина. Це відбувається у тому випадку, коли членом декартова добутку є булеан.

⚠ ***Множина перших проекцій відповідності $G \subseteq A \times B$ є областю визначення відповідності G.***

Зауважимо, що це визначення діє саме для відповідностей, які мають два члена. Як відповідності, які мають більшу кількість членів можна звести до відповідностей, які мають два члени розглянуто у наступному параграфі цієї книги.

Приклад 2.12. Область визначення для відповідності $f(A, B, G)$ $A = \{a; b; c; d\}$ $B = \{1,2,3,4,5\}$ $G = \{(a; 1); (a; 2); (a, 3), (b, 3)\}$ *є множина* $\{a, b\}$

Приклад 2.13. Якщо є множина будинків на якійсь вулиці та множина цілих позитивних чисел. Якщо вважати адресою лише номер будинку, тоді множина використовуваних для адреси цілих позитивних чисел є областю визначення для відповідності «адреса-будинок».

⚠ ***Множина других проекцій відповідності $G \subseteq A \times B$ є областю значень відповідності G***

Приклад 2.14. Область значень для відповідності $f(A, B, G)$ $A = \{a; b; c; d\}$ $B = \{1,2,3,4,5\}$ $G = \{(a; 1); (a; 2); (a, 3), (b, 3)\}$ *є множина* $\{1; 2; 3\}$

Приклад 2.15. Якщо розглядати множину будинків з Приклад 2.13, то вона є областю значень для відповідності «адреса-будинок»

Властивості відповідностей. Для відповідностей існують чотири основних властивості: всюди визначеність, сюр'єктивність, функціональність, ін'єктивність.

⚠ ***Відповідність для якої область відправлення співпадає з областю визначення є всюди визначеною***

Приклад 2.16. Відповідність $f(A,B,G)$ $A=\{a;b;c;d\}$ $B=\{1,2,3,4,5\}$ $G=\{(a;1);(a;2);(a,3),(b,3)\}$ *не є всюди визначеною, оскільки* $\{a,b\}\neq A$

Приклад 2.17 Відповідність $f(A,B,G)$ $A=\{a;b;c;d\}$ $B=\{1,2,3,4,5\}$ $G=\{(a;1);(b;2);(c,3),(d,3)\}$ *є всюди визначеною, оскільки* $\{a,b,c,d\}=A.$

Таким чином, на одних та тих самих областях відправлення та прибуття може бути визначена як всюди визначена, так і не всюди визначена відповідність. Всюди визначеність відповідності буде залежати від того, яким є графік цієї відповідності.

⚠ ***Відповідність для якої область прибуття співпадає з областю значень є сюр'єктивною.***

Відповідності наведені у Приклад 2.16, Приклад 2.17 не є сюр'єктивними, оскільки в обох випадках область значень відповідності не дорівнює множині B, яка наведена у цих прикладах

Приклад 2.18. Відповідність $\gamma(A,B,G)$ $A=\{a,b,c\}$ $B=4,5,6,7$ $G=(a,4);(b,4);(b;5);(c;6);(c;7)$ *є сюр'єктивною, оскільки множина* $\{4,5,6,7\}=B$

Приклад 2.19. Відповідність «адреса-будинок» є сюр'єктивною, оскільки для кожного будинку призначена відповідна адреса.

Для введення у розгляд інших властивостей відповідностей, визначимо далі два нових поняття – образ та прообраз

⚠ ***Множина всіх*** $\boldsymbol{b\in B}$ ***які відповідають елементу*** $\boldsymbol{a\in A}$ ***на графіку*** $\boldsymbol{G}$ ***є образом*** $\boldsymbol{a}$ ***у*** $\boldsymbol{B}$ ***при відповідності*** $\boldsymbol{\delta(A,B,G)}$

Чи не найвідомішим прикладом множини образів деякого елементу множини є світлини у соцмережах на яких відмічена деяка особа. Формально це можливо представити наступним чином:

Приклад 2.20. Нехай $Human$ *– множина людей.* $Photos$ *– множина світлин у будь-якій соцмережі. Розглянемо відповідність* $\rho(Human, Photos, S)$ *із графіком* S*, який має наступну характеристичну властивість* $S=\{(h,p)$: особа h відмічена на світлині $p\}$*. Тоді образ особи* h *буде множиною його світлин, чим він і є у реальному світі.*

Приклад 2.21. Розглянемо відповідність $f(A,B,G)$*, де* $A=\{1,2,3,4\}$ $B=5,6,7$ $A,B\subset Z$*,* $G=(1,5),(1,6),(1,7),(2,5),(3,5),(2,6),(4,7)$*. Образом елементу «1» буде вся множина* B*. Образом елементу «2» буде множина* $5,6$

Приклад 2.22. Задана множина D *дисциплін, які викладають студентам, та множина оцінок з цих дисциплін* M*. Введемо у розгляд декартовий добуток* $Exam=D\times M$*, який буде позначати оцінку підсумкового контролю з цієї дисципліни. Зрозуміло, що з кожної дисципліни має бути лише одна оцінка, тому*

введемо у розгляд також підмножину $Exam \subset Exam$. Уведемо у розгляд множину $Student$ студентів та декартів добуток $Course = Student \times \mathcal{P}(Exam)$. Булеан множини $Exam$ є, в цьому сенсі, є заліковою книжкою цього студента. Оскільки у студента може бути лише один варіант залікової книжки, то має місце відповідність $Course_0 \subset Course$. Отже, можна стверджувати, що в цій відповідності залікова книжка є образом студента.

⚠ ***Множина всіх $a \in A$ які відповідають елементу $b \in B$ на графіку G є прообразом b у A при відповідності $\delta(A, B, G)$***

Найвідомішим прикладом прообразу є вислів з літературної критики: «прообразом цього літературного героя був….». Це приклад використання терміну прообраз у тому самому значенні, у якому він був застосований у цьому визначенні. Розглянемо цю ситуацію на прикладі.

Приклад 2.23 Нехай H_0 – множина людей, яка використовувалась автором для формування літературних героїв власної нової книги (прообрази літературних героїв). H_b – множина літературних героїв цієї книги. Розглянемо відповідність $\delta(H_0, H_b, G)$. Графік цієї відповідності має наступну характеристичну властивість: $G = \{(h_0, h_b)$: h_0 прообраз герою книги *$h_1\}$.*

Отже, поняття прообразу у математиці та у реальному житті дуже схожі. Поняття образу та прообразу використовується при визначенні властивостей функціональності та ін’єктивності відповідностей.

⚠ ***Відповідність є ін’єктивною, якщо будь-який елемент області значень цієї відповідності має єдиний прообраз***

Приклад 2.24 Відповідність $Course_0$ з Приклад 2.22 є ін'єктивною тому, що кожна залікова книжка належить єдиному студенту.

Приклад 2.25. Відповідність, яка наведена у Приклад 2.21 не є ін'єктивною, оскільки, наприклад, елемент «5» має три різні прообрази.

Приклад 2.26. У Приклад 2.23 також розглянуто не ін'єктивну відповідність, оскільки герой будь-якої книги може мати власними літературними прообразами декількох людей з реального життя.

Зі шкільного курсу математики відомо поняття функції як залежності між змінними. Найчастіше у шкільному курсі розглядаються аналітичні залежності, такі, як $y = 2 * sin\,x + cos\,x$. Але будь-яка залежність між змінними може бути задана й у формі таблиці, тобто перерахуванням пар.

З суворо математичної точки зору будь-яка функціональна залежність є відповідністю із властивістю функціональності

⚠ ***Відповідність є функціональною, якщо будь-який елемент області визначення має єдиний образ***

Отже, усі функціональні залежності, що розглядаються у шкільному курсі математики є функціональними відповідностями $f: Q \to Q$. Запис відповідності, що характеризує функцію, досить часто називається видом або типом функції.

Приклад 2.27 Ще одним прикладом функціональної відповідності є відповідність людини та номеру або номері її білетів на залізничний транспорт. . На кожну людину може бути придбаний не менше ніж один квиток. За прізвищем, номером потягу та датою відправлення завжди однозначно можна визначити за номерами квитки, які ця людина купила. В цьому випадку, відповідність

має наступний формальний запис: $f: (Human;\ \mathcal{P}(Ticket); G)$. *Зауважимо, що в цьому випадку немає жодних обмежень стосовно кількості куплених квитків*

Слід зазначити, що будь-яка зміна графіка відповідності f є, фактично, купівлею якогось квитка. Також формально визначено «зайця», тобто безбілетного пасажира. «Зайця» формально буде визначати вектор з множини G, який має другою проекцією пусту множину.

Відомо, що з метою недопущення «зайців» у потяги провадиться процедура перевірки квитків, яка полягає у тому, що визначається, чи належить множина білетів одній конкретній людині. Ця дія може бути визначена через відповідність, яка має вид $\delta(\mathcal{P}(Ticket); Human; F)$. Такі відповідності мають назву зворотних відповідностей

⚠ ***Зворотньою до заданої відповідності є відповідність у якої область відправлення є областю прибуття заданої відповідності, а область прибуття є областю відправлення заданої відповідності***

Отже, якщо $f(A, B, G)$ - задана відповідність, то відповідність g, яка є зворотньою для f буде мати наступний вигляд $g(B, A, F)$ де множина F має наступну характеристичну властивість:

$$F = \{(b, a): (a, b) \in G\} \tag{2.6}$$

У $F = \{(b, a): (a, b) \in G\}$(2.6) множина G – графік відповідності f, множина F – графік відповідності G.

Зверніть увагу на цікавий факт. У розглянутому Приклад 2.27 перевірка наявності білетів представлена зворотньою відповідністю до відповідності, яка

характеризує пасажирів із квитками. Також, у тексті перед прикладом, було зазначено, що контроль є процедурою перевірки. Оскільки зворотня функціональна відповідність є, фактично, узагальненням зворотньої арифметичної дії, то наша зворотня відповідність повністю відповідає поняттю перевірки, як зворотньої дії

Розглянемо властивості зворотніх відповідностей. Перш за все, звернемо увагу на те, що у властивостях відповідностей дві стосуються областей прибуття та відправлення, а ще дві – областей визначення та значень. Отже, виходячи з вищевикладеного сформулюємо два правила, які дозволять встановлювати властивості зворотних відповідностей, якщо відомі властивості прямих відповідностей

⚠ ***Якщо відповідність всюди визначена – то зворотня їй відповідність сюр'єктивна. Якщо відповідність сюр'єктивна – то зворотня їй відповідність – всюди визначена***

Справедливість цього твердження випливає безпосередньо з визначення зворотньої відповідності. Таке саме твердження існує й для пари «функціональність-ін'єктивність

⚠ ***Якщо відповідність функціональна – то зворотня їй відповідність – ін'єктивна. Якщо відповідність ін'єктивна – то зворотня їй відповідність – функціональна.***

Дві вказані властивості дозволяють швидко встановлювати властивості зворотньої відповідності за властивостями вихідної відповідності.

Операції над відповідностями. Оскільки відповідності, насамперед, є множинами, то для них справедливі основні теоретико-множинні операції.

Об'єднання двох відповідностей $\gamma(A, B, G_1)$ та $\omega(C, D, G_1)$ є відповідність, що визначається за наступною формулою

$$\gamma \cup \omega = \delta(A \cup C, B \cup D, G_1 \cup G_2) \tag{2.7}$$

Перетином двох відповідностей $\gamma(A, B, G_1)$ та $\omega(C, D, G_1)$ є відповідність, що визначається за наступною формулою

$$\gamma \cap \omega = \delta(A \cap C, B \cap D, G_1 \cap G_2) \tag{2.8}$$

Доповненням відповідності $\gamma(A, B, G_1)$ є відповідність, що задається наступною формулою

$$\bar{\gamma} = \delta(A, B, (A \times B) \backslash G_1) \tag{2.9}$$

Для створення нових відповідностей на базі вже існуючих використовується операція композиції. Композиція двох відповідностей $\alpha(A, B, G_1)$ та $\beta(B, C, G_2)$ є відповідність, яка задається наступним співвідношенням:

$$\alpha^{\circ}\beta = (A, C, G_1{}^{\circ}G_2) \tag{2.10}$$

Отже, як слідує з $\alpha^{\circ}\beta = (A, C, G_1{}^{\circ}G_2)$ (2.10), відповідність, що є результатом композиції містить як область відправлення область відправлення першого аргументу, як область прибуття приймається область прибуття другого аргументу, а графіком відповідності є множина, яка визначається за наступною характеристичною властивістю:

$$G_1 \circ G_2 = \{(a,c): a \in A, b \in B, c \in C\ , (a,b) \in G_1, (b,c) \in G_2\} \quad (2.11)$$

Для композиції відповідностей справедливі декілька властивостей. Композиція відповідностей не є комутативною.

$$f \circ g \neq g \circ f \quad (2.12)$$

Композиція відповідностей асоціативна.

$$(f \circ g) \circ h = f \circ (g \circ h) = f \circ g \circ h \quad (2.13)$$

Справедливість цієї властивості випливає з того, що, враховуючи визначення композиції відповідностей неважливо у якому порядку обчислювати цю композицію

Для зворотніх відповідностей існує наступна властивість композиції.

$$(f \circ g)^{-1} = f^{-1} g^{-1} \quad (2.14)$$

Існують декілька спеціальних типів відповідностей, які класифікуються за наявністю або відсутністю деяких властивостей. Розглянемо більш детально ці типи відповідностей

Спеціальні типи відповідностей. Окремим класом відповідностей є всюди визначені та функціональні відповідності.

⚠ ***Відображення – всюди визначена та функціональна відповідність***

Зауважимо, що в різних літературних джерелах трактування поняття відображення можуть відрізнятися. Насамперед це стосується вимоги всюди визначеності відображення. Зокрема, ця вимога може й не зазначатися у визначенні. В

цьому випадку, не виділяється окремо область відправлення, а множина A відразу вважається областю визначення. Те саме стосується й області значень, якою відразу вважається множина B.

⚠ ***Оператор – це відображення, у якому на множині визначення та значень задана додаткова структура, порядок тощо***

Найпростішим математичним прикладом оператору є оператор диференціювання або лінійний оператор. У програмуванні виділяються оператори умовного та безумовного переходу, оператори циклу. У випадку програмування відношенням порядку може бути порядок слідування самих операторів у програмному коді. Оператори переходу та циклу обирають наступний блок коду для виконання. Процедура вибору може відбуватися з урахуванням умов до даних предметної області, або без них.

⚠ ***Бієкція – всюди визначена, функціональна, ін'єктивна, сюр'єктивна відповідність.***

Одним з найпростіших прикладів бієкції є відповідність людини та її місця, наприклад, у залі театру чи кінотеатру. Також бієкцією є відповідність людини та номеру її паспорту.

Якщо застосувати до бієкції поняття, пов'язані зі зворотними відповідностями можна дійти наступного висновку

⚠ ***Зворотня до бієкції відповідність - бієкція***

Окремим класом відповідностей є відповідності, у яких область відправлення та прибуття є однією та тією ж самою множиною за характеристичною властивістю. Ці відповідності отримали назву відношень

2.3 Поняття відношення

Визначення відношення. Особливим випадком відповідності є поняття відношення.

⚠ ***Відношення – підмножина декартова ступеню: $R \subseteq A^n$. Величина n є арністю відношення***

Відношення, як і множини, позначаються великими латинськими літерами. У математичній літературі [] найчастіше зустрічається позначення відношення за допомогою латинської букви R (від англійського слова – relation, що означає «відношення»).

Приклад 2.28. Чи не найбільш зрозумілим прикладом того, чим є відношення є будь-які відношення на множині людей. Ми досить часто чуємо «у них дружні відносини», «в них добрі професійні відносини». Ці слова є спорідненими зі словом відношення та мають на увазі, щонайменше, двох людей. Нехай $Human$ – множина людей. Тоді множина людей, які мають хороші професійні відносини можна вважати відношенням $R \subset Human^2$. Знак суворої підмножини використано тому, що далеко не з усіма людьми, як правило, можливо мати хороші професіональні відношення.

Для відношень справедливі всі ці ж самі засоби завдання, що й для відповідностей та усі поняття пов'язані з ними. Це справедливо, насамперед, тому, що декартовий ступінь є частковим випадком декартова добутку. Так само для відношень справедливі всі властивості відповідностей, які були розглянуті у попередньому розділі. Поряд із цим, у відповідностей є більше властивостей.

Властивості відношення. Властивості відношень розглядаються, насамперед, для бінарних відношень, тому що відношення з більшою кількістю членів зводяться до бінарних.

Для завдання бінарних відношень досить часто використовуються матриці відношення. Матриця відношення – це прямокутна таблиця, у якій кожному рядку і стовпцю відповідає елемент множини, на якій задане відношення. На перетину рядка та стовпця стоять «1», якщо пара належить відношенню, пуста клітинка у іншому випадку.

Приклад 2.29 Розглянемо відношення $R \subset A^2$ *на множині* $A = \{a, b, c, d\}$. *Графік цього відношення має наступний вигляд* $R = \{(a, a), (a, b), (c, c), (c, d)\}$. *Матриця відношення має наступний вид:*

	A	B	C	d
A	1	1		
B				
C			1	1
D				

Для бінарних відношень існують три основних властивості: рефлексивність, симетричність та транзитивність

⚠ ***Відношення називається рефлексивним, якщо для будь-якого елементу*** $\boldsymbol{a} \in \boldsymbol{A}$ ***справедливо, що*** $(\boldsymbol{a}, \boldsymbol{a}) \in \boldsymbol{R}$***, де*** $\boldsymbol{R} \subseteq \boldsymbol{A^2}$

Рефлексивне відношення містить пари, які склажаються з однакових компонентів. Рефлексивне відношення стверджує, що об'єкт сам із собою знаходиться у заданому відношенні.

Приклад 2.30. Відношення «бути студентом деякої групи» є рефлексивним, оскільки кожний студент деякої групи знаходиться із самим собою у цьому відношенні.

Приклад 2.31 Відношення $G \subset A^2$ *де* $A = \{a, b, c\}$, $G = \{(a, a); (a, b); (b, b); (c, c)\}$ *є рефлексивним тому, що воно включає як пари усі можливі вектори з однаковими першими та другими компонентами.*

Якщо розглядати більшість відношень у реальному світі, вони є рефлексивними. А вилучення цієї властивості не впливає істотно на зміст та суть відношення, що розглядається Тому дуже часто рефлексивну властивість вилучають з розгляду.

⚠ ***Відношення називається симетричним, якщо наслідком того, що*** $(\boldsymbol{a}, \boldsymbol{b}) \in \boldsymbol{R}$ ***є*** $(\boldsymbol{b}, \boldsymbol{a}) \in \boldsymbol{R}$ ***(для будь-яких*** $\boldsymbol{a}, \boldsymbol{b} \in \boldsymbol{A}$**,** $\boldsymbol{R} \subseteq \boldsymbol{A^2}$***)***

Трактовка симетричного відношення не відрізняється від трактовки симетрії, наприклад, у геометрії. Випадок геометричної симетричності полягає в тому, що дві точки знаходяться на однакових відстанях одна від одної. Це означає, що геометричні вектори, які мають початком точку симетрії або точку на осі симетрії, а кінцем – задані точки, мають однакову довжину.

З курсу середньої школи відомо, що довжина (модуль) вектору обчислюється за формулою $\sqrt{x^2+y^2}$, де x та y – координати вектору. Якщо розглядати дві симетричні пари $(a,b) \in R$ та $(b,a) \in R$, R $\subset Q^2$, то модуль вектору для обох цих пар буде однаковим, тобто має місце випадок геометричної симетричності

Приклад 2.32. Відношення на множині $St = \{st: st\ -$ студент $\}$ *«бути в одній групі» є симетричним, оскільки незалежно від того, як ми перерахуємо студентів, це відношення буде симетричним.*

Приклад 2.33 Відношення $G \subset A^2$ *де* $A = \{a,b,c\}$, $G = \{(a,a);(a,b);(b,b);(c,c)\}$ *не є симетричним тому, що воно має пару* (a,b)*, але не має пари* (b,a)

⚠ ***Транзитивним називається відношення у якому для будь-яких трьох елементів*** $\boldsymbol{a,b,c \in A}$ ***з того, що*** $\boldsymbol{(a,b) \in R}$ ***та*** $\boldsymbol{(b,c) \in R}$ ***виходить, що*** $\boldsymbol{(a,c) \in R}$**,** $\boldsymbol{R \subseteq A^2}$

Слово «транзитивне» є спорідненим зі словом «транзит». Дійсно, якщо уважно перечитати визначення, то приналежність пари $(a;c) \in R$ випливає своєрідним «транзитом» через елемент b.

Слід зазначити, що для цих трьох властивостей їх відсутність позначається за допомогою префікса «анті-».

Спеціальні типи відношень. У залежності від того, яким набором властивстей є декілька спеціальних типів відношень.

⚠ ***Антирефлексивне, антисиметричне та транзитивне відношення є суворим порядком***

Найпростішим відношенням суворого порядку є відомі відношення на множині чисел «більше» або «менше». Узагальненням цих відношень є відношення «передування» для будь-яких об'єктів, зокрема знайоме вже відношення суворого включення, яке задається на булеані множин. Родинні зв'язки також є прикладом відношення суворого порядку, якщо враховувати дату народження

⚠ ***Рефлексивне, антисиметричне, транзитивне відношення є відношенням несуворого порядку***

Для множини чисел таким відношенням є відношення «більше або дорівнює, менше або дорівнює».

⚠ ***Рефлексивне, симетричне, транзитивне відношення є відношенням еквівалентності***

Поняття еквівалентності значно ширше, аніж загальновідоме поняття рівності. Досить важливим у програмній інженерії є поняття класу. З математичної точки зору поняття класу.

⚠ ***Підмножина, усі елементи якої мають деяку загальну властивість або набір властивостей називається класом***

Приклад 2.34. Задана множина пасажирських залізничних вагонів $RailCarr$ *, яка задається наступним аналітичним виразом* $RailCarr = Num \times Seats \times Firm$ *. У цьому виразі* Num *– множина номерів вагонів, яка визначається за формулою* $Num = \{num : num \in Z_+\}$ *,* $Firm$ *– множина виробників ,* $Seats$ *–*

кількість місць у вагоні, $Seats = \{s: s \in Z_+ \ \& s \in [1; 52]\}$ *. У випадку, коли кількість місць дорівнює 36 (число місць у купейному вагоні), можна говорити, що в рамках* $RailCarr$ *існує клас «купейний вагон».*

Приклад 2.35. Якщо розглядати множину студентів St *, що аналітично задається виразом* $St = F \times Adr \times \mathcal{P}(Tel)$ *, де* F *– множина прізвищ,* Adr *– адреса проживання,* $\mathcal{P}(Tel)$ *– булеан множини номерів телефонів студента. Якщо вважати, що в цій моделі, наприклад, колір волосся не вказаний лише тому, що він однаковий для всіх студентів, то* St *є класом, який заданий на множині* $St = F \times Adr \times \mathcal{P}(Tel) \times ColorHair$ *.*

Приклад 2.35 ілюструє, фактично, поняття класу так, як воно, наразі, трактується у об'єктно-орієнтованих мовах програмування.

Розглянемо далі клас купейних вагонів, який наведено у Приклад 2.34. Якщо розглядати класи купейних вагонів за фірмою-виробником, то можна утворити множину класів (систему класів) у рамках класу «купейні вагони». Розглянемо властивості цієї системи класів.

Елементи цієї системи класів не перетинаються, оскільки кожен конкретний вагон може містити лише єдиного виробника. Безпосередньо з цього випливає, що ці класи не перетинаються, знов таки тому, що в кожного вагону є лише єдиний виробник. І так само зрозуміло, що вагони різних виробників для нас, в цьому сенсі, будуть різними.

Побудована нами система класів є класами еквівалентності.

⚠ ***Задана система класів*** $\mathrm{C_1}, \mathrm{C_2}, \mathrm{C_3}, \ldots, \mathrm{C_i}$ ***на заданій множині M , яка має наступні властивості: 1) Класи не перетинаються 2) Будь-які два елементи з***

одного класу еквівалентні 3) Будь-які два елемента з різних класів не еквівалентні. Така система класів називається системой класів еквівалентності

Введемо далі у розгляд поняття порівняння векторів. Нехай задані два вектори $(a_1, a_2, a_3, \dots, a_n), (b_1, b_2, b_3, \dots, b_n) \in R$, R $\subset A^n$. Визначимо відношення $<$, $\leq$ та $=$ наступним чином:

$(a_1, a_2, a_3, \dots, a_n) > (b_1, b_2, b_3, \dots, b_n)$ якщо $a_1 > b_1, a_2 > b_2, \dots, a_n > b_n$

$(a_1, a_2, a_3, \dots, a_n) \geq (b_1, b_2, b_3, \dots, b_n)$ якщо $a_1 \geq b_1, a_2 \geq b_2, \dots, a_i > b_i, \dots, a_n > b_n$

$(a_1, a_2, a_3, \dots, a_n) = (b_1, b_2, b_3, \dots, b_n)$ якщо $a_1 = b_1, a_2 = b_2, \dots, a_i = b_i, \dots, a_n = b_n$

Зауважимо, що порівнювати можна лише вектори, які мають однакову довжину (кількість елементів). В іншому випадку між цими векторами немає відношення.

Відношення рівності має наступну властивість

⚠ ***Усі класи еквівалентності за відношенням рівності складаються з єдиного елементу***

Наведений у Приклад 2.35 випадок коли множина студентів є класом еквівалентності за рівним кольором волосся в усіх студентів є класом еквівалентності за відношенням рівності. У той же час, випадок у Приклад 2.34, хоча й використовує клас еквівалентності за відношенням рівності, у той же час не є класом еквівалентності за відношенням рівності, оскільки за номером вагону може не виконуватись умова рівності двох векторів, яка наведена вище.

Сфера застосування відповідностей та відношень у програмній інженерії досить велика. Пов'язане це із тим, що на базі понять «відповідність» та «відношення» у математиці вводиться у розгляд багато похідних понять, на яких заснована розробка програмних систем: предикати, операції, булеві функції та деякі інші. Під час розгляду цих понять буде розглядатись також їх застосування у програмній інженерії

2.4 Завдання до розділу 2

Завдання 2.1. Виконайте дії

а) $\{1,2,3\} \times \{a, b, c\}$

б) $\{a, b, c\} \times \{e, f\}$

в) $\{a, b\} \times \{a, b\}$

г) $\{a\} \times \{a, b, c\}$

Завдання 2.2 Визначте властивості відповідності, яка задана таблично.

а)

B / A	1	2	3	4
A	1			
B	1		1	
C			1	1
D				

б)

B	1	2	3	4

A				
A	1	1		
B			1	
C				1
D				

в)

B / A	1	2	3	4
A	1			
B	1		1	
C		1		
D			1	

г)

B / A	1	2	3	4
a	1			
b	1	1		
c		1		
d			1	1

Завдання 2.3.Визначте властивості відношень, що завдані на множині дійсних чисел. У завданні літерою x позначено значення першої проекції відношення, літерою y – другої компоненти відношення

а) $y = \sin x$

б) $y = \frac{1}{x+1}$

в) $y = \sqrt{x+2}$

г) $y = \sqrt{x^2 + 2x - 3}$

Завдання 2.4. Визначте властивості бінарного відношення, що задане наступною матрицею

а)

A / A	a	B	C	d
a	1			
b	1	1	1	
c		1	1	
d			1	1

б)

A / A	a	b	C	D	e	f
A	1					
B	1	1	1			1
C		1	1		1	
D			1	1		
E			1			
F		1				1

в)

A	a	b	C	D	e	f

A						
A		1			1	
B	1			1	1	
C						
D		1				
E	1	1				
F						

г)

A / A	a	b	c
A	1		
B	1	1	1
C		1	1

д)

A / A	a	b	c	D
a	1			
b		1	1	
c			1	
d			1	1

3 ОПЕРАЦІЇ. АЛГЕБРА. РИШІТКА.

3.1 Поняття операції

Визначення операції. Одним з головних застосувань відношень у математиці є визначення поняття операції.

⚠ ***Нехай A - довільна непорожня множина і n - натуральне число. Будь-яке відображення*** $\omega: A^n \to A$ ***називають n-арною (або n-місною) операцією на множині A***

Таким чином, згідно з наведеним визначенням, n-арна операція ω кожному кортежу $(a_1, \ldots, a_n) \in A$ однозначно зіставляє елемент $b \in A$. Компоненти кортежу називають при цьому аргументами операції ω; а b - результатом застосування операції ω до аргументів $a_i, \ldots, a_n$.

Для n-арної операції використовують позначення:

$$b = \omega(a_1, \ldots, a_n) \tag{3.1}$$

або

$$b = a_1, \ldots, a_n \tag{3.2}$$

Запис операції згідно формули $b = \omega(a_1, \ldots, a_n)$ (3.1) є префіксним записом операції ω, запис за формулою $b = a_1, \ldots, a_n$ (3.2) є постфіксним записом операції. При двох аргументах операції $n = 2$ говорять про бінарну операцію. При одному аргументі операції говорять про унарну операцію. Для бінарних операцій часто використовують позначення $a_1\ \omega\ a_2$.

Наведемо деякі приклади операцій на різних множинах:

Приклад 3.1 Найпростішим прикладом математичних операцій є операція «додавання». Це бінарна операція, визначена на множині дійсних чисел Q. З точки зору визначення операції це можна записати так: $\varphi: Q^2 \to Q$.

Приклад 3.2. Усі теоретико-множинні операції визначені на булеані. Основою побудови булеану є розглядаємий нами універсум. Отже, операція перетину та об'єднання можуть бути записані у формі визначення операції наступним чином: $\varphi: \left(\mathcal{P}(U)\right)^2 \to \mathcal{P}(U)$.

Зазначимо, що у багатьох випадках універсум нам вже заданий зазделегідь. В цьому випадку замість символа U може використовуватись символ множини, яка має аналітичне визначення. Частіше за все, для цілей моделювання предметних областей для розробки програмного забезпечення та для опису обчислювальних процесів, у якості визначення для універсуму використовується відповідний декартів добуток

Приклад 3.3 Розглянемо множину студентів, що задається наступним аналітичним виразом $St = F \times Adr \times Tel$, де $F = \{f: \text{прізвище}\}$, $Adr = \{adr: \text{домашня адреса}\}$, $Tel = \{tel: \text{контактний телефон}\}$. Вочевидь, будь яка група студентів є елементом булеану. Тобто $Gr = \mathcal{P}(St)$. Враховуючи цю рівність можна стверджувати, що, наприклад, зарахування групи студентів є бінарною операцією над групою студентів $\varphi: (Gr)^2 \to Gr$, яка співпадає із теоретико-множинною операцією об'єднання. Цей факт випливає з того, що усі можливі варіанти академічної групи співпадають з поняттям булеану студентів, тобто має місце рівність $Gr = \mathcal{P}(St)$.

Для ілюстрації виконання операції частіше за все використовуються таблиці Келі. Приклад таблиці для операції доповнення, якщо задана множина $A = \{a, b, c\}$ як універсальна множина, наведено у Таблиця 3.1

Таблиця 3.1 Таблиця Келі для операції доповнення

Вихідна множина $B \subseteq A$	$\overline{B}$
$\overline{\emptyset}$	$\{a, b, c\}$
{a}	{b, c}
{b}	{a, c}
{c}	{a, b}
{a, b}	{c}
{a, c}	{b}
{b, c}	{a}
{a, b, c}	∅

Для операції перетину для множини $\mathcal{P}(A)$, таблиця Келі має вигляд:

Таблиця 3.2 - Таблиця Келі для операції перетину

	∅	{a}	{b}	{c}	{a, b}	{a, c}	{b, c}	{a, b, c}
∅	∅	∅	∅	∅	∅	∅	∅	∅
{a}	∅	{a}	∅	∅	{a}	{a}	∅	{a}
{b}	∅	∅	{b}	∅	{b}	∅	{b}	{b}
{c}	∅	∅	∅	{c}	∅	{c}	{c}	{c}
{a, b}	∅	{a}	{b}	∅	{a, b}	{a}	{b}	{a, b}
{a, c}	∅	{a}	∅	{c}	{a}	{a, c}	{c}	{a, c}
{b, c}	∅	∅	{b}	{c}	{b}	{c}	{b, c}	{b, c}
{a, b, c}	∅	{a}	{b}	{c}	{a, b}	{a, c}	{b, c}	{a, b, c}

Зрозуміло, що, якщо операція задана на нескінченній множині, то таблиця Келі для неї також буде нескінченною.

Розглянемо ще один приклад [3]. Нехай наведено квадрат зі вершинами a_1, a_2, a_3, a_4

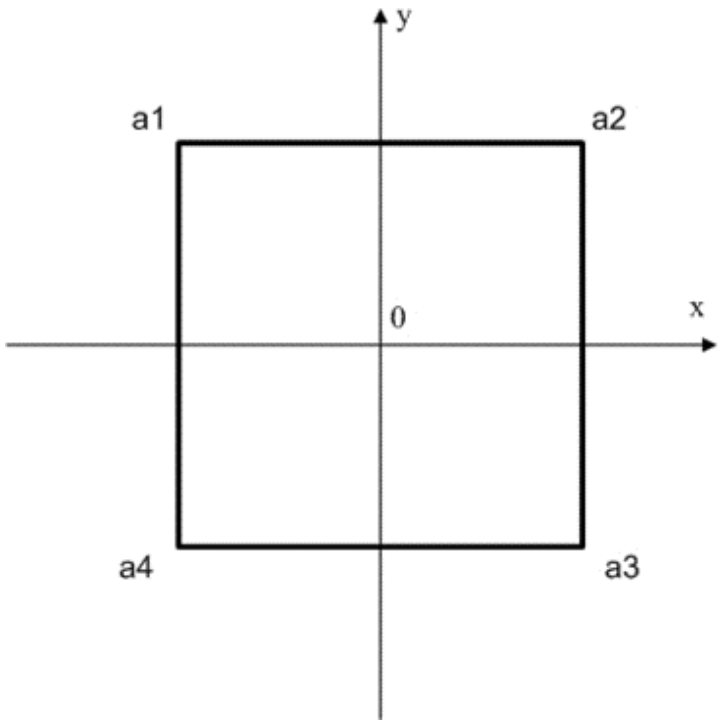


Рисунок 3.1 - Квадрат з вершинами a_1, a_2, a_3, a_4

Введемо у розгляд чотири операції $\alpha, \beta, \gamma, \delta$. Ці операції будуть, відповідно, означати: «поворот квадрата на 0 рад» (операція α), «поворот квадрата на $\frac{\pi}{2}$ рад» (операція β), «поворот квадрата на π рад» (операція γ), «поворот квадрата на $\frac{3\pi}{2}$ рад» (операція δ). Таблиця Келі для цього прикладу буде мати вигляд, що наведений у Таблиця 3.3

Звернемо увагу, що операція α не змінює вершину квадрату. Тобто результатом операції $\alpha(a_2)$ буде знов вершина a_2. Ця операція є прикладом тотожньої операції.

Таблиця 3.3 - Таблиця Келі для операції повороту квадрату

	α	B	γ	δ
a_1	a_1	a_2	a_3	a_4
a_2	a_2	a_3	a_4	a_1
a_3	a_3	a_4	a_1	a_2
a_4	a_4	a_1	a_2	a_3

⚠ ***Тотожня операція – це операція, результатом якої для будь-якого аргумента є цей аргумент.***

Розрізняють також нульарну операцію

⚠ ***Нульарна операція – це довільний елемент множини A***

Прикладом нульарної операції є фіксація нуля по відношенню до операції додавання.

Найбільш дослідженими у математиці та найбільш застосованими є унарні та бінарні операції. Розглянемо далі їх властивості

Властивості бінарних та унарних операцій. До властивостей бінарних операцій $\varphi: A^2 \to A$, які задані на множині A, належать асоціативність, дистрибутивність, комутативність та ідемпотентність. Для унарних операцій існує властивість інвалютивності.

Властивість комутативності полягає у незалежності результатів бінарної операції від порядку задавання конкретних значень аргументів. Операція $\varphi: A^2 \to$ A , яка має властивість комутативності, є комутативною операцією:

$$a\varphi b = b\varphi a \tag{3.3}$$

Для того, щоб спростити розуміння властивостей, досить часто використовується поняття «позначення операції», тобто використання якогось символу замість грецької букви (див. формулу $a\varphi b = b\varphi a$ (3.3)). Надалі ми будемо використовувати для позначення операції символ «·». Тоді рівняння $a\varphi b = b\varphi a$ (3.3) буде преобразовано до вигляду:

$$a \cdot b = b \cdot a \tag{3.4}$$

Прикладом комутативних операцій є операціїї додавання та множення на множині дійсних чисел. Прикладами некомутативних операцій є операція декартова добутку. Комутативній операції властива симетрична відносно головної діагоналі таблиця Келі.

Властивість асоціативності характеризує поведінку бінарної операції у випадку її послідовного застосування до трьох та більшого числа операндів. Властивість асоціативності декларує незалежність результату від порядку виконання бінарної операції.

$$a \cdot b \cdot c = (a \cdot b) \cdot c = a \cdot (b \cdot c) \tag{3.5}$$

Як і для випадку комутативної операції, крапка у формулі $a \cdot b \cdot c = (a \cdot b) \cdot c = a \cdot (b \cdot c)$ (3.5) символізує деяку бінарну операцію $\varphi: A^2 \to A$, яка визначена на множині A. Прикладами асоціативних операцій можуть бути перетин на булеані множин, додавання на множині чисел та інші.

Властивість ідемпотентності характеризує поведінку бінарної операції по відношенню до рівних між собою аргументів. Будь-яка операція є ідемпотентною,

якщо на двох, рівних між собою, аргументах, вона має результат, рівний цим аргументам.

$$a \cdot a = a \tag{3.6}$$

Ідемпотентними є усі теоретико-множинні операції. Операції, що визначені на множинах чисел не є ідемпотентними.

Для двох бінарних операцій справедлива властивість дистрибутивності. Саме властивість дистрибутивності дозволяє проводити операції «розкриття дужок», «групування» та «приведення подібних членів» у алгебрі чисел. У загальному вигляді властивість дистрибутивності можна представити наступною рівністю.

$$a \cdot (b * c) = (a \cdot b) * (a \cdot c) \tag{3.7}$$

У формулі $a \cdot (b * c) = (a \cdot b) * (a \cdot c)$ (3.7) символом «·» позначена операція $\varphi: A^2 \to A$, а символом «*» операція $\gamma: A^2 \to A$, яка так само визначена на множині A, але має іншу суть. Прикладом таких операцій є, наприклад, додавання та множення на множині чисел або перетин та об'єднання на деякому булеані. У випадку множини чисел маємо дистрибутивність множення по відношенню до додавання, а у випадку булеану множин є справедливою дистрибутивність перетину відносно об'єднання та об'єднання відносно перетину.

Поняття «одиниці» та «нуля». З курсу математики середньої школи відомі тотожності по відношенню до чисел 0 та 1. Зокрема, для множення справедливі наступні тотожності: $0 * a = 0$ (анулююча властивість нуля) та $a * 1 = a$. Тому елементи множини A, які ведуть себе так само по відношенню до бінарної операції $\varphi: A^2 \to A$, як нуль та одиниця по відношенню до операції множення,

називають, відповідно, нулем операції φ та одиницею операції φ. Досить часто термін «одиниця операції» замінюють терміном «нейтральний елемент по відношенню до операції φ».

З властивостей теорії множин видно, що універсум є нулем по відношенню до об'єднання та одиницею по відношенню до перетину. Пуста множина є нулем по відношенню до перетину та одиницею по відношенню до об'єднання. Нуль у операції додавання дійсних чисел відіграє роль одиниці для цієї операції.

Властивість елементу бути нулем чи одиницею деякої операції може істотно залежить від позиції цього елементу по відношенню до цієї операції. Найбільш поширені ці ситуації у теорії матриць. Розглянемо цю ситуацію на прикладі [2]

Приклад 3.4. На множині квадратних матриць виду $\begin{pmatrix} a & 0 \\ b & 1 \end{pmatrix}$, *де* $a, b \in \mathbb{Q}$, *будь-яка матриця* $\begin{pmatrix} 0 & 0 \\ d & 1 \end{pmatrix}$ $d \in \mathbb{Q}$ *буде правим нулем по відношенню до операції множення цих матриць, тому що:*

$$\begin{pmatrix} a & 0 \\ b & 1 \end{pmatrix}\begin{pmatrix} 0 & 0 \\ d & 1 \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ d & 1 \end{pmatrix}$$

Зазначимо, що лівого нуля у цій множині не існує.

Для нуля та одиниці будь-якої операції справедливе наступне твердження

⚠ ***Якщо для деякої операції у множині існує одночасно лівий та правий ноль (ліва та права одиниця), то ці ноль та одиниця є єдиними для цієї операції.***

Це твердження зводить, фактично, до часткового випадку існування більш ніж одного нуля чи одиниці у множині по відношенню до операції.

Використання поняття операції для моделювання дій у предметних областях. Найчастіше операції використовуються для моделювання дій у предметних областях. Слід зазначити, що цій самій меті слугують й функції (функціональні відповідності). Але практично в усіх випадках, функції можуть бути замінені на операції. Розглянемо декілька прикладів використання операцій для моделювання предметних областей.

Приклад 3.5. Розглянемо множину вантажівок, яка характеризується наступним аналітичним виразом $Truck = Dr \times Num \times Cargo$. *Множини, які входять у опис вантажівки мають наступні характеристичні властивості* $Dr = \{dr: \text{водій}\}$, $Num = \{num: \text{державний реєстраційний номер}\}$, $Cargo = \{c: \text{найменування вантажу}\}$. *Тоді операція зміна водія вантажівки буде формально описуватись як* $\varphi: Truck^2 \to Truck$. *Першим аргументом цієї операції буде вантажівка, для якої потрібно змінити водія. Другим аргументом буде «віртуальна вантажівка», у якій буде міститися фамілія нового водія. Результатом цієї операції буде нова вантажівка, яка містить потрібну фамілію водія.*

Зазначимо, що підхід, який використовується у цьому прикладі, зокрема, використання у якості другого аргумента об’єкта в цілому, дуже розповсюджений у об'єктно-орієнтованих мовах програмування та відомий у них, як перевизначення операцій.

Той самий результат можна отримати, якщо визначити φ як функцію типу: $\varphi: \text{Truck} \times \text{Dr} \to \text{Truck}$. Саме цей випадок використовується у процедурних мовах. Другий параметр є параметром функції мови програмування, область прибуття цієї функціональної відповідності є значенням, яке повертає функція. Перша проекція області відправлення, як правило, є глобальною змінною у мові

програмування, яку також використовує реалізація функції. Слід зазначити, що цю змінну можна передати і у якості параметра функції. Те саме можна зробити також з операцією.

Таким чином, цей підхід дозволяє представити будь яку дію у мовах програмування.

3.2 Поняття алгебри. Ришітка

Поняття алгебри. Як було показано у попередніх розділах, будь-яка операція задається на деякій множині значень. Досить часто таку множину значень розглядають і як частину більш складного математичного елементу – алгебри.

⚠ ***Двійка множин (A, Ω) де A – довільна множина, Ω – множина операцій (сигнатура), що визначена на цій множині, є алгеброю***

Звернемо увагу, що, фактично, алгебра є формальною системою, що поєднує у собі множину та операції на цій множині.

Усі алгебри класифікуються за декількома принципами: за кількістю основ, за арністю операцій сигнатури та за властивостями цих операцій. За кількістю основ алгебри поділяються на одноосновні та багатоосновні. Багатоосновні алгебри можуть мати декілька основ.В цьому випадку аргументи операцій сигнатури можуть належати декільком різним основам. Саме багатоосновні алгебри лежать у основі поняття абстрактного типу даних, який визначається, за стандартом ДСТУ як:

⚠ ***Datatype – defined set of data objects of a specified data structure and a set of permissible operations, such that these data objects act as operands in the execution of any of these operations.***

Це визначення наведене англійською у стандарті ДСТУ ISO 2382:2015. Він введений в дію с 1 січня 2019 року.

Іншим способом класифікації алгебр є класифікація за арністю операцій. Якщо в сигнатурі присутня одна унарна та одна бінарна операція, то це алгебра типу «2-1». Якщо в сигнатурі присутні дві бінарні операції, то це є алгеброю типу «2-2».

Найбільш важливою, розивненою та використовуваною на даний момент є класифікація алгебр за властивостями операцій. Надалі ми будемо її використовувати для будь-якого випадку визначення конкретної алгебри.

Розглянуті нами операції теорії множин належать до важливого класу алгебр – ришітка. Формальне визначення ришітки, з точки зору класифікації алгебр, виглядає наступним чином:

⚠ ***Алгебра, яка містить дві бінарні операції, які є комутативними, асоціативними та ідемпотентними є ришіткою.***

Розглянемо ришітки більш детально. Для цього введемо у розгляд поняття «у-множини»

Поняття у-множини. Поняття у-множини визначається наступним чином:

⚠ ***Множину, на якій задано відношення несуворого (або, як частковий випадок, суворого) порядку будемо надалі називати упорядкованою множиною або, коротше, у-множиною.***

Виділяють окремий частковий випадок у-множини – лінейно впорядкована множина або ланцюг.

⚠ ***У-множина A, яка задовольняє властивості лінейності, а саме: $x \geq y$ або $x \leq y$ для усіх $x, y \in A$, є лінейно упорядкованою у-множиною або ланцюгом***

Головною відмінністю ланцюга від будь-якої іншої у-множини є той факт, що у ланцюгу усі елементи є такими, що можна порівнювати. У загальному випадку в у-множині можуть бути елементи, які не можна порівняти між собою. Якщо жоден елемент не множна порівняти з іншим елементом, такий випадок називають антиланцюгом [4].

Приклад 3.6. Найпростішим прикладом ланцюга є послідовність цілих чисел. Серед них немає елементів, які неможливо порівняти.

Приклад 3.7 Прикладом множини, яка не є ланцюгом, однак є у-множиною є булеан деякої підмножини.

Для у-множин існує поняття порядку

⚠ ***Порядком $n(P)$ у-множини P є кількість її елементів. Якщо це кінцеве число, то говорять про кінцеву у-множину.***

Надалі введемо у розгляд поняття «найменший елемент», «найбільший елемент», «максимальний елемент», «мінімальний елемент». Раніше, у алгебрі чисел ці поняття були фактично тотожні. Але, з точки зору інших розділів математики, зокрема, загальної алгебри та теорії ришіток – це різні поняття

⚠ ***Якщо в у-множині P існує єдиний елемент $a \in P$, такий, що для будь-якого $x \in P$ є вірною нерівність $a \leq x$ то a називається найменшим елементом у-множини.***

З цього випливає два важливих наслідки. По-перше, якщо цей елемент існує, то він є єдиним, по-друге – всі елементи множини P можна порівняти між собою. Остання особливість є досить важливою. Вона випливає з того, що для визначення поняття найменшого елементу використовується поняття «приналежності у-множині» будь-якого елементу. Наслідком цього є порівнянність усіх елементів у-множини між собою.

Але не завжди всі елементи однієї й тієї ж у-множини можна порівнювати між собою, навіть, якщо мова йде про множини, елементами яких є числа. Найкращим прикладом цього є у-множина, де діє відношення порядку $x \leq y$ у розумінні «х ділить у». Прикладом такої множини є множина $A = \{2, 3,6,12,24\}$ [4]. Але у цій множині існують числа, які не перебувають між собою у цьому відношенні. Це, наприклад, 2 та 3. В цьому випадку не можна застосовувати поняття «найменший елемент», бо, згідно визначення, він має бути єдиним. Тому використовується поняття «мінімального елементу».

⚠ ***Мінімальним елементом у-множини P є такий елемент $a \in P$, що ні для якого $x \in P$ не виконується умова $x \leq a$***

Для розглянутої нами множини A існують мінімальні елементи 2 та 3. Але не існує найменшого елементу, тому що його не можна обрати між 2 та 3.

Аналогічно вводяться поняття найбільшого та максимального елементу.

⚠ ***Якщо в у-множині P існує єдиний елемент $a \in P$, такий, що для будь-якого $x \in P$ є вірною нерівність $b \geq x$ то b називається найменшим елементом у-множини.***

Так само, якщо не можна визначити найменший елемент використовується поняття максимального елементу.

⚠ ***Максимальним елементом у-множини P є такий елемент $a \in P$, що ні для якого $x \in P$ не виконується умова $x \leq a$***

Зазначимо, що, наприклад, у множині A є найбільший (24), але відсутній найменший елемент. Також слід зазначити, що далеко не завжди порівнянність елементів у-множини між собою є умовою для появи найбільшого та найменшого елементу. Але можна стверджувати, що якщо у множині єдиний мінімальний чи максимальний елемент – він одночасно є найбільшим або найменшим. Саме тому, для теорії множин, справедливе, що в них існують найбільший (універсум) та найменший (пуста множина) елементи. У загальному випадку найбільший елемент називають одиницею у-множини. Одиницю у-множини не плутати з одиницею – нейтральним елементом алгебри, але одиниця у-множини може бути нейтральним елементом. Одиниця у-множини позначається символом I. Відповідно, найменший елемент у-множини є нулем у-множини та позначається символом O.

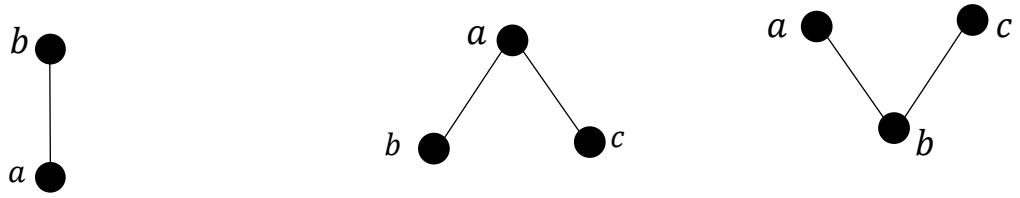
Так само, не слід плутати нуль у-множини та нуль для операції, але нуль у-множини може виступати нулем для операції у алгебрі.

Графічне позначення у-множин. Діаграми Хассе. Елементи у-множини можна позначити графічно на діаграмі Хассе. Ця діаграма містить точки, які позначають елементи у-множини та відрізки, які з'єднують точки, що можна порівнювати. Точки можливо маркувати відповідними буквами або іншими знаками.

Слід зазначити, що якщо на діаграмах Хассе позначити всі можливі випадки для відношення порядку в у-множині, є ризик отримати дуже складний рисунок. Тому в математиці досить часто використовується діаграма Хассе, із додатковим урахуванням відношення покриття на у-множині

⚠ ***В у-множині із відношенням порядку елемент b покриває елемент a, якщо $a < b$ та не існує такого елементу x, що виконується рівність $a < x < b$.***

Розглянемо далі приклади діаграм Хассе [4] (Рисунок 3.2)



а) б) в)

а) b покриває а; б) a покриває b та c ; в) а та с покривають b

Рисунок 3.2 - Приклади діаграм Хассе

У випадку б), зокрема, маємо у-множину, що має 2 мінімальних елементи та найбільший елемент.

Поняття ришітки. Для у-множин вводять поняття нижньої та верхньої грані

⚠ ***Елемент u є нижньою гранню елементів a та b, якщо $u \leq a$ та $u \leq b$ за відношенням порядку в у-множині***

Аналогічно визначається поняття верхньої грані

⚠ ***Елемент u є верхньою гранню елементів a та b, якщо $u \geq a$ та $u \geq b$ за відношенням порядку в у-множині***

З цих визначень є зрозумілим, що в у-множині є декілька верхніх та нижніх граней. Відповідно, у будь-якій у-множині окремо визначають точну верхню та точну нижню грань.

⚠ ***Елемент x є найбільшою нижньою гранню (або точною нижньою гранню) елементів a та b, якщо він є їх нижньою гранню, та для будь-якої нижньої грані u $u \leq x$***

Найменша нижня грань для елементів a, b позначається як $x = inf\{a, b\}$ або $x = a \wedge b$. Слід зазначити, що поняття грані (верхньої та нижньої) є досить поширеним у математиці, та використовується у різних розділах математики, зокрема у рядах. Так само визначається точна верхня грань. Єдине, що змінюється у цьому формулюванні – знак нерівності.

⚠ ***Елемент y є найменшою верхньою гранню (або точною верхньою гранню) елементів a та b, якщо він є їх верхньою гранню, та для будь-якої верхньої грані u $u \geq x$***

Точна верхня грань позначається як $y = sup\{a, b\}$ або $y = a \vee b$. Не для кожної у-множини існує точна верхня та точна нижня грані. Наприклад, для у-множини, діаграма Хассе для якої наведена на рис., не існує точна грань[4].

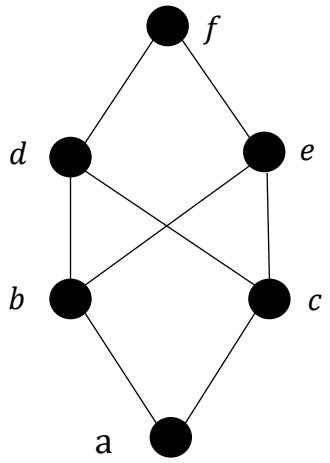


Рисунок 3.3 – Приклад у-множини, де відсутня точна грань

З діаграми Хасе на рис.3.3 видно, що для елемнтів d та e нижніми гранями будуть елементи b, c, a. Для елементу b це випливає з наступних нерівностей b ≤ d, b ≤ e. Для елементу c це випливає з нерівностей c ≤ d, c ≤ e. Відповідно, для елементу a справедливі наступні нерівності $a \leq d, a \leq e, a \leq b, a \leq c$. Виходячи з визначення, та діаграми Хасе «кандидатами» на найбільшу нижню грань є елементи b та c. Але між ними на діаграмі Хасе немає ребра. Такий випадок означає неможливість порівняти елементи b та c між собою, отже для елементів d та e немає найбільшої верхньої грані.

Інший випадок для пошуку найбільшої нижньої грані наведений на рис. 3.4

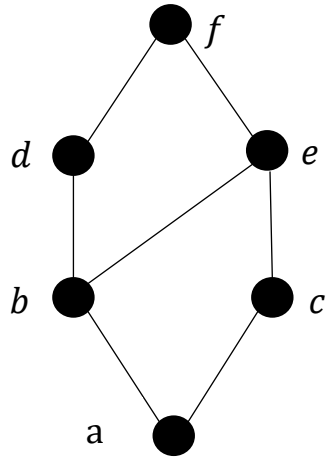


Рисунок 3.4 - Приклад у-множини

На рис. 3.4. нижньою гранню для елементів d та e є елементи b та a, оскільки $b \leq d$, $b \leq e$ та $a \leq d$, $a \leq e$. Водночас, $a \leq b$, і це обумовлює, що b є точною нижньою гранню для елементів d та e.

Також, для елементів e та c $c \leq e$ та a ≤ e. Оскільки a ≤ c, то найбільшою нижньою гранню елементів e та c є елемент c

Введемо у розгляд визначення ришітки:

⚠ ***У-множина, у якій для кожних двох елементів існує точна верхня та точна нижня грані називаються ришітками.***

Операцію знаходження точної верхньої та точної нижньої грані часто називають, відповідно, «перетином» та «об'єднанням», за аналогією з відомими теоретико-множинними операціями. Оскільки операції знаходження цих граней мають сенс лише для у-множин, то алгебра, яка має, як основу, у-множину та операції об'єднання та перетину так само мають назву ришіток.

Найпростішим прикладом ришітки є алгебра множин. Оскільки усі теоретико-множинні операції визначені на булеані, у випадку кінцевої множини-універсуму, можливо побудувати діаграму Хасе для множин. Наприклад, для трьохелементної множини $A = \{a, b, c\}$ така діаграма Хасе буде мати наступний вигляд

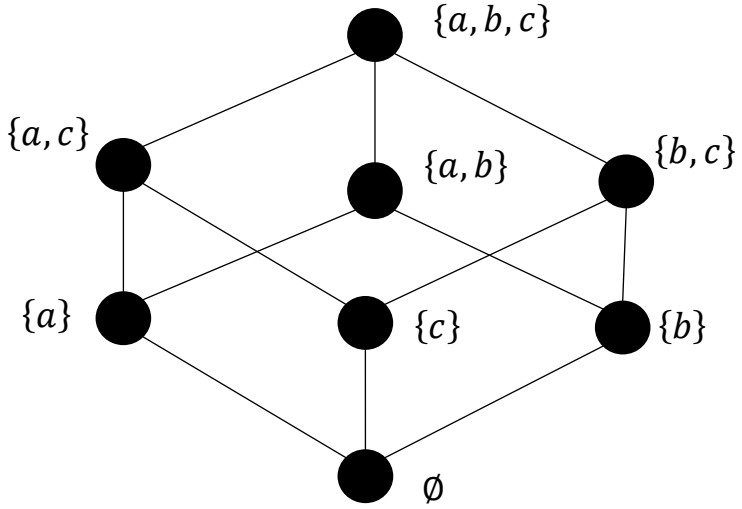


Рисунок 3.5 - Приклад діаграми Хасе для кінцевої множини

Аналізуючи діаграму Хасе можна прийти до висновку, що вона ілюструє ришітку, яка заснована на алгебрі множин, в основі якій булеан множини $\{a, b, c\}$. Пропонуємо виконати це самостійно.

Приклади у-множин для чисел наведено на рис. 3.6.

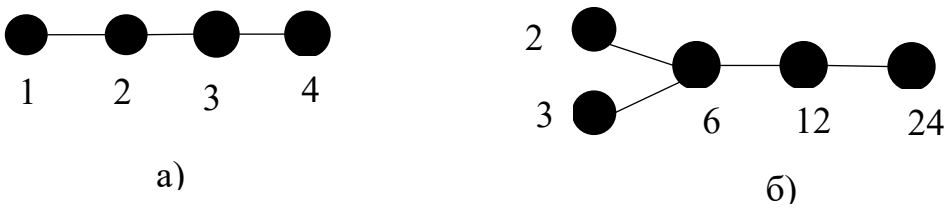


Рисунок 3.6 - Приклади у-множин для чисел

На рис. 3.6а зображено найпростіший приклад ланцюга для цілих чисел, який будується за допомогою відношення «х більше у». Діаграма Хасе на рис. 3.6б побудована за відношенням «х ділить у».

На множині цілих чисел може бути задано й ришітку. Наприклад, на множині $A = \{1,2,3,5,6,10,15,30\}$ із заданим на ньому відношенням «х дільник у» є ришіткою, де точна нижня грань – це найбільший спільний дільник х та у, а точна верхня грань – це найменше спільне кратне х та у.

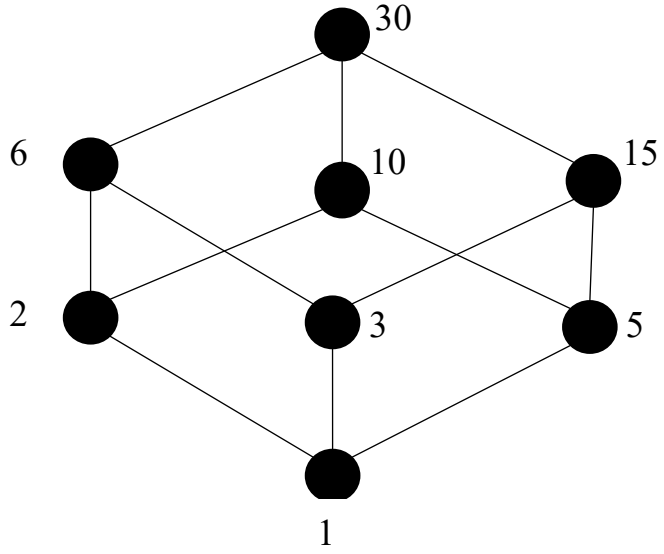


Рисунок 3.7 - Діаграма Хасе "х ділить у"

3.3 Завдання до розділу 3

Завдання 3.1. Визначити формально операцію «Посадка пасажирів у автобус».

Завдання 3.2. Визначити формально операцію «Завезення нових товарів на склад»

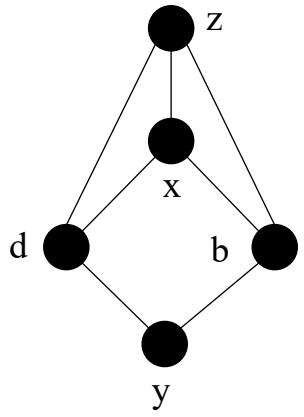
Завдання 3.3. Визначити формально операцію «Паркування таксі у таксопарку»

Завдання 3.4. Визначити формально операцію «Зміна фамілії студента»

Завдання 3.5.Визначити формально операцію «Виставлення оцінки з дисципліни».

Завдання 3.6. Визначити формально операцію «Завантаження товарів на склад із машини».

Завдання 3.7. Визначити точну нижню та точну верхню грань (за наявності) для елементів d та b у наступній діаграммі Хассе для y-множини із відношенням $a > b$ *(більші за значенням елементи розташовані у верхній частині діаграми)*



4 ПРЕДИКАТИ. БУЛЕВІ ФУНКЦІЇ

4.1 Предикати

Визначення предикату. Важливу роль під час моделювання об'єктів реального світу відіграють предикати. Саме предикати дозволяють перевіряти у моделі наявність чи відсутність деяких властивостей у елементів множині.

⚠ ***Будь-яке відображення*** $\boldsymbol{P}: \boldsymbol{A} \to \{\mathbf{0}; \mathbf{1}\}$ ***довільної множини*** $\boldsymbol{A}$ ***у двоелементну множину*** $\{\mathbf{0}; \mathbf{1}\}$ ***називається предикатом.***

У залежності від того, чи є множина А декартовим добутком інших множин, розрізнюють одномісні та багатомісні предикати. Наприклад, $P: A \to \{0; 1\}$ – одномісний предикат, $P: A \times B \to \{0; 1\}$ – двомісний предикат, $P: A^2 \to \{0; 1\}$ також двомісний предикат.

Якщо множина А дорівнює декартовому добутку множин – предикат однорідний. Якщо множина А дорівнює декартовому ступеню множин – предикат неоднорідний.

⚠ ***Множина значень, на якій предикат є істинним називається областю істинності предикату.***

Приклад 4.1. Визначимо предикат, який є істинним для всіх власників банківських рахунків, які обслуговуються більше п'яти років та мають депозитні рахунки терміном два роки або мають суму вкладу вище за 200 000 грн.

Для визначення цього предикату побудуємо формальний опис вкладника. $Client = F \times Sum \times Date_0$ *де* $Client$ *– множина клієнтів банку,* $Sum = \{s: s \in \mathbb{Z} \text{ та } s \in [0; 10^9]\}$ *– множина, яка визначає суму вкладів,* $Date_0$ *– множина дат відкриття вкладів. Дата відкриття виражається за формулою:* $Date_0 \subset Date$, *де* $Date = D \times M \times Y$, *де* $D = \{d: d \in \mathbb{Z} \text{ та } d \in [1; 31]\}$, $M = \{m: m \in \mathbb{Z} \text{ та } m \in [1; 12]\}$, $Y = \{y: y \in \mathbb{Z} \text{ та } y \in [1900; 2100]\}$. *Визначимо заданий предикат наступним чином:* $P: Client \times Date_0 \to \{0; 1\}$. *Область істинності заданого предикату* P *є множиною двоелементних векторів* $\{(client; date): \text{пр}_2 client > 200000 \text{ та } date - \text{пр}_3 client > 732\}$. *Операція різниці вживається у розумінні різниці між двома датами.*

У розглянутому прикладі предикат $\text{P: Client} \times Date \to 0; 1$ можна розглядати й по іншому. Можна вважати, що предикат $\text{P: Client} \times Date \to 0; 1$ для деяких $\text{client} \in Client$ та $\text{date} \in Date_0$ (це позначається як $\text{P}(client, date)$) є істинним, якщо для клієнта з множини Client має значення функція $\text{f}(client)$ її значення дорівнює date. Узагальнено це можна сформулювати у вигляді наступного твердження

⚠ ***Будь-якій функції*** $\boldsymbol{f: M^n \to M}$ ***можна поставити у відповідність*** $\boldsymbol{n+1}$***-місний предикат*** $\boldsymbol{P: M^{n+1} \to \{0; 1\}}$

Фактично, це твердження показує, що будь-яку область істинності предикату можна задавати за допомогою функціональних відповідностей, що й було зроблено у попередньому прикладі.

Введемо у розгляд поняття висловлювання.

⚠ ***n*-місний предикат $P(x_1, x_2, \dots, x_n)$ у якого $x_1 = a_1$, $x_1 = a_2$, …, $x_i = a_i$, …, $x_n = a_n$ $(a_1, a_2, \dots, a_i, \dots, a_n = const)$ є висловлюванням**

Приклад 4.2 У Приклад 4.1 вектор $client \in Client$, $client = (\text{Іванов}, 3000, (23; 10; 2018)$ *та* $date \in Date_0$ $date = (23; 10; 2019)$ *перетворює предикат P з цього прикладу у висловлювання.*

Якщо аналізувати області істинності з точки зору теорії множин, стає зрозумілим співвідношення між логічними сполучниками «ТА», «АБО» та логічною часткою «НІ» та відомими теоретико-множинними операціями перетину, об'єднання та доповнення. Область істинності предикату P, який є логічною комбінацією предикатів P_1 та P_2 зі сполучником «ТА» ($P = P_1 \wedge P_2$), є перетином областей істинності предикатів P_1 та P_2. Область істинності предикату P, який є логічною комбінацією предикатів P_1 та P_2 зі сполучником «АБО» ($P = P_1 \vee P_2$), є об'єднанням областей істинності предикатів P_1 та P_2. Нарешті, областю істинності предикату, P перед яким стоїть логічна частка «НІ» є доповненням області істинності предикату P. Предикати, які з'єднані логічними сполучниками надалі будемо називати складними предикатами.

Поняття квантору. Важливим поняттям теорії предикатів є поняття квантору. Квантор є узагальненням логічних сполучників «ТА» та «АБО». Існують два типи кванторів: загальності та існування. Квантор загальності стверджує справедливість предикату **P** для всіх елементів деякої множини **x**. Це позначається як $\forall \boldsymbol{x}\mathbf{P}(\boldsymbol{x})$. З цього запису має бути відома область визначення **x**. Квантор існування стверджує, що для деяких елементів, які належать області істинності предикату **P** справедливе твердження цього предикату. Цей випадок відповідно позначається $\exists \boldsymbol{x}\mathbf{P}(\boldsymbol{x})$.

Якщо є деяка кінцева множина $X = x_1, x_2, \dots, x_n$, то можна встановити наступне співвідношення між логічними сполучниками «ТА», «АБО» та кванторами загальності та існування відповідно:

$$\forall x \mathrm{P}(x) \sim \mathrm{P}(x_1) \wedge P(x_2) \wedge \dots \wedge P(x_n) \tag{4.1}$$

$$\exists x \mathrm{P}(x) \sim \mathrm{P}(x_1) \vee P(x_2) \vee \dots \vee P(x_n) \tag{4.2}$$

Позначення $\sim$ (еквівалентність) у цьому випадку означає, що ці записи є еквівалентними за змістом, але різними за формою запису.

У випадку узагальнення поняття кванторів на нескінченні множині, формули $\forall x \mathrm{P} x \sim \mathrm{P}(x_1) \wedge P(x_2) \wedge \dots \wedge P(x_n)$ (4.1) та $\exists x \mathrm{P}(x) \sim \mathrm{P}(x_1) \vee P(x_2) \vee \dots \vee P(x_n)$(4.2) можливо переписати у наступному вигляді:

$$\forall x \mathrm{P}(x) \sim \bigwedge_{i=1}^{\infty} P(x_i) \tag{4.3}$$

$$\exists x \mathrm{P}(x) \sim \bigvee_{i=1}^{\infty} P(x_i) \tag{4.4}$$

Проілюструємо далі використання кванторів на прикладі.

Приклад 4.3. Раніше, у Приклад 4.1 ми ввели у розгляд формальний опис вкладника. Введемо далі формальний опис банківського відділення та сформулюємо предикат, який дозволяє визначити VIP-відділення. VIP-відділення будемо визначати за наступною ознакою: у VIP-відділенні усі клієнти відповідають вимогам, які задані для предикату, який сформульовано у Приклад 4.1. Формальний опис відділення введемо за допомогою наступної формули: $Branch = N \times Adr \times \mathcal{P}(Client)$. *Перша проекція вектору* $br \in Branch$ *містить номер відділення, друга проекція містить адресу відділення, третя проекція містить*

булеан множини клієнтів. Множина клієнтів була визначена у Приклад 4.1. Сформулюємо далі предикат для VIP-відділення: $P_{vip}: Branch \to 0;1$. *Область істинності цього предикату має наступну характеристичну властивість:* $M_{vip} = br: \forall \text{пр}_3 br P(\text{пр}_3 br)$. *У цьому прикладі* P *є предикатом, визначеним у Приклад 4.1*

Для кванторів доведено декілька властивостей, які дозволяють переходити від квантору загальності до квантору існування та навпаки.

$$\overline{\forall x P(x)} \sim \overline{\exists x P(x)} \tag{4.5}$$

Формула $\forall x P x \sim \overline{\exists x P(x)}$ (4.5) стверджує, що заперечення квантора існування є еквівалентним запереченню квантора загальності. Справедливість формули $\forall x P x \sim \overline{\exists x P(x)}$ (4.5) випливає безпосередньо з формул $\forall x \text{P}(x) \sim \text{P}(x_1) \wedge P(x_2) \wedge \ldots \wedge P(x_n)$ (4.1) $\exists x \text{P} x \sim \text{P}(x_1) \vee P(x_2) \vee \ldots \vee P(x_n)$ (4.2), а також із самої суті квантору загальності.

Наведемо ще декілька властивостей кванторів:

$$\forall x \vdash (P\_1 \vdash (x \dashv) \& P\_2 \vdash (x \dashv) \dashv) \sim \forall x P\_1 \vdash (x \dashv) \& \forall x P\_2 \vdash (x \dashv) \tag{4.6}$$

$$\exists x \big(P_1(x) \vee P_2(x)\big) \sim \exists x P_1(x) \vee \exists x P_2(x) \tag{4.7}$$

Ці властивості схожі на властивості, які справедливі для похідної суми двух функцій. Також ці властивості мають умовну назву «дистрибутивність для кванторів».

Очевидно, що предикат, який з'єднаний сполучником «ТА» (позначений у формулі $\forall x \vdash (P\_1 \vdash (x \dashv) \& P\_2 \vdash (x \dashv) \dashv) \sim \forall x P\_1 \vdash (x \dashv) \& \forall x P\_2 \vdash (x \dashv)$ (4.6) знаком &) та до змінної якого застосовано квантор загальності, буде вірним лише в

тому випадку, якщо обидва предикати будуть вірними для елементу, який знаходиться під квантором. Те саме стосується й формули $\exists xP1x \vee P_2(x) \sim \exists xP_1(x) \vee \exists xP_2(x)$ (4.7).

Якщо у вказаних співвідношеннях замінити квантор загальності на квантор існування і навпаки, то отримаємо наступні вирази:

$$\exists x \vdash(P\_1 \vdash(x\dashv)\&P\_2 \vdash(x\dashv)\dashv) \to \exists xP\_1 \vdash(x\dashv)\&\exists xP\_2 \vdash(x\dashv) \quad (4.8)$$

$$\forall x\big(P_1(x) \vee P_2(x)\big) \to \forall xP_1(x) \vee \forall xP_2(x) \quad (4.9)$$

Дійсно з того, що існують x які задоволняють складеному предикату випливає їх існування або загальність для хоча б одного з наведених предикатів.

Наведемо ще декілька формул, які справедливі для кванторів, не наводячи їх докази:

$$\forall x\forall y\mathrm{P}(x,y) \sim \forall y\forall x\mathrm{P}(x,y) \quad (4.10)$$

$$\exists x\exists y\mathrm{P}(x,y) \sim \exists y\exists x\mathrm{P}(x,y) \quad (4.11)$$

Властивості $\forall x\forall y\mathrm{P}(x,y) \sim \forall y\forall x\mathrm{P}(x,y)$ (4.10) та $\exists x\exists y\mathrm{P}x,y \sim \exists y\exists x\mathrm{P}(x,y)$ (4.11) ілюструють те, що порядок змінних, які розташовані у кванторах не впливає на значення квантору.

Нехай Y – предикат, який не містить x. Тоді для цього предикату справедливі наступні твердження:

$$\forall x(\mathrm{P} \vdash(x\dashv)\&\mathrm{Y}) \sim \forall x\mathrm{P} \vdash(x\dashv)\&\mathrm{Y} \quad (4.12)$$

$$\forall x(P(x) \vee Y) \sim \forall x\mathrm{P}(x) \vee Y \quad (4.13)$$

$$\exists x \vdash (P \vdash (x \dashv) \& Y \dashv) \sim \exists x \mathrm{P} \vdash (x \dashv) \& \mathrm{Y} \quad (4.14)$$

$$\exists x (P(x) \vee Y) \sim \exists x \mathrm{P}(x) \vee Y \quad (4.15)$$

Справедливість цих властовостей обумовлена тим, що оскільки Y не залежить від x він не впливає на значення виразу у дужках у лівій частині виразу. Тому його можна виносити безпосередньо за дужки.

Застосування предикатів у програмній інженерії. Головною сферою застосування предикатів та кванторів у програмній інженерії є визначення умов альтернатив та циклів, а також встановлення порядку вкладення альтернатив та циклів. Розглянемо це на прикладах

Необхідно визначити приналежність точки одній з наступних областей (Рис.)

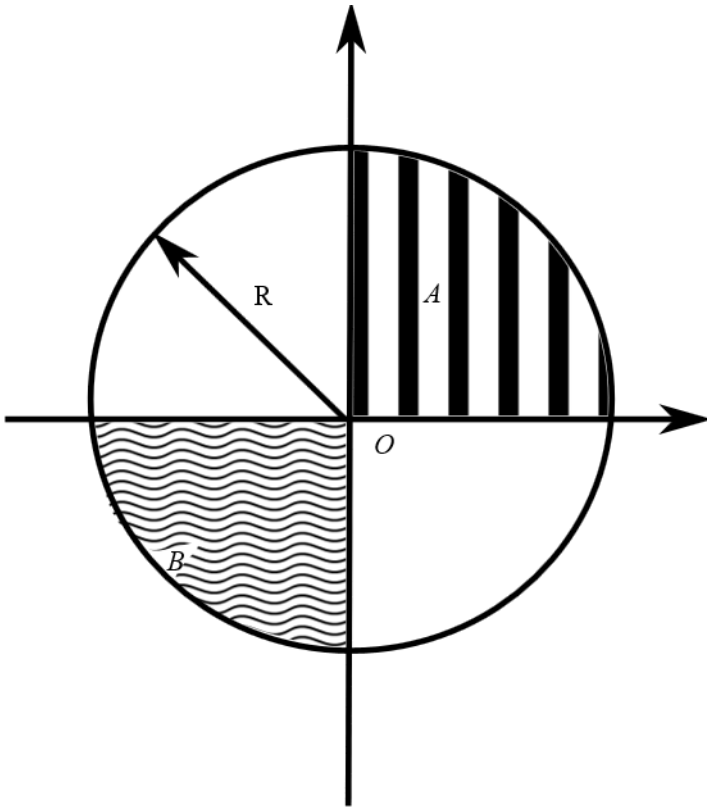


Рисунок 4.1 - Області для яких треба визначити приналежність

Уся координатна площина задається наступним декартовим добутком: $\mathrm{Coord} = Q \times Q$. На рис. позначено, що областю A є область, яка знаходиться у першій координатній чверті (ця область позначена широкими вертикальними смугами). Цю координатну чверть задає відповідність $Coord_0 \subset C\mathrm{oord}$. Розглянемо побудову характеристичної властивості цієї відповідності. Для побудови характеристичної властивості будемо використовувати предикати та їх області істинності. Область A характеризується комбінацією декількох простих предикатів. Перший з них має вид $\mathrm{P_{a1}}\colon \mathrm{Q}^{\mathbb{2}} \to 0; 1$ та характеризується наступною областю істинності: $\mathrm{M_{a1}} = (\mathrm{x}; \mathrm{y})\colon \mathrm{x} \geq 0$. Другий з них має вид $\mathrm{P_{a2}}\colon \mathrm{Q}^{\mathbb{2}} \to 0; 1$ та область істинності $\mathrm{M_{a2}} = (\mathrm{x}; \mathrm{y})\colon \mathrm{y} \geq 0$. Третій з них має вид $\mathrm{P_{a3}}\colon \mathrm{Q}^{\mathbb{3}} \to 0; 1$ та область істинності $\mathrm{M_{a3}} = (\mathrm{x}; \mathrm{y}, \mathrm{r})\colon \mathrm{x}^2 + \mathrm{y}^{2\backslash \mathrm{ler}^2}$. Усі перераховані області істинності мають перетинатись для утворення множини A, тобто $\mathrm{A} = \mathrm{M}_{\mathrm{a1}\backslash \mathrm{capM}_{\mathrm{a2}\backslash \mathrm{capM}_{\mathrm{a3}}}}$. Відповідно, предикат, який відповідає цій характеристичній властивості має вигляд $\mathrm{P_a} = \mathrm{P}_{\mathrm{a1}\backslash \mathrm{landP}_{\mathrm{a2}\backslash \mathrm{landP}_{\mathrm{a3}}}}$. Для області B (виділена на рис. хвилястими лініями) вводяться наступні предикати:$\mathrm{P_{b1}}\colon \mathrm{Q}^{\mathbb{2}} \to 0; 1$, $\mathrm{P_{b2}}\colon \mathrm{Q}^{\mathbb{2}} \to 0; 1$, $\mathrm{P_{b3}}\colon \mathrm{Q}^{\mathbb{3}} \to 0; 1$, із областями істинності відповідно $\mathrm{M_{b1}} = (\mathrm{x}; \mathrm{y})\colon \mathrm{x} < 0$; $\mathrm{M_{b2}} = (\mathrm{x}; \mathrm{y})\colon \mathrm{y} < 0$; $\mathrm{M_{b3}} = (\mathrm{x}; \mathrm{y}; r)\colon \mathrm{x}^2 + \mathrm{y}^{2\backslash \mathrm{ler}^2}$. З характеристичних властивостей областей істинності випливає наступне твердження: $\mathrm{M_{b3}} = \mathrm{M_{a3}}$. Наслідком цієї рівності є еквівалентність предикатів $\mathrm{P_{b3}}$ та $\mathrm{P_{a3}}$. Аналогічно до області A, область B задається перетином областей істинності відповідних предикатів, а саме: $\mathrm{B} = \mathrm{M}_{\mathrm{b1}\backslash \mathrm{capM}_{\mathrm{b2}\backslash \mathrm{capM}_{\mathrm{a3}}}}$. Відповідно предикати мають наступний вид: $\mathrm{P} = \mathrm{P}_{\mathrm{b1}\backslash \mathrm{landP}_{\mathrm{b2}\backslash \mathrm{landP}_{\mathrm{a3}}}}$. Відповідь на питання, чи належить точка заштрихованим областям, чи ні, формулюється як питання приналежності вектору (x, y): $(\mathrm{x}, \mathrm{y}) \in \mathrm{A} \cup \mathrm{B}$.

Замінимо множини A та B на їх аналітичне представлення: $\left(\mathrm{M}_{\mathrm{a1}\backslash \mathrm{capM}_{\mathrm{a2}\backslash \mathrm{capM}_{\mathrm{a3}}}}\right) \cup \left(\mathrm{M}_{\mathrm{b1}\backslash \mathrm{capM}_{\mathrm{b2}\backslash \mathrm{capM}_{\mathrm{a3}}}}\right)$. Застосуємо до цього виразу формулу

$A \cap B \cup C = (A \cap B) \cup (A \cap C)$ (1.13) у зворотньому напрямку, отримаємо: $M_{a3} \cap \left(\{(M\}_{a1 \backslash capM_{a2}}) \cup \left(M_{b1 \backslash capM_{b2}}\right)\right)$. Враховуючи з характеристичних властивостей областей істинності, що $M_{b1} = \overline{M_{a1}}$ та $M_{b2} = \overline{M_{a2}}$ отримаємо кінцевий результат: $M_{a3} \cap \left(\{(M\}_{a1 \backslash capM_{a2}}) \cup (\overline{M_{a1}} \cap \overline{M_{a2}})\right)$. Переходячи, відповідно, до предикатів, отримаємо. $P_{a3} \wedge \left(\left(P_{a1 \backslash landP_{a2}}\right) \vee (\overline{P_{a1}} \wedge \overline{P_{a2}})\right)$.

Для програмної реалізації цього прикладу оберемо мову Сі. Кожен тип цієї мови є, по суті, множиною значень змінної. При визначенні типу даних, який відтворює множину значень з реального світу, слід враховувати інтерпретацію відношення включення, а також той факт, що множина значень обраного типу даних має бути найбільшою за потужністю та мати бієктивну відповідність зі значеннями з предметної області.

Згідно задачі, формальний опис значень, які обробляються має вигляд $Q^{\mathbb{3}}$ Це означає, що у програмі має бути використано три змінних. Якщо говорити про тип змінних, то вимогам до такого типу відповідає тип double мови С. Враховуючи все вищевикладене, можна отримати наступний вихідний код (враховуючи формальні представлення операцій вводу-виводу, наведені у попередньому розділі)

```
#include <stdio.h>
#include <math.h>
int main()
{
double x;
double y;
double r;
printf("Enter x:\n");
scanf("%lf",&x);
printf("Enter y:\n");
scanf("%lf",&y);
printf("Enter r:\n");
```

```
scanf("%lf",&r);

if ((pow(x,2)+pow(y,2))<=pow(r,2))
{
   if (((x>=0) && (y>=0)) || ((x<0) && (y<0)))
   {
        if ((x>=0) || (y>=0))
        {
            printf("Point in area A");
        }
        else
        {
            printf("Point in area B");
        }
    }
    else
    {
      printf("Point not in area A or B, but in circle");
    }

}
else
{
printf("The point does not belong to either region A or region B");
}
return 0;
}
```

Для компіляції цього коду можна використати компілятор Qt або будь-який інший компілятор, який відповідає стандарту C99. Такий самий код можна отримати й на будь-якій іншій мові програмування, якщо врахувати, що предикати мають бути присутніми в умовних операторах, відомий синтаксис запису логічних сполучників та операторів вводу-виводу даних.

Якщо поданий код аналізувати з точки зору логічних виразів, то можна побачити, що послідовне вкладене виконання операторів цілком вкладається у виконання операції «ТА»

Розглянемо запис умовного оператору у вигляді:

```
if P(x)
{
}
if P(y)
{
}
```

Такий запис умовного оператору буде еквівалентний логічній операції «АБО».

Квантори загальності та існування можуть свідчити у формальному запису про необхідність циклу, його вид та положення умови предикату у даному циклі.

Раніше було розглянуто представлення будь-якої множини у пам'яті комп'ютера, вона представляється як відповідність вигляду $\mathrm{Array} \subset \mathbb{Z} \times A$. Перша проекція вектору $\mathrm{array} \in A\mathrm{rray}$ цієї множини символізує адресу у пам'яті комп'ютера, друга є елементом довільної множини A. Вказана відповідність обов'язково є бієкцією. Враховуючи такий опис пам'яті, застосування квантору існування буде призводити або до повного перебору масиву за допомогою циклу for та вкладеної умови, або до часткового перебору, за допомогою циклу із постумовою, яка є предикатом, на якій навішаний квантор. Узагальнена інформація щодо використання квантору існування у програмуванні наведена у Таблиця 4.1

Таблиця 4.1 - Використання квантору існування у програмуванні

Математична характеристика	Знаходження усіх елементів у множині, що відповідають предикату	Знаходження хоча б одного елементу, який задоволняє предкату

$\exists el$: P(el)
Де P – предикат з областю істинності Array. el ∈ Array
	for (i=1;i<$\|A\|$;i++)
	{
	If (P(el))
	{// Обробка чергового знайденого елементу}
	}
		Do
		{
		}while (P(el)) & (i > $\|A\|$))
		If (P(el)) {// Обробка знайденого елементу};

У Таблиця 4.1 використовується наближений до більшості мов програмування синтаксис операторів альтернативи та циклу. Вихід з циклу передбачається за істинним значенням умови циклу. Символом $|A|$ позначається потужність множини A, i ∈ Z, el ∈ Array

Квантор загальності передбачає, що предикат є справедливим обов'язково для усіх елементів відповідної множини. Тому, в цьому випадку використовується трохи інший прийом, який наведено у Таблиця 4.2

Таблиця 4.2 – Використання квантору загальності у програмуванні

Математичний зміст квантору загальності	Фрагмент коду, що відповідає перевірці квантору загальності
$\forall el$: P(el) Де P – предикат з областю істинності Array. el ∈ Array. Array ⊂ $Z \times A$	Flag:=1 Do { flag = flag & P(el) } while (flag) & i > $\|A\|$

Значення операторів та правила виходу з циклу прийняті такі сами, як і у Таблиця 4.1. Додатково використовується оператор присвоєння значення змінної.

Таким чином, якщо у формальному описі даних для деякої предметної області присутні квантори, це однозначно вказує на те, що при обробці предикату повинен бути присутній цикл, а головною сферою використання предикатів у мовах програмування є умовні оператори різних типів (умови циклу, умови альтернативи).

Однією з головних проблем під час опису умовних операторів є велика кількість роботи, яку необхідно робити, якщо використовувати підхід, пов'язаний із областями істинності цих предикатів. На початку цього параграфу був проілюстрований саме такий підхід. Більш гнучким методом побудови скалдних умов є використання алгебри булевих функцій

4.2 Булеві функції

Визначення булевої функції. Ще одним інструментом для побудови логічних умов є булева функція.

⚠ ***Булева функція від n змінних (функція алгебри логіки) – це довільне відображення виду $f: \{0; 1\}^n \to \{0; 1\}$***

Булева функція приймає значення 0 або 1 на множині векторів довжини n. Зрозуміло,що множина цих векторів є кінцевою множиною, як і множина значень. Отже, булева функція належить до класу кінцевих функцій.

⚠ ***Кінцева функція – це відображення однієї кінцевої множини в іншу***

Аргументами булевої функції є булеві змінні.

⚠ ***Булева змінна – це змінна, що приймає значення з множини*** $\{\mathbf{0};\mathbf{1}\}$

Аргументи булевої функції являють собою вектор, який зветься двійковим набором. Якщо $n = 2$ існує лише 4 варіанту наборів $\{(0;0);(0;1);(1;0);(1;1)\}$. Якщо $\mathrm{n} = 1$ то існує лише два набори – $\{0;1\}$. Останнє твердження випливає з того, що перший декартовий ступінь будь-якої множини є самою цією множиною. Деякій двійковий набір для булевої функції будемо позначати грецькою літерою з тильдою над нею, наприклад: $\tilde{\alpha}$, $\tilde{\beta}$ тощо.

Булеві функції однієї та двох змінних. Виходячи з числа двійкових наборів, а також з того, що булева функція є кінцевою функцією, можна встановити, що існує 4 булевих функції однієї змінної:

Таблиця 4.3 - Булеві функції однієї змінної

x	f_0	f_1	f_2	f_3
0	0	0	1	1
1	0	1	0	1
	Константа «0»	Зберігання x	$\bar{x}$	Константа «1»

Якщо уважно подивитись на другий та третій рядок таблиці, то можна побачити, що вектори значень усіх наведених у таблиці функцій є двоелементним двійковими векторами. Як зазначалося вище, потужність множини цих векторів дорівнює 4, що й обумовлює саме таку кількість функцій.

Якщо за тим самим принципом розглядати функції двох змінних, отримаємо таблицю з 16 можливих функцій

Таблиця 4.4 - Таблиця булевих функцій двох змінних

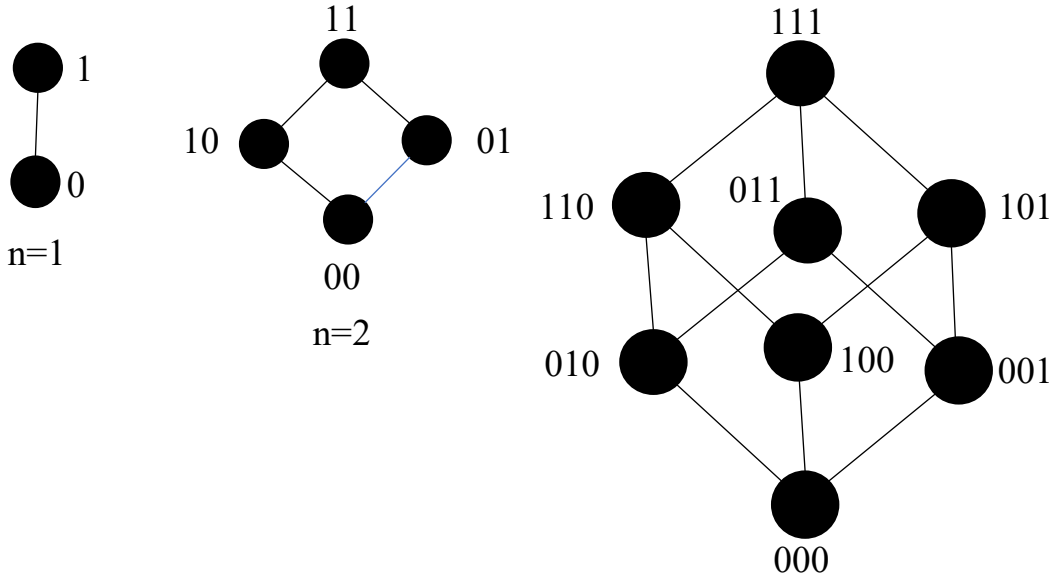
x_1	x_2	f_0	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}	f_{12}	f_{13}	f_{14}	f_{15}
0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

Таблиця 4.5 - Назви булевих функцій двох змінних

Позначення в таблиці 4.4	Назва	Позначення	Позначення в таблиці 4.4	Назва	Позначення
f_0	Константа «0»	0	f_{15}	Константа «1»	1
f_1	Кон'юнкція	$x_1 \wedge x_2$	f_{14}	Штріх Шеффера	$x_1 \mid x_2$
f_2	Інверсія імплікації	$x_1 \not\Rightarrow x_2$	f_{13}	Імлікація	$x_1 \rightarrow x_2$
f_3	Збереження x_1	x_1	f_{12}	Заперечення x_1	$\bar{x}_1$
f_4	Інверсія зворотньої імлікації	$x_1 \not\Leftarrow x_2$	f_{11}	Зворотня імлікація	$x_1 \leftarrow x_2$
f_5	Збереження x_2	x_2	f_{10}	Заперечення x_2	$\bar{x}_2$
f_6	Додавання за модулем 2	$x_1 \oplus x_2$	f_9	Еквівалентність	$x_1 \sim x_2$
f_7	Диз'юнкція	$x_1 \vee x_2$	f_8	Стрілка Пірса	$x_1 \downarrow x_2$

Якщо подивитись на Таблиця 4.5 можна побачити досить цікаве співвідношення. У кожній парі функцій у цій таблиці кожна функція є інверсією (доповненням) функції, яка знаходиться із нею в парі. Таке явище зветься подвійністю булевих функцій

Будемо розглядати множину $\{0; 1\}$ як у-множину. Відношення порядку на цій у-множині визначає, що 0 передує 1. В цьому випадку операції кон'юнкції та диз'юнкції ведуть себе аналогічно до відповідних операції «перетину» та «об'єднання» у ришітках. Отже, алгебра булевих функцій, яка складається з диз'юнкції та кон'юнкції є ришіткою у звичному розумінні цього слова. Відповідно, аргументами n-місної булевої функції будуть вектори довжини n, які є елементами множини $\{0; 1\}^n$. Ці вектори мають назву булевих векторів. Їх можна представити у вигляді наступних діаграм Хасе



Функції диз'юнкції, кон'юнкції та заперечення разом утворюють стандартний базис булевих функцій. Отже, постає питання зв'язку стандартного базису булевих функцій із іншими булевими функціями однієї та двох змінних. Це завдання може бути вирішене за допомогою аналітичної мінімізації булевих функцій та нормальних форм булевих функцій.

4.3 Аналітична мінімізація булевих функцій

Основні властивості булевих функцій. Як вже вказувалося вище, алгебра булевих функцій є ришіткою, так само як і алгебра множин. За пріорітетом операції цієї алгебри розподіляються наступним чином: найвищий пріорітет має унарна операція інверсії, слідом йде операція кон'юнкції та останньою виконується операція диз'юнкції. Оскільки алгебра булевих функцій належить до того ж самого класу алгебр, що й алгебра множин, для неї справедливі всі властивості алгебри множин, якщо встановити наступну відповідність між операціями: логічна інверсія відповідає доповненню множин, кон'юнкція відповідає перетину множин, а диз'юнкція – об'єднанню множин. Розглянемо докладно ці властивості.

Операції диз'юнкції та кон'юнкції є комутативними:

$$x_1 \vee x_2 = x_2 \vee x_1 \tag{4.16}$$

$$x_1 \wedge x_2 = x_2 \wedge x_1 \tag{4.17}$$

Операції кон'юнкції та диз'юнкції є асоціативними:

$$x_1 \wedge x_2 \wedge x_3 = x_1 \wedge (x_2 \wedge x_3) = (x_1 \wedge x_2) \wedge x_3 \tag{4.18}$$

$$x_1 \vee x_2 \vee x_3 = x_1 \vee (x_2 \vee x_3) = (x_1 \vee x_2) \vee x_3 \tag{4.19}$$

Зауважимо, що операція кон'юнкції двох змінних може бути записана без застосування знаку кон'юнкції, а операція диз'юнкції змінних може бути записана за допомогою знака «+». Тоді властивості комутативності та асоціативності записуються у наступному вигляді:

$$x_1 + x_2 = x_2 + x_1$$

$$x_1 x_2 = x_2 x_1$$

$$x_1 x_2 x_3 = x_1 (x_2 x_3) = (x_1 x_2) x_3$$

$$x_1 + x_2 + x_3 = x_1 + (x_2 + x_3) = (x_1 + x_2) + x_3$$

Наведені дві форми запису є цілком еквівалентними, тому надалі будуть використовуватися одночасно.

Операції диз'юнкції та кон'юнкції дистрибутивні по відношенню одна до одної:

$$x_1 \wedge (x_2 \vee x_3) = (x_1 \wedge x_2) \vee (x_1 \wedge x_3) \tag{4.20}$$

$$x_1 \vee (x_2 \wedge x_3) = (x_1 \vee x_2) \wedge (x_1 \vee x_3) \tag{4.21}$$

Операції кон'юнкції та диз'юнкції мають властивість ідемпотентності:

$$x \wedge x = x \tag{4.22}$$

$$x \vee x = x \tag{4.23}$$

Для операції логічної інверсії визначена властивість інвалютивності

$$\bar{\bar{x}} = \mathrm{x} \tag{4.24}$$

Також для кон'юнкції та диз'юнкції справедливі наступні твердження:

$$1 \vee x = 1 \tag{4.25}$$

$$0 \vee x = \text{x} \tag{4.26}$$

$$1 \wedge \text{x} = \text{x} \tag{4.27}$$

$$0 \wedge \text{x} = 0 \tag{4.28}$$

Ці властивості нагадують за змістом властивості нуля та одиниці для теоретико-множинних операцій перетину та об'єднання, якщо вважати універсум – логічною одиницею, а пусту множину – логічним нулем. Так само існують властивості констант для інверсії:

$$\bar{1} = 0 \tag{4.29}$$

$$\bar{0} = 1 \tag{4.30}$$

Так само, як для операцій перетину та об'єднання, для операцій кон'юнкції та диз'юнкції існує властивість поглинання, яка записується у наступному вигляді

$$x_1 \vee (x_1 \wedge x_2) = x_1 \tag{4.31}$$

$$x_1 \wedge (x_1 \vee x_2) = x_1 \tag{4.32}$$

Властивість Де-Моргана для кон'юнкції та диз'юнкції записується наступним чином:

$$\overline{x_1 \wedge x_2} = \overline{x_1} \vee \overline{x_2} \tag{4.33}$$

$$\overline{x_1 \vee x_2} = \overline{x_1} \wedge \overline{x_2} \tag{4.34}$$

Додаткові властивості булевих функцій. Окрім перелічених властивостей, булеві функції мають додаткові властивості, що пов'язують більш складні логічні операції зі стандартним базисом булевих функцій. Розглянемо ці співвідношення більш детально.

Логічні функції еквіваленції ($\sim$) та суми за модулем 2 ($\oplus$) представляються у стандартному базисі у вигляді наступних співвідношень

$$x_1 \sim x_2 = \bar{x}_1\bar{x}_2 + x_1x_2$$

$$x_1 \oplus x_2 = \bar{x}_1x_2 + x_1\bar{x}_2$$

Пряма та зворотня імплікації записуються у наступному вигляді

$$x_1 \to x_2 = \bar{x}_1 + x_2$$

$$x_1 \leftarrow x_2 = x_1 + \bar{x}_2$$

Операції інверсії прямої та зворотньої імплікації записуються у стандартному базисі наступним чином:

$$x_1 \bar{\to} x_2 = x_1\bar{x}_2$$

$$x_1 \leftarrow x_2 = \bar{x}_1x_2$$

Операції штрих Шеффера та стрілка Пірса можливо у стандартному базисі записати наступним чином:

$$x_1 \downarrow x_2 = \overline{x_1 + x_2} = \bar{x}_1\bar{x}_2$$

$$x_1|x_2 = \overline{x_1x_2} = \bar{x}_1 + \bar{x}_2$$

Виконання аналітичної мінімізації (спрощення) проводиться аналогічно спрощенню виразів у алгебрі множин. Головним інструментом є виконання символьної частини виразу на тотожній їй вираз згідно відповідної тотожності. Розглянемо цю заміну на прикладі.

Приклад 4.4 Спростити аналітично вираз $(x_1 \downarrow x_2) \to x_3$. *У дужках знаходиться операція стрілка Пірса. Розкриємо її за визначеням. Отримаємо наступний результат* $(\overline{x_1 + x_2}) \to x_3$. *Імплікацію, першим аргументом якої є дужки,*

а другим – змінна x_3 *також розкладемо за визначенням. Отримаємо наступний результат:* $\left(\overline{\overline{x_1 + x_2}}\right) + x_3$. *Далі, для дужок використовуємо властивість інвалютивності та отримаємо вираз* $(x_1 + x_2) + x_3$. *Застосовуючи властивість асоціативності будемо мати наступний кінцевий варіант виразу* $x_1 + x_2 + x_3$

Виконання саме аналітичної мінімізації вимагає завдання булевої функції аналітично, що можливо далеко не завжди. Тому існують методи мінімізації, що базуються на таблиці істинності булевої функції. Результатом цих методів буде аналітичне завдання булевої функції у стандартному базисі, що зветься нормальною формою булевої функції. Розглянемо ці методи більш детально

4.4 Нормальні форми булевих функцій.

Поняття імліканти. Для визначення цих понять введемо у розгляд поняття літералу

⚠ ***Літерал – це позначення булевої змінної x або її заперечення*** $\bar{x}$

Досить часто у математичній літературі використовується позначення літералу за допомогою значень з двоелементної множини $\{0; 1\}$. Конкретне значення з цієї множини у цьому випадку позначається грецькою літерою σ. Тоді сам літерал буде визначено за допомогою наступною формули:

$$x^{\sigma} = \begin{cases} x\,, \sigma = 1 \\ \bar{x}\,, \sigma = 0 \end{cases} \tag{4.35}$$

Також замість цієї формули використовують скорочене позначення $\tilde{x}$, розуміючи під цим позначенням будь-який з двох літералів x або $\bar{x}$.

Використовуючи поняття літералу, визначимо далі поняття елементарної кон'юнкції та елементарної диз'юнкції

⚠ ***Формула, що має вигляд $\widetilde{x_1}\widetilde{x_2}\widetilde{x_3}\ldots\widetilde{x_n}$ де колжна зі змінних може входити до формули лише один раз, називається елементарною кон'юнкцією літералів $\widetilde{x_1}, \widetilde{x_2}, \widetilde{x_3}, \ldots \widetilde{x_n}$***

Аналогічно визначається поняття елементарної диз'юнкції.

Формула, що має вигляд $\widetilde{x_1} + \widetilde{x_2} + \widetilde{x_3} + \ldots + \widetilde{x_n}$ де кожна зі змінних може входити до формули лише один раз, називається елементарною диз'юнкцією літералів $\widetilde{x_1}, \widetilde{x_2}, \widetilde{x_3}, \ldots \widetilde{x_n}$

Текстове формулювання цих визначень ще раз ілюструє принцип подвійності булевих функцій. Зокрема, у випадку цих визначень прицип подвійності полягає у можливості заміни понять «диз'юнкція» та «кон'юнкція» по всьому тексту визначення без втрати справедливості цих визначень. У цьому легко пересвідчитись, проаналізувавши тексти цих визначень.

Диз'юнктивна та кон'юнктивна нормальна форма. Спираючись на визначення елементарної диз'юнкції, кон'юнкції та літералу введемо у розгляд поняття диз'юнктивної та кон'юнктивної нормальної форми

⚠ ***Диз'юнктивна нормальна форма (ДНФ) булевої функції - це вираз виду $K_1 + K_2 + + \ldots + K_n$ де змінні $K_1, K_2, \ldots, K_n$ позначають елементарні***

кон'юнкції. Якщо до елементарної кон'юнкції K_i для кожного номеру $j = \overline{1;n}$ входить один та тільки один літерал $\tilde{x}_j$, така диз'юнктивна нормальна форма є досконалою диз'юнктивною нормальною формою (ДДНФ)

Зверніть увагу на істотну відмінність між визначенням елементарної кон'юнкції та досконалої диз'юнктивної нормальної форми. Елементарна кон'юнкція вимагає лише різниці змінних, у той час, як ДДНФ вимагає додатково присутності або змінної, або її інверсії, отже, ця вимога є більш суворою.

Використовуючи принцип двоїстості так само можна сформулювати визначенення КНФ та ДКНФ

⚠ ***Кон'юнктивна нормальна форма (КНФ) булевої функції - це вираз виду $D_1 D_2 D_3 \dots D_n$ де змінні $D_1, D_2, \dots, D_n$ позначають елементарні диз'юнкції. Якщо до елементарної диз'юнкції D_i для кожного номеру $j = \overline{1;n}$ входить один та тільки один літерал $\tilde{x}_j$, така кон'юнктивна нормальна форма є досконалою кон'юнктивною нормальною формою (ДКНФ)***

Методи формальної побудови ДНФ та КНФ засновані на поняттях констітуенти одиниці та нуля та понятті імпліканти.

⚠ ***Констітуентою одиниці булевої функції f, яка не є константою 0, це набір α для якого $f(\alpha) = 1$***

Зрозуміло, що для булевої функції, яка є константою нуля, немає значення одиниці, а отже поняття констітуенти одиниці в цьому випадку не має сенсу.

⚠ ***Констітуентою нуля булевої функції f, яка не є константою 1, це набір α, для якого $f(\alpha) = 0$***

Зрозуміло, що для булевої функції, яка є константою одиниці не може існувати набору, на якому вона приймає значення нуль.

Приклад 4.5 Булева функція задана наступною таблицею істинності

Таблиця 4.6 Таблиця інстинності для булевої функції трьох змінних

x_1	x_2	x_3	$\mathrm{f}(x_1, x_2, x_3)$
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

У цьому випадку констітутентами нуля вважаються елементи наступної множини $\mathrm{C}_0 = \{(0;0;0),(0;1;1);(1;0;0);(1;0;1)\}$ *. Відповідно, інші набори будуть вважатися констітуентами одиниці.*

Введемо далі у розгляд важливе для побудови ДНФ та КНФ поняття імпліканти.

⚠ ***Імплікантою булевої функції $f(x)$ називається булева функція $g(x)$ для якої справедливе, що для будь-яких значень змінних з того, що $g(x) = 1$ слідує, що $f(x) = 1$***

Поняття імпліканти означає, що функція $g(x)$ приймає значення, щонайменше, на тих самих наборах, що й функція $f(x)$. Зазначимо, що при завданні побудови імплікант функція $f(x)$ частіше за все задається у таблічній формі, а функція g(x) – аналітично.

Приклад 4.6 Розглянемо побудову імліканти для булевої функції, яка задана наступною таблицею

Таблиця 4.7 Таблиця значень булевої функції

x_1	x_2	x_3	R
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	0

Побудуємо відповідну функцію-імліканту для цієї функції. Для цього будемо використовувати безпосередньо визначення імліканти та стандартний базис. З визначення імпліканти відомо, що ця функція має приймати значення «1»

на тих самих наборах, що й функція задана таблично. На інших наборах вона повинна мати значення «0» Для представлення кожної констітуенти одиниці будемо використовувати функцію кон'юнкції, оскільки тільки у випадку всіх значень «1» вона має також значення «1». Така взаємооднозначна відповідність дозволяє побудувати для кожного елементу множини констітуент одиниці функції (див. табл 4.7) вираз, який буде мати значення одиниці лише тоді, коли значення змінних буде відповідати елементу множини констітуент одиниці для функції, заданої таблицею 4.7. Вирази, що відповідають констітуентам одиниці, наведено у таблиці 4.8

Таблиця 4.8 Елементарні кон'юнкції для конституент одиниці

x_1	x_2	x_3	Елементарна кон'юнкція
0	0	0	$\bar{x}_1\bar{x}_2\bar{x}_3$
0	0	1	$\bar{x}_1\bar{x}_2x_3$
0	1	1	$\overline{x_1}x_2x_3$
1	0	0	$x_1\bar{x}_2\bar{x}_3$
1	0	1	$x_1\bar{x}_2x_3$

Неважко зрозуміти, що значення «1» ці кон'юнкції приймають тільки на наборах зі значеннями булевих змінних, що відповідають кожному з рядків цієї таблиці

Далі, використовуючи визначення диз'юнкції, а також визначення імліканти, досить легко побудувати функцію імпліканту $g(x)$ для поданої табличної функції: $g(x) = \bar{x}_1\bar{x}_2\bar{x}_3 + \bar{x}_1\bar{x}_2x_3 + \overline{x_1}x_2x_3 + x_1\bar{x}_2\bar{x}_3 + x_1\bar{x}_2x_3$. Функція $g(x)$ буде приймати значення «1» виключно на тих двікових наборах, на яких функція

$f(x)$, що задана таблично приймає значення «1». Це випливає з визначення та пріорітетності операцій у стандартному логічному базисі

Подібним чином можуть бути побудовані імліканти із використанням нульових наборів. Тоді у таблиці 4.8 будуть присутні елементарні диз'юнкції, які у свою чергу будуть поєднані між собою операціями кон'юнкції.

Зверніть увагу на особливості цієї імпліканти. По-перше, побудована нами імпліканта цілком відповідає визначенню ДНФ. Отже, ми можемо говорити про те, що ми побудували ДНФ для функції f(x). Зі ішого боку, можна побачити що цю ДНФ ще можна спростити завдяки тому, що, наприклад, перша та четверта елементарна кон'юнкція мають спільний множник $\bar{x}_2\bar{x}_3$. Отже, за аналогією зі спрощеннями у звичайній алгебрі чисел, існує проблема отримання мінімальної за кількістю членів, форми булевої функції. Для оцінки подальших спрощень, введемо у розгляд поняття довжини для ДНФ та КНФ.

⚠ ***Довжина ДНФ – це кількість елементарних кон'юнкцій, з яких складається ця ДНФ***

⚠ ***Довжина КНФ – це кількість елементарних диз'юнкцій, з яких складається ця КНФ***

Та ДНФ, яка серед усіх інших ДНФ має найменшу довжину є найкоротшою ДНФ. Поняття ж мінімальної ДНФ визначається виходячи з кількості змінних, які використовуються у запису цієї ДНФ.

⚠ ***ДНФ (КНФ) є мінімальною, якщо вона має найменшу кількість літералів серед усіх ДНФ (КНФ), які їй еквівалентні***

Проілюструємо основні відмінності понять мінімальна ДНФ або КНФ та найкоротша ДНФ або КНФ на основі ДНФ, що побудована у Приклад 4.6. Цей вираз має вигляд $\bar{x}_1\bar{x}_2\bar{x}_3 + \bar{x}_1\bar{x}_2x_3 + \overline{x_1}x_2x_3 + x_1\bar{x}_2\bar{x}_3 + x_1\bar{x}_2x_3$. Цей вираз являє собою ДНФ. За визначенням, довжина цієї ДНФ має значення 5. У той же час вона не є мінімальною, оскільки є можливість спрощення виразу за допомогою тотожності склеювання. Зокрема, за цією тотожністю можна спростити першу та другу елементарну диз'юнкцію.

Таким чином, актуальним є завдання мінімізації представлення булевих функцій. Для виконання мінімізації булевих функцій використовуються методи карт Карно, метод Квайна-МакКласки та аналітичні методи мінімізації булевих функцій.

4.5 Методи мінімізації булевих функцій

Мінімізація методом карти Карно. Карта Карно є прямокутною таблицею, де сусідні клітини мають властивість склейки. Значення у карті Карно заповнюються відповідно до таблиці істинності булевої функції. Для булевої функції 2-х змінних карта Карно має 4 клітини, для булевої функції 3х змінних – 8 клітин, 4-х – 16 клітин. Взагалі, для булевої функції, яка складається з **n** змінних карта Карно містить $\mathbf{2^n}$ клітин. Приклад карти Карно наведений на рис. 4.2

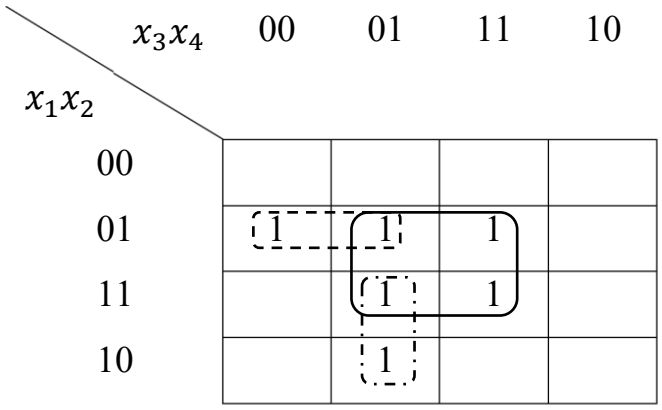


Рисунок 4.2 - Приклад карти Карно

На рис. 4.2 наведено приклад мінімізації методом карт Карно булевої функції яка має шість констітуент одиниці. При виконанні склеювання на карті Карно дозволяється використовувати 1,2,4,8…. сусідніх клітин. Головна вимога – кількість сусідніх клітин має бути цілим ступенем числа 2. В цьому випадку значення констітуенти одиниці у сусідніх клітинах відрізняється на один розряд.

Принцип мінімізації за картами Карно полягає у використанні тотожності склеювання та понятя простої імліканти. Тотожність склеювання виглядає так: $K\bar{x} \vee Kx = K(\bar{x} \vee x)$. На карті Карно це виглядає як склейка двох сусідніх клітин. Для прикладу, який наведено на рисунку 4.2, якщо виконується мінімізація по одинцях, результат буде виглядати так: $x_2x_4 + \overline{x_1}x_2 + \overline{x_3}x_4$.

Мінімізацію можна проводити й іншими способами, об'єднуючи наприклад, замість чотирьох клітин. дві по дві. Для досягнення більш кращих результатів мінімізації можна дотримуватись наступного приципу: кожний наступний прямокутник на карті Карно охоплює якомога більшу кількість одиниць і кожний новий прямокутник містить хоча б одну нову клітину. Нових клітин до прямокутників бажано включати якомога більше.

Дотримання таких принципів дозволяє значною мірою полегшити побудову нормальних форм булевих функцій, значно спростивши кількість кроків для їх мінімізації.

Мінімізація методом Квайна-Маккласки. Другим методом є метод Квайна-Маккласки, який є більш складним у застосуванні. Суть методу полягає у тому, що для деякої булевої функції спочатку відбираються відповідні набори, на яких функція приймає значення одиниці.

Таблиця 4.9 - Приклад булевої функції

x_1	x_2	x_3	x_4	$f(x_1, x_2, x_3, x_4)$
0	0	0	0	1
0	0	0	1	0
0	0	1	0	1
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	0
0	1	1	1	1
1	0	0	0	1
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	1
1	1	0	1	0
1	1	1	0	1
1	1	1	1	1

Для булевої функції, яка наведена на у таблиці 4.9, будемо далі розглядати набори, які відповідають значенню «1» для булевої функції. Ці набори у таблиці позначені сірим кольором. Для наведеної у таблиці функції, це буде множина наступних значень: $\mathrm{C} =$. Надалі розділимо ці набори на класи за кількістю одиниць: $\mathrm{C}_0 = \{0000\}$; $C_1 = \{0010,1000\}$, $C_2 = \{0011,0101,1100\}$, $\mathrm{C}_3 = \{1110,1011,0111\}$, $\mathrm{C}_4 = \{1111\}$, де множина C_n відповідає класу з n одиниць. Далі вони розташовуються у порядку збільшення кількості одиниць у класі, після чого виконується операція склеювання (рис. 4.3).

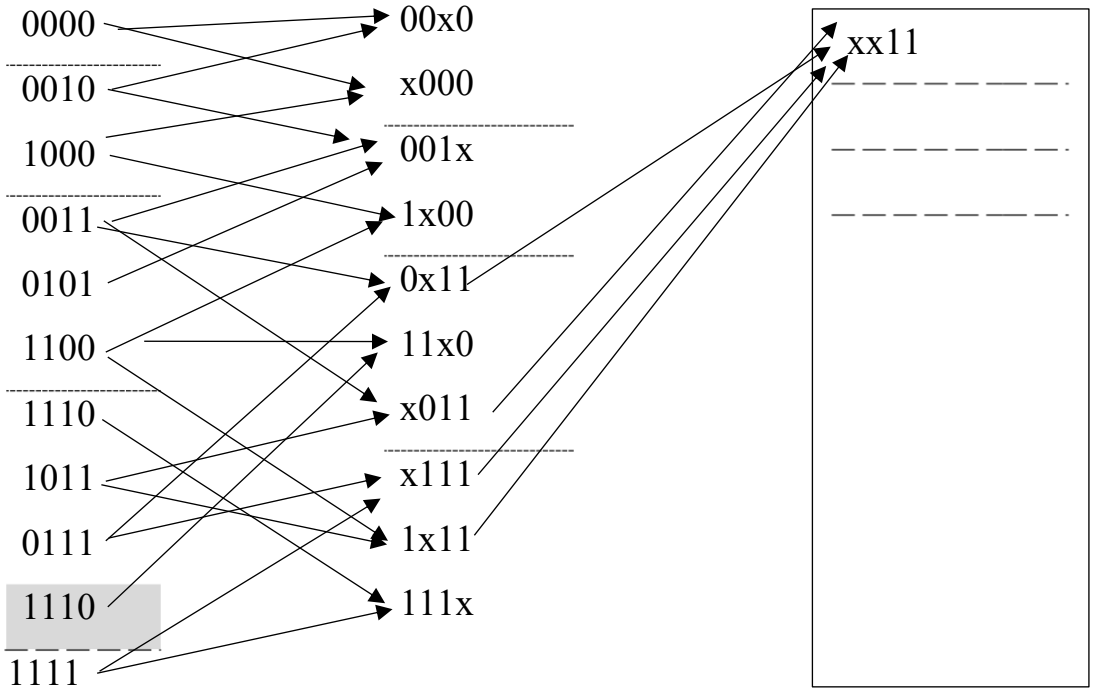


Рисунок 4.3 - Приклад побудови простих імплікант за методом Квайна Маккласкі

Таким чином, на рисунку 4.3 отримано так званий комплекс кубів. Ця назва бере свій початок від поняття, що алгебра булевих функцій є ришіткою. Як

відомо, для кожної ришітки можна побудувати діаграму Хассе, яка для булевої функції зветься булевим кубом.

Далі побудуємо покриття кубами усіх одиничних наборів

	0000	0010	0011	0111	1000	1010	1011	1100	1110	1111
00x0		V				V				
x000	V				V					
001x		V	V							
1x00					V			V		
0x11			V	V						
11x0								V	V	
x011			V				V			
x111				V						V
1x11							V			V
111x									V	V
xx11			V	V			V			V

Як і у методі карт Карно, рядкам, позначеним сірим кольором відповідають елементарні кон'юнкції, а їх комбінація відповідає диз'юнктивній нормальній формі, яка записується наступним чином $\overline{x_1x_2x_4} + \overline{x_2x_3x_4} + \overline{x_1x_2}x_3 + x_1\overline{x_3x_4} + x_1x_2\overline{x_4} + x_1x_2x_3 + x_3x_4$.

Для мінімізації методом Квайна-МакКласкі можна надати наступні рекомендації. При використанні методу до диз'юнктивної чи кон'юнктивної нормальної форми слід додавати спочатку рядки, які здатні покрити найбільшу кількість одиничних наборів. Це еквівалентно виділенню найбільшого прямокутника на карті

Карно. Після цього до ДНФ чи КНФ слід додавати ті рядки, які мають найбільшу кількість непокритих одиничних наборів.

Метод Квайна-Маккласкі, завдяки використанню у ньому двомірної таблиці, може бути легко автоматизований за допомогою програмного забезпечення.

Методи мінімізації булевих функцій у програмній інженерії. Методи мінімізації булевих функцій у програмній інженерії використовуються, насамперед при побудові умов у програмному забезпеченні. Для цього, спочатку будується предикат, що представляє просту умову для деякого об'єкту, а потім використовується комбінація побудованих предикатів у вигляді булевих функцій.

Розглянемо приклад побудови такої булевої функції на прикладі предметної області

Приклад 4.7. Розглянемо предметну область «Вищий навчальний заклад». Об'єкт «студент» характеризується прізвищем (множина F), датою народження (DateBorn) та номером курсу ($Course$) та набором балів з дисципліни. Набор балів з дисципліни характеризується атрибутами: «Назва дисципліни» ($DiscName$) та «Оцінка» ($Mark$). $St = F \times DateBorn \times Course \times \mathcal{P}(DiscName \times Mark)$. *Введемо у розгляд два предиката з наступними областями істинності* $P_1(St) = \{(f, db, c, Z): c = 1\}$ *– студенти першого курсу, та* $P_2: DiscName \times Mark \times St \to 0; 1, P_2 = (Dn, Mark): M >= Mark \ \& (Dn, M) \in \text{пр}_4 St\}$ *та* $P_3: DiscName \times Mark \times St \to 0; 1, P_2 = (Dn, Mark): M <= Mark \ \& (Dn, M) \in \text{пр}_4 St$ *На цій основі сконструюємо більш складний предикат:* $P_1(St)$ & $P_2(\text{КДМ}, 90, St)$*, який буде відбирати студентів першого курсу, які мають оцінку 90 з дисципліни «Комп'ютерна дискретна математика». Узагальнена таблиця істинності для цього предикату виглядає наступним чином.*

Номер набору	P_1	P_2	P_3	$f(P_1, P_2, P_3)$
0	0	0	0	0
1	0	0	1	0
2	0	1	0	0
3	0	1	1	0
4	1	0	0	X
5	1	0	1	0
6	1	1	0	1
7	1	1	1	1

Хрести у таблиці істинності позначають невизначені набори, тобто такі, що не мають місця у реальній предметній області. Такі набори можна використовувати у якості як нульових, так і одиничних наборів на карті Карно. Приклад карти Карно для мінімізації булевої функції з цього прикладу наведено на рисунку

P_3 \ P_1P_2	00	01	11	10
0	0	0	0	0
1	X	0	1	1

Відповідь наведемо у формі КНФ: $P_1 + P_3$

У наведеному прикладі проілюстровано мінімізацію дуже розповсюдженого класу булевих функцій – частково визначеної булевої функції

4.6 Функціональна повнота базису булевих функцій

Постановка завдання. Оскільки будь-яку булеву функцію можна представити за допомогою диз'юнктивної та/або кон'юнктивної нормальної форми, то базис, що складається з диз'юнкції, кон'юнкції та інверсії є функціонально повним базисом булевої функції, тобто за його допомогою можна представити будь-яку булеву функцію.

Звідти постає питання, чи є такий функціонально-повний базис єдиним, або існують інші функціонально повні базиси булевих функцій. Ця проблема була сформульована та вирішена Постом, шляхом запровадження декількох спеціальних класів булевих функцій та формулюванням критерію функціональної повноти на базі цих класів булевих функцій. Розглянемо більш детально ці класи

Клас булевих функцій, що зберігають константу 0 та 1. Ці клас визначається наступним чином

⚠ ***Булева функція $f(x_1, x_2, \dots, x_n)$ зберігає константу 0 (константу 1) якщо вона приймає нульове значення на нульовому (одиничному) наборі змінних***

Нульовим набором функції $f(x_1, x_2, \dots, x_n)$ є набір на якому $x_1 = x_2 = \cdots = x_n = 0$, тобто всі аргументи функції приймають нульове значення. Відповідно, одиничним набором є набір, на якому $x_1 = x_2 = \cdots = x_n = 1$, тобто всі аргументи булевої функції приймають нульове значення.

Приклад 4.8. Функція кон'юнкції є такою, що зберігає константи 0 та 1, оскільки на нульовому наборі вона приймає значення 0, а на одиничному наборі – значеннння 1.

Приклад 4.9. Функція логічної інверсії є такою, що не зберігає константу 0 та 1. Оскільки $f(0) = 1$*, та навпаки* $f(1) = 0$

Приклад 4.10. Функція трьох змінних,що наведена нижче, не зберігає константу «0» та зберігає константу «1»

X1	*X2*	*X3*	*f(x1,x2,x3)*
0	*0*	*0*	*1*
0	*0*	*1*	*1*
0	*1*	*0*	*0*
0	*1*	*1*	*1*
1	*0*	*0*	*0*
1	*0*	*1*	*1*
1	*1*	*0*	*0*
1	*1*	*1*	*1*

Множину всіх функцій, що зберігають константу «0» позначають T_0. Множину всіх функцій, що зберігають константу «1» позначають T_1.

Подвійність булевих функцій. Поняття подвійності булевої функції наведено у наступному визначенні

⚠ ***Булеву функцію g називають подвійною до булевої функції f, якщо для будь-якого набору*** $\tilde{\alpha} \in \{0;1\}^n$ ***(n>0) справедлива рівність*** $g(\tilde{\alpha}) = \bar{f}(\bar{\tilde{\alpha}})$

Також вважається, що константа «0» є подвійною до константи «1» та навпаки. Тобто подвійна функція приймає інверсні значення на інверсних наборах змінних. Проілюструємо це на прикладах

Приклад 4.11.Стрілка Пірса є подвійною до функції штрих Шеффера. Це випливає з наступних перетворень: $x \downarrow y = \overline{x \vee y} = \overline{\overline{\bar{x} \wedge \bar{y}}} = \overline{\bar{x}|\bar{y}}$

Окремо відзнаючають клас самоподвійних функцій. Це функції, які визначаються наступним чином:

⚠ ***Самоподвійна функція – це функція, що подвійна до самої себе, тобто для якої виконується рівність*** $\boldsymbol{f(\tilde{\alpha}) = \bar{f}(\bar{\tilde{\alpha}})}$

Іншою мовою, булева функція самоподвійна тоді і лише тоді, коли вона на взаємо протилежних наборах приймає взаємо протилежні значення.

Приклад 4.12 Мажоритарна функція, яка задається наступною таблицею істинності

x1	*x2*	*x3*	*f(x1,x2,x3)*
0	*0*	*0*	*0*
0	*0*	*1*	*0*
0	*1*	*0*	*0*
0	*1*	*1*	*1*
1	*0*	*0*	*0*

1	*0*	*1*	*1*
1	*1*	*0*	*1*
1	*1*	*1*	*1*

є самоподвійною, оскільки вона приймає на взамопротилежних наборах аргументів взаємо протилежні значення.

Монотонні булеві функції. Визначення монотонної булевої функції формулюється наступним чином

⚠ ***Функцію називають монотонною, якщо для будь-яких наборів*** $\boldsymbol{\alpha}, \boldsymbol{\beta} \in \{\mathbf{0}; \mathbf{1}\}^{\boldsymbol{n}}$, $\boldsymbol{n} \geq \mathbf{0}$ ***виконується твердження: з того, що*** $\boldsymbol{\alpha} \geq \boldsymbol{\beta}$ ***слідує*** $\boldsymbol{f}(\boldsymbol{\alpha}) \geq \boldsymbol{f}(\boldsymbol{\beta})$

При використанні визначення монотонної функції порівняння двійкових векторів здійснюється справа наліво і так само, як виконується порівняння будь-яких інших векторів з множини декартова добутку

Приклад 4.13 Функція кон'юнкції є монотонною. Це можна встановити, аналізуючи її таблицю істинності (див табл. 4.4 функція f_1*) на приклад виконання визначення монотонної функції*

Приклад 4.14. Функція еквівалентності не є монотонною (див табл 4.4 функція f_9*). На наборі (0;0) вона приймає значення 1, а на наборі (0;1) значення 0, що суперечить твердженню монотонності:* $(0;1) \geq (0;0)$*, але* $f_9(0;1) \leq f_9(0;0)$.

Алгебра Жегалкіна. Лінійні функції. Алгебра Жегалкіна складається з операції кон'юнкції, додавання за модулем 2 та логічної константи «1». Через неї

досить легко виразити основні функції стандартного булевого базису. Кон'юнкція вже входить до цього базису, інверсія представляється твердженням $\bar{x} = 1 \oplus x$; $x \vee y = x \oplus y \oplus xy$. Таким чином, алгебру Жегалкіна можна вважати функціонально повним базисом булевих функцій, тобто системою, за допомогою якої можна побудувати представлення будь-якої булевої функції

У рамках алгебри Жегалкіна цю побудову можна здійснити за допомогою методу невизначених коефіцієнтів для полінома Жегалкіна. Поліном Жегалкіна для функції двох змінних має вигляд

$$f(x_1; x_2) = a_{12}x_1x_2 \oplus a_1x_1 \oplus a_2x_2 \oplus a_0$$

Відповідно для трьох змінних поліном має вигляд

$$f(x_1; x_2; x_3) = a_{123}x_1x_2x_3 \oplus a_{12}x_1x_2 \oplus a_{23}x_2x_3 \oplus a_{13}x_1x_3 \oplus a_1x_1 \oplus a_2x_2 \oplus a_3x_3 \oplus a_0$$

За аналогією будуються поліноми Жегалкіна для функцій, які мають кількість зміннних більше ніж три.

Розглянемо побудову полінома Жегалкіна методом невизначених коефіцієнтів на прикладі функції, що задана таблично (табл. 4.10)

Таблиця 4.10 - Приклад булевої функції двох змінних

x1	x2	f(x1,x2)
0	0	1
0	1	0
1	0	1
1	1	1

Беремо нульовий набір, та підставляємо його до поліному двох змінних. Оскільки аргументи в цьому випадку мають нульове значення, то всі члени полінома, що залежать від аргументів також отримають нульове значення, виходячи із властивостей кон'юнкції. Загалом отримаємо наступний вираз

$$1 = 0 \oplus 0 \oplus 0 \oplus a_0$$

З цього виразу видно, що для того, щоб рівність була вірною необхідно, щоб коефіцієнт a_0 мав значення 1. Це значення надалі будемо використовувати у якості значення цього коефіцієнту.

На наборі (0;1) функція приймає значення 0. Беручи до уваги властивості функції кон'юнкції та значення коефіцієнту $a_0 = 1$ отримаємо наступний поліном

$$0 = 0 \oplus 0 \oplus a_2 \oplus 1$$

Звідки отримаємо, що коефіцієнт a_2 повинен мати значення 1. Тоді наш поліном буде мати наступний вигляд

$$a_{12}x_1x_2 \oplus a_1x_1 \oplus x_2 \oplus 1$$

Надалі підставляємо до отриманого полінома набір (1;0), на якому функція приймає значення «1». Отримаємо наступний вид поліному

$$1 = 0 \oplus a_1 \oplus 0 \oplus 1$$

З цього випливає, що значення коефіцієнту a_1 буде дорівнювати 0. Отже наш полином приймає наступний вид

$$a_{12}x_1x_2 \oplus x_2 \oplus 1$$

Для отриманння кінцевого виду полінома підставляємо до його поточного вигляду набір (1;1) на якому він приймає значення «1». Отримаємо наступний вираз:

$$1 = a_{12} \oplus 1 \oplus 1$$

З цього випливає, що значення коефіцієнту $a_{12} = 1$. Таким чином, представлення цієї булевої функції у вигляді поліному Жегалкіна має вид:

$$f(x_1; x_2) = x_1x_2 \oplus x_2 \oplus 1$$

Особливим класом поліномів Жегалкіна є клас лінійних поліномів, які визначаються за наступною формулою

$$f(x_1, \ldots, x_n) = \sum_{i=1}^{n} (mod\ 2) a_i x_i \oplus a_0$$

Відповідно, визначимо клас булевих функцій – лінійні функції

⚠ ***Лінійна булева функція – це функція, яку можливо представити у вигляді лінійного полінома Жегалкіна***

Крітерій Поста функціональної повноти булевої функції. Розглянуті нами класи булевих функцій, а саме: функції, що зберігають константу «0» (позначається T_0), клас функцій, який зберігає константу «1» (позначається T_1), клас самоподвійних функцій (позначається S), клас монотонних функцій (позначається M), клас лінійних функцій (позначається L), є класами Поста. Використовуючи ці класи Е. Пост сформулював крітерій функціональної повноти базиса булевих функцій, який зараз формулюється наступним чином

⚠ ***Базис (множина) булевих функцій є функціонально повним, якщо він не міститься цілком у жодному з класів Поста***

Іншими словами, базис є функціонально повним, якщо він містить хоча б одну функцію, яка не зберігає константу «0», хоча б одну функцію, яка не зберігає константу «1», хоча б одно не самоподвійну функцію, хоча б одну не монотонну функцію, хоча б одну не лінійну функцію. За допомогою функцій з функціонально повного базису можливо представити будь-який логічний вираз.

Слід зазаначити, що при перевірці функціональної повноти базису булевих функцій варто притримуватись послідовності перевірки «від простого до складного»: спочатку перевірити, чи містить система функції, що не зберігають

константи «0» або «1», потім чи не містить вона хоча б однієї не самоподвійної чи не монотонної функції, та, на останок, чи не містить вона хоча б однієї не лінійної функції

Приклад 4.15 Функція стрілка Пірса (функція f_8 у табл. 4.4) («заперечення диз'юнкції», функція «або-ні») є функціонально повною, оскільки: 1) не зберігає константу «0» ($f_8(0;0)=1$), 2) не зберігає константу «1» $f_8(1;1)=0$; 3) є не монотонною ($(0;0)\leq(0;1)$ але $f_8(0;0)\geq f_8(0;1)$) 4) є несамоподвійною, оскільки $f_8(1;0)=f_8(0;1)=0$.

Для визначення лінійності цієї функції розглянемо побудову для неї поліному Жегалкіна. $f_8(0;0)=1$, отже за методом невизначених коефіцієнтів для поліному Жегалкіна отримаємо $1=0\oplus0\oplus0\oplus a_0$, звідки $a_0=1$. $f_8(0;1)=0$. Поліном Жегалкіна отримає вигляд: $0=0\oplus0\oplus a_2\oplus1$, звідки $a_2=1$. $f_8(1;0)=0$. Відповідно поліном Жегалкіна має вигляд $0=0\oplus a_1\oplus0\oplus1$ звідки $a_1=1$. $f_8(1;1)=0$, звідки поліном Жегалкіна має вигляд $0=a_{12}\oplus1\oplus1\oplus1$, звідки $a_{12}=1$. Таким чином, поліном Жегалкіна для цієї функції має вигляд $f_8(x_1;x_2)=x_1x_2\oplus x_1\oplus x_2\oplus1$, що свідчить про нелінійність цієї функції

Зважаючи на функціональну повноту системи з однієї функції стрілка Пірса, покажемо, як функції стандартного базису можуть бути утворені з цієї. $f_8(x,x)=\bar{x}$, $f_8(f_8(x_1,x_2),f_8(x_1,x_2))=x_1\vee x_2;$, $f_8(f_8(x_1,x_1),f_8(x_2,x_2))=x_1\wedge x_2$.

Звернемо увагу, що в усіх випадках у записах виразів присутні виключно функції системи та змінні. Замість змінних можуть бути використані знов функції з цієї системи. В цьому випадку кажуть про функцію суперпозиції змінних.

4.7 Завдання до розділу 4

Завдання 4.1 Мінімізувати булеву функцію, яка задана наступною таблицею істинності

X1	*X2*	*X3*	*F(x1,x2,x3)*
0	*0*	*0*	*1*
0	*0*	*1*	*1*
0	*1*	*0*	*0*
0	*1*	*1*	*0*
1	*0*	*0*	*1*
1	*0*	*1*	*1*
1	*1*	*0*	*0*
1	*1*	*1*	*0*

Завдання 4.2 Мінімізувати булеву функцію, яка задана наступною таблицею істинності

X1	*X2*	*X3*	*F(x1,x2,x3)*	*X1*	*X2*	*X3*	*F(x1,x2,x3)*
0	*0*	*0*	*0*	*1*	*0*	*0*	*1*
0	*0*	*1*	*1*	*1*	*0*	*1*	*1*
0	*1*	*0*	*0*	*1*	*1*	*0*	*0*
0	*1*	*1*	*0*	*1*	*1*	*1*	*0*

Завдання 4.3. Мінімізувати булеву функцію, яка задана наступною таблицею істинності

X1	*X2*	*X3*	*X4*	*F(X1,X2,X3,X4)*
0	*0*	*0*	*0*	*0*

0	*0*	*0*	*1*	*1*
0	*0*	*1*	*0*	*1*
0	*0*	*1*	*1*	*0*
0	*1*	*0*	*0*	*0*
0	*1*	*0*	*1*	*0*
0	*1*	*1*	*0*	*1*
0	*1*	*1*	*1*	*1*
1	*0*	*0*	*0*	*1*
1	*0*	*0*	*1*	*1*
1	*0*	*1*	*0*	*1*
1	*0*	*1*	*1*	*1*
1	*1*	*0*	*0*	*0*
1	*1*	*0*	*1*	*0*
1	*1*	*1*	*0*	*0*
1	*1*	*1*	*1*	*0*

Завдання 4.4 Визначити функціональну повноту системи функцій, яка складається з імплікації та еквівалентності

Завдання 4.5 Визначити функціональну повноту системи функцій, що задана наступними таблицями істинності

x1	x2	x3	f(x1, x2, x3)
0	0	0	1
0	0	1	1
0	1	0	0

0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

x1	x2	x3	f(x1, x2, x3)
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

Завдання 4.6. Мінімізувати булеву функцію, яка приймає значення «1» на наступних наборах аргументів: 0000; 0001; 0010; 1101; 1010; 1110

Завдання 4.7. Є множина товарів на складі. Визначити предикат, що буде визначати наявність конкретного найменування товару на складі.

Завдання 4.8. Є множина автомобілів таксі. Визначити предикат, який буде визначати максимальний з пробігів для двох автомобілів.

5 ЕЛЕМЕНТИ ТЕОРІЇ ГРАФІВ

5.1 Визначення та основні поняття теорії графів

Визначення поняття графу. Уведемо у розгляд множини V вершин графу та E – множину пар вершин, яка позначатимете ребро графу

⚠ ***Графом G будемо називати пару множин: довільну множину вершин V та множину E впорядкованих чи невпорядкованих пар ребер***

У випадку, коли множина ребер містить впорядковані пари кажуть про орієнтований граф. Коли множина ребер містить невпорядковані пари кажуть про неорієнтований граф. Приклади орієнтованого і неорієнтованого графу наведено на рис.

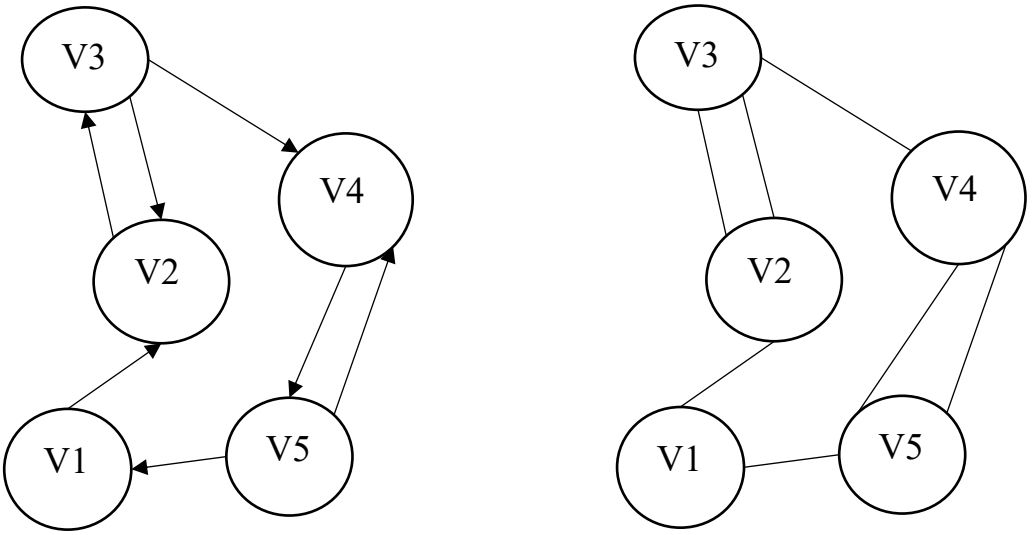


Рисунок 5.1 - Типи графів (а - орієнтований граф, б - неорієнтований граф)

Якщо $E = \emptyset$ кажуть про нуль-граф, якщо ж $E = V^2$ кажуть про повний граф. Повні графи розрізняють як графи з петлями та графи без петель. Графи з петлями містять у якості ребер V^2, а графи без петель – відношення $V^2 \backslash R$, де R – рефлексивне відношення на V^2. Підграфом деякого графу є граф, який містить деяку підмножину вершин графу та деяку підмножину інцедентиних їм ребер. Приклад підграфів для графів (див. рис. 5.1) наведено на рис. 5.2

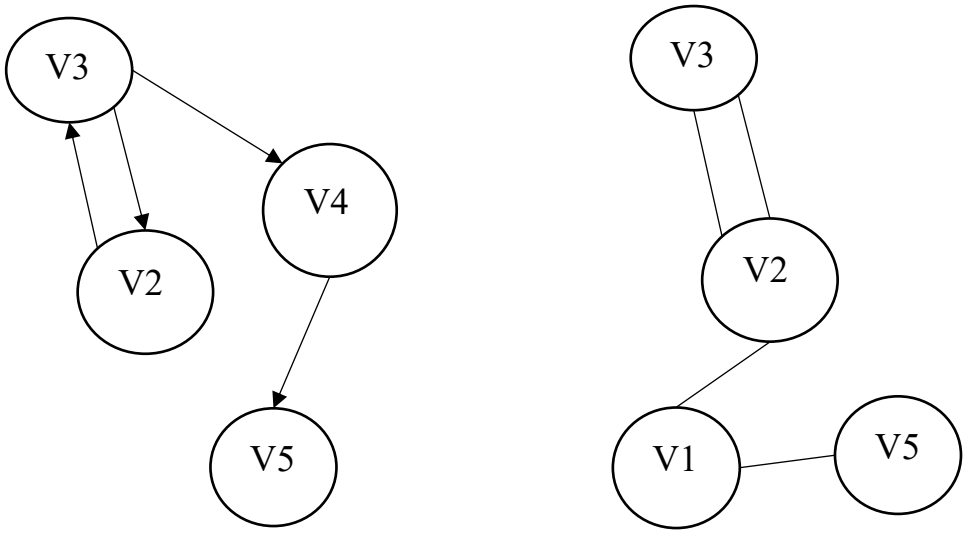


Рисунок 5.2 - Приклад підграфів

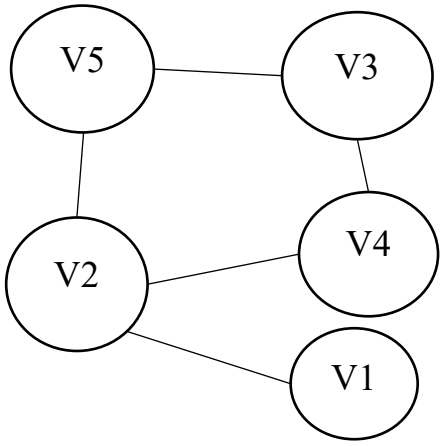
На рис 5.2а наведено приклад підграфу для графу на рис 5.1а, а на рис 5.2б наведено підграф для рис. 5.1б

Мультиграфом G є граф, у якому існують декілька різних типів ребер. Зазвичай, ці ребра промарковані або різними кольорами, або різними символами. У випадку, коли дві вершини v_1, v_2утворюють ребро $e = (v_1, v_2)$, говорять про інцедентність ребра e вершинам v_1 та v_2. Відповідно вершини v_1 та v_2 є суміжними

вершинами. Ступенем вершини графу є число ребер, інцедентних цій вершині. Ступінь вершини v позначається d(v) чи deg(v). Якщо $d(v) = 0$ вершина є ізольованою. Якщо $d(v) = 1$ вершина є листом графу або висячою вершиною. Ребро інцедентне такій вершині є висячим ребром.

Для орієнтованих графів додатково існує поняття напівступінь виходу та напівступінь входу. Напівступінь виходу вершини v – це кількість ребер, що виходить з цієї вершини. Напівступінь входу вершини v – це кількість ребер, що входять до вершини v. Напівступінь виходу позначається $d^-(v)$, а напівступінь входу $d^+(v)$. Вочевидь, для орієнтованого графу $d(v) = d^+(v) + d^-(v)$.

Поняття інцедентності та суміжності слугують основним поняттями при матричному способі завдання графів Якщо кожному стовпцю та рядку матриці зіставлена вершина графу – маємо матрицю суміжності графу. Приклад матриці суміжності наведено на рис. 5.3



а)

	V1	V2	V3	V4	V5
V1		1			
V2	1			1	1
V3				1	1
V4		1	1		
V5		1	1		

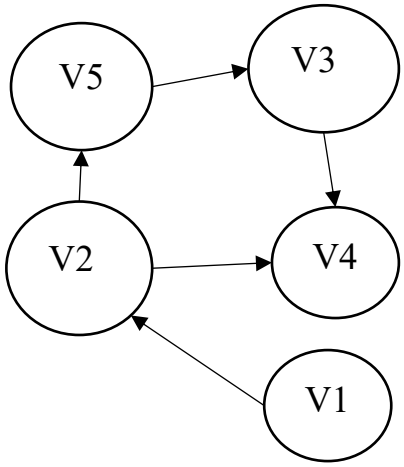
б)

Рисунок 5.3 - Матриця суміжності для неорієнтованого графу

Матриця суміжності для неорієнтованого графу, що зображено на рис. 5.3а, наведено на рис. 5.3б. Зазначимо, що матриця суміжності для неорієнтованого

графу є симетричною. Елементами матриці суміжності є числа, які позначають кількість ребер. У випадку простого графу це 1.

Приклад матриці суміжності для орієнтованого графу наведено на рис. 5.4.



а)

	V1	V2	V3	V4	V5
V1		1			
V2				1	1
V3				1	
V4					
V5			1		

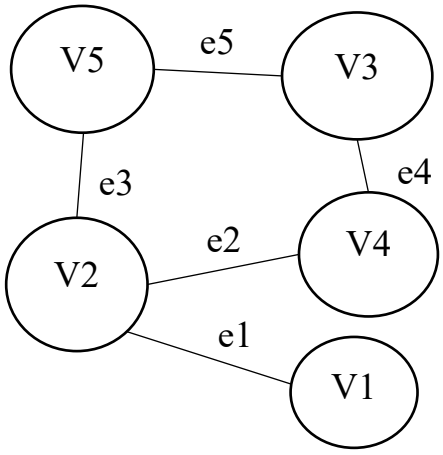
б)

Рисунок 5.4 - Матрриця суміжності для орієнтованого графу

Матриця суміжності для орієнтованого графу на рис. 5.4а наведено на рис. 5.4б. На відміну від матриці суміжності неорієнтованого графу, ця матриця не є симметричною. Це пов’язане із тим, що в орієнтованому графі суміжність вершин визначається із урахуванням напряму ребер, які інцедентні вершинам.

Іншою матрицею, яка використовується для завдання графів є матриця інцедентності. Рядками цієї матриці слугують вершини, а стовпцями – ребра графу. Одиниця на перетині строки та стовпця ставиться у тому випадку, коли ребро інцедентно відповідній вершині

Приклад матриці інцедентності для неорієнтованого графу наведено на рис. 5.5.



а)

	e1	e2	e3	e4	e5
V1	1				
V2	1	1	1		
V3				1	1
V4		1		1	
V5			1		1

б)

Рисунок 5.5 - Приклад матриці інцедентності для графу

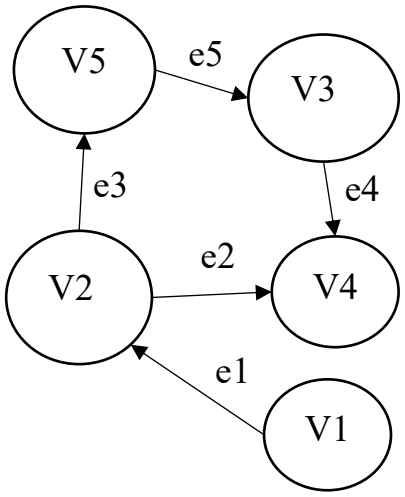
На рис. 5.5.б наведена матриця інцедентності для графу, який зображено на рис. 5.5а. Слід зазначити, що у кожному стовпці матриці інцедентності можуть знаходитись не більше двох значень, що відповідають вершинам, яким інцедентно ребро. Також для неорієнтованого графу (яким є граф на рис. 5.5а) усі значення є додатніми числами, на відміну від орієнтованого графу, де кінцевим вершинам присвоюється значення «-1», як показано на рис. 5.6.

З використанням матриць інцедентності можливе обчислення таких значень, як ступінь вершини та напівступінь виходу та входу. У випадку неорієнтованого графу ступінь будь-якої вершини є арифметичною сумою одиниць, що стоять у рядку матриці, якому відповідає вершина, для якої обчислюється ступінь. Формально це можна визначити наступним виразом

$$d(v_i) = \sum_{j=1}^{n} e_{ij} \qquad (5.1)$$

де e_{ij} – елемент, що стоїть на перетині i-ого рядка та j-ого стовпця, n – кількість стовпців матриці

Приклад 5.1. Для графу, який наведено на рис. 5.5 $d(v_1) = 1$, $d(v_2) = 3$, $d(v_3) = d(v_4) = d(v_5) = 2$.



а)

	e1	e2	e3	e4	e5
V1	-1				
V2	1	-1	-1		
V3				-1	1
V4		1		1	
V5			1		-1

б)

Рисунок 5.6 - Матриця інцедентності для орієнтованого графу

У випадку орієнтованого графу, матриця інцедентності може бути використана для підрахунку не тільки ступеню вершини, а й напівступенів входу та виходу. Для підрахунку цих показників необхідно врахувати, що вихід ребра до вершини позначається від'ємним значенням у матриці, а вхід – додатнім. Отже, сума за модулем всіх від'ємних значень дасть напівступінь виходу, а сума за модулем всіх додатніх – напівступінь входу

Приклад 5.2. У графі на рис. 5.6а напівступінь входу вершини V2 $d^+(v_2) = 1$ *(одна додатна одиниця у відповідному рядку матриці інцедентності (рис.*

5.6б)), а напівступінь виходу тієї ж вершини $d^-(v_2) = 2$ (дві від'ємних одиниці у відповідному рядку матриці інцедентності (див. рис. 5.6б))

Уведемо у розгляд поняття порядку графу та розміру графу

⚠ ***Порядок графу – це кількість його вершин***

⚠ ***Розмір графу – це кількість його ребер***

Операції над графами. Нехай задано два графи $\boldsymbol{G_1(V_1, E_1)}$ та $\boldsymbol{G_2(V_2, E_2)}$. Розглянемо можливі операції над цими графами.

⚠ ***Об'єднанням графів G_1, G_2 $G = G_1 \cup G_2$ є граф $G(V_1 \cup V_2, E_1 \cup E_2)$, тобто граф, що містить об'єднання вершин графів та ребер графів.***

Приклад об'єднання графів наведено на рис 5.7

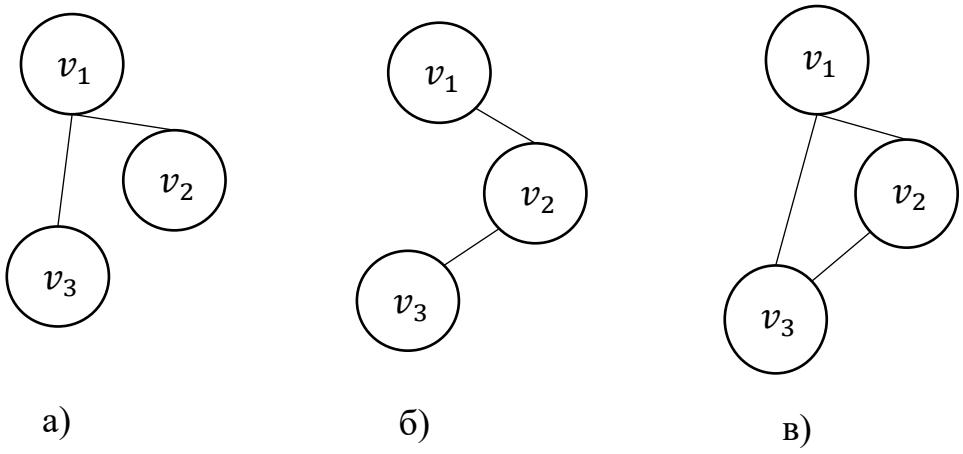


Рисунок 5.7 - Об'єднання графів

На рис. 5.7 наведено приклад виконання операції об’єднанння графів. Граф G, зображений на рис. 5.7в є об’єднанням графів G_1 та G_2, що зображені відповідно на рис 5.7а (граф G_1) та рис 5.7б (граф G_2)

⚠ ***Перетином графів G_1, G_2 $G = G_1 \cap G_2$ є граф $G(V_1 \cap V_2, E_1 \cap E_2)$, тобто перетин графів утворюють перетини їх вершин та ребер.***

Приклад виконання операції перетину графів наведено на рис. 5.8

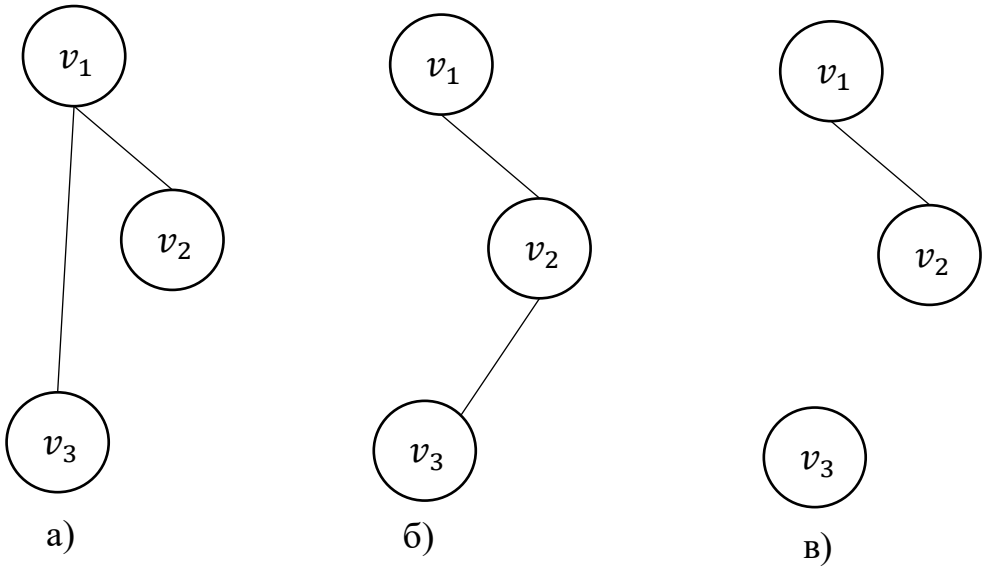


Рисунок 5.8 - Перетин графів

На рис. 5.8в зображено перетин графів, що зображені на рис.5.8а та рис.5.8б

⚠ ***Доповненням графу $G(V, E)$ є граф $G'(V, \overline{E})$. Граф, що є доповненням містить ті самі вершини, що й вихідний граф, але містить ребра, що не входять до вихідного графу.***

Приклад доповнення графу наведно на рис. 5.9

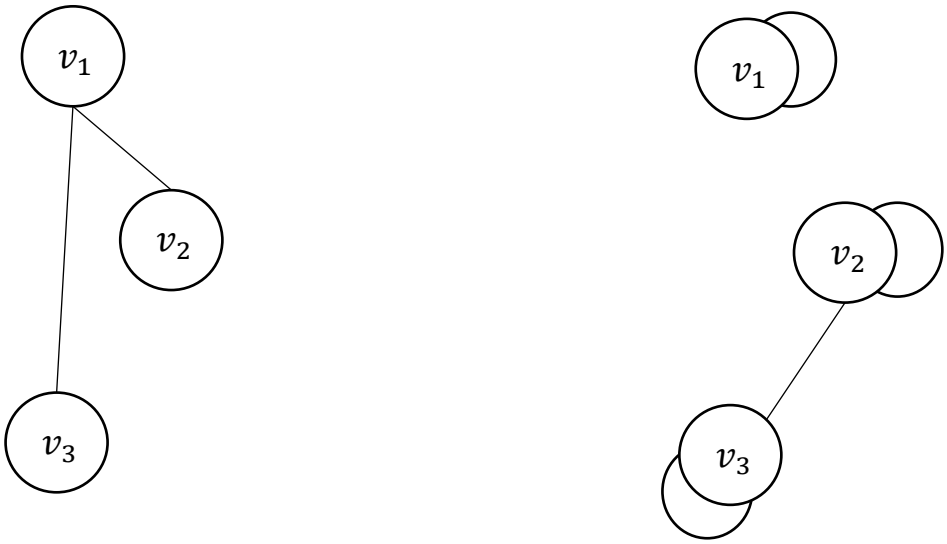


Рисунок 5.9 - Доповнення графу

Граф, що наведено на рис 5.9б є доповненням графу, що наведено на рис 5.9а.

5.2 Поняття шляху у графі. Дерева.

Поняття шляху у графі. Введемо у розгляд поняття шляху у деякому графі

⚠ ***Шляхом у графі G є послідовність вершин, у якому кожна наступна вершина з'єднана із наступним ребром.***

Якщо у шляху кожне ребро зустрічається не більше, ніж один раз, він називається ланцюгом. Приклади ланцюга неведено на рис. 5.10 (ланцюг позначено пунктиром).

Поняття шляху тісно пов’язане з орієнтованістю та неорієнтованістю у графі. У випадку неорієнтованого графу, шлях існує у випадку, коли є ребро між вершинами. В іншому випадку шлях існує лише, якщо у відповідному напрямку існує направлене ребро (рис. 5.10).

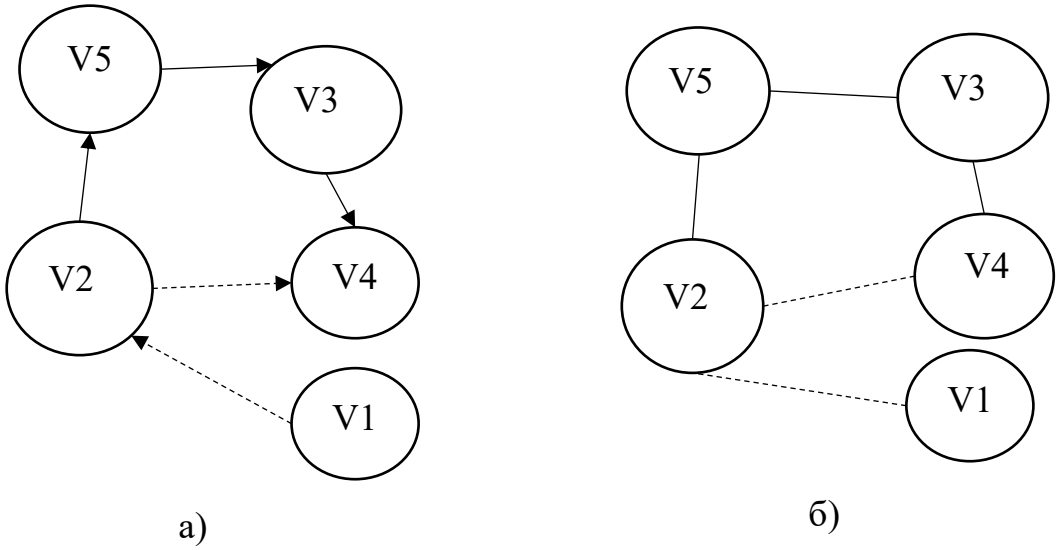


Рисунок 5.10 - Приклади шляхів у графі (Позначено пунктиром)

На рис 5.10 а пунктиром позначено наступний шлях $\{(v_1; v_2); (v_2, v_4)\}$. Звернемо увагу, що зворотнього шляху не існує, оскільки немає направлених у зворотній бік ребер. На рис 5.10б пунктиром показано шляхи $\{(v_1; v_2); (v_2, v_4)\}$ та $\{(v_4; v_2); (v_2, v_1)\}$. Довжиною шляху є кількість ребер, що задає цей шлях. У випадку, наведеному на рис. 5.10а довжина шляху дорівнює 2.

Окремо розглядаються поняття циклічного шляху

⚠ ***Циклічним шляхом у орієнтованому графі є такий шлях, у якому $v_0 = v_k$, де v_0 – початкова вершина циклічного шляху, v_k – кінцева вершина циклічного шляху.***

Приклад циклічного шляху наведено на рис. 5.11

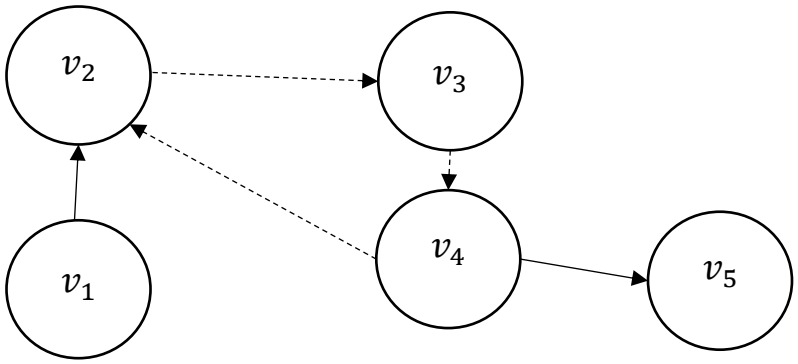


Рисунок 5.11 - Циклічний шлях у графі (позначено пунктиром)

На рис. 5.11 циклічним шляхом є шлях $\{(v_2, v_3); (v_3, v_4); (v_4; v_2)\}$Якщо у циклічному шляху ребро зустрічається не більше, ніж один раз, він називається циклом. Так, циклічний шлях на рис. 5.11 є циклом. Графи без циклів називають ациклічними. Множина вершин, між якими існує шлях, називається компонентою зв'язності графу. На рис. 5.12 зображено граф, у якому існують три компоненти зв'язності: $\{v_1, v_2, v_3\}$, $\{v_4\}$, $\{v_5, v_6, v_7\}$. Граф, який містить лише одну компоненту зв'язності називається зв'язним графом.

Для орієнтованого графу існує поняття компоненти сильної зв'язності. Орієнтований граф називається сильно зв'язаним, якщо будь-які дві вершини s та t сильно зв'язані, тобто, якщо існує орієнтований шлях з s у t та зворотньо із t у s. Компонента сильної зв'язності орієнтованого графу – це найбільший сильнозв'язаний підграф цього орієнтованого графу.

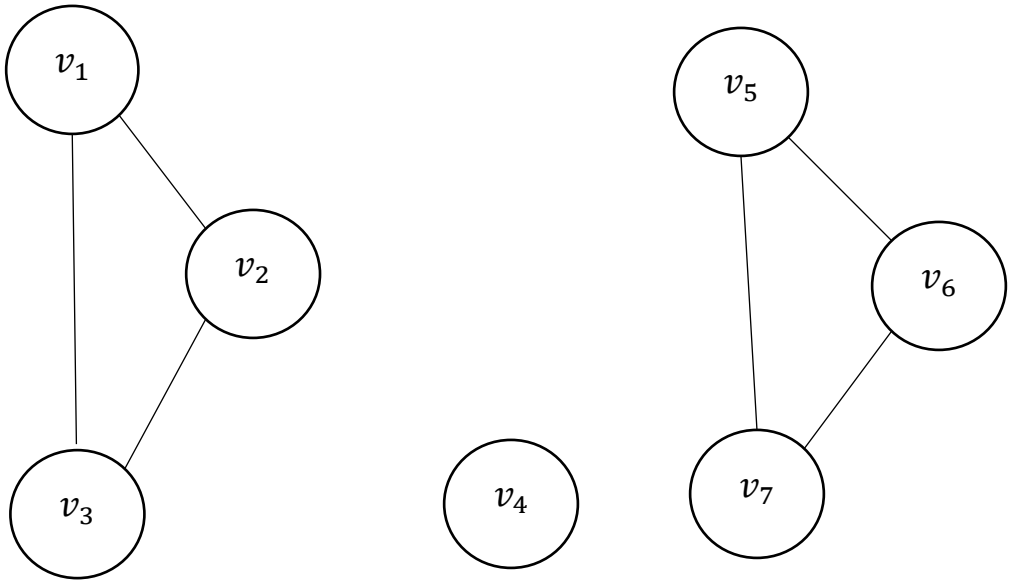
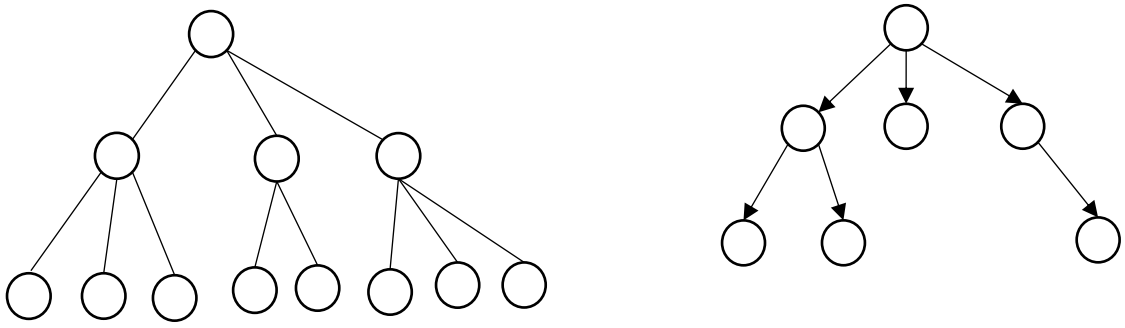


Рисунок 5.12 - Граф із трьома компонентами зв'язності

Дерева. Дерево – це зв’язний ациклічний граф. Ациклічність позначає відсутність циклів Дерево може бути орієнтованим та не орієнтованим. Приклад орієнтоваого та неорієнтованого дерева наведено на рис. 5.13



а) б)

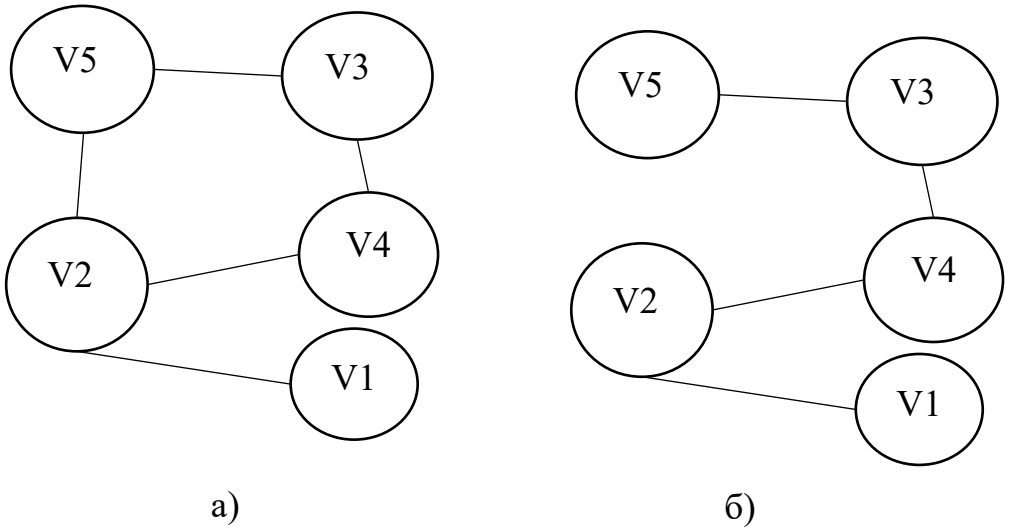
Рисунок 5.13 - Приклади дерев

На рис. 5.6 а зображене неорієнтоване дерево, на рис. 5.6 б орієнтоване дерево. У направленому (орієнтованому) дереві лише одна вершина має нульовий ступінь заходу, всі інші мають ступінь заходу 1. Вершина із нульовим ступенем

заходу зветься коренем дерева, а вершини із нульовим ступенем виходу зветься листям дерева або кінцевими вершинами)

Для неорієнтованих дерев існує декілька важливих властивостей. У неорієнтованому дереві існує щонайменше дві кінцеві вершини. Між будь-якими двома вершинами дерева існує тільки один шлях. Дерево з n вершинами має n-1 ребро.

Зрозуміло, що у неорієнтованому дереві кожний шлях простий. У будь-якому зв'язному графі G знайдеться підграф, що є деревом. Якщо таке дерево включає в себе усі вершини графу G таке дерево зветься остовним деревом. Для побудови остовного дерева у графі з n вершинами необхідно використати $n-1$ ребро. Приклад остовного дерева для графу G наведено на рис. 5.14



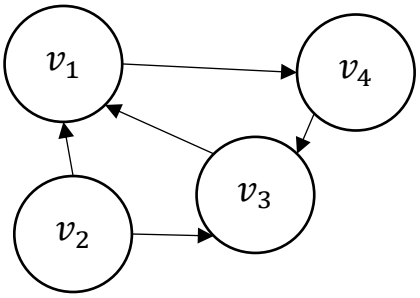
а) б)

Рисунок 5.14 - Приклад побудови остовного дерева графу

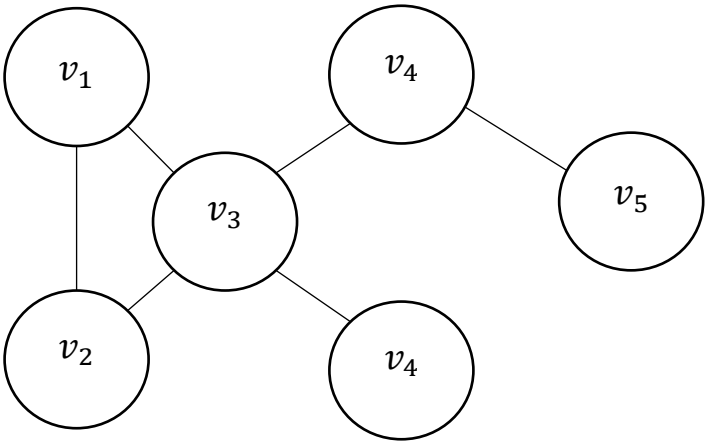
На рис. 5.14а зображено граф, для якого потрібно побудувати остовне дерево, а відповідне йому остовне дерево зображено на рис. 5.14б.

5.3 Завдання до розділу 5

Завдання 5.1 Визначте тип графу (орієнтований/неорієнтований), побудуйте матрицю суміжності для графу

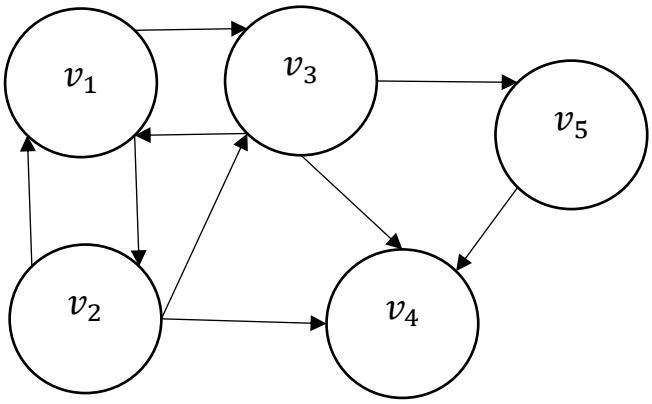


а)



б)

в)



Завдання 5.2. Побудуйте граф за матрицею суміжності

а)

	v_1	v_2	v_3	v_4
v_1		*1*		
v_2		*1*		
v_3	*1*			
v_4			*1*	*1*

б)

	v_1	v_2	v_3	v_4	v_5
v_1		*1*			*1*
v_2	*1*				
v_3		*1*	*1*	*1*	*1*
v_4			*1*		
v_5	*1*	*1*			*1*

в)

	v_1	v_2	v_3	v_4	v_5
v_1	*1*		*1*		*1*
v_2				*1*	*1*
v_3			*1*		*1*
v_4			*1*		
v_5	*1*	*1*		*1*	*1*

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. O. Levin Discrete Mathematics: An Open Introduction 414p – доступно за посиланням: https://discrete.openmathbooks.org/pdfs/dmoi3-tablet.pdf

2. Susanna S. Epp Discrete Mathematics with applications: Forth Edition – BrooksCole 993 p – доступно за посиланням: https://archive.org/details/2-discrete-mathematics-with-applications-by-susanna-s.-epp-4th-edition/mode/2up

3. Балога С.І. Дискретна математика. Навчальний посібник. – Ужгород: ПП «АУТДОРШАРК», 2021. – 124 с. – доступно за посиланням: https://www.uzhnu.edu.ua/uk/infocentre/get/42936

4. Matthew Towers MATH0005 Algebra 1, 2022. –127 p. . – доступно за посиланням https://www.ucl.ac.uk/~ucahmto/0005_2021/MATH0005_lecture_notes.pdf

5. S. Burrisand H.P. SankappanavarA Course in Universal Algebra.The Millennium Edition–331 p. – доступно за посиланням https://www.math.uwaterloo.ca/~snburris/htdocs/UALG/univ-algebra.pdf

6. Ryan O'Donnell Analysis of boolean functions, arxiv.org, 419 p. –доступно за посиланням https://arxiv.org/pdf/2105.10386.pdf

7. О.Л. Темнікова Дискретна математика. Частина 2 – К. «КПІ», 2019, 128с – доступно за посиланням https://ela.kpi.ua/bitstream/123456789/42842/1/LectureDM2Temnikova.pdf