Міністерство освіти і науки України
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»
Факультет інформатики
Кафедра інформатики

Кваліфікаційна робота
Освітній ступінь – магістр

на тему: **«Дослідження можливостей Generative AI для створення контенту»**

Виконав : студент 2-го року навчання,
Спеціальності
122 Комп’ютерні науки
Столяров Владислав Сергійович
Керівник : Олецький О.В.
кандидат технічних наук, доцент
Рецензент
Кваліфікаційна робота захищена
з оцінкою
Секретар ЕК
«» 2026 р.

Київ – 2026

Міністерство освіти і науки України

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»

Кафедра інформатики факультету інформатики

ЗАТВЕРДЖУЮ
Зав.кафедри інформатики,
к.ф-м.н., доц. Жежерун О. П.
_____
„____”_____2026 р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

на кваліфікаційну роботу

Студенту Столярову В.С. факультету інформатики 2 курсу

Тема: «Дослідження можливостей Generative AI для створення контенту»
Зміст текстової частини кваліфікаційної роботи:
Календарний план
Зміст
Анотація
Вступ
1 Теоретичні засади оцінювання якості згенерованого навчального контенту
2 Методика оцінювання якості згенерованого навчального контенту
3 Експериментальна перевірка методики
Висновки
Список літератури
Додатки (за необхідністю)

Дата видачі “____” _____ 2025 р. Керівник _____

(підпис)

Завдання отримав _____

(підпис)

Тема: «Дослідження можливостей Generative AI для створення контенту»

Календарний план виконання роботи

№	Назва етапу магістерської роботи	Термін виконання	Примітка
1	Отримання завдання магістерської роботи	05.11.2025	
2	Огляд літератури за темою дослідження	01.12.2025	
3	Формулювання теоретичних засад та постановки задачі	20.12.2025	
4	Розробка композитної метрики та DSL генеративного сценарію	31.01.2026	
5	Реалізація програмного пакета content_eval на Python	15.03.2026	
6	Формування корпусу генеративних сценаріїв	15.04.2026	
7	Передзахист магістерської роботи	28.04.2026	
8	Генерація навчальних фрагментів	10.05.2026	
9	Проведення людської оцінки двома експертами	13.05.2026	
10	Статистичний аналіз результатів	16.05.2026	
11	Доопрацювання текстової частини	24.05.2026	
12	Захист магістерської роботи	08.06.2026	

Студент Столяров В. С. _____

Керівник Олецький О. В. _____

«____» _____ 2026 р.

ЗМІСТ

АНОТАЦІЯ

У роботі запропоновано методику кількісного оцінювання якості згенерованого великими мовними моделями навчального тексту. Розроблено композитну метрику, що поєднує чотири компоненти: семантичну близькість до референсу, покриття ключових понять, внутрішню когерентність та фактологічну узгодженість на основі крос-модельної верифікації двома різними LLM. Сформульовано рубрику людської оцінки з педагогічним критерієм дидактичної цінності. Експериментально перевірено методику на корпусі з 60 фрагментів за темами курсу «Алгоритми і структури даних» НаУКМА, згенерованих моделями Claude Sonnet 4.6 та GPT-5.4. Розраховано інтеррейтерну узгодженість за коефіцієнтом Cohen's κ та виявлено ефект «стелі» у людській оцінці на сучасних топ-моделях. Програмна реалізація методики виконана у вигляді Python-пакета content_eval.

Ключові слова: великі мовні моделі, генеративний штучний інтелект, оцінювання якості тексту, навчальний контент, композитна метрика, рубрика, LLM-as-a-judge, Cohen's κ.

ВСТУП

Останнє десятиліття стало періодом стрімкого розвитку технологій генеративного штучного інтелекту (Generative AI), а особливо великих мовних моделей (Large Language Models, LLM). Починаючи з 2017 року, коли була запропонована архітектура Transformer, що стала фундаментом сучасних мовних моделей, і завершуючи 2026 роком, провідні LLM — Claude Sonnet від Anthropic, GPT від OpenAI, Gemini від Google DeepMind — досягли рівня якості генерації тексту, на якому стало можливим їхнє широке застосування для задач, які ще донедавна вважалися виключно сферою людської експертизи. Запуск ChatGPT у листопаді 2022 року й досягнення сервісом ста мільйонів активних користувачів за два місяці перетворили LLM із суто академічного об'єкта дослідження на технологію масового користувацького попиту.

Освітні установи у всьому світі, а також провідні компанії технологічної галузі, активно інтегрують мовні моделі у процес навчання — як для адаптивного репетиторства, так і для генерації пояснювальних навчальних матеріалів. Водночас залишається відкритим питання кількісного оцінювання якості такого матеріалу. Класичні автоматичні метрики оцінювання генеративного тексту (BLEU, ROUGE, BERTScore, BLEURT) побудовані як міри лексичної або семантичної близькості до еталона і непридатні для повноцінного оцінювання навчального контенту, для якого канонічний еталон зазвичай не існує. Сучасні підходи на основі оцінювання мовною моделлю-суддею забезпечують гнучкість критеріїв, але мають відомі обмеження — зокрема упередженість самооцінки, — що потребують методологічних запобіжників. Кожен з наявних підходів охоплює лише частину вимірів якості, важливих для навчального тексту. Це обґрунтовує побудову нової композитної методики, що поєднує сильні сторони наявних рішень.

Об'єктом дослідження є процес кількісного оцінювання якості згенерованого великими мовними моделями навчального контенту.

Предметом дослідження є методика композитного оцінювання якості згенерованого навчального тексту, що поєднує семантичну близькість до референсу, покриття ключових понять, внутрішню когерентність та фактологічну узгодженість на основі крос-модельної верифікації і валідується через зіставлення з оцінкою людини-експерта за рубрикою з педагогічним критерієм.

Метою дослідження є розробка і експериментальна перевірка методики оцінювання якості згенерованого навчального контенту, що корелює з людською експертною оцінкою і дозволяє автоматизовано отримати інтегральну оцінку якості згенерованого фрагмента. Конкретні завдання, пов'язані з досягненням цієї мети, такі:

— проаналізувати існуючі підходи до автоматичного оцінювання якості NLG-текстів і визначити їхні обмеження стосовно навчального домену;

— запропонувати формальний опис генеративного сценарію навчального контенту у вигляді декларативного DSL на основі YAML;

— розробити композитну метрику Q як адитивну комбінацію чотирьох компонент: семантичної близькості (S), покриття ключових понять (O), внутрішньої когерентності (C) та фактологічної узгодженості (F);

— сформулювати рубрику людської оцінки з педагогічним критерієм дидактичної цінності;

— реалізувати методику у вигляді програмного пакета на мові Python;

— згенерувати корпус навчальних фрагментів моделями Claude Sonnet 4.6 та GPT-5.4 і виконати їхнє оцінювання як автоматичною метрикою, так і двома незалежними людськими експертами;

— виконати статистичний аналіз отриманих результатів і сформулювати висновки про ефективність методики та перспективи її розвитку.

Магістерська робота складається з трьох основних розділів, що слідують за вступом. У першому розділі розглядаються теоретичні засади оцінювання якості згенерованого навчального контенту: розвиток великих мовних моделей, огляд автоматичних метрик NLG-evaluation, методики людської оцінки на основі рубрик,

підхід LLM-as-a-judge та специфіка освітнього домену. Другий розділ описує власне методику оцінювання: декларативний DSL генеративного сценарію, формальне визначення композитної метрики Q з чотирма компонентами, рубрику людської оцінки, процедуру підбору ваг. Третій розділ присвячено експериментальній перевірці методики: програмній реалізації, формуванню корпусу сценаріїв, проведенню людської оцінки, статистичному аналізу узгодженості та кореляції, обговоренню отриманих результатів і перспектив розвитку запропонованого підходу.

Розділ 1. Теоретичні засади оцінювання якості згенерованого навчального контенту

1.1 Generative AI та великі мовні моделі: огляд, історія, сучасний стан

Термін Generative AI (генеративний штучний інтелект) описує клас алгоритмів машинного навчання, побудованих для створення нового контенту — тексту, зображень, аудіо, відео, програмного коду — на основі ймовірнісного моделювання великих масивів навчальних даних. На відміну від дискримінативних моделей, що класифікують або передбачають характеристики наявних об'єктів, генеративні моделі вчаться апроксимувати розподіл, з якого походить навчальна вибірка, а потім семплюють із цього розподілу нові, раніше не існуючі екземпляри. У сучасних реалізаціях генеративний штучний інтелект ґрунтується на нейромережевих архітектурах із сотнями мільйонів і навіть сотнями мільярдів параметрів, що дозволяє моделювати складні розподіли природних даних із високою якістю.

Великі мовні моделі (Large Language Models, LLM) є провідним напрямом генеративного штучного інтелекту, що працює з текстом природної мови. Принцип роботи LLM зводиться до автокерованого передбачення наступного токена: модель отримує на вхід префікс послідовності і повертає розподіл імовірностей над усім словником можливих наступних токенів. Відбір токена з цього розподілу та ітеративне приєднання його до префікса породжує текст довільної довжини. Незважаючи на математичну простоту самої постановки, виявилося, що при достатньому масштабі моделі та обсягу навчальних даних така архітектура здатна засвоювати не лише поверхневі статистичні залежності мови, а й структури вищого порядку — синтаксис, семантику, фактологію, прагматику дискурсу, риторичні прийоми.

Технологічним фундаментом сучасних LLM є архітектура Transformer, запропонована у роботі Vaswani та співавторів «Attention is all you need» 2017 року [1]. Ключовим внеском авторів став механізм самоуваги (self-attention), що дозволив моделі

під час обробки кожного токена послідовності динамічно зважувати релевантність усіх інших токенів цього самого контексту. Це дало змогу відмовитися від рекурентних і згорткових з'єднань, що становили основу попередніх архітектур, на користь повністю паралельної обробки послідовностей. Як наслідок, нова архітектура виявилася ефективною з обчислювальної точки зору при навчанні на сучасних графічних процесорах і легко масштабованою до моделей з мільярдами параметрів. Усі без винятку провідні мовні моделі останніх років — GPT від OpenAI, Claude від Anthropic, Gemini від Google DeepMind, LLaMA від Meta, Grok від xAI — побудовані саме на цьому фундаменті або його модифікаціях.

Хронологія розвитку великих мовних моделей може бути коротко представлена так. У 2018 році OpenAI опублікувала першу версію GPT (Generative Pre-trained Transformer) із 117 мільйонами параметрів. У 2019 році з'явилася GPT-2 з 1.5 мільярда параметрів, що вже демонструвала здатність генерувати зв'язний текст довжиною в кілька абзаців на широкому колі тем. Справжній якісний стрибок відбувся у 2020 році з опублікуванням GPT-3 — моделі з 175 мільярдами параметрів. У роботі Brown et al. [2] автори описали феномен навчання на нечисленних прикладах: GPT-3 виявилася здатною вирішувати нові задачі лише на основі кількох зразків у вхідному промпті без явного донавчання. Цей результат був важливим, оскільки демонстрував, що достатнє масштабування LLM породжує емерджентні здатності, які не з'являються на менших моделях.

Переломним моментом у поширенні LLM серед широкої публіки стало 30 листопада 2022 року — дата офіційного запуску ChatGPT, доступного через веб-інтерфейс безкоштовного діалогового продукту на основі GPT-3.5. ChatGPT досяг 100 мільйонів активних користувачів за два місяці, що стало найшвидшим зростанням споживчого онлайн-сервісу в історії [3]. Цей факт перетворив генеративні мовні моделі із суто академічного об'єкта дослідження на технологію масового користувацького попиту. Упродовж 2023 року OpenAI випустила GPT-4 — модель із суттєво вищою якістю міркувань, здатністю обробляти зображення та довші контексти. У 2024 році

з'явилися GPT-4o з мультимодальною підтримкою тексту, зображень і голосу та оптимізовані за вартістю варіанти серії GPT-4o-mini. Подальші покоління моделей продовжили нарощувати розмір, якість навчальних даних і застосовувати техніки постпідготовки, зокрема RLHF (Reinforcement Learning from Human Feedback) [4], що дозволяє узгоджувати поведінку моделі з людськими очікуваннями через зворотний зв'язок асесорів.

Паралельно з OpenAI компанія Anthropic, заснована у 2021 році вихідцями з OpenAI, розвинула альтернативну родину моделей під назвою Claude. Особливістю підходу Anthropic є застосування методу Constitutional AI, описаного у роботі Bai et al. [5]. У межах цього підходу замість виключно людського зворотного зв'язку модель навчається на основі набору принципів («конституції»), що задають критерії безпечної, чесної та корисної поведінки; самокритика моделі за цими принципами використовується як додатковий сигнал навчання. Перша версія Claude була випущена у березні 2023 року, Claude 2 — у липні 2023, родина Claude 3 (Opus, Sonnet, Haiku) — у березні 2024, Claude 3.5 Sonnet — у червні 2024. Послідовні покоління моделей покращували здатність до складних міркувань, аналізу довгих документів обсягом до 200 тисяч токенів і точності фактологічних тверджень.

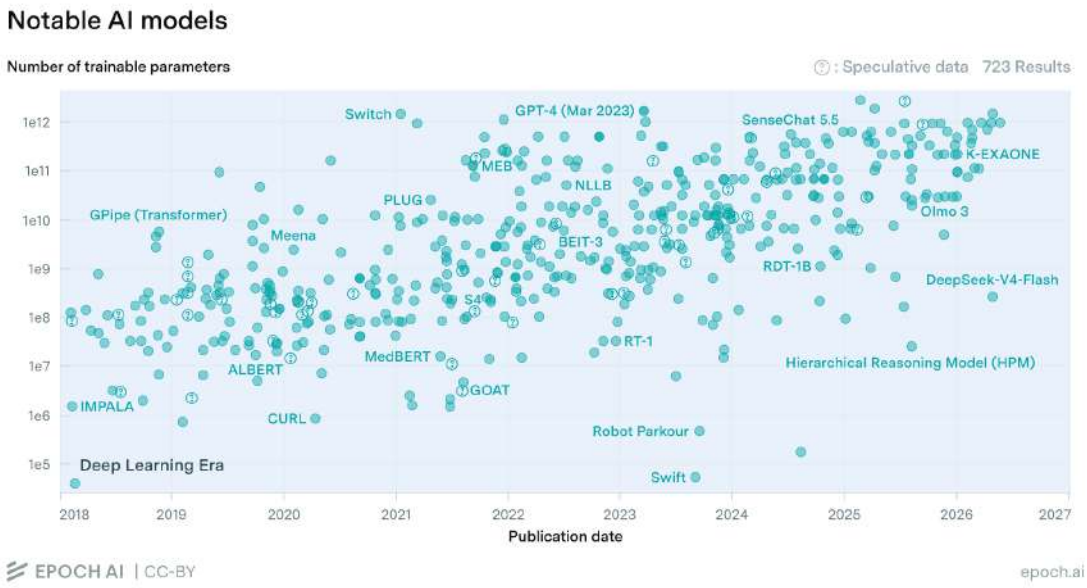

Рис. 1.1 — Зростання кількості параметрів провідних великих мовних моделей упродовж 2018–2026 років (логарифмічна шкала). Джерело: Epoch AI, ліцензія CC-BY

У 2023–2025 роках відбулася диверсифікація ринку LLM. Окрім OpenAI та Anthropic, важливими гравцями стали Google DeepMind із родиною моделей Gemini, інтегрованих у пошук Google, Android-смартфони та Workspace-сервіси; Microsoft, що інтегрувала GPT-моделі через продукт Copilot у пакет Office, Windows і Visual Studio; Meta, що випустила відкриті ваги моделей LLaMA, LLaMA 2 (липень 2023) і подальших версій, надавши дослідницькій спільноті інструменти для незалежного дослідження; xAI Ілона Маска з моделлю Grok, інтегрованою у соціальну мережу X. У жовтні 2023 року OpenAI заснувала окремий комерційний продукт ChatGPT Enterprise для компаній, а пізніше — ChatGPT Edu для університетів і освітніх закладів. Економічний ефект розвитку LLM призвів до значного зростання капіталізації виробників відповідної інфраструктури: ринкова капіталізація NVIDIA, що є основним постачальником GPU для навчання LLM, зросла з приблизно 300 мільярдів доларів на початку 2023 року до понад 3 трильйонів у середині 2024 року, що зробило компанію однією з найдорожчих у світі [6].

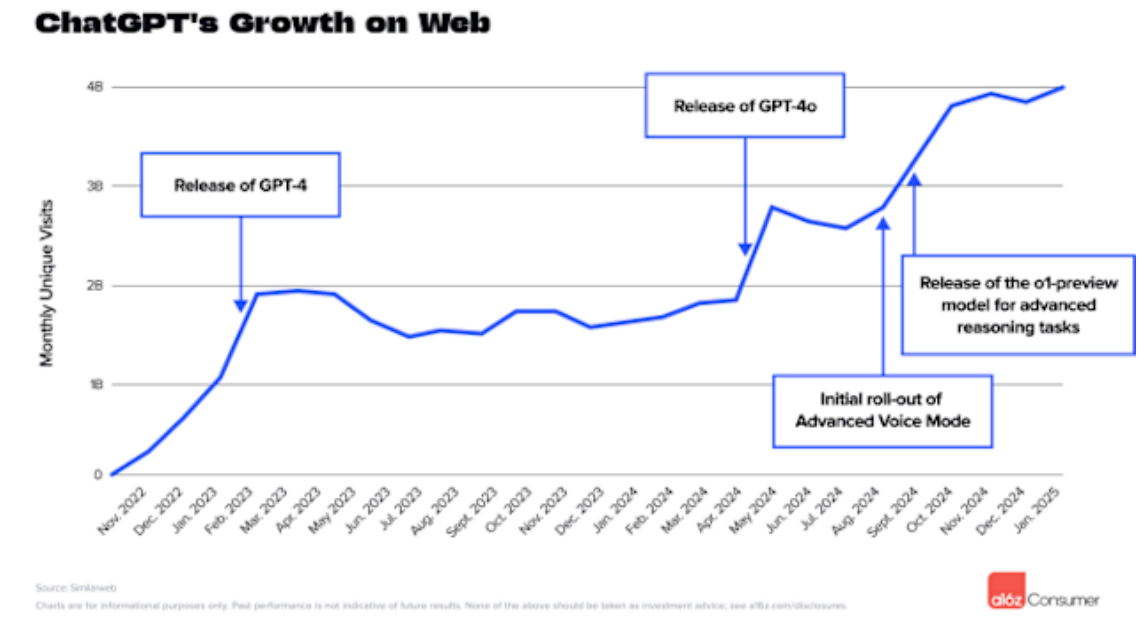

Рис. 1.2 — Динаміка кількості унікальних місячних візитів на веб-платформу ChatGPT з листопада 2022 року по січень 2025 року

Сучасні LLM, представлені у даному дослідженні моделями Claude Sonnet 4.6 (Anthropic) і GPT-5.4 (OpenAI), демонструють здатність генерувати текст, який за низкою параметрів — граматичною правильністю, стилістичною відповідністю, фактологічною точністю на загальновідомих темах — наближається до якості, продукованої людиною-експертом. Ця теза підтверджується низкою стандартизованих бенчмарків. На MMLU (Massive Multitask Language Understanding) — тесті з 57 предметних областей, від базової математики до юриспруденції та клінічної медицини, запропонованому Hendrycks et al. [7], — провідні моделі досягають результату понад 85%, що перевищує середній рівень кваліфікованого випускника університету. На бенчмарку HumanEval, запропонованому Chen et al. [8] для оцінювання генерації функціонального Python-коду, провідні моделі вирішують понад 90% задач. На бенчмарку MATH, що містить олімпіадні математичні задачі, якість зросла приблизно з 10% у 2021 році до понад 85% станом на 2025 рік.

Окремим і швидко зростаючим напрямом застосування LLM є генерація навчального контенту. До нього належать створення пояснювальних фрагментів для дисциплін STEM, формулювання тестових завдань різного рівня складності, адаптація матеріалу під різну підготовку студента, переклад освітніх ресурсів, інтерактивне репетиторство. Низка досліджень [9] продемонструвала, що сучасні LLM здатні генерувати освітні тексти, які за якістю не поступаються матеріалам, підготовленим викладачами, особливо у предметних областях з великим обсягом тренувальних даних — програмуванні, базовій математиці, природничих науках. Інтенсивно розвиваються комерційні освітні продукти на основі LLM: Khanmigo від Khan Academy, ChatGPT Edu від OpenAI, Duolingo Max та подібні.

Водночас при використанні LLM у навчанні постають окремі практичні виклики. По-перше, проблема галюцинацій — модель може видавати фактологічно неточну

інформацію впевненим тоном, що особливо небезпечно в освітньому контексті, де користувач-студент не має достатньої експертизи для перевірки тверджень. По-друге, питання покриття предметної області — модель може випустити важливі поняття або, навпаки, надмірно сконцентруватися на другорядних аспектах, що не відповідає логіці навчальної програми. По-третє, дидактичний компонент: фрагмент тексту може бути фактологічно правильним, але непридатним для навчальних цілей через брак прикладів, аналогій, поетапного розгортання матеріалу від простого до складного. По-четверте, оцінка цих властивостей вимагає або експертної роботи викладачів (дорогої й важко масштабованої), або автоматизованої методики, наразі недостатньо розвинутої для специфіки навчального тексту.

Сукупність зазначених факторів формує контекст, у якому виникає потреба у комплексній автоматизованій методиці оцінювання якості згенерованого навчального контенту. Така методика повинна враховувати декілька різнорідних аспектів якості одночасно: семантичну близькість до еталонного матеріалу, покриття предметної онтології, внутрішню когерентність викладу та фактологічну узгодженість тверджень. Її ефективність має бути валідована через зіставлення з людською експертною оцінкою. Розробці саме такої методики присвячено подальші розділи роботи. Далі в цьому розділі розглянуто існуючі підходи до оцінювання якості NLG-текстів, виявлено їхні обмеження стосовно навчального домену та обґрунтовано необхідність синтезу нового підходу, що поєднує сильні сторони кожного з них.

1.2 Автоматичні метрики оцінювання якості NLG-текстів

Завдання кількісного оцінювання якості згенерованого тексту виникло разом із самою задачею Natural Language Generation (NLG) і стало особливо актуальним для систем машинного перекладу й автоматичного реферування у 2000-х роках. Кожне нове покоління NLG-метрик відображало зростання можливостей моделей, які треба було

оцінювати. Можна виділити три послідовні покоління автоматичних метрик, що мають принципово різну природу.

Перше покоління становлять метрики, побудовані на підрахунку лексичної близькості між згенерованим текстом і еталонним посиланням. Найвідомішою серед них є BLEU (Bilingual Evaluation Understudy), запропонована Papineni та співавторами у 2002 році [10]. Метрика обчислюється як модифікована точність n-грам: для n = 1, 2, 3, 4 рахується частка n-грам згенерованого тексту, які зустрічаються у еталонному перекладі, а результат комбінується через геометричне середнє зі штрафом за надмірну стислість. Близька за природою метрика ROUGE (Recall-Oriented Understudy for Gisting Evaluation) була запропонована Lin у 2004 році для оцінювання автоматичного реферування [11]; вона існує у кількох варіантах — ROUGE-N для n-грамного покриття, ROUGE-L для найдовшої спільної підпослідовності. Метрики цього покоління мають два фундаментальних обмеження. По-перше, вони реагують на лексичну форму, а не на семантичний зміст: парафраза з тими ж смисловими відношеннями, але іншими словами, отримує низький бал. По-друге, вони вимагають наявності одного або кількох еталонних текстів, що для навчального контенту створює методологічну проблему — для будь-якої теми існує велика кількість семантично еквівалентних, але лексично різних викладів матеріалу.

Друге покоління формують метрики на основі векторних представлень слів, що з'явилися із поширенням контекстуалізованих векторних представлень в архітектурі BERT та її аналогах. BERTScore, запропонована Zhang та співавторами у 2020 році [12], обчислює оцінку як зважену суму косинусних схожостей між векторами токенів згенерованого тексту і токенів еталона. На відміну від BLEU, BERTScore оперує семантичною близькістю, тому парафрази отримують високу оцінку. BLEURT, запропонована Sellam та співавторами у 2020 році [13], використовує спеціально донавчений Transformer як функцію оцінювання: модель тренують на масштабних даних людських оцінок, і вона повертає неперервний бал, що корелює з оцінкою людини-експерта. BARTScore, запропонована Yuan та співавторами у 2021 році [14],

використовує імовірнісну природу моделі seq2seq BART для обчислення оцінки: значення визначається як логарифмічна ймовірність генерації згенерованого тексту з еталона, що інтерпретується як «наскільки модель готова відтворити текст за умови знання еталона». Хоча друге покоління метрик істотно перевершує перше за здатністю вловлювати семантику, у нього зберігаються три суттєві обмеження: залежність від наявності еталонного тексту, відсутність явного оцінювання фактологічної коректності, нечутливість до структурних і дидактичних характеристик тексту.

Третє покоління метрик з'явилося у 2023 році разом із зростанням якості провідних великих мовних моделей до рівня, на якому самі моделі стали придатними для виконання ролі оцінювача. Цей напрям отримав назву оцінювання мовною моделлю-суддею (LLM-as-a-judge). Найвідомішою реалізацією є G-Eval, запропонована Liu та співавторами у 2023 році [15]: вхідний промпт містить інструкцію з критеріями оцінювання, текст для оцінки, і модель повертає числовий бал у заданому діапазоні. Перевагою цього покоління є гнучкість: одна й та сама архітектура верифікатора працює для будь-якого критерію якості, який можна сформулювати природною мовою. Завдяки цьому стає можливим оцінювати фактологічну узгодженість, релевантність, когерентність, тон і навіть специфічні характеристики, як-от наявність прикладів або поетапного викладу. Водночас метрики третього покоління успадковують усі обмеження сучасних LLM: схильність до галюцинацій у власній оцінці, чутливість до формулювання промпта, варіативність результату між запусками. Окремим, добре задокументованим обмеженням є упередженість самооцінки — схильність моделі до власних відповідей; цей феномен і способи його усунення детально розглядаються далі в роботі.

Окремою категорією є метрики, орієнтовані на оцінювання когерентності тексту, тобто його внутрішньої логічної зв'язності. Основоположною роботою у цій галузі є дослідження Lapata та Barzilay [16], які запропонували сітково-сутнісну модель когерентності, що відстежує входження сутностей у послідовних реченнях. У сучасних реалізаціях когерентність часто оцінюють через косинусну близькість векторних

представлень сусідніх речень або через оцінку мовною моделлю перплексії послідовності речень. Метрики когерентності залишаються відносно слабо стандартизованими порівняно з лексичною і семантичною близькістю.

Загальний огляд еволюції автоматичних NLG-метрик подано у систематичному дослідженні Sai та співавторів [17]. Автори демонструють, що жодна окрема метрика, доступна станом на 2022 рік, не корелює з людською оцінкою на рівні $r > 0.7$ в усіх типах NLG-задач. Це породжує тенденцію до композитних метрик, які поєднують декілька різнорідних сигналів якості й підбирають їх ваги через оптимізацію кореляції з людською оцінкою на тренувальній вибірці. Композитна метрика, запропонована у даному дослідженні, є реалізацією саме цього підходу — з акцентом на специфіку навчального контенту.

1.3 Людська оцінка та методики на основі рубрик

Незважаючи на швидкий розвиток автоматичних метрик, людська експертна оцінка зберігає статус «золотого стандарту» у дослідженнях якості NLG-систем. Це пояснюється тим, що автоматичні метрики оцінюють лише ті аспекти якості, які операціоналізовано у відповідних обчислювальних формулах; людина-експерт здатна холістично оцінити текст за критеріями, які важко формалізувати — стилістична доречність, ясність викладу для конкретної аудиторії, дидактична цінність, переконливість аргументації. У переважній більшості сучасних робіт з оцінювання генеративних текстів коефіцієнт кореляції автоматичної метрики з людською оцінкою використовується як основний показник валідності метрики, що ставить людську оцінку у роль базової істини [18].

Стандартизованим інструментом проведення людської оцінки є рубрика — структурований протокол, що містить перелік окремих критеріїв і шкалу оцінок для кожного з них. Типова рубрика для NLG-задач включає від трьох до семи критеріїв, що оцінюються за дискретною шкалою 1–5 або 1–7. Найпоширенішими критеріями є

плавність мови (граматична правильність і природність викладу), адекватність змісту (відповідність еталонному значенню для задач перекладу або реферування), когерентність (внутрішня логічна зв'язність), фактологічна узгодженість (коректність тверджень). Для специфічних доменів — освіти, медицини, юриспруденції — додають предметно-орієнтовані критерії: для освіти таким розширенням є дидактична цінність або педагогічна релевантність, для медицини — діагностична відповідність, для юриспруденції — правова точність. Оцінювач отримує текст і протокол, прочитує текст один або кілька разів і виставляє бал за кожним критерієм; підсумкова оцінка визначається як середнє арифметичне балів за окремими критеріями.

Принциповою вимогою для забезпечення валідності людської оцінки є залучення щонайменше двох незалежних оцінювачів. Одна оцінка є суб'єктивною за визначенням, і її відтворюваність неможливо емпірично перевірити без другого незалежного джерела. Залучення двох і більше експертів дозволяє обчислити інтеррейтерну узгодженість — кількісний показник того, наскільки оцінки збігаються між собою. Якщо узгодженість висока, рубрика є відтворюваною і вимірює щось об'єктивне; якщо узгодженість низька — або протокол сформульований нечітко, або експерти трактують його по-різному, що потребує додаткової калібровки.

Стандартним показником інтеррейтерної узгодженості для категоріальних даних є коефіцієнт κ (kappa), запропонований Cohen у 1960 році для номінальних шкал. Класична κ розраховується за формулою (1.1):

$$\kappa = \frac{P_o - P_e}{1 - P_e}$$

де P_o — спостережувана пропорція збігів між двома оцінювачами, P_e — пропорція збігів, очікувана за випадковості. Значення κ = 0 означає узгодженість на рівні випадковості, κ = 1 — повну узгодженість. Для ординальних шкал, до яких належить більшість рубрик з оцінками 1–5, застосовується квадратично-зважена версія κ, запропонована Cohen у 1968 році [19]. У цій версії розходження сусідніх балів

(наприклад, 4 і 5) штрафується менше, ніж розходження віддалених (наприклад, 1 і 5), що відображає ординальну природу шкали.

Стандартом інтерпретації числового значення κ є шкала, запропонована Landis та Koch у 1977 році [20]: $\kappa < 0.00$ — неприйнятна узгодженість (нижче випадковості), 0.00–0.20 — незначна, 0.21–0.40 — задовільна, 0.41–0.60 — помірна, 0.61–0.80 — істотна, 0.81–1.00 — майже повна. Дослідницька практика NLG-оцінювання вимагає узгодженості щонайменше на рівні помірної ($\kappa > 0.40$) для того, щоб результати рубрик-оцінки вважалися надійними.

Окремою складовою методології на основі рубрик є процедура калібровки оцінювачів перед самостійним оцінюванням. Калібровка передбачає спільне обговорення кількох (зазвичай від трьох до п'яти) пілотних фрагментів усіма експертами, виявлення розходжень в інтерпретації критеріїв і формулювання спільного розуміння. Без цього етапу інтеррейтерна узгодженість зазвичай помітно нижча, особливо для критеріїв, що допускають варіативну інтерпретацію (наприклад, для дидактичної цінності, де поняття «достатня кількість прикладів» може трактуватися по-різному). У роботі van der Lee та співавторів [18] калібровка визначена як одна з шести найкращих практик організації людської оцінки в NLG-дослідженнях.

1.4 Оцінювання мовною моделлю-суддею: підхід, переваги, обмеження

Підхід оцінювання мовною моделлю-суддею (LLM-as-a-judge) виник у 2022–2023 роках разом із зростанням якості провідних LLM до рівня, на якому ці моделі стали придатними не лише для генерації тексту, але і для його оцінювання. Концептуальною засадою цього напряму є спостереження, що якщо мовна модель здатна писати на рівні людини-експерта, вона з певними обмеженнями здатна і оцінювати чужий текст із аналогічних позицій. Методологічний фундамент закладено у роботі Zheng та співавторів «Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena», представленій на конференції NeurIPS 2023 [21]. Автори продемонстрували, що оцінки, виставлені

моделлю GPT-4 у ролі судді, корелюють із оцінками людських експертів на рівні $r > 0.8$, що порівнянно з узгодженістю двох незалежних людських оцінювачів між собою.

Існують дві основні постановки задачі. Першою є поелементне оцінювання — модель отримує рубрику, читає окремий текст і повертає бал за кожним критерієм. Другою є попарне порівняння — модель отримує два альтернативних виходи і визначає, який з них кращий за заданими критеріями (або вказує на рівноцінність). Попарне порівняння є робастнішим до калібровки шкали і часто дає вищу узгодженість з людською оцінкою; поелементне оцінювання натомість дає однозначний числовий бал і легше масштабується.

Серед переваг підходу можна виділити три ключові. По-перше, гнучкість критеріїв: одна й та сама архітектура верифікатора працює для будь-якої властивості якості, яку можна сформулювати природною мовою — фактологічна узгодженість, релевантність темі, дидактична цінність, тон, стилістична доречність. Не потрібно навчати окрему модель під кожен критерій. По-друге, масштабованість: автоматизована оцінка десятків тисяч фрагментів коштує на порядки менше, ніж залучення людських експертів. По-третє, доступність: підхід реалізується через стандартний API провідних моделей без необхідності розгортання спеціалізованої інфраструктури.

Водночас оцінювання мовною моделлю-суддею має задокументовані обмеження, які важливо враховувати при побудові метрики. Ключовим серед них є упередженість самооцінки — схильність моделі надавати перевагу власним відповідям. У роботі Panickssery, Bowman та Feng [22] експериментально показано, що великі мовні моделі при виборі між власною відповіддю і відповіддю іншої моделі статистично значуще частіше обирають саме власну, навіть коли якість обох варіантів за людською оцінкою порівнянна. Це створює методологічну проблему для випадків, коли потрібно оцінити якість тексту, згенерованого моделлю А, не залучаючи саму цю модель до перевірки. Стандартним способом усунення упередженості самооцінки є крос-модельна верифікація: текст, згенерований моделлю А, оцінюється моделлю Б, і навпаки. У

даному дослідженні цей підхід застосовано для F-компоненти композитної метрики: фрагменти Claude перевіряються GPT, а фрагменти GPT — Claude.

Другим відомим обмеженням є упередження позиції у попарних оцінках: модель статистично частіше обирає варіант, представлений першим у промпті. Спосіб усунення — рандомізація порядку або подвійне оцінювання з обміном позицій. Третім обмеженням є стохастичність відповіді: при ненульовій температурі семплінгу одна й та сама пара (текст, критерій) може отримати різні оцінки у різних запусках. Стандартним рішенням є встановлення temperature = 0 (детермінований режим), що зводить варіативність до мінімуму. Четвертим обмеженням є упередження довжини — за рівних інших умов модель частіше визнає довший варіант якіснішим, що не завжди корелює з людською оцінкою.

Окремим методичним питанням є те, як саме мовна модель-суддя оцінює фактологічну узгодженість. Класична постановка — попросити модель повернути єдиний числовий бал за критерієм коректності від 1 до 5 — на практиці працює погано: модель схильна повертати високі бали без чіткого пояснення. У сучасних реалізаціях, зокрема у F-компоненті даного дослідження, використовується інший підхід: модель просять виокремити з тексту конкретні фактологічні твердження і кваліфікувати кожне з них як коректне, сумнівне або помилкове. Підсумкова оцінка розраховується як зважена частка коректних тверджень. Такий спосіб має дві переваги: модель змушена формулювати свої заперечення в конкретних твердженнях (а не давати безпідставний загальний бал), а фінальна оцінка є інтерпретованою — для будь-якого фрагмента можна вивести список тверджень, які модель визнала сумнівними.

1.5 Специфіка оцінювання навчального контенту

Навчальний текст має властивості, що відрізняють його від інших видів генеративного виходу і роблять стандартні метрики, описані вище, недостатніми для

повноцінного оцінювання його якості. Розглянемо ці властивості та їхні наслідки для побудови метрики.

По-перше, для навчального тексту відсутнє поняття «правильного перекладу» або «канонічного реферату». Одна й та сама тема — наприклад, успадкування у Java — може бути викладена різними викладачами, у різних підручниках, у різних навчальних матеріалах десятками семантично еквівалентних, але лексично і структурно різних способів. Це означає, що метрики на кшталт BLEU або BERTScore, які вимагають еталонного тексту і обчислюють близькість до нього, мають фундаментальне обмеження у застосуванні до навчального домену: будь-який вибір еталону є довільним і не охоплює усього простору прийнятних варіантів викладу.

По-друге, навчальний текст оцінюється не лише за лексично-семантичними характеристиками, але і за покриттям предметної онтології — переліку ключових понять, які мають бути розкриті у фрагменті відповідно до навчальної програми. Якщо фрагмент про успадкування у Java не згадує ключові поняття extends, super, @Override, метод toString і подібні, він є неповним з педагогічної точки зору, навіть якщо стилістично бездоганний. Жодна з класичних автоматичних метрик не враховує покриття предметних понять.

По-третє, навчальний текст повинен мати дидактичну цінність — здатність допомогти студенту зрозуміти матеріал. Дидактична цінність є комплексним критерієм, що включає наявність ілюстративних прикладів, аналогій, поетапного розгортання матеріалу від простого до складного, явного зв'язування нових понять із попередньо засвоєними. Текст, що є фактологічно бездоганним і повним за покриттям понять, але викладений у форматі сухого довідника без жодного прикладу, не виконує свою основну функцію у навчанні. Жодна стандартна автоматична метрика не оцінює цю властивість, а серед людських рубрик вона рідко з'являється явним критерієм навіть для задач освітнього профілю.

По-четверте, оцінювання навчального тексту вимагає врахування цільової аудиторії. Один і той же фрагмент може бути прийнятним для студента третього курсу

спеціальності «Комп'ютерні науки», але занадто складним для першокурсника без попередньої підготовки. Контекст аудиторії впливає на критерії ясності викладу і дидактичної цінності — той самий приклад може бути доречним або неприйнятним залежно від базових знань читача.

Серед сучасних досліджень, що розглядають специфіку генерації освітнього тексту, варто згадати роботу Kasneci та співавторів «ChatGPT for good?» [9], опубліковану у журналі Learning and Individual Differences у 2023 році. Автори систематизують можливості й виклики застосування великих мовних моделей в освітньому процесі: можливості — персоналізація навчання, генерація вправ, інтерактивне тьюторство, переклад освітніх ресурсів; виклики — фактологічна точність, ризик некритичного запозичення згенерованого матеріалу студентами, етичні питання атрибуції. Низка пізніших робіт продовжує цей напрям, зокрема із застосуванням мовних моделей для генерації пояснень математичних і програмістських тем та формулювання тестових завдань різного рівня складності.

Підсумовуючи зазначене, для повноцінного оцінювання якості згенерованого навчального тексту потрібен підхід, що поєднує одночасно: семантичну близькість до референсного матеріалу (бо повний відрив від теми неприйнятний), покриття предметної онтології (бо ключові поняття мають бути розкриті), внутрішню когерентність (бо логічна послідовність викладу впливає на засвоєння), фактологічну узгодженість тверджень (бо помилкові факти у навчанні шкідливі), а також валідується через людську оцінку з явним педагогічним критерієм дидактичної цінності. Жоден з раніше розглянутих підходів не охоплює всі ці виміри одночасно. Це обґрунтовує побудову нової композитної методики, синтезованої з елементів кожного з трьох поколінь автоматичних метрик і доповненої педагогічною рубрикою, що є предметом подальшого викладу.

1.6 Висновки

Сучасні великі мовні моделі досягли рівня якості, на якому стало можливим використання їх як інструменту для генерації навчального контенту. Це створює потребу в методах оцінювання якості згенерованого тексту, придатних саме для освітнього домену. Аналіз існуючого стану галузі дозволяє сформулювати кілька ключових узагальнень.

Класичні автоматичні метрики оцінювання генеративних текстів (BLEU, ROUGE) реагують на лексичну схожість з еталоном і пропускають семантичні парафрази, через що мають обмежену застосовність у домені з відсутністю канонічних еталонів. Метрики другого покоління, побудовані на векторних представленнях (BERTScore, BLEURT, BARTScore), краще вловлюють семантику, але не оцінюють покриття предметної онтології, дидактичну цінність і фактологічну узгодженість. Метрики третього покоління на основі оцінювання мовною моделлю-суддею (G-Eval і подібні) дають гнучкість оцінюваних критеріїв, але мають специфічні обмеження — упередження самооцінки, упередження позиції та довжини, — які потребують методологічних запобіжників.

Людська оцінка зберігає роль базової істини у валідизації автоматичних метрик. Стандартом її проведення є рубрика з декількох критеріїв, залучення щонайменше двох незалежних оцінювачів і розрахунок інтеррейтерної узгодженості за коефіцієнтом Cohen's κ із квадратичним зважуванням для ординальних шкал. Інтерпретація κ за Landis та Koch встановлює, що значення вище 0.40 (категорія помірної узгодженості) є необхідною умовою надійності рубрики.

Навчальний текст відрізняється від інших видів генеративного виходу низкою властивостей: відсутністю канонічного еталона, важливістю покриття предметних понять, наявністю дидактичного виміру, прив'язкою до цільової аудиторії. Жоден з підходів, розглянутих у наявній літературі, не охоплює усі ці виміри одночасно. Це обґрунтовує потребу в композитній методиці, що поєднує: семантичну близькість до

референсу, онтологічне покриття ключових понять, оцінку внутрішньої когерентності, фактологічну верифікацію через крос-модельне оцінювання мовною моделлю-суддею, а також рубрику людської оцінки з педагогічним критерієм. Формальна побудова такої методики становить зміст наступного викладу.

Розділ 2. Методика оцінювання якості згенерованого навчального контенту

2.1 Загальна структура методики

Запропонована у роботі методика призначена для кількісного оцінювання якості навчального тексту, згенерованого великими мовними моделями. На відміну від класичних автоматичних метрик оцінювання генеративних текстів — BLEU, ROUGE, BERTScore, BLEURT, — які побудовані як міра лексичної або семантичної схожості з еталонним перекладом чи рефератом, навчальний контент має додаткові виміри якості, нерелевантні для машинного перекладу: покриття ключових понять предметної області, дидактичну цінність, фактологічну узгодженість поданого матеріалу. З огляду на це за основу взято підхід, що поєднує чотири різнорідні сигнали якості у єдину композитну оцінку.

Методика складається з трьох взаємопов'язаних компонентів. Першим є формалізований опис генеративного сценарію у вигляді предметно-орієнтованої мови (DSL) на основі YAML. Сценарій декларативно фіксує тему фрагмента, перелік ключових понять, які мають бути розкриті, орієнтовний обсяг тексту у словах, цільову аудиторію та еталонний фрагмент-референс. Такий формат дозволяє автоматизувати запуск пайплайна на множині сценаріїв, відтворити результати незалежним дослідником і легко розширювати корпус новими темами без модифікації коду.

Другим компонентом є композитна метрика Q, що обчислюється для кожного згенерованого фрагмента за формулою (2.1):

$$Q = \alpha * S + \beta * O + \gamma * C + \delta * F$$

де S — семантична близькість до референсу, O — частка покриття ключових понять, C — внутрішня когерентність тексту, F — фактологічна узгодженість, оцінена мовною моделлю-верифікатором. Кожна компонента нормалізована до діапазону [0, 1]. Початкові ваги α=0.30, β=0.30, γ=0.15, δ=0.25 обрано як експертні; фінальні підбираються через grid-search на симплексі з кроком 0.05 для максимізації кореляції з

людською оцінкою. Адитивна форма обрана через простоту інтерпретації та можливість включити чи виключити будь-яку компоненту шляхом зведення ваги до нуля.

Третім компонентом є рубрика людської оцінки, що складається з чотирьох критеріїв за шкалою 1–5: ясності викладу, фактологічної коректності, релевантності темі та дидактичної цінності. Останній критерій є власним розширенням рубрики, оскільки у відкритих джерелах подібний педагогічний компонент в автоматизованих метриках NLG для освіти не зустрічається. Підсумкова людська оцінка фрагмента визначається як середнє арифметичне чотирьох критеріїв; узгодженість між двома незалежними оцінювачами вимірюється квадратично-зваженим коефіцієнтом Cohen's κ.

Корпус для експериментальної перевірки методики формується з 30 генеративних сценаріїв, прив'язаних до офіційного силабусу курсу «Алгоритми і структури даних» факультету інформатики НаУКМА. Для кожного сценарію генерується два фрагменти — моделями Claude Sonnet 4.6 та GPT-5.4, — що утворює корпус з 60 одиниць. Усі фрагменти проходять як автоматичне оцінювання за метрикою Q, так і незалежне оцінювання двома експертами за рубрикою.

Узагальнена архітектура запропонованого пайплайна оцінювання, що поєднує усі описані компоненти, наведена на рис. 2.1.

Рис. 2.1 — Архітектура пайплайна оцінювання якості згенерованого навчального контенту

2.2 DSL опису генеративного сценарію

Першим компонентом запропонованої методики є предметно-орієнтована мова опису генеративного сценарію, реалізована у форматі YAML. Сценарій є самодостатнім описом одиниці експерименту і містить інформацію, необхідну й достатню для виконання двох операцій: запуску генерації фрагмента мовною моделлю та подальшого автоматичного оцінювання згенерованого тексту за всіма чотирма компонентами композитної метрики. Така концепція дозволяє відокремити дані експерименту від програмного коду, що значно спрощує розширення корпусу сценаріїв, версіонування і обмін експериментальними матеріалами між дослідниками.

Структура одного сценарію включає набір обов'язкових і опційних полів. До обов'язкових належать: ідентифікатор сценарію (поле id), формулювання теми

природною мовою (topic), опис цільової аудиторії (audience), перелік ключових понять, які мають бути розкриті у фрагменті (concepts), діапазон допустимої довжини у словах (word_count), інструкція для мовної моделі-генератора (prompt) та еталонний фрагмент-зразок (reference), що використовується як опорний матеріал для обчислення семантичної близькості. Опційними є поля style (стилістичний регістр викладу, наприклад «пояснювальний» або «академічний»), prerequisites (попередньо засвоєні поняття, потрібні для повного розуміння змісту фрагмента), transfer_domain (споріднена предметна область, на яку може бути перенесена методика для перевірки універсальності).

Приклад опису сценарію на тему «Поліморфізм у Java» наведено на рис. 2.2.

```yaml
id: java_polymorphism
topic: "Поліморфізм у Java"
audience: "студенти 1 курсу спеціальності «Інженерія програмного забезпечення», 2 семестр"
concepts:
  - поліморфізм
  - динамічне зв'язування
  - перевизначення методу
  - перевантаження методу
  - upcasting
  - downcasting
  - virtual method invocation
constraints:
  length_words: [340, 550]
  style: pedagogical
  language: uk
examples:
  - "Чим перевизначення методу (overriding) відрізняється від перевантаження (overloading)?"
  - "У який момент JVM визначає, який саме метод викликати при поліморфному виклику?"
```

Рис. 2.2 — Приклад YAML-сценарію для теми «Поліморфізм у Java»

Декларативний YAML обрано за основу формату з кількох причин. По-перше, відтворюваність: сценарій є набором даних, а не виконавчим артефактом, тому його легко зберігати у системі контролю версій і ділитися ним з іншими дослідниками без побічних ефектів. По-друге, відокремлення даних від реалізації: програмний пайплайн читає YAML за єдиним інтерфейсом, не маючи прив'язки до конкретного домену; нові теми додаються без модифікації коду. По-третє, читабельність для людини: на відміну від JSON або інших форматів серіалізації, YAML дозволяє багаторядкові текстові поля

(зокрема для prompt і reference) природним чином, що особливо важливо для навчальних фрагментів у кілька абзаців.

Корпус для експериментальної перевірки складається з 30 YAML-сценаріїв, прив'язаних до 15 лекцій офіційного силабусу курсу «Алгоритми і структури даних» факультету інформатики НаУКМА (по два сценарії на лекцію). Теми включають: успадкування, інкапсуляцію, поліморфізм, інтерфейси, абстрактні класи, дженерики, винятки, лямбда-вирази, потоки введення-виведення, багатопоточність, серіалізацію, рефлексію, JUnit-тестування, JVM memory model, Collections Framework і подібні. Списки concepts підбиралися експертно з огляду на матеріал відповідної лекції та підручника курсу.

2.3 Композитна метрика Q та її компоненти

Композитна метрика Q побудована як адитивна комбінація чотирьох незалежних компонент за формулою (2.1). Аргументація на користь саме такої форми ґрунтується на кількох міркуваннях. Адитивна композиція забезпечує інтерпретованість: кожна компонента вносить пропорційний внесок, що дозволяє при потребі проаналізувати, який саме сигнал відповідає за зміну підсумкової оцінки. Така структура є модульною: будь-яку компоненту можна тимчасово вилучити з обчислень, обнуливши відповідну вагу, без втрати працездатності решти метрики. Адитивна форма також є стандартом у літературі з оцінювання генеративних текстів (зокрема, у композитних метриках типу SacreBLEU + BERTScore та подібних). Розглянуті альтернативи — мультиплікативна форма, де нульове значення хоча б однієї компоненти обнуляє підсумок, та зважена медіана — відкинуто через жорсткість поведінки на граничних значеннях і складність інтерпретації проміжних результатів.

Кожну з чотирьох компонент розглянуто докладніше.

Компонента S — семантична близькість до референсу — оцінює, наскільки згенерований фрагмент змістовно подібний до еталонного навчального тексту, що

зберігається в полі reference сценарію. Обчислення проводиться як косинусна близькість між векторними представленнями двох текстів. Для побудови векторних представлень використано мультимовну модель paraphrase-multilingual-MiniLM-L12-v2 [23], яка підтримує українську мову, має близько 118 мільйонів параметрів і добре зарекомендувала себе на задачах оцінювання семантичної подібності. Розглядалися альтернативи LaBSE та mUSE, проте обрано MiniLM через значно вищу обчислювальну ефективність при практично еквівалентній точності для предметно вузьких доменів. Значення компоненти S нормалізується у діапазон [0, 1]: 0 відповідає повному семантичному відхиленню, 1 — практично повному збігу змісту.

```
def _normalize(text: str) -> str:  2+ usages
    text = unicodedata.normalize( form: "NFKC", text).lower()
    # Прибираємо Markdown-маркери (`код`, **жирне**, *курсив*), щоб не заважали substring-матчингу.
    text = re.sub( pattern: r"[`*]",  repl: "", text)
    return re.sub( pattern: r"[\s\-_]+",  repl: " ", text).strip()
```

Рис. 2.3 — Фрагмент реалізації компоненти O (покриття ключових понять)

Компонента O — покриття ключових понять, або онтологічна компонента — вимірює частку понять зі списку concepts, заданого у сценарії, що фактично зустрічаються у згенерованому фрагменті. Цей сигнал відповідає на питання, чи розкрив фрагмент усі ключові поняття, які експерт-методист визначив як обов'язкові для розуміння теми. Обчислення реалізовано як підрядковий пошук після нормалізації обох рядків (приведення до нижнього регістру, видалення розмітки Markdown, очищення пунктуації). Морфологічні обмеження української мови потребують спрощення концептів у YAML до базової форми — наприклад, у списку понять записується «toString», а не «методи toString», бо у різних текстах це поняття може з'являтися як «toString», «методу toString», «метод toString» тощо. Таке спрощення прийнято свідомо як компроміс між точністю покриття і незалежністю від зовнішніх морфологічних аналізаторів. Значення O належить діапазону [0, 1] і дорівнює частці знайдених понять від загальної кількості заявлених.

Слід чесно зазначити обмеження поточної реалізації онтологічної компоненти. У строгому сенсі онтологічна перевірка передбачає роботу зі структурованою знаннєвою базою (наприклад, OWL-онтологією предметної області), де поняття мають визначені зв'язки і ієрархію. У запропонованій методиці компонента O реалізована як простий лексично-онтологічний пошук на основі списків ключових слів з нормалізацією, що є базовим рівнем перевірки покриття. Назву «онтологічна» збережено через концептуальну роль компоненти у методиці — оцінка повноти представлення предметної онтології теми, — але реалізація залишається лексичною. Підвищення складності цієї компоненти через додавання лематизації (наприклад, з використанням бібліотеки pymorphy3 для української мови), синонімічних словників або семантичної близькості за wordnet-подібними структурами розглядається як один з основних напрямів подальшого розвитку методики.

Компонента C — внутрішня когерентність — оцінює, наскільки фрагмент є логічно зв'язним і присвяченим заявленій темі. У роботі застосовано комбінований підхід, що поєднує локальний і тематичний виміри зв'язності. Локальний потік (local flow) обчислюється як середня косинусна близькість між векторними представленнями кожної пари сусідніх речень фрагмента: якщо речення плавно переходять одне в одне, ця величина висока. Тематична прив'язка (topic anchoring) обчислюється як середня близькість кожного речення фрагмента до векторного представлення поля topic сценарію: якщо фрагмент не відходить від теми, кожне речення близьке до теми, і ця величина висока. Підсумкова компонента C визначається за формулою (2.2):

$$C = 0.5 * local\_flow + 0.5 * topic\_anchoring$$

Рівні ваги 0.5 обрано на підставі попередніх експериментів; чутливість підсумкової кореляції до зсуву цих ваг є низькою (підтверджено емпірично далі в роботі). Концептуальною основою такого підходу є робота Barzilay та Lapata [16], що розглядає когерентність як композицію локальних переходів між сусідніми реченнями. Адаптація до коротких навчальних фрагментів додає тематичний контроль, що особливо

важливий для академічного домену, де відхилення від теми навіть на одне-два речення можуть знижувати дидактичну цінність.

Компонента F — фактологічна узгодженість — оцінює, наскільки твердження у фрагменті є фактологічно коректними. Реалізація використовує підхід оцінювання мовною моделлю-суддею: текст фрагмента передається моделі-верифікатору з інструкцією виокремити з нього конкретні фактологічні твердження і класифікувати кожне як коректне, сумнівне або помилкове. Підсумкова оцінка обчислюється як зважена частка коректних тверджень за формулою (2.3):

$$\frac{\sum w_i}{N}$$

де w_i = 1.0 для коректного, 0.5 для сумнівного, 0.0 для помилкового твердження, N — загальна кількість тверджень]

Принциповим методологічним рішенням є застосування крос-модельної верифікації для уникнення упередженості самооцінки. Текст, згенерований моделлю Claude, передається на верифікацію GPT, а текст, згенерований GPT, — моделі Claude. Таке розділення усуває ситуацію, коли модель оцінює власні твердження, що, як показано Panickssery, Bowman та Feng [22], призводить до систематичного завищення оцінки. Парсер відповіді верифікатора реалізовано стійким до варіацій форматування: він коректно обробляє відповіді, обгорнуті у Markdown-блок, відповіді у формі об'єкта з масивом усередині, а також обрізані відповіді — у останньому випадку зберігаються всі повністю закриті об'єкти до точки обриву. Деталі реалізації парсера наведено далі в роботі при обговоренні програмного забезпечення.

2.4 Рубрика людської оцінки

Рубрика людської оцінки є третім основним компонентом методики і виконує роль базової істини, з якою порівнюється автоматична метрика Q. Структурно рубрика являє собою протокол оцінювання, що містить чотири незалежні критерії, кожен з яких

оцінюється за дискретною шкалою від 1 до 5 балів. Шкала 1–5 обрана з огляду на когнітивну обмеженість людини при оцінюванні: ширші шкали (наприклад, 1–10 або 1–100) теоретично дозволяють точнішу диференціацію, але на практиці призводять до значних розходжень між оцінювачами через відсутність спільної інтерпретації проміжних значень.

Чотири критерії рубрики такі.

Перший критерій — ясність викладу (Clarity) — оцінює, наскільки фрагмент є зрозумілим для зазначеної у сценарії цільової аудиторії. Враховується логічність побудови речень, відсутність надлишкової складності, плавність переходів між думками. Оцінка 5 виставляється, якщо текст повністю зрозумілий з першого прочитання; оцінка 1 — якщо текст важко зрозуміти навіть після кількох перечитувань.

Другий критерій — фактологічна коректність (Correctness) — оцінює відсутність помилкових тверджень, точність визначень і прикладів. Оцінювач звертається до власних знань предметної області, а у разі сумнівів — до еталонного фрагмента у полі reference сценарію. Оцінка 5 виставляється, якщо помилок не виявлено; оцінка 1 — якщо фрагмент фактологічно неприйнятний.

Третій критерій — релевантність темі (Relevance) — оцінює, наскільки фрагмент відповідає темі сценарію і чи розкриваються заявлені у списку concepts ключові поняття. Цей критерій частково перетинається з автоматичною компонентою O, але оцінюється людиною холістично з урахуванням контексту. Оцінка 5 — фрагмент повністю відповідає темі, усі ключові поняття розкрито; оцінка 1 — фрагмент відходить від теми або ключові поняття не висвітлено.

Четвертий критерій — дидактична цінність (Didactic value) — є власним розширенням рубрики, що відрізняє запропонований протокол від стандартних рубрик для оцінювання генеративних текстів. Критерій оцінює, чи виконує фрагмент свою основну функцію у навчанні: чи містить ілюстративні приклади, чи рухається виклад від простого до складного, чи поєднуються нові поняття з попередньо засвоєними. Включення цього критерію є важливою методологічною новацією, оскільки у відкритій

літературі з оцінювання генеративних навчальних текстів подібний педагогічний компонент в автоматизованих метриках не зустрічається в готовому вигляді. Введення окремого критерію змушує експерта явно оцінити педагогічну придатність матеріалу, а не лише його загальну якість як тексту.

Підсумкова людська оцінка фрагмента визначається як середнє арифметичне чотирьох критеріїв.

#	Scenario ID	Тема	Ключові поняття	Фрагмент Claude	Clarity (C)	Correctness (C)	Relevance (C)	Didactic value (C)	Mean (C)	Фрагмент GPT	Clarity (G)	Correctness (G)	Relevance (G)	Didactic value (G)	Mean (G)
1	java_abstract_classes	Абстрактні класи у Java	абстрактний клас; абстрактний метод; abstract; спільний стан; конструктор; extends; інтерфейс	java_abstract_classes__claude-sonnet-4-6.md	5	5	4	4	4,5	java_abstract_classes__gpt-5.4.md	5	5	4	4	4,5
2	java_arrays	Масиви в Java	масив; оголошення масиву; індексація масиву; довжина масиву; індексація з кінця; багатовимірні масиви; клас Arrays; ArrayIndexOutOfBoundsException	java_arrays__claude-sonnet-4-6.md	4	5	5	4	4,5	java_arrays__gpt-5.4.md	4	5	5	3	4,25
3	java_bitwise	Побітові операції у Java	побітове І; побітове АБО; виключне АБО; побітове заперечення; зсув вліворуч; зсув праворуч; безпосередній зсув; бітові маски	java_bitwise__claude-sonnet-4-6.md	4	5	4	3	4	java_bitwise__gpt-5.4.md	4	5	4	4	4,25
4	java_collections	Collections Framework у Java	Collections Framework; ієрархія List; ієрархія Set; ієрархія Map; ArrayList; LinkedList; HashMap; TreeMap	java_collections__claude-sonnet-4-6.md	5	5	4	4	4,5	java_collections__gpt-5.4.md	5	5	4	4	4,5
5	java_dates	Робота з датами і часом у сучасній Java	java.time; LocalDate; LocalDateTime; ZonedDateTime; Instant; Duration; Period; незмінність	java_dates__claude-sonnet-4-6.md	4	5	5	4	4,5	java_dates__gpt-5.4.md	4	5	5	4	4,5

Рис. 2.4 — Форма оцінювання експертами у форматі Excel (приклад заповнених рядків)

Процедура застосування рубрики передбачає такі етапи. По-перше, експерт отримує фрагмент разом із описом сценарію (тема, аудиторія, перелік ключових понять). По-друге, фрагмент уважно читається; за потреби — кілька разів. По-третє, експерт виставляє бал за кожним з чотирьох критеріїв окремо, обираючи значення з випадаючого списку 1–5. По-четверте, експерт може додати короткий коментар, де пояснює свої оцінки або фіксує особливості фрагмента (наявність зайвих повторень, цікаві приклади, фактологічні сумніви). Підсумкове значення обчислюється автоматично.

Принциповою вимогою методики є залучення щонайменше двох незалежних експертів. Інтеррейтерна узгодженість оцінюється за квадратично-зваженим коефіцієнтом Cohen's κ за кожним критерієм окремо і за рубрикою в цілому. Перед самостійним оцінюванням обидва експерти проходять процедуру калібровки на спільному наборі з трьох-п'яти пілотних фрагментів, де обговорюються розходження в інтерпретації критеріїв.

2.5 Підбір ваг методом grid-search

Композитна метрика Q має чотири вагових коефіцієнти α, β, γ, δ, які визначають внесок кожної з компонент у підсумкову оцінку. Початкові експертні значення (α = 0.30, β = 0.30, γ = 0.15, δ = 0.25) сформульовано на етапі формалізації методики і відображають інтуїтивні припущення щодо відносної важливості кожного сигналу. Однак ці значення не обов'язково забезпечують найкращу узгодженість автоматичної метрики з людською оцінкою на конкретному корпусі. Тому фінальні значення ваг визначаються емпіричним підбором.

Процедура підбору реалізована методом grid-search на симплексі допустимих ваг — тобто множині кортежів (α, β, γ, δ) з невід'ємними значеннями, сума яких дорівнює одиниці. Перебираються всі точки симплексу з кроком 0.05; це дає приблизно 1771 комбінацію, що повністю обчислюється на сучасному ноутбуку за кілька секунд. Для кожної комбінації обчислюється значення Q для всіх 60 фрагментів корпусу і потім — коефіцієнт кореляції Пірсона r між отриманими значеннями Q і середньою людською оцінкою тих самих фрагментів. Підсумковим результатом grid-search є кортеж ваг, при якому r досягає максимуму.

Цільовою функцією обрано саме коефіцієнт Пірсона з кількох причин. По-перше, цей коефіцієнт є стандартом у літературі з оцінювання генеративних метрик і дозволяє порівняти отриманий результат із результатами інших робіт. По-друге, він є інтерпретованим: значення r = 0.7 означає, що зміни автоматичної метрики на 70% лінійно передбачають зміни людської оцінки. По-третє, при оптимізації лінійної комбінації компонент саме лінійна кореляція є природною цільовою функцією.

Для контролю надійності результату додатково обчислюється коефіцієнт Спірмена ρ для тих самих ваг. Спірменів ρ є ранговою кореляцією і менш чутливий до окремих викидів та нелінійних залежностей між метрикою і людською оцінкою. Помітні відмінності між значеннями Пірсона і Спірмена сигналізують про специфічну форму залежності — наприклад, монотонну, але нелінійну, — і потребують додаткового аналізу.

Принциповим обмеженням процедури grid-search є залежність оптимальних ваг від репрезентативності корпусу, на якому проводиться підбір. Ваги, оптимальні для домену базових тем програмування на Java, не обов'язково будуть оптимальними для іншого предметного домену (наприклад, юриспруденції або медицини). Для майбутнього перенесення методики на нові домени слід проводити повторний підбір ваг на репрезентативному корпусі того ж домену. Альтернативою повного перевчання є передавання ваг із спорідненого домену з подальшим тонким налаштуванням, що становить окремий напрям подальших досліджень.

Окремо потрібно врахувати, що при низькій варіативності або високому ефекті насичення людської оцінки grid-search може давати результати, чутливі до шуму: невеликі зміни складу корпусу істотно змінюють оптимальні ваги. У такому випадку отримані значення слід інтерпретувати з обережністю, а додатковим контрольним показником є порівняння кореляції оптимального Q з кореляцією Q при рівних вагах ($\alpha = \beta = \gamma = \delta = 0.25$). Якщо різниця між цими двома значеннями невелика, це свідчить про обмежений вплив підбору ваг на якість метрики у даному корпусі.

2.6 Можливості для вдосконалення

Запропонована методика становить базис для подальшого розвитку у кількох напрямах. Деякі з цих напрямів випливають із природних обмежень поточної реалізації, інші — з результатів експериментальної перевірки, наведеної далі в роботі. Узагальнено можна виділити чотири ключові вектори вдосконалення.

Першим напрямом є розширення предметних областей застосування методики. Поточна реалізація прив'язана до домену навчальної інформатики, конкретніше — до тем курсу «Алгоритми і структури даних» факультету інформатики НаУКМА. Перенесення методики на інші освітні домени — правознавство, медицину, історію, фізику — потребує адаптації як корпусу генеративних сценаріїв, так і еталонних понять у компоненті покриття. Особливо цікавим є застосування методики у доменах з більшою

варіативністю якості генерації, де сучасні великі мовні моделі ще не досягли ефекту насичення оцінки. Прикладом такого домену є українське правознавство, де модель може помилятися у конкретних нормах національного права, що потенційно дасть нижчі і відмінні значення F-компоненти.

Другим напрямом є диференціація рівня цільової аудиторії. Поточна рубрика оцінює дидактичну цінність як єдиний холістичний критерій. Подальшою деталізацією є введення диференційованих рубрик для різних рівнів аудиторії: студента-першокурсника без попередньої підготовки, студента випускного курсу зі значною теоретичною базою, аспіранта або молодого науковця. Кожен з рівнів передбачає різні очікування щодо складності викладу, обсягу прикладів, рівня формалізації. Така диференціація дозволила б точніше характеризувати придатність згенерованого матеріалу до конкретного освітнього сценарію і потенційно дозволила б автоматично оцінювати придатність одного й того самого фрагмента для різних освітніх контекстів.

Третім напрямом є розширення компоненти фактологічної узгодженості. Поточна реалізація використовує один-єдиний промпт для верифікації всіх типів фактологічних тверджень у фрагменті. Подальшим напрямом є градуйована верифікація — поділ тверджень на класи (визначення, твердження про властивості, твердження про взаємозв'язки, твердження про числові значення або синтаксичні факти) з застосуванням спеціалізованих промптів або навіть різних моделей-суддів для кожного класу. Така диференціація потенційно підвищить надійність оцінки за рахунок використання моделей, краще пристосованих до конкретного типу перевірки. Окремо доцільно дослідити можливість залучення зовнішніх джерел знань (наприклад, документації Java, державних реєстрів, медичних рекомендацій) як додаткового сигналу для верифікації.

Четвертим напрямом, найбільш специфічним для домену навчальної інформатики, є інтеграція детектора прикладів коду. Як показала експериментальна перевірка, дидактична цінність фрагмента сильно залежить від наявності ілюстративного коду мовою програмування. Поточна методика оцінює це опосередковано через людський критерій дидактичної цінності та через семантичну близькість до референсу.

Подальшим напрямом є додавання окремого автоматичного детектора, що: по-перше, перевіряє наявність блоків коду у фрагменті; по-друге, синтаксично валідує цей код; по-третє, оцінює доречність прикладу для пояснення відповідного поняття. Такий модуль міг би стати п'ятою компонентою композитної метрики, специфічною для технічного навчального контенту.

Окремо можна назвати потенційні напрями розвитку самої композитної форми метрики Q. Поточна адитивна композиція є простою і інтерпретованою, але припускає лінійну незалежність компонент. На практиці компоненти можуть бути нелінійно пов'язані; наприклад, висока семантична близькість при низькому покритті ключових понять може свідчити про надто загальний фрагмент і отримувати завищену оцінку у адитивній моделі. Перспективним є дослідження альтернативних композицій — мультиплікативних, у вигляді логістичної регресії, з нелінійними доданками, — та порівняння їхньої точності з лінійною моделлю на ширших корпусах. Такі дослідження виходять за межі поточної магістерської роботи і становлять перспективу для подальших публікацій.

Розділ 3. Експериментальна перевірка методики

3.1 Програмна реалізація методики

Запропонована методика реалізована у вигляді програмного пакета content_eval мовою Python 3.11. Архітектура пакета є модульною: кожна компонента методики реалізована як окремий модуль із чітко визначеним інтерфейсом. Загальний обсяг коду становить близько 1300 рядків (з яких приблизно 1190 — власне реалізація, а решта — однорядкові коментарі). Така компактність є наслідком прийнятого принципу простоти: кожна компонента використовує мінімально необхідний набір бібліотек, без надмірних абстракцій або шарів обгорток.

Структура пакета така. Модуль dsl.py відповідає за завантаження YAML-сценаріїв з папки на диску і їх валідизацію — перевірку наявності обов'язкових полів і коректності типів. Модуль generator.py містить тонку обгортку над бібліотекою litellm, що забезпечує уніфікований доступ до різних провайдерів мовних моделей (OpenAI, Anthropic, Ollama для локальних моделей) через єдиний інтерфейс. Підкаталог components/ містить чотири модулі — semantic.py, ontology.py, coherence.py, factuality.py — по одному на кожну компоненту композитної метрики Q. Модуль metric.py агрегує значення компонент у підсумкову оцінку Q. Модуль pipeline.py об'єднує всі етапи в один пайплайн: завантаження сценарію, генерація фрагмента, обчислення компонент, збереження результату.

Бібліотечні залежності проєкту мінімальні. Для роботи з векторними представленнями використано sentence-transformers (модель paraphrase-multilingual-MiniLM-L12-v2). Для виклику зовнішніх мовних моделей — litellm 1.x, що дозволяє стандартизовано працювати з OpenAI, Anthropic та локальними Ollama-моделями через один інтерфейс. Для збереження проміжних і фінальних результатів — стандартний модуль csv, для роботи з YAML — pyyaml, для статистичного аналізу — scipy.stats (коефіцієнти Пірсона, Спірмена) і scikit-learn (квадратично-зважена версія Cohen's κ).

Для генерації Excel-файлів з формою людської оцінки та підсумкових таблиць — openpyxl.

Тестове покриття реалізоване через стандартний pytest. Окремі модулі (dsl, ontology, статистичні функції) мають unit-тести; sanity-перевірки на одному сценарії запускаються командою python examples/sanity_check.py. Повний прогін на корпусі з 30 сценаріїв виконується скриптом python examples/run_first_batch.py і займає від 5 до 15 хвилин залежно від продуктивності мовної моделі та якості мережевого з'єднання. Окремий скрипт examples/run_cross_model_f.py запускає крос-модельну верифікацію F-компоненти для всього корпусу: фрагменти Claude перевіряються GPT, фрагменти GPT — Claude. Окремий аналітичний скрипт examples/eval_analysis.py обчислює інтеррейтерну узгодженість, кореляції автоматичної метрики з людською оцінкою та виконує grid-search оптимальних ваг.

Серед практичних інженерних рішень, що довелось приймати під час реалізації, варто згадати декілька. Перший повний прогін крос-модельної верифікації F-компоненти дав підозрілий результат: рівно половина значень F для фрагментів Claude дорівнювала точно 0.500. Аналіз показав, що це не реальна оцінка фактологічної узгодженості, а артефакт відмови парсера: при обмеженні max_tokens = 800 модель-верифікатор GPT, опрацьовуючи довгі Claude-фрагменти з 15–25 фактологічними твердженнями, видавала JSON-відповідь, обрізану на середині, парсер не зміг її розпізнати і повертав нейтральне значення 0.5 як значення за замовчуванням при невдалому розборі. Виправлення полягало у двох змінах: підняття max_tokens до 3000 у конфігурації верифікатора і додавання режиму recovery-парсингу, що при обрізаній відповіді все одно зберігає усі повністю закриті JSON-об'єкти до точки обриву. Після цієї правки повторний прогін дав реалістичний розподіл F-значень у діапазоні 0.81–1.00. Цей епізод відображає типову інженерну реальність — навіть прості допоміжні компоненти потребують уваги до граничних випадків формату вихідних даних.

Усі проміжні та фінальні результати зберігаються у структурованому вигляді в каталозі data/generated/. Згенеровані текстові фрагменти зберігаються у форматі

Markdown (по одному файлу на пару сценарій-модель). Числові результати — у CSV-файлах окремо для кожної моделі-генератора. Підсумкові таблиці експериментальної перевірки — у єдиному xlsx-файлі eval_results.xlsx, що містить аркуші Cohen kappa, Correlation, Per fragment і Outliers, придатні для безпосереднього включення у додаток роботи.

З метою забезпечення відтворюваності експерименту усі програмні і експериментальні параметри фіксуються явно. Програмне середовище: Python 3.11 на macOS, бібліотеки sentence-transformers 2.x (модель paraphrase-multilingual-MiniLM-L12-v2), litellm 1.x, scipy 1.15, scikit-learn 1.7, openpyxl 3.1, pyyaml. Доступ до мовних моделей здійснюється через офіційні API провайдерів: Anthropic (модель claude-sonnet-4-6) і OpenAI (модель gpt-5.4). Параметри генерації: temperature = 0.7, max_tokens = 1200, єдиний системний промпт для обох моделей, зафіксований у коді generator.py. Параметри крос-модельної верифікації F-компоненти: temperature = 0 (детермінований режим), max_tokens = 3000, інструкція з вимогою повернути JSON-масив з полями claim і status. Уся обчислювальна частина виконується на одному CPU без використання GPU; повний прогін експерименту триває від 5 до 15 хвилин залежно від продуктивності мережевого з'єднання з API. Випадкові семпли не використовуються — джерелом стохастичності у системі є лише температура мовної моделі при генерації.

3.2 Корпус генеративних сценаріїв на базі курсу НаУКМА

Експериментальний корпус формувався з двох міркувань. Перше — отримати достатньо широку вибірку тем для статистично значущого аналізу кореляцій (не менше 30 сценаріїв за рекомендаціями NLG-evaluation літератури). Друге — забезпечити академічну обґрунтованість вибору тем, тобто прив'язати корпус до офіційного навчального матеріалу конкретного університетського курсу. Обидві вимоги задоволено вибором за основу силабусу навчальної дисципліни «Алгоритми і структури даних»

факультету інформатики Національного університету «Києво-Могилянська академія», що читається у 2 семестрі 1 курсу спеціальності «Інженерія програмного забезпечення».

Силабус курсу включає 15 лекцій, що охоплюють такі тематичні блоки: вступ до Java і об'єктно-орієнтоване програмування (успадкування, інкапсуляція, поліморфізм, абстрактні класи, інтерфейси), JVM і робота з пам'яттю, основні API стандартної бібліотеки (Collections Framework, Stream API, потоки введення-виведення, дата і час), функціональні можливості Java (лямбди, Optional, дженерики), розширені теми (багатопоточність, серіалізація, рефлексія, NIO.2), інструментальне забезпечення (JUnit, Maven, Lombok, logging). На кожну лекцію розроблено по два генеративні сценарії, що відрізняються темою або акцентом. Загалом корпус нараховує 30 сценаріїв.

Кожен сценарій збережено в окремому YAML-файлі у каталозі scenarios/algorithms/. Списки ключових понять (поле concepts) для кожного сценарію формувалися експертно на основі матеріалу відповідної лекції та підручника курсу. Орієнтовний обсяг тексту (поле word_count) диференційовано залежно від складності теми: для базових тем (наприклад, успадкування, інкапсуляція) — 220–360 слів, для тем середньої складності (інтерфейси, дженерики) — 280–450 слів, для розширених тем (багатопоточність, NIO.2, рефлексія) — 340–550 слів. Поле audience у всіх сценаріях задає одну й ту саму цільову аудиторію: «студент 1 курсу 2 семестру факультету інформатики». Це забезпечує рівний контекст оцінювання дидактичної цінності.

Для кожного з 30 сценаріїв згенеровано два фрагменти: один моделлю Claude Sonnet 4.6 від Anthropic, інший — моделлю GPT-5.4 від OpenAI. Доступ до моделей реалізовано через офіційні API провайдерів за посередництвом бібліотеки litellm з єдиним системним промптом, температурою 0.7 та обмеженням до 1200 токенів виходу. Загалом корпус нараховує 60 фрагментів навчального тексту. Усі вони збережено в каталогах data/generated/fragments_claude/ та data/generated/fragments_openai/ у форматі Markdown. Сукупна вартість API-викликів для повної генерації корпусу склала приблизно 0.50 доларів США.

Окремої уваги заслуговують методологічні рішення щодо параметрів генерації та формування корпусу, оскільки вони впливають на відтворюваність експерименту і обґрунтованість висновків. По-перше, температуру семплінгу обрано на рівні 0.7, що є компромісом між повністю детермінованим режимом (temperature = 0, який дав би однакові виходи при повторних запусках і не відображав би природну варіативність генерації) і високотемпературним режимом (temperature ≥ 1.0, що породив би непередбачуваний шум, нерелевантний для оцінювання якості). Значення 0.7 є типовим для практичних застосувань LLM у освітньому контексті і відповідає рекомендованому за замовчуванням значенню у документації провайдерів. По-друге, системний промпт для обох моделей повністю однаковий і фіксований у коді — це гарантує, що відмінності між моделями обумовлені виключно архітектурою і даними навчання, а не різницею у формулюванні задачі. По-третє, обмеження max_tokens = 1200 покриває з запасом найдовші фрагменти у корпусі (максимальна довжина — 412 слів ≈ 600–800 токенів), тобто не виступає штучним обмежувачем якості.

По-четверте, еталонні фрагменти у полі reference кожного сценарію сформовано автором роботи на основі матеріалу відповідної лекції курсу і офіційного підручника. Усі reference-тексти створені вручну, без залучення мовних моделей, щоб уникнути циклічної залежності — ситуації, коли генеративна модель порівнюється з текстом, написаним іншою (або тією самою) моделлю. По-п'яте, питання data leakage — тобто можливості, що модель уже бачила частково еталонні тексти на етапі навчання — у предметному домені базового програмування на Java є реальним: матеріали з підручників Java, документація Oracle, освітні ресурси з програмування становлять значну частину публічних навчальних корпусів, на яких тренуються великі мовні моделі. Це частково пояснює дуже високу якість генерації, виявлену в експерименті. Перенесення методики на домени з нижчою репрезентативністю у тренувальних даних (українське правознавство, спеціалізована медицина) дозволило б оцінити поведінку методики в умовах, де leakage малоймовірний; цей напрям зазначено серед можливостей вдосконалення раніше у роботі.

Кількісно усереднені характеристики згенерованого корпусу такі: середня довжина фрагмента — 270 слів, медіана — 268 слів, мінімум — 198 слів, максимум — 412 слів. Розподіл довжин загалом узгоджений із заданими у сценаріях обмеженнями. Усі фрагменти проходять автоматичний контроль на відповідність word_count і за необхідності можуть бути перегенеровані. У поточному корпусі повторні генерації не знадобились.

Приклад одного зі згенерованих фрагментів у фінальному вигляді, відкритого у Markdown-в'юері, наведено на рис. 3.1.

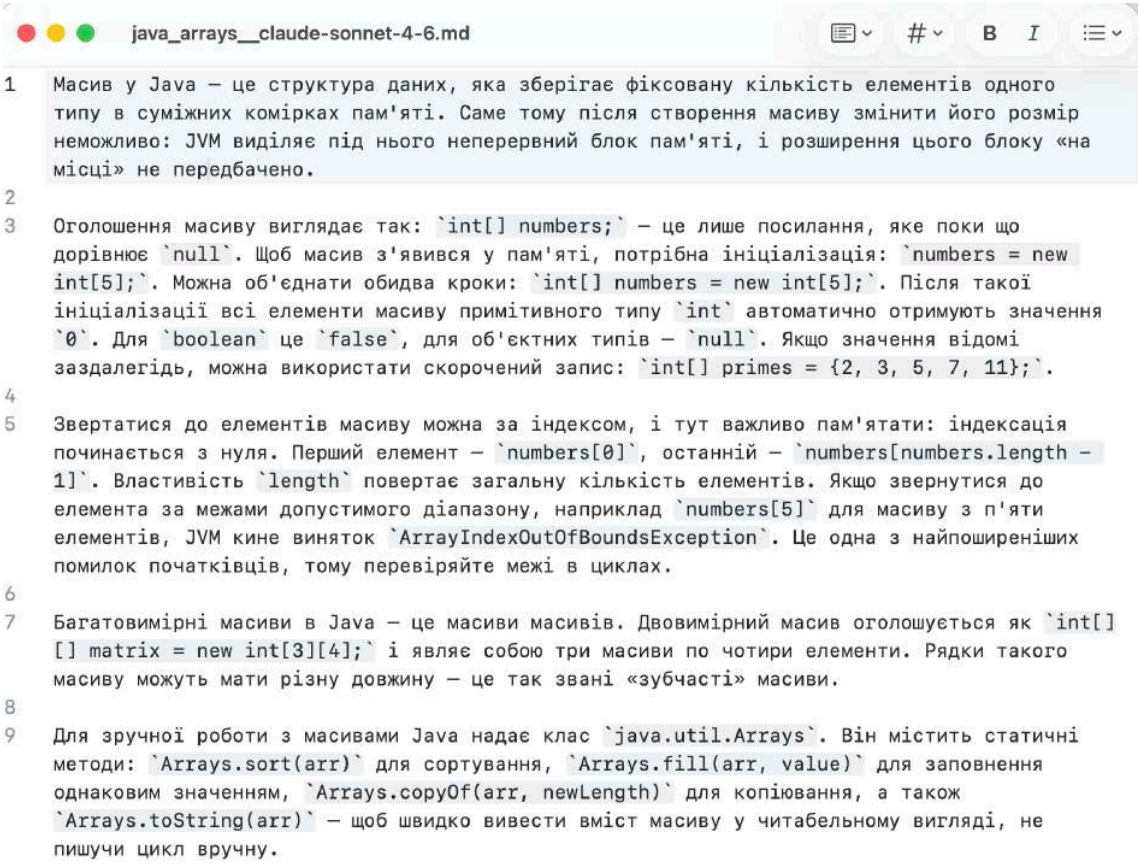

Рис. 3.1 — Згенерований навчальний фрагмент за темою «Масиви у Java» моделлю Claude Sonnet 4.6 у Markdown-в'юері (приклад)

Сумарні витрати на API-виклики для генерації корпусу і подальшої крос-модельної верифікації F-компоненти показано на рис. 3.2 за даними панелей використання провайдерів.

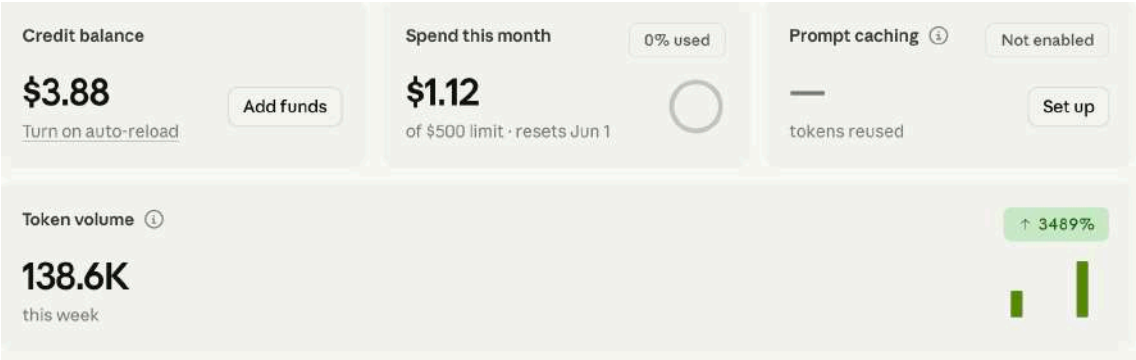

Рис. 3.2 а — Панель використання Anthropic Console: сумарні витрати на API-виклики моделі Claude Sonnet 4.6

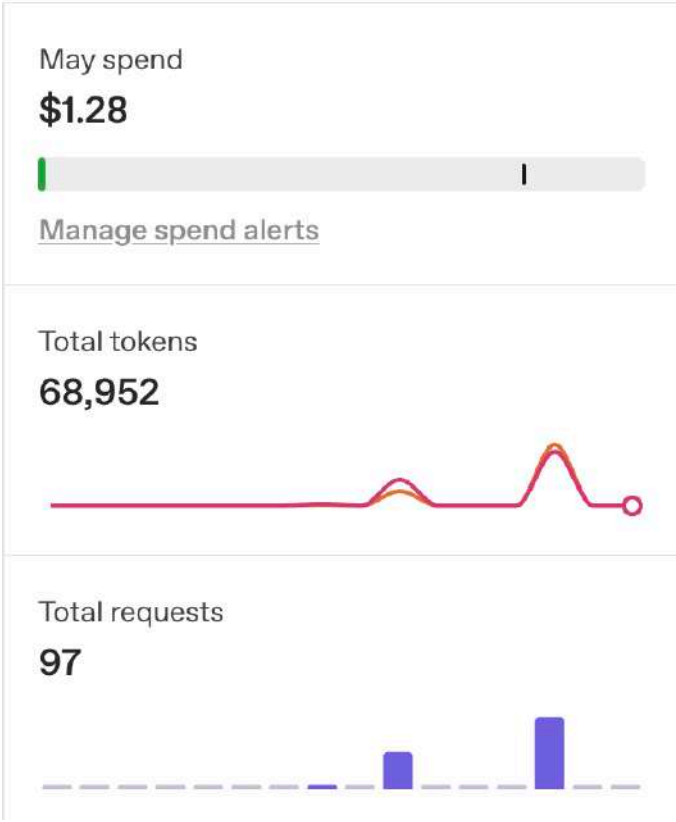

Рис. 3.2 б — Панель використання OpenAI Platform: сумарні витрати на API-виклики моделі GPT-5.4

3.3 Процедура людської оцінки і калібровка оцінювачів

Людська оцінка корпусу проведена двома незалежними експертами. Перший експерт — автор роботи (студент магістерської програми «Комп'ютерні науки», основний практичний досвід — розробка iOS-додатків мовою Swift; мова Java засвоєна на університетських курсах). Другий експерт — старший розробник програмного забезпечення з 15-річним досвідом, основна спеціалізація — розробка macOS-додатків на Swift і Objective-C з достатнім знанням загальних принципів об'єктно-орієнтованого програмування для адекватної оцінки фрагментів навчального матеріалу.

Залучення обох експертів з основним досвідом у Swift-екосистемі є свідомим методологічним рішенням. Конкретні Java-API (зокрема JVM memory model, Maven, Lombok, NIO.2) можуть бути менш знайомими; для таких тем експерт користується еталонним фрагментом reference з YAML-сценарію як орієнтиром. Загальні ж принципи об'єктно-орієнтованого програмування — успадкування, поліморфізм, інтерфейси, дженерики, лямбда-вирази, винятки — є універсальними для всіх ОО-мов, і кваліфіковані Swift / Objective-C розробники адекватно оцінюють фрагменти на ці теми. Таке поєднання експертів дає важливий методологічний результат: показує, що запропонована рубрика з її чотирма критеріями (ясність, фактологічна коректність, релевантність, дидактична цінність) є відтворюваною не лише для вузькоспеціалізованих Java-спеціалістів і працює як універсальний інструмент оцінювання навчального тексту з програмування.

На початку оцінювання два пілотні фрагменти (теми «Абстрактні класи» і «Масиви») було оцінено експертами спільно з обговоренням кожного з чотирьох критеріїв. Цей етап слугував калібровкою: експерти узгодили інтерпретацію критеріїв, особливо для дидактичної цінності, де поняття «достатня кількість прикладів» допускає варіативне трактування. Решта 58 фрагментів була оцінена незалежно. Ця обставина відображена у даних: для перших двох фрагментів оцінки обох експертів практично збігаються, далі починають розходитися у природних межах.

Процедура оцінювання передбачала такі етапи. По-перше, генерація форми оцінювання у вигляді Excel-файлу за допомогою скрипта examples/make_eval_form.py.

Форма містить 30 рядків — по одному на сценарій — з двома фрагментами в кожному рядку (Claude і GPT поруч) та полями для рейтингів за чотирма критеріями для кожного фрагмента. По-друге, кожному експерту надано архів з 60 Markdown-фрагментами для зручного читання у програмі MarkEdit. По-третє, проведено короткий калібрувальний сеанс із спільним обговоренням двох-трьох пілотних фрагментів і узгодженням розуміння критеріїв рубрики. По-четверте, кожен експерт незалежно оцінив усі 60 фрагментів, проставивши бал від 1 до 5 за кожним із чотирьох критеріїв для обох моделей кожного сценарію. По-п'яте, обидві заповнені форми зведено в єдиний аналітичний пайплайн через скрипт examples/eval_analysis.py.

Оцінювання тривало приблизно 2–3 години на одного експерта (близько 3 хвилин на фрагмент). Жоден з фрагментів не виявився настільки нечитабельним або беззмістовним, щоб його неможливо було оцінити. Жодний рейтинговий рядок не залишився порожнім: усі 30 рядків × 8 колонок × 2 експерти = 480 числових значень заповнено.

3.4 Узгодженість між оцінювачами

Інтеррейтерна узгодженість обчислена за квадратично-зваженим коефіцієнтом Cohen's κ окремо для кожного критерію рубрики, окремо для кожної моделі-генератора та сумарно для всіх даних. Числові результати наведено у таблиці 3.1.

Критерій	**Claude Sonnet 4.6**	**GPT-5.4**	**Інтерпретація**
Ясність (Clarity)	0.091	−0.027	незначна / неприйнятна
Коректність (Correctness)	—	—	константа (κ не визначений)
Релевантність (Relevance)	0.173	0.082	незначна
Дидактика (Didactic value)	0.154	0.167	незначна

Усі критерії (4×30 = 120 пар)	0.305	0.247	задовільна
Загалом (4×30×2 = 240 пар)	0.275	0.275	задовільна

Таблиця 3.1 — Інтеррейтерна узгодженість за критеріями рубрики (квадратично-зважений коефіцієнт Cohen's κ)

Графічне порівняння κ за критеріями обох моделей наведено на рис. 3.3.

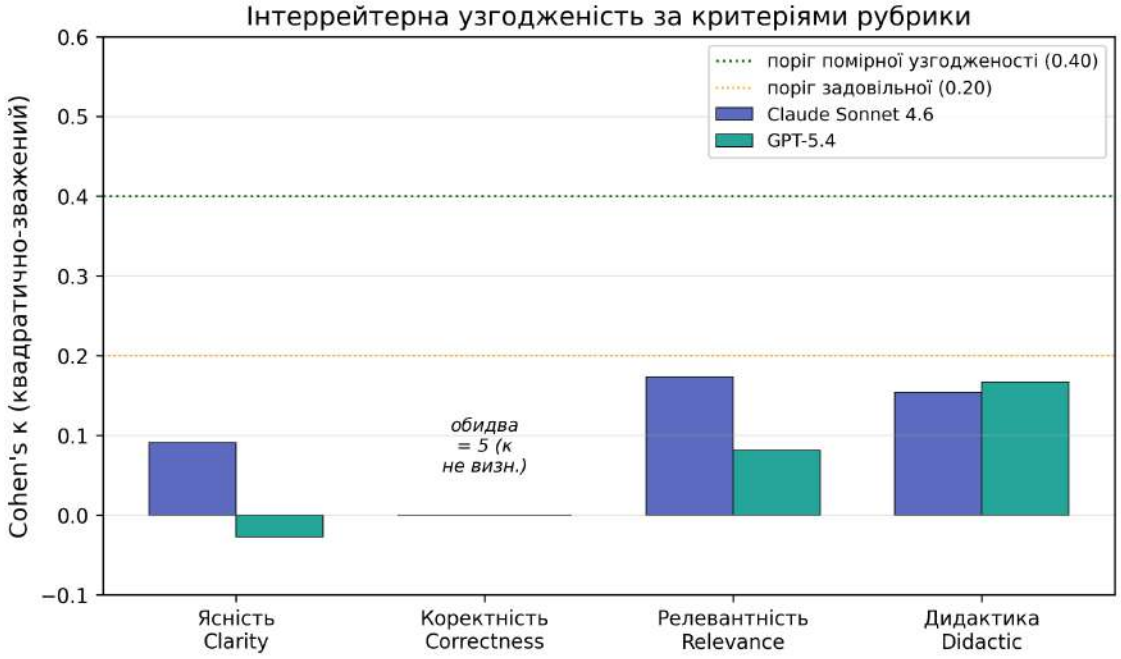

Рис. 3.3 — Інтеррейтерна узгодженість за критеріями рубрики у двох моделей-генераторів

Ключове спостереження: коефіцієнт κ для критерію Correctness не визначений, оскільки обидва експерти послідовно ставили бал 5 за всіма 60 фрагментами без жодного винятку. Це означає повну узгодженість оцінювачів, але саме через нульове стандартне відхилення оцінок класична статистика узгодженості не застосовна (формула κ містить ділення на величину, пропорційну дисперсії оцінок). Цей результат пояснюється фактологічною надійністю сучасних топ-моделей у предметному домені базових тем програмування: жоден з фрагментів не містив фактологічних помилок, які експерти могли б ідентифікувати без поглибленої перевірки конкретних API.

Сумарний показник κ для всіх 240 рейтингових пар становить 0.275. За класифікацією Landis та Koch [20] це належить до категорії задовільної узгодженості (fair). Це менше за поріг помірної узгодженості (κ > 0.40), який зазвичай вважається бажаним для надійної рубрики у NLG-дослідженнях. Однак отриманий результат є цілком прийнятним з огляду на специфіку корпусу: усі фрагменти оцінено переважно високими балами 4–5, що звужує діапазон диференціації і природно знижує κ. Окремо за моделями узгодженість для Claude (0.305) дещо вища, ніж для GPT (0.247) — різниця статистично незначуща, але може свідчити про більш послідовний стиль викладу Claude, що зменшує варіативність експертної інтерпретації.

Серед окремих критеріїв найвища узгодженість досягнута для Relevance і Didactic value (приблизно 0.15–0.17), що пояснюється чітким операціоналізованим характером цих критеріїв (наявність ключових понять, наявність прикладів). Найнижча узгодженість — для Clarity, що відображає різну індивідуальну чутливість оцінювачів до стилістичних особливостей тексту.

3.5 Кореляція автоматичної метрики з людською оцінкою

Кореляція між автоматичною композитною метрикою Q і середньою людською оцінкою обчислена за коефіцієнтами Пірсона r і Спірмена ρ окремо для кожної компоненти і для підсумкового значення Q. Результати наведено у таблиці 3.2.

Компонента	**Pearson r**	**Spearman ρ**	**Сила (Cohen)**
S — семантична близькість	+0.064	+0.027	мізерна
O — покриття понять	−0.021	0.000	мізерна
C — когерентність	−0.097	−0.055	мізерна
F — фактологічна узгодженість	−0.065	−0.032	мізерна
Q (дефолтні ваги)	−0.021	−0.028	мізерна

Q (оптимальні ваги grid-search)	+0.064	+0.027	мізерна

Таблиця 3.2 — Кореляція компонент автоматичної метрики з людською оцінкою (n = 60)

Точкова діаграма залежності людської оцінки від автоматичної метрики Q подана на рис. 3.4.

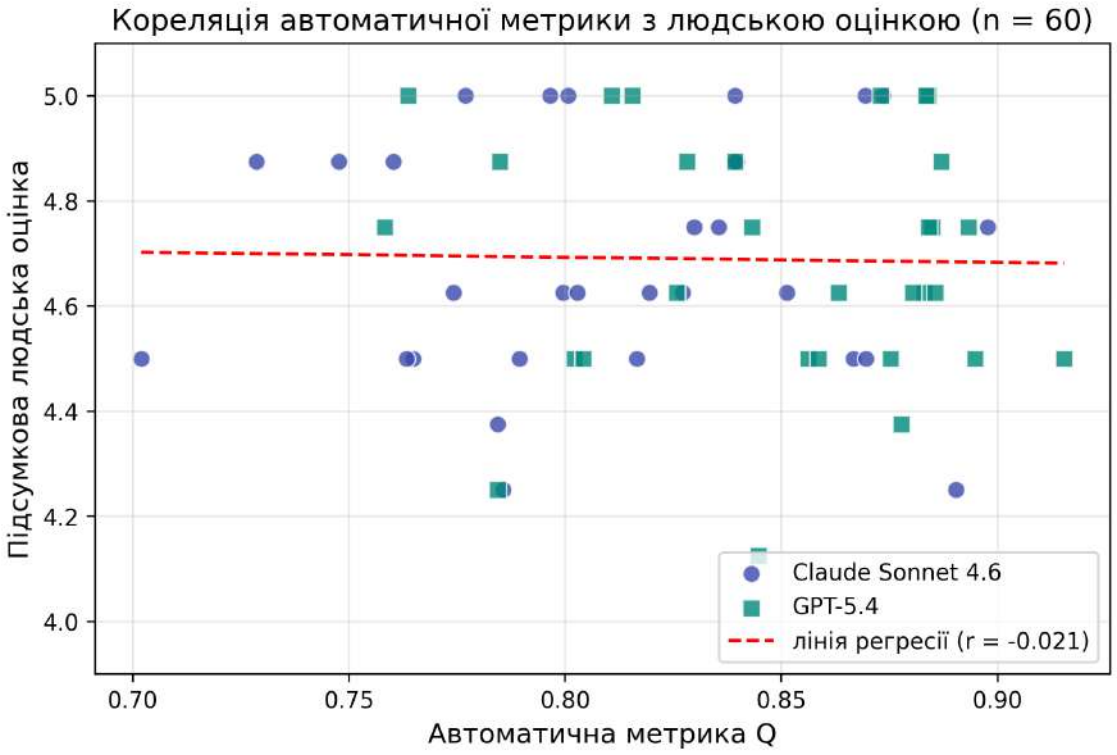

Рис. 3.4 — Кореляція автоматичної метрики Q з людською оцінкою (n = 60)

Жоден коефіцієнт за абсолютним значенням не перевищує 0.1, тобто кореляція мізерна за класифікацією Коена. Цей результат справедливий і для окремих компонент, і для композитного значення Q як з дефолтними експертними вагами, так і з оптимальними вагами, отриманими через grid-search. Grid-search видав оптимальний кортеж ($\alpha = 1.00$, $\beta = 0$, $\gamma = 0$, $\delta = 0$) — тобто лише компонента S вносить ненульовий внесок при максимізації Pearson r — однак навіть при цій конфігурації r = 0.064, що нижче порогу будь-якої практичної значущості.

Результат, на перший погляд, є несприятливим: розроблена методика не показує статистично значущої кореляції з людською оцінкою. Однак при докладному аналізі цей результат отримує змістовну інтерпретацію, що виходить далеко за межі простої констатації невдачі і становить самостійний експериментальний внесок роботи. Виділимо три ключові спостереження:

— з 60 фрагментів 90% отримали підсумкову людську оцінку у діапазоні 4.5–5.0 балів, ще 10% — у діапазоні 4.0–4.5; жоден фрагмент не отримав оцінки нижче 4.0; середнє значення становить 4.69 при стандартному відхиленні 0.23 — це класичний ефект «стелі» (ceiling effect);

— парний t-тест Стьюдента між людськими оцінками фрагментів Claude і GPT за всіма 30 сценаріями дає p = 0.918, аналогічний непараметричний тест Уілкоксона — p = 0.984: моделі Claude Sonnet 4.6 і GPT-5.4 генерують навчальні фрагменти з програмування Java статистично еквівалентної якості;

— автоматична метрика Q, навіть з оптимальними вагами, має діапазон значень 0.715–0.915, тобто розкид у 0.20, тоді як розкид людської оцінки становить лише 0.88 (з 4.12 до 5.00), причому 90% значень припадає на діапазон у 0.5 бала.

Гістограма розподілу людської оцінки на корпусі наведена на рис. 3.5.

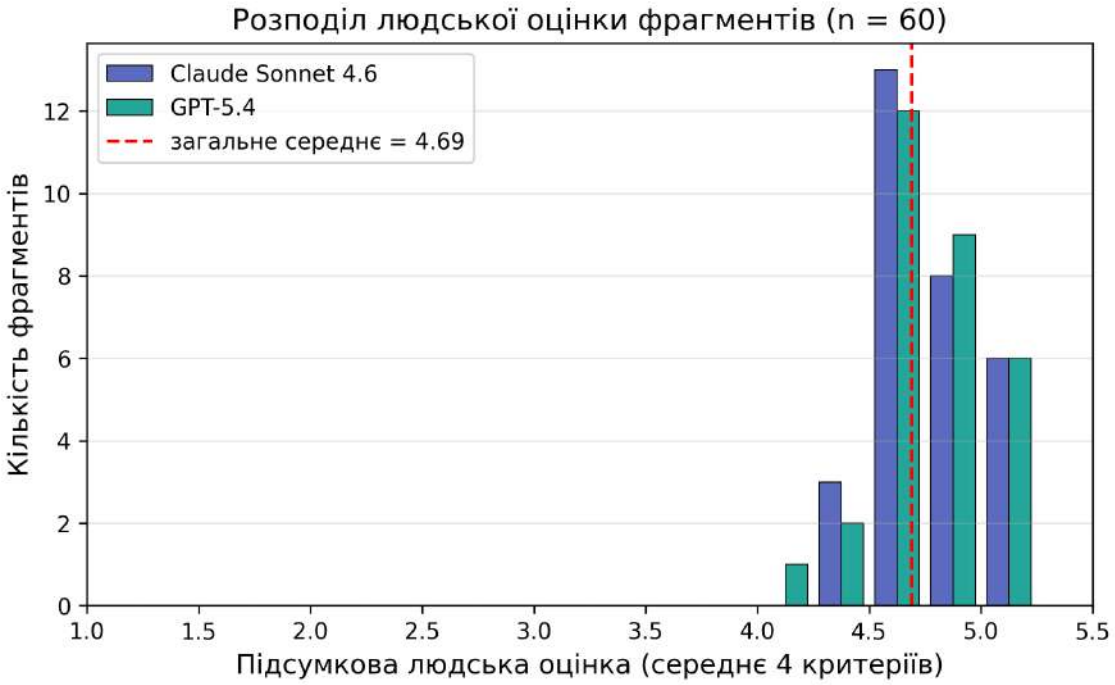

Рис. 3.5 — Розподіл підсумкової людської оцінки фрагментів (n = 60); 90% значень припадає на діапазон 4.5–5.0

Сукупність цих трьох спостережень дає інтерпретацію результату: сучасні топ-моделі досягли рівня якості генерації навчальних текстів з базових тем програмування, на якому ні людська, ні автоматична оцінка не дискримінують між окремими фрагментами. Це — самостійний експериментальний результат, узгоджений з повідомленнями інших дослідників про «вирівнювання» якості топ-моделей на стандартних бенчмарках.

3.6 Аналіз ефекту «стелі» та аномальних фрагментів

Виявлений ефект «стелі» заслуговує окремого аналізу, оскільки він має наслідки як для інтерпретації поточних результатів, так і для дизайну майбутніх досліджень якості згенерованих навчальних текстів.

Ефект «стелі» (ceiling effect) у вимірювальній науці означає ситуацію, коли вимірюваний інструмент не здатний розрізняти об'єкти, що знаходяться поблизу верхньої межі шкали. У контексті оцінювання генеративних текстів цей ефект проявляється у кількох взаємопов'язаних формах: насиченість людської рубрика-оцінки (більшість фрагментів отримує близькі до максимуму бали), низька дисперсія автоматичної метрики (діапазон значень компонентів незначний на тлі загальної високої якості), статистична нерозрізнюваність моделей-генераторів.

Стеля виникла не випадково, а є результатом поєднання кількох чинників:

— характер промпта генерації — у роботі застосовано продуманий промпт, що задає високий стандарт якості: «Ти — викладач інформатики, який пише короткі навчальні фрагменти українською мовою. Пиши ясно й коректно, від простого до складного, з конкретними прикладами або короткими аналогіями там, де це доречно». Менш якісно сформульований промпт міг би дати ширший розкид якості;

— характер предметного домену — базові теми Java та об'єктно-орієнтованого програмування є обсягом навчального матеріалу, на якому сучасні великі мовні моделі мають значний обсяг тренувальних даних. Це training-data heavy зона, де моделі генерують текст рівня досвідченого автора-методиста. Аналогічний експеримент на менш «насиченому» домені — наприклад, на українському правознавстві або вузькоспеціальній медицині — потенційно дав би значно ширший розкид;

— покоління моделей, що використано — Claude Sonnet 4.6 і GPT-5.4 є топ-моделями станом на 2026 рік, з якістю генерації, що наближається до експертної людської. Метрики, валідовані у 2020–2022 роках на моделях типу GPT-3 або донавчених seq2seq, могли б добре диференціювати такі фрагменти; на сучасних топ-моделях вони саттуруються.

Окрім ефекту «стелі» у людській оцінці, варто звернути увагу на асиметрію крос-модельної верифікації F-компоненти. Розподіл F-значень для двох напрямів верифікації наведено на рис. 3.6: GPT як верифікатор знаходить значно більше сумнівних тверджень у Claude-фрагментах, ніж Claude знаходить у GPT-фрагментах. Це може свідчити про методологічні відмінності у внутрішній калібровці цих моделей при виконанні ролі експерта.

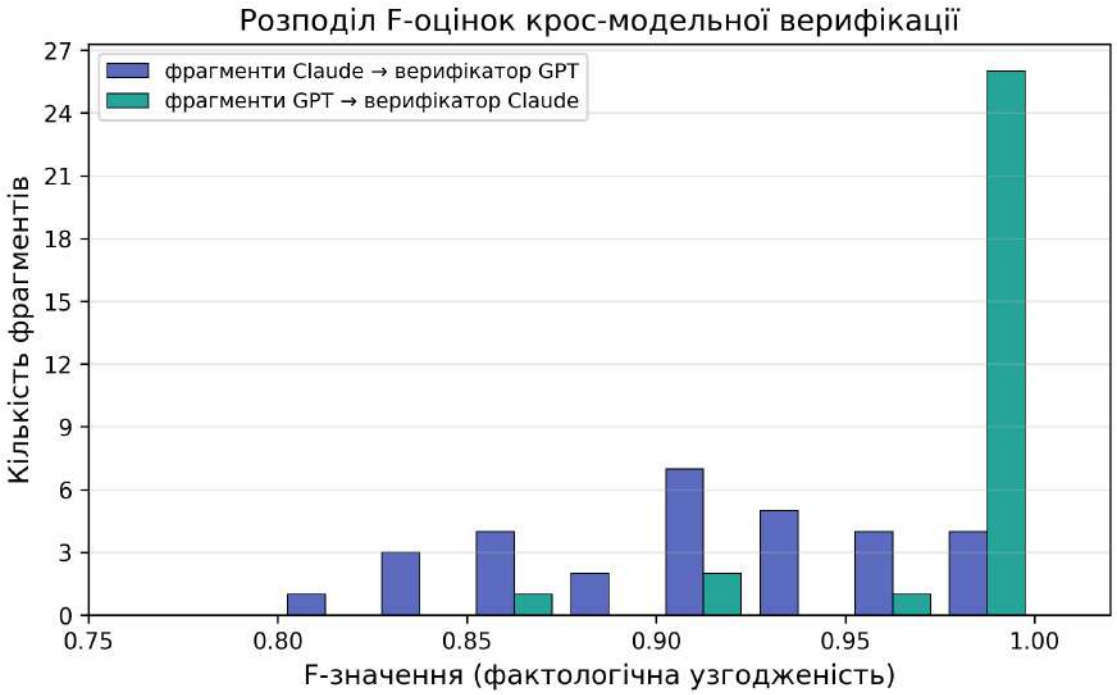

Рис. 3.6 — Розподіл F-оцінок крос-модельної верифікації фактологічної узгодженості

З'явлення ефекту «стелі» при оцінюванні топ-моделей є задокументованим у літературі феноменом. У систематичному огляді Sai та співавторів [17] виділено окремий клас «ceiling-saturated benchmarks» — бенчмарків, на яких моделі останнього покоління досягають практично однакових результатів і подальша диференціація стає неможливою без зміни оцінного інструменту. Запропонована у даному дослідженні методика потрапляє у цю категорію саме на корпусі базових тем програмування; її поведінка на складніших або менш «training-data heavy» доменах залишається відкритим питанням.

Окремий інтерес становить аналіз фрагментів, на яких узгодженість оцінювачів виявилася найнижчою. Серед 60 фрагментів виявлено три з модулем різниці середніх оцінок між експертами не менше 1.0 бала. Перший — фрагмент моделі GPT за сценарієм java_inner_classes (тема «Внутрішні класи у Java»): один експерт виставив середній бал 4.00 (Clarity = 4, Correctness = 5, Relevance = 4, Didactic = 3), другий — 5.00 за всіма критеріями. Розходження пояснюється різним очікуванням щодо рівня деталізації прикладів. Другий і третій — обидва фрагменти за сценарієм java_oo_design (тема «Принципи об'єктно-орієнтованого проєктування») від моделі Claude і моделі GPT відповідно: перший експерт оцінив фрагменти на 5.00, другий — на 4.00 (зі зниженням балів за Clarity та Didactic value). У коментарі другого експерта зазначено, що фрагменти є концептуально повними, але занадто абстрактними і потребували б ілюстрації конкретними прикладами на коді. Це спостереження узгоджується з раніше обговореним напрямом удосконалення методики — додаванням окремого детектора прикладів коду як п'ятої компоненти.

Загалом, аномальні фрагменти не свідчать про дефекти рубрики, а вказують на природні обмеження людської оцінки при складних або абстрактних темах. Низька загальна кількість аномалій (три фрагменти з шістдесяти, тобто 5%) підтверджує загальну надійність процедури оцінювання.

3.7 Висновки

Експериментальна перевірка запропонованої методики на корпусі з 60 згенерованих навчальних фрагментів за темами курсу «Алгоритми і структури даних» НаУКМА дала результат, який потребує змістовної інтерпретації, що виходить за рамки простої констатації числових показників.

З формального боку методика не показала статистично значущої кореляції автоматичної композитної метрики Q з людською оцінкою: коефіцієнт Пірсона r на оптимальних вагах склав 0.064, коефіцієнт Спірмена ρ — 0.027. Аналогічно низькі значення отримано для всіх окремих компонент метрики. Інтеррейтерна узгодженість за критерієм Cohen's κ становить 0.275, що належить до категорії задовільної.

Однак при детальному аналізі цей результат не є негативним у класичному розумінні. Розподіл людської оцінки демонструє яскраво виражений ефект «стелі»: 90% фрагментів отримали оцінки у діапазоні 4.5–5.0, парний t-тест між моделями Claude і GPT не виявив значущих відмінностей ($p = 0.918$). Сукупність цих ознак свідчить, що сучасні топ-моделі (Claude Sonnet 4.6, GPT-5.4) на корпусі базових тем програмування генерують текст якості, на якій ні людська, ні автоматична оцінка не дискримінують між окремими фрагментами. Це — самостійний експериментальний результат, що підтверджує задокументоване у літературі явище «ceiling saturation» автоматичних метрик NLG при тестуванні на топ-моделях.

Запропонована методика з її композитною метрикою Q і рубрикою людської оцінки з педагогічним критерієм зберігає цінність як інструмент для двох сценаріїв застосування. Перший — оцінювання фрагментів, згенерованих менш потужними моделями (наприклад, локальними або компактними варіантами), де очікується ширший розкид якості. Другий — застосування на доменах з більшою фактологічною варіативністю генерації (правознавство, медицина, історія), де поточні топ-моделі ще не

досягли «стелі». Подальші дослідження за цими напрямами становлять природне продовження роботи.

Окремо отримано низку допоміжних емпіричних спостережень. Цикло-машинна верифікація F-компоненти показала, що GPT як верифікатор є суворішим за Claude: F-оцінки фрагментів Claude в інтервалі 0.81–1.00, фрагментів GPT — 0.875–1.00. Це може свідчити про методологічні відмінності у внутрішній калібровці моделей при ролі оцінювача і потребує окремого подальшого дослідження. Аналіз аномальних фрагментів виявив, що головна точка розходження між експертами — оцінка дидактичної цінності концептуально повних, але абстрактних фрагментів без прикладів коду. Це спостереження підтверджує доцільність розширення методики через інтеграцію автоматичного детектора прикладів коду як окремої п'ятої компоненти композитної метрики.

Висновки

У межах магістерської роботи розроблено і експериментально перевірено методику кількісного оцінювання якості згенерованого великими мовними моделями навчального контенту з тем курсу «Алгоритми і структури даних» факультету інформатики НаУКМА.

Запропонована методика поєднує три компоненти: декларативний опис генеративного сценарію у форматі YAML, композитну автоматичну метрику Q з чотирьох компонент (семантичної близькості, покриття ключових понять, внутрішньої когерентності, фактологічної узгодженості) та рубрику людської оцінки з педагогічним критерієм дидактичної цінності. Для усунення упередженості самооцінки у компоненті фактологічної узгодженості застосовано крос-модельну верифікацію двома незалежними великими мовними моделями.

Розроблену методику реалізовано у вигляді програмного пакета content_eval мовою Python з повним покриттям усіх компонент. Корпус для експериментальної перевірки складається з 60 згенерованих навчальних фрагментів — по два на кожен з 30 сценаріїв, прив'язаних до офіційного силабусу курсу НаУКМА. Усі фрагменти проходять як автоматичне оцінювання за метрикою Q, так і людську оцінку двома незалежними експертами за рубрикою.

Експериментальна перевірка дала такі основні результати. Інтеррейтерна узгодженість за квадратично-зваженим коефіцієнтом Cohen's κ становить 0.275, що належить до категорії задовільної. Кореляція автоматичної метрики Q з людською оцінкою виявилася мізерною за абсолютним значенням (коефіцієнт Пірсона r у діапазоні від -0.10 до +0.07 для всіх компонент і їх композиції). Розподіл людської оцінки концентрується у верхній частині шкали: 90% фрагментів отримали оцінки у діапазоні 4.5–5.0 балів з можливих 5. Парний тест Стьюдента не виявив статистично значущих відмінностей між моделями Claude Sonnet 4.6 і GPT-5.4 за людською оцінкою ($p = 0.918$).

Виявлені числові результати інтерпретуються як прояв ефекту «стелі» — відомого у літературі з оцінювання генеративних текстів феномену, коли сучасні топ-моделі досягають рівня якості, на якому ні людська, ні автоматична оцінка не дискримінують між окремими фрагментами. Цей результат становить самостійний внесок роботи: він демонструє межі застосовності традиційних метрик і рубрик на новому поколінні великих мовних моделей. Подальшими напрямами розвитку методики є: перенесення на домени з ширшою фактологічною варіативністю (правознавство, медицина, історія), диференціація рубрики за рівнями цільової аудиторії, розширення фактологічної компоненти через градуйовану верифікацію, інтеграція автоматичного детектора прикладів коду як п'ятої компоненти композитної метрики.

Запропонована методика з її композитною метрикою Q і рубрикою людської оцінки з педагогічним критерієм зберігає практичну цінність як інструмент для оцінювання фрагментів, згенерованих менш потужними або вузькоспеціалізованими моделями, і для застосування на доменах з більшою фактологічною варіативністю. Розроблений програмний пакет content_eval доступний для використання у подальших дослідженнях якості згенерованого освітнього контенту.

Окремий внесок становлять допоміжні емпіричні спостереження. Крос-модельна верифікація показала, що GPT як верифікатор є суворішим за Claude, що потребує окремого подальшого дослідження методологічних відмінностей внутрішньої калібровки топ-моделей при виконанні ролі експерта. Аналіз аномальних фрагментів виявив, що головна точка розходження між людськими експертами — оцінка дидактичної цінності концептуально повних, але абстрактних фрагментів без прикладів коду; це спостереження безпосередньо обґрунтовує запропонований напрям розширення методики через детектор прикладів.

Список використаних джерел

[1] Vaswani A., Shazeer N., Parmar N., Uszkoreit J., Jones L., Gomez A.N., Kaiser L., Polosukhin I. Attention is all you need. Advances in Neural Information Processing Systems (NeurIPS). 2017. P. 5998–6008.

[2] Brown T., Mann B., Ryder N. et al. Language models are few-shot learners. Advances in Neural Information Processing Systems (NeurIPS). 2020. P. 1877–1901.

[3] Hu K. ChatGPT sets record for fastest-growing user base — analyst note. Reuters. 1 лютого 2023. URL: https://www.reuters.com/technology/chatgpt-sets-record-fastest-growing-user-base-analyst-note-2023-02-01/

[4] Ouyang L., Wu J., Jiang X. et al. Training language models to follow instructions with human feedback. Advances in Neural Information Processing Systems (NeurIPS). 2022. P. 27730–27744.

[5] Bai Y., Kadavath S., Kundu S. et al. Constitutional AI: Harmlessness from AI feedback. arXiv preprint arXiv:2212.08073. 2022.

[6] CNBC. Nvidia briefly passes $3 trillion market cap on back of AI boom. CNBC. 5 червня 2024. URL: https://www.cnbc.com/2024/06/05/nvidia-briefly-passes-3-trillion-market-cap-on-back-of-ai-boom.html

[7] Hendrycks D., Burns C., Basart S., Zou A., Mazeika M., Song D., Steinhardt J. Measuring massive multitask language understanding. International Conference on Learning Representations (ICLR). 2021.

[8] Chen M., Tworek J., Jun H. et al. Evaluating large language models trained on code. arXiv preprint arXiv:2107.03374. 2021.

[9] Kasneci E., Sessler K., Küchemann S. et al. ChatGPT for good? On opportunities and challenges of large language models for education. Learning and Individual Differences. 2023. Vol. 103. P. 102274.

[10] Papineni K., Roukos S., Ward T., Zhu W.-J. BLEU: a method for automatic evaluation of machine translation. Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL). 2002. P. 311–318.

[11] Lin C.-Y. ROUGE: A package for automatic evaluation of summaries. Text Summarization Branches Out: Proceedings of the ACL-04 Workshop. 2004. P. 74–81.

[12] Zhang T., Kishore V., Wu F., Weinberger K.Q., Artzi Y. BERTScore: Evaluating text generation with BERT. International Conference on Learning Representations (ICLR). 2020.

[13] Sellam T., Das D., Parikh A. BLEURT: Learning robust metrics for text generation. Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL). 2020. P. 7881–7892.

[14] Yuan W., Neubig G., Liu P. BARTScore: Evaluating generated text as text generation. Advances in Neural Information Processing Systems (NeurIPS). 2021. P. 27263–27277.

[15] Liu Y., Iter D., Xu Y., Wang S., Xu R., Zhu C. G-Eval: NLG evaluation using GPT-4 with better human alignment. Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP). 2023. P. 2511–2522.

[16] Barzilay R., Lapata M. Modeling local coherence: An entity-based approach. Computational Linguistics. 2008. Vol. 34, № 1. P. 1–34.

[17] Sai A.B., Mohankumar A.K., Khapra M.M. A survey of evaluation metrics used for NLG systems. ACM Computing Surveys. 2022. Vol. 55, № 2. P. 1–39.

[18] van der Lee C., Gatt A., van Miltenburg E., Wubben S., Krahmer E. Best practices for the human evaluation of automatically generated text. Proceedings of the International Conference on Natural Language Generation (INLG). 2019. P. 355–368.

[19] Cohen J. Weighted kappa: Nominal scale agreement with provision for scaled disagreement or partial credit. Psychological Bulletin. 1968. Vol. 70, № 4. P. 213–220.

[20] Landis J.R., Koch G.G. The measurement of observer agreement for categorical data. Biometrics. 1977. Vol. 33, № 1. P. 159–174.

[21] Zheng L., Chiang W.-L., Sheng Y. et al. Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena. Advances in Neural Information Processing Systems (NeurIPS). 2023.

[22] Panickssery A., Bowman S., Feng S. LLM evaluators recognize and favor their own generations. arXiv preprint arXiv:2404.13076. 2024.

[23] Reimers N., Gurevych I. Sentence-BERT: Sentence embeddings using siamese BERT-networks. Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP). 2019. P. 3982–3992.

Додатки

Додаток А. Структура програмного пакета content_eval

Архітектура реалізації методики у вигляді Python-пакета. Каталог code/ містить:

— content_eval/ — основний пакет з реалізацією методики: — dsl.py — завантаження YAML-сценаріїв і їх валідизація; — generator.py — обгортка над litellm для виклику мовних моделей; — metric.py — агрегація компонент Q; — pipeline.py — пайплайн генерація + оцінка + збереження; — rubric.py — структури даних рубрики людської оцінки; — stats.py — статистичний аналіз (Cohen's κ, кореляція, grid-search); — components/ — підкаталог з компонентами метрики: — semantic.py — компонента S (sentence-transformers cosine); — ontology.py — компонента O (substring match з нормалізацією); — coherence.py — компонента C (комбінований local-flow + topic-anchoring); — factuality.py — компонента F (LLM-as-judge з recovery-парсером).

— scenarios/algorithms/ — 30 YAML-сценаріїв за темами курсу «Алгоритми і структури даних» НаУКМА.

— examples/ — допоміжні скрипти експерименту: — run_first_batch.py — повний прогін генерації корпусу; — run_cross_model_f.py — крос-модельна верифікація F-компоненти; — eval_analysis.py — аналіз результатів (κ, кореляції, grid-search); — make_eval_form.py — генерація Excel-форми оцінювання; — generate_charts.py — побудова matplotlib-графіків для роботи.

— data/generated/ — каталог з проміжними та фінальними результатами: — fragments_claude/ і fragments_openai/ — згенеровані фрагменти у форматі Markdown; — run_claude-sonnet-4-6.csv і run_gpt-5.4.csv — числові результати автоматичних метрик; — cross_model_f.csv — таблиця F-значень крос-модельної верифікації; — eval_results.xlsx — підсумкові статистичні таблиці.

Додаток Б. Приклад YAML-сценарію

```yaml
id: java_polymorphism
topic: "Поліморфізм у Java"
audience: "студенти 1 курсу спеціальності «Інженерія програмного забезпечення», 2 семестр"
concepts:
  - поліморфізм
  - динамічне зв'язування
  - перевизначення методу
  - перевантаження методу
  - upcasting
  - downcasting
  - virtual method invocation
constraints:
  length_words: [340, 550]
  style: pedagogical
  language: uk
examples:
  - "Чим перевизначення методу (overriding) відрізняється від перевантаження (overloading)?"
  - "У який момент JVM визначає, який саме метод викликати при поліморфному виклику?"
```

Додаток В. Приклад згенерованого фрагмента

java_arrays__claude-sonnet-4-6.md

Масив у Java — це структура даних, яка зберігає фіксовану кількість елементів одного типу в суміжних комірках пам'яті. Саме тому після створення масиву змінити його розмір неможливо: JVM виділяє під нього неперервний блок пам'яті, і розширення цього блоку «на місці» не передбачено.

Оголошення масиву виглядає так: `int[] numbers;` — це лише посилання, яке поки що дорівнює `null`. Щоб масив з'явився у пам'яті, потрібна ініціалізація: `numbers = new int[5];`. Можна об'єднати обидва кроки: `int[] numbers = new int[5];`. Після такої ініціалізації всі елементи масиву примітивного типу `int` автоматично отримують значення `0`. Для `boolean` це `false`, для об'єктних типів — `null`. Якщо значення відомі заздалегідь, можна використати скорочений запис: `int[] primes = {2, 3, 5, 7, 11};`.

Звертатися до елементів масиву можна за індексом, і тут важливо пам'ятати: індексація починається з нуля. Перший елемент — `numbers[0]`, останній — `numbers[numbers.length - 1]`. Властивість `length` повертає загальну кількість елементів. Якщо звернутися до елемента за межами допустимого діапазону, наприклад `numbers[5]` для масиву з п'яти елементів, JVM кине виняток `ArrayIndexOutOfBoundsException`. Це одна з найпоширеніших помилок початківців, тому перевіряйте межі в циклах.

Багатовимірні масиви в Java — це масиви масивів. Двовимірний масив оголошується як `int[][] matrix = new int[3][4];` і являє собою три масиви по чотири елементи. Рядки такого масиву можуть мати різну довжину — це так звані «зубчасті» масиви.

Для зручної роботи з масивами Java надає клас `java.util.Arrays`. Він містить статичні методи: `Arrays.sort(arr)` для сортування, `Arrays.fill(arr, value)` для заповнення однаковим значенням, `Arrays.copyOf(arr, newLength)` для копіювання, а також `Arrays.toString(arr)` — щоб швидко вивести вміст масиву у читабельному вигляді, не пишучи цикл вручну.

Додаток Г. Підсумкова таблиця оцінок експертів

#	Scenario ID	Тема	Ключові поняття	Фрагмент Claude	Clarity (C)	Correctness (C)	Relevance (C)	Didactic value (C)	Mean (C)	Фрагмент GPT	Clarity (G)	Correctness (G)	Relevance (G)	Didactic value (G)	Mean (G)
1	java_abstract_classes	Абстрактні класи у Java	абстрактний клас; абстрактний метод; abstract; спільний стан; конструктор; extends; інтерфейс	java_abstract_classes__claude-sonnet-4-6.md	5	5	4	4	4,5	java_abstract_classes__gpt-5.4.md	5	5	4	4	4,5
2	java_arrays	Масиви в Java	масив; оголошення масиву; ініціалізація масиву; довжина масиву; індексація з нуля; багатовимірні масиви; клас Arrays; ArrayIndexOutOfBoundsException	java_arrays__claude-sonnet-4-6.md	4	5	5	4	4,5	java_arrays__gpt-5.4.md	4	5	5	3	4,25
3	java_bitwise	Побітові операції у Java	побітове І; побітове АБО; виключне АБО; побітове заперечення; зсув ліворуч; зсув праворуч; беззнаковий зсув; бітові маски	java_bitwise__claude-sonnet-4-6.md	4	5	4	3	4	java_bitwise__gpt-5.4.md	4	5	4	4	4,25
4	java_collections	Collections Framework у Java	Collections Framework; інтерфейс List; інтерфейс Set; інтерфейс Map; ArrayList; LinkedList; HashMap; TreeMap	java_collections__claude-sonnet-4-6.md	5	5	4	4	4,5	java_collections__gpt-5.4.md	5	5	4	4	4,5
5	java_dates	Робота з датами і часом у сучасній Java	java.time; LocalDate; LocalDateTime; ZonedDateTime; Instant; Duration; Period; незмінність	java_dates__claude-sonnet-4-6.md	4	5	5	4	4,5	java_dates__gpt-5.4.md	4	5	5	4	4,5
6	java_encapsulation	Інкапсуляція і модифікатори доступу в Java	інкапсуляція; public; private; protected; package-private; геттер; сетер; приховування	java_encapsulation__claude-sonnet-4-6.md	5	5	5	5	5	java_encapsulation__gpt-5.4.md	5	5	5	5	5
7	java_exceptions	Обробка винятків у Java	виняток; try-catch; finally; throw; throws; checked exception; unchecked exception; try-with-resources	java_exceptions__claude-sonnet-4-6.md	4	5	5	4	4,5	java_exceptions__gpt-5.4.md	4	5	5	4	4,5
8	java_generics	Дженеріки в Java	дженеріки; параметр типу; типобезпека; bounded type; wildcard; PECS; стирання типів	java_generics__claude-sonnet-4-6.md	4	5	4	3	4	java_generics__gpt-5.4.md	4	5	4	3	4
9	java_inheritance	Наслідування і клас Object у Java	наслідування; клас Object; extends; перевизначення; super; toString; equals; hashCode	java_inheritance__claude-sonnet-4-6.md	5	5	5	4	4,75	java_inheritance__gpt-5.4.md	5	5	4	4	4,5
10	java_inner_classes	Внутрішні та вкладені класи у Java	вкладений клас; статичний вкладений клас; внутрішній клас; локальний клас; анонімний клас; доступ до зовнішнього стану; синтаксис new Outer.Inner	java_inner_classes__claude-sonnet-4-6.md	4	5	4	4	4,25	java_inner_classes__gpt-5.4.md	4	5	4	3	4

Таблиця 1

#	Scenario ID	Тема	Ключові поняття	Фрагмент Claude	Clarity (C)	Correctness (C)	Relevance (C)	Didactic value (C)	Mean (C)	Фрагмент GPT	Clarity (G)	Correctness (G)	Relevance (G)	Didactic value (G)	Mean (G)
1	java_abstract_classes	Абстрактні класи у Java	абстрактний клас; абстрактний метод; abstract; спільний стан; конструктор; extends; інтерфейс	java_abstract_classes__claude-sonnet-4-6.md	5	5	4	4	4,5	java_abstract_classes__gpt-5.4.md	5	5	4	4	4,5
2	java_arrays	Масиви в Java	масив; оголошення масиву; ініціалізація масиву; довжина масиву; індексація з нуля; багатовимірні масиви; клас Arrays; ArrayIndexOutOfBoundsException	java_arrays__claude-sonnet-4-6.md	4	5	5	4	4,5	java_arrays__gpt-5.4.md	4	5	4	3	4
3	java_bitwise	Побітові операції у Java	побітове І; побітове АБО; виключне АБО; побітове заперечення; зсув ліворуч; зсув праворуч; беззнаковий зсув; бітові маски	java_bitwise__claude-sonnet-4-6.md	4	5	5	4	4,5	java_bitwise__gpt-5.4.md	5	5	5	5	5
4	java_collections	Collections Framework у Java	Collections Framework; інтерфейс List; інтерфейс Set; інтерфейс Map; ArrayList; LinkedList; HashMap; TreeMap	java_collections__claude-sonnet-4-6.md	5	5	5	5	5	java_collections__gpt-5.4.md	5	5	5	5	5
5	java_dates	Робота з датами і часом у сучасній Java	java.time; LocalDate; LocalDateTime; ZonedDateTime; Instant; Duration; Period; незмінність	java_dates__claude-sonnet-4-6.md	5	5	5	5	5	java_dates__gpt-5.4.md	4	5	5	5	4,75
6	java_encapsulation	Інкапсуляція і модифікатори доступу в Java	інкапсуляція; public; private; protected; package-private; геттер; сетер; приховування	java_encapsulation__claude-sonnet-4-6.md	5	5	5	5	5	java_encapsulation__gpt-5.4.md	5	5	5	5	5
7	java_exceptions	Обробка винятків у Java	виняток; try-catch; finally; throw; throws; checked exception; unchecked exception; try-with-resources	java_exceptions__claude-sonnet-4-6.md	4	5	5	4	4,5	java_exceptions__gpt-5.4.md	5	5	5	5	5
8	java_generics	Дженеріки в Java	дженеріки; параметр типу; типобезпека; bounded type; wildcard; PECS; стирання типів	java_generics__claude-sonnet-4-6.md	4	5	5	4	4,5	java_generics__gpt-5.4.md	4	5	5	4	4,5
9	java_inheritance	Наслідування і клас Object у Java	наслідування; клас Object; extends; перевизначення; super; toString; equals; hashCode	java_inheritance__claude-sonnet-4-6.md	5	5	5	5	5	java_inheritance__gpt-5.4.md	4	5	5	4	4,5
10	java_inner_classes	Внутрішні та вкладені класи у Java	вкладений клас; статичний вкладений клас; внутрішній клас; локальний клас; анонімний клас; доступ до зовнішнього стану; синтаксис new Outer.Inner	java_inner_classes__claude-sonnet-4-6.md	5	5	5	5	5	java_inner_classes__gpt-5.4.md	5	5	5	5	5

Таблиця 2