НОВА УКРАЇНСЬКА ШКОЛА

Джон Ендрю Біос

ІНФОРМАТИКА 9

ЦИФРОВІ ПІДЛІТКИ

Містить цифровий контент



Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

НОВА УКРАЇНСЬКА ШКОЛА

Джон Ендрю Біос

ІНФОРМАТИКА 9

Підручник для 9 класу
закладів загальної середньої освіти

Рекомендовано Міністерством освіти і науки України

ЦИФРОВІ ПІДЛІТКИ



Київ
Видавництво «Лінгвіст»
2026

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

УДК 004*кл9(075.3)
Б 65

Рекомендовано Міністерством освіти і науки України
(наказ Міністерства освіти і науки України від від 27.01.2026 № 110)

Підручник створено відповідно до модельної навчальної програми «Інформатика. 7–9 класи» для закладів загальної середньої освіти (автори: О. В. Пасічник, Л. З. Козак, А. В. Ворожбит)

Біос Джон Ендрю

Б 65 Інформатика : підруч. для 9 кл. закл. загальн. середн. освіти. / Джон Ендрю Біос. — К. : Видавництво «Лінгвіст», 2026. — 208 с. : іл.

ISBN 978-617-8146-65-8

Підручник призначений для вивчення інформатики у 9 класі, розроблений згідно з Державним стандартом базової середньої освіти. Зміст підручника має практичне спрямування та реалізує компетентнісний підхід у навчанні. Розвиває наскрізні вміння та компетенції.

УДК 004*кл9(075.3)

Інтерактивний електронний додаток до підручника можна знайти за посиланням:
https://lms.e-school.net.ua/courses/course-v1:Linguist+IF09+2025/course/

ISBN 978-617-8146-65-8

Усі права застережено. All rights reserved
Copyright © 2026 by Binary Logic
Published by arrangement with Binary Logic

Підручник охороняється авторським правом.
Виключне майнове право на використання твору в Україні належить ТОВ «Видавництво "Лінгвіст"». Використання твору, а також його фрагментів без дозволу правовласника та видавця заборонено

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

ЗМІСТ

Розділ 1. Життя з розумними пристроями — 6

- Тема 1. Моніторинг і керування — 7
- Тема 2. Мікроконтролери, взаємодія з ними — 13
- Тема 3. Штучний інтелект — 17
- Тема 4. Цифрові пристрої та здоров'я — 25

Розділ 2. Дані, інформація, безпека — 27

- Тема 1. Дані, інформація, знання — 28
- Тема 2. Збір даних — 31
- Тема 3. Типи даних і кодування — 35
- Тема 4. Великі дані — 45
- Тема 5. Сучасні технології — 48
- Тема 6. Навчання в цифрову епоху — 54

Розділ 3. Упорядкування даних — 57

- Тема 1. Структурування інформації — 58
- Тема 2. Форми для введення даних — 65
- Тема 3. Запити — 69
- Тема 4. Звіти — 75
- Тема 5. Імпорт та експорт даних — 80

Розділ 4. Програмні проєкти — 86

- Тема 1. Списки та кортежі — 87
- Тема 2. Алгоритми впорядкування масивів / списків — 93
- Тема 3. Опрацювання текстових даних — 101
- Тема 4. Бібліотеки мови програмування — 105
- Тема 5. Технологія створення програмних продуктів — 114
- Тема 6. Документація програмного проєкту — 122
- Тема 7. Тестування та налагодження програмних проєктів — 126

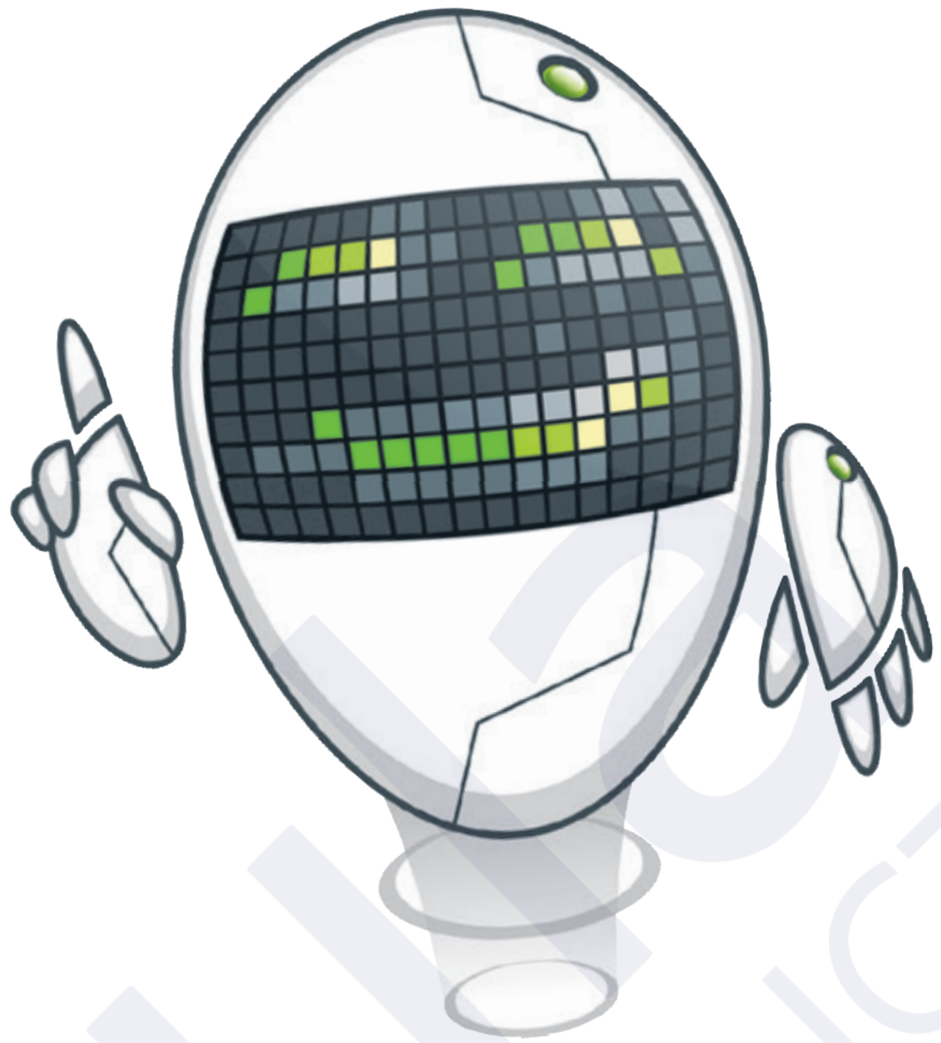
Розділ 5. 3D-моделювання — 132

- Тема 1. Моделі — 133
- Тема 2. Методи тривимірного моделювання — 135
- Тема 3. Етапи створення 3D-зображення — 141
- Тема 4. Графічний редактор Blender — 146
- Тема 5. Об'єктне моделювання у Blender — 155
- Тема 6. Полігональне моделювання у Blender — 159
- Тема 7. Основи візуалізації у Blender — 168
- Додаткова тема. Основи 3D-моделювання — 187
- Додаткова тема. 3D-друк — 198

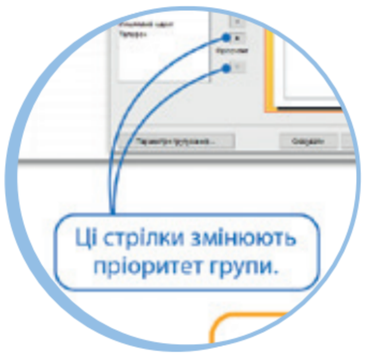
Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Привіт!

Разом ми реалізуємо місію – зрозуміти ЦИФРОВИЙ світ і встигнути за його швидкими змінами! Вмикайте свій комп'ютер та вирушайте за мною!



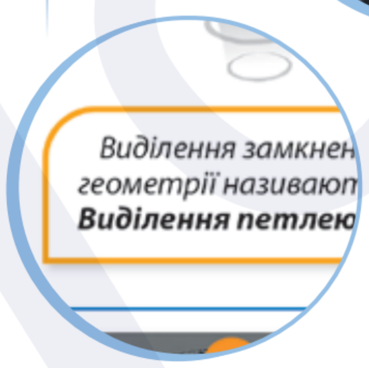
У підручнику ви натрапите на такі позначки



Розкриває елемент інтерфейсу та функціональні можливості інструментів.



Вказує на елемент або інструмент, який треба обрати чи натиснути на нього.



Звертає увагу на важливі поняття і термінологію.



ІСТОРІЯ – містить факти з історії розвитку ІКТ.

ПІДКАЗКА – акцентує увагу на альтернативних варіантах виконання поставлених завдань і додаткових інструкціях.

ДОВІДКА – розкриває додаткові можливості.

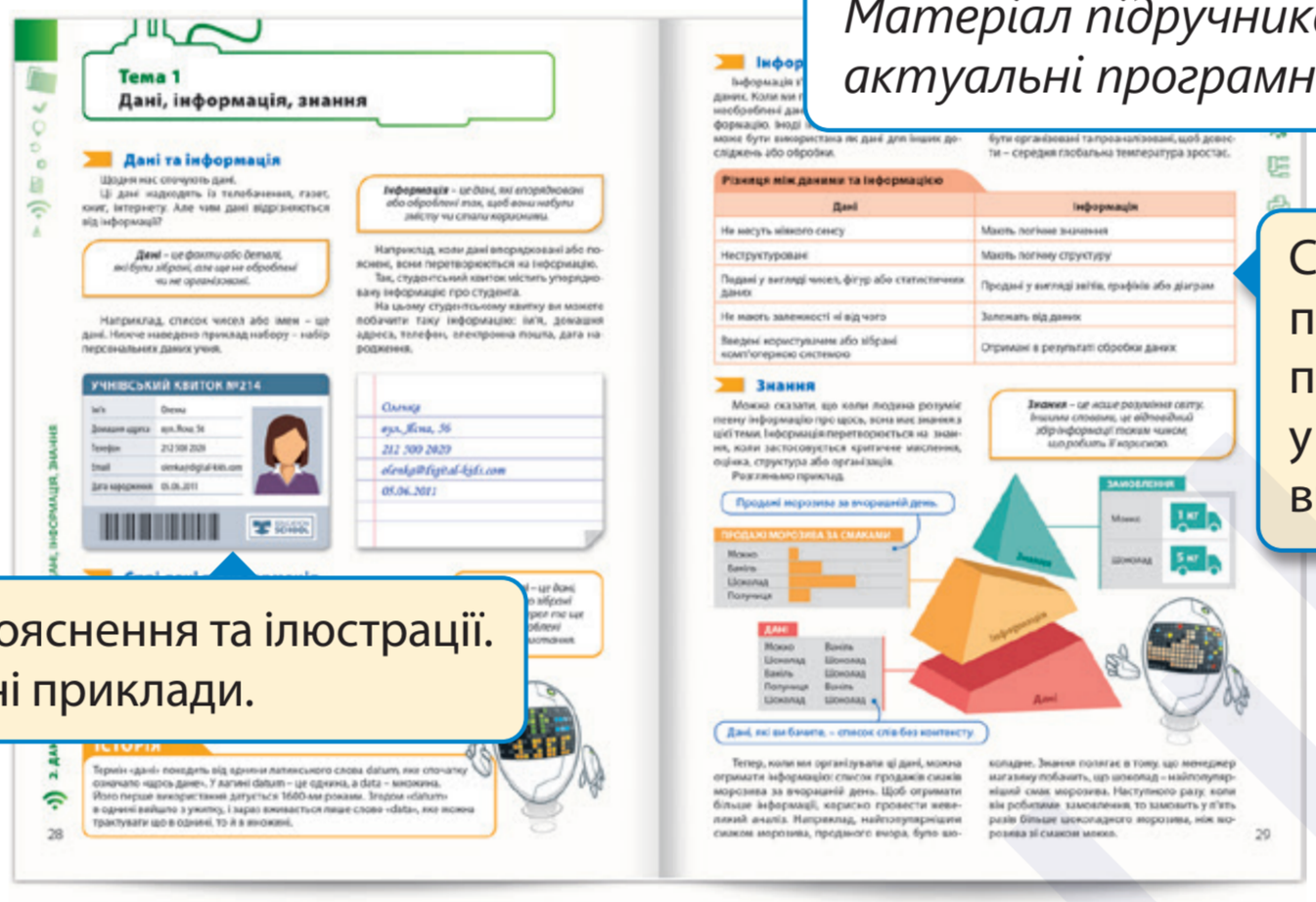


Відвідайте цифрові ресурси та знайдіть ті, які допоможуть вам працювати з підручником. Повторюйте те, чого навчилися, у будь-який час та в будь-якому місці. Використовуйте для цього зручний для вас цифровий пристрій.

https://formula.education/online-ict-9-klas/

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

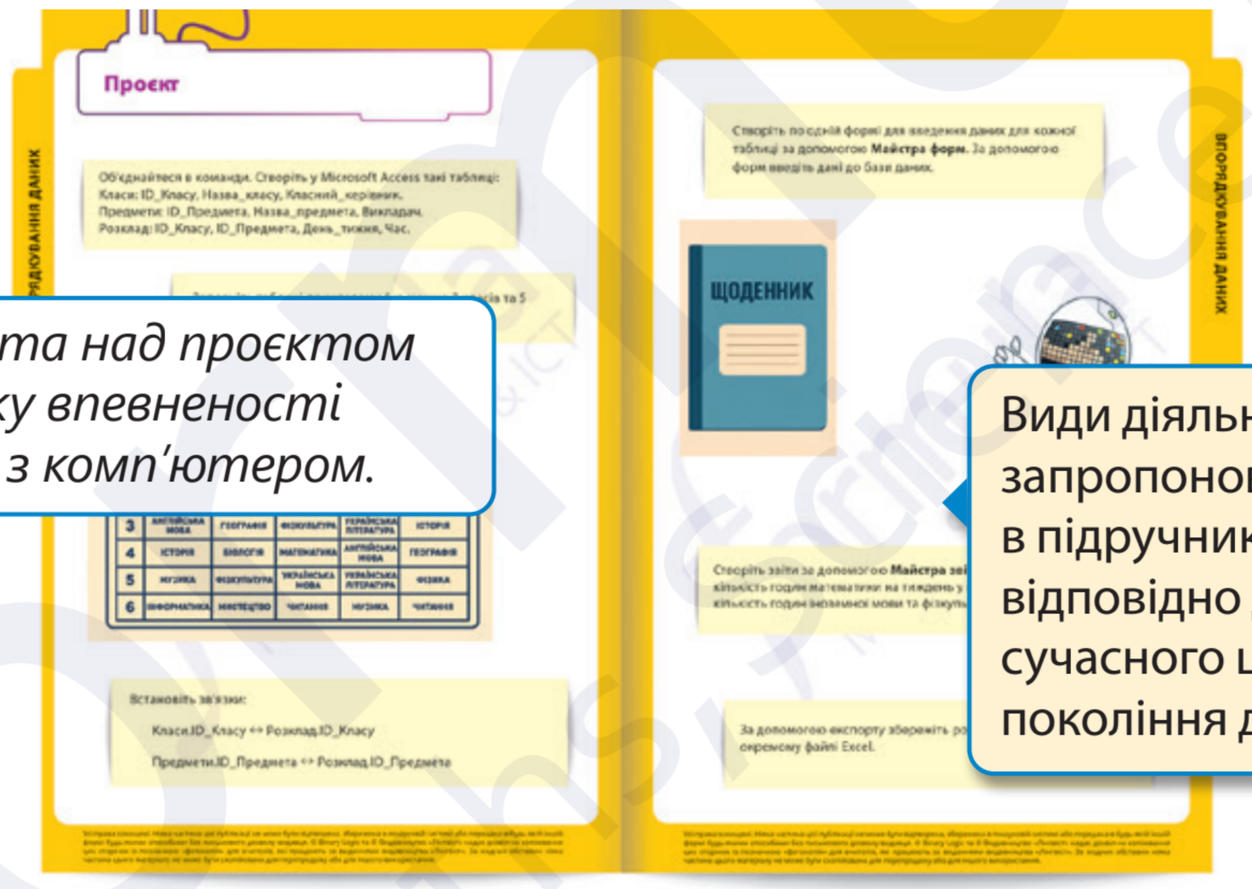
Познайомимось!



Матеріал підручника містить актуальні програмні продукти.

Спіральна структура підручника дає змогу повторювати матеріал у класі щоразу на більш високому рівні.

Чіткі пояснення та ілюстрації. Сучасні приклади.



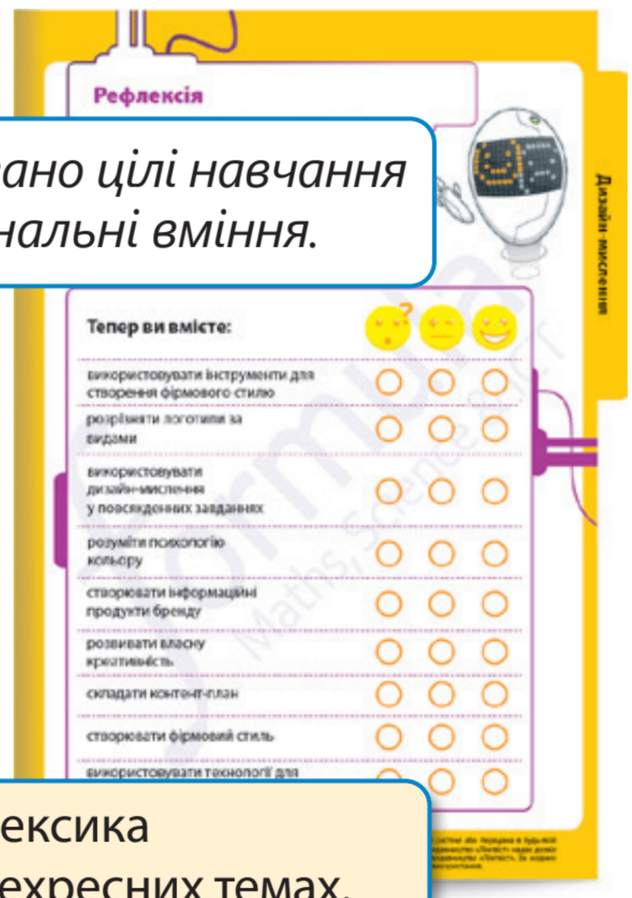
Командна робота над проєктом сприяє розвитку впевненості під час роботи з комп'ютером.

Види діяльності, запропоновані в підручнику, підібрано відповідно до потреб сучасного цифрового покоління дітей.



Сформульовано цілі навчання та функціональні вміння.



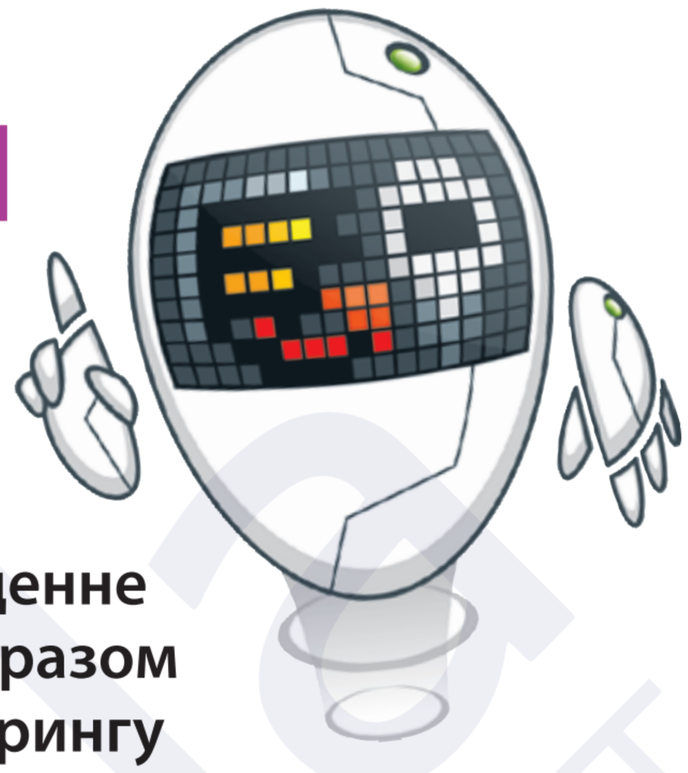


Розділ **Інші платформи** відкриває для здобувачів і здобувачок освіти доступні альтернативи програм та їхніх інструментів.

Нова лексика на перехресних темах.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

1. Життя з розумними пристроями



У наш час технології розвиваються швидкими темпами та докорінно змінюють наше повсякденне життя. У цьому розділі поговоримо про життя разом з розумними пристроями, про системи моніторингу та управління, про використання датчиків.

Дізнаєтеся більше про дані та системи машинного навчання, штучний інтелект, робототехніку та їх впровадження.

З'ясуєте, як використання цифрових пристроїв впливає на людей та зрозумієте наслідки цього впливу.

У цьому розділі ви навчитеся:

- розрізняти типи датчиків;
- розуміти, де та як можна застосовувати роботів;
- відрізняти застосування штучного інтелекту від систем машинного навчання;
- розуміти етичні наслідки застосування цифрових пристроїв;
- правильно користуватися цифровими пристроями.

Програмне забезпечення та обладнання:

- робот Edison;
- EdPy;
- Arduino IDE;
- плата Arduino;
- makecode.microbit.org;
- плата MicroBit;
- Python;
- плата Raspberry Pi.

У цьому розділі ви дізнаєтеся:

- що таке система моніторингу та контролю;
- які існують датчики;
- як використання роботів та сучасних технологій покращує наше життя;
- як машинне навчання покращує комп'ютери та наше життя;
- де застосовується штучний інтелект;
- які загрози для безпеки та приватного життя несуть розумні пристрої;
- про фізичні та психологічні наслідки використання цифрових пристроїв.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Тема 1
Моніторинг і керування

Системи моніторингу

Система моніторингу призначена для спостереження та передавання даних на сервер, мережевий пристрій або іншу систему. Спостереження в системах здійснюється за допомогою датчиків, дані з них синхронізуються. Система відстежує ці дані, аналізує їх та надсилає результати. Найпоширенішим прикладом системи моніторингу є охоронна сигналізація. Системи моніторингу дедалі частіше інтегруються із системами керування, які утворюють об'єднані системи моніторингу й керування.



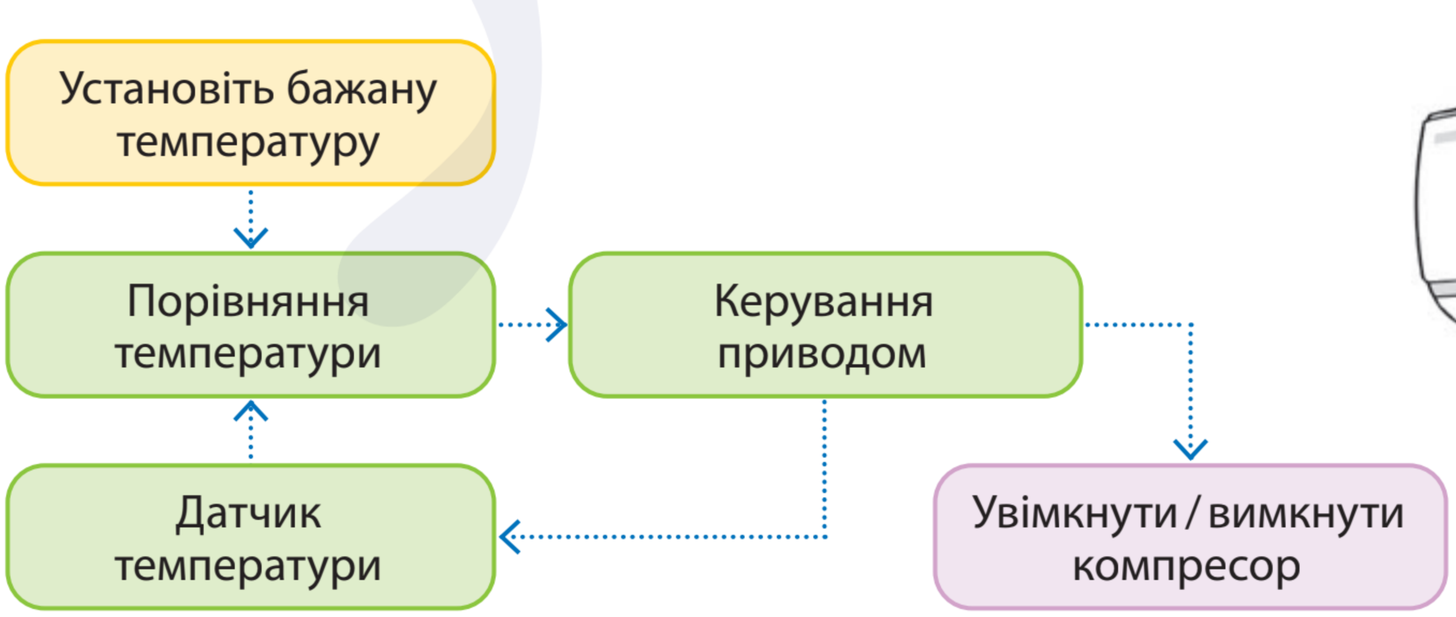
Системи керування

Залежно від сфери застосування та мети, елементи керування можуть варіюватися від простих вимикачів до повністю автоматизованих комп'ютерних систем. Такі системи широко використовують в побуті й промисловості, як-от у пральних машинах, кондиціонерах, системах охоронної сигналізації тощо. Існує два основні типи систем керування: система з відкритим контуром та система із замкнутим контуром. Характеристики кожного типу наведені в таблиці.

Система керування – *це сукупність взаємопов'язаних компонентів, які здатні керувати процесом або змінною.*

Типи систем керування

Система з відкритим контуром	Система із замкненим контуром
Вихідний результат не впливає на вхідні дані у процесі керування	На виході фіксується результат, і система використовує ці дані як зворотний зв'язок для керування та саморегулювання
Немає порівняння між фактичними та бажаними значеннями	Відстежує вихідні значення і повертає частину інформації назад для досягнення бажаного результату
Погано пристосована до зміни умов, що знижує ефективність виконання поставлених завдань	Призначена для автоматичного досягнення та підтримки заданих умов
Системи без зворотного зв'язку	Система зі зворотним зв'язком



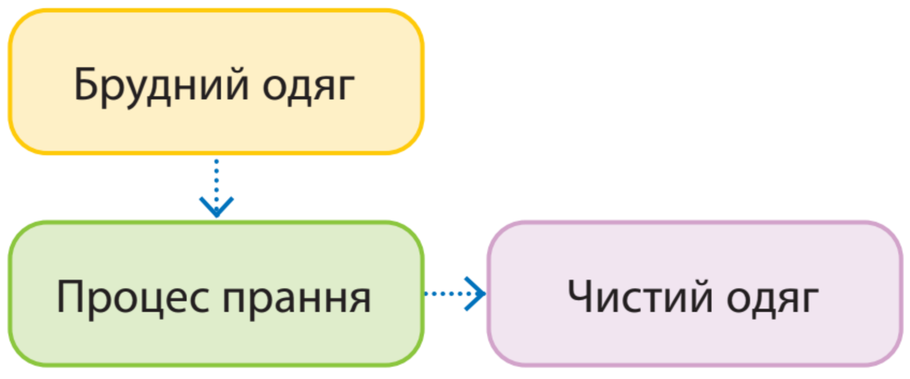


Кондиціонер
блок-схема системи із замкненим контуром

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Пральна машина

блок-схема системи з відкритим контуром





Системи моніторингу й керування *– це системи, які відстежують середовище і керують ним. Вони використовують апаратні засоби, як-от датчики – для моніторингу, виконавчі механізми – для керування.*

Системи моніторингу й керування

Ці системи постійно зчитують показники від датчиків і вживають заходи залежно від отриманих даних. Коли датчики фіксують певні події, вони надсилають сигнал до центральної станції моніторингу, де система керування здійснює відповідні дії. Прикладом такої системи є система, яка контролює температуру та автоматично вмикає або вимикає обігрівачі.





Датчики

Функція датчика – збирати дані та надсилати їх до комп'ютерної системи, що їх обробляє та вживає відповідних заходів. Датчики є одним з основних елементів систем моніторингу й контролю. Ми живемо в оточенні датчиків, які є частиною нашого повсякденного життя. Наприклад, вони вмикають світло, регулюють температуру, виявляють дим, відчиняють двері тощо.

Датчик *– це пристрій, який виявляє фізичний сигнал, як-от рух, світло, тиск, температуру тощо, та генерує вихідний сигнал.*

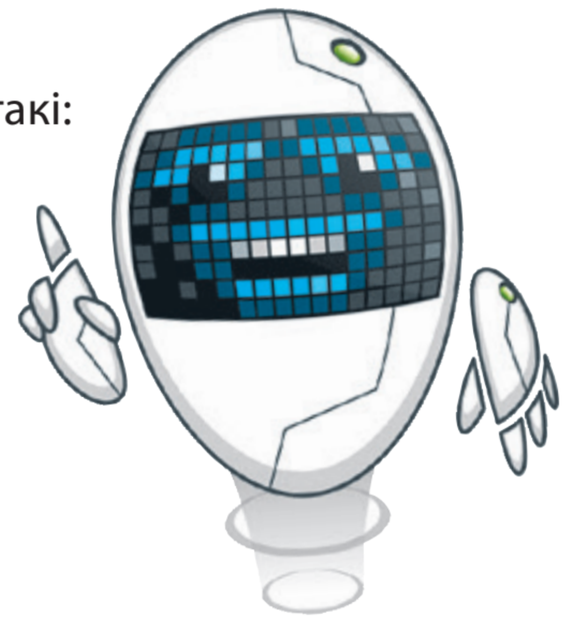
Типи датчиків

Існує багато різних типів датчиків, але найчастіше використовують такі:



Датчики температури

Це пристрої, які вимірюють температуру навколишнього середовища. Ви можете знайти такі датчики в системах опалення та охолодження, у холодильниках, печах тощо. Медичні термометри також мають датчик температури для її вимірювання.





Датчики освітленості

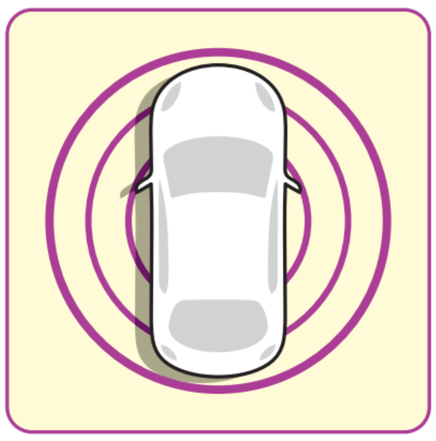
Це електронні пристрої, які використовуються для виявлення світла. Існує багато різних типів датчиків освітленості, і їх можна знайти в багатьох пристроях. Мобільні телефони та смарттелевізори мають датчики, які визначають рівень освітлення в кімнаті та автоматично підвищують або знижують яскравість екрана. Інший приклад – вуличні ліхтарі, які автоматично вмикаються, коли темніє. Деякі датчики освітленості виявляють випромінювання, невидиме для людського ока, як-от рентгенівське, інфрачервоне або ультрафіолетове.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.



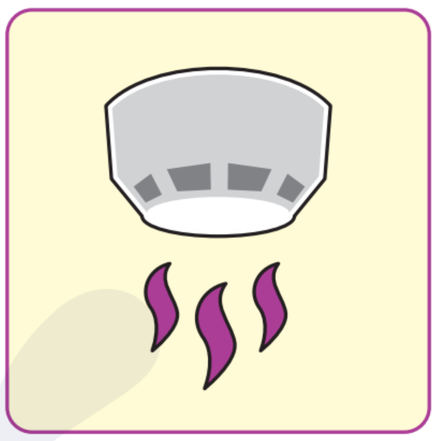
Датчики тиску

Це пристрої, які вимірюють тиск. Вимірювання тиску зазвичай пов'язане з рідинами чи газами. Датчик тиску діє як перетворювач, він генерує сигнал як функцію прикладеного тиску. Ви можете знайти такі датчики в метеостанціях, літаках, автомобілях та іншому обладнанні, де реалізовано функцію вимірювання тиску.



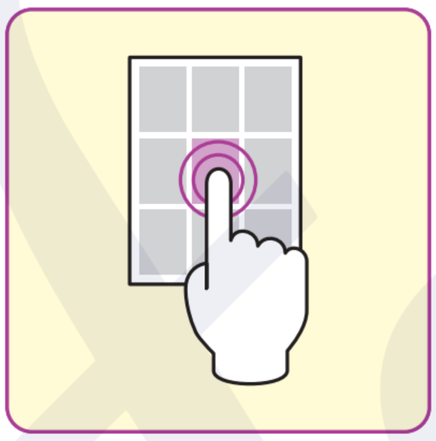
Датчики наближення

Вони визначають присутність об'єктів без фізичного контакту. Для цього використовують електромагнітні поля, світло та звук. Типовий приклад – парктронік в автомобілях. Ці датчики використовують в роботизованих машинах, щоб уникнути зіткнення з об'єктами. Ще одним пристроєм, який використовує датчики наближення, є мобільний телефон, який вимикає екран, коли ви підносите телефон до вуха під час дзвінка.



Датчики диму

Це пристрої, які виявляють дим як індикатор пожежі. Зазвичай їх підключають до центральної системи пожежної сигналізації, вони вмикають тривогу, коли виявляють дим. У кожному будинку повинен бути принаймні один датчик диму. Найчастіше датчики диму встановлюють на кухні та в спальнях.



Сенсорні датчики (датчики дотику)

Вони чутливі до дотику, сили або тиску. Їхній принцип роботи подібний до простого вимикача. При контакті з поверхнею сенсорного датчика ланцюг усередині датчика замикається, виникає струм. Коли контакт розривається, розмикається ланцюг, струм зникає. Такі датчики найчастіше використовують у мобільних пристроях, як-от смартфони, планшети, ноутбуки. Їх можна зустріти в пультах управління та пультах дистанційного керування. У майбутньому вони замінять більшість механічних кнопок і перемикачів.



Датчики руху

Вони використовують одну або декілька технологій для виявлення присутності рухомого об'єкта, який потрапляє в поле зору датчика. Найчастіше датчики руху використовуються для охорони. Датчик виявляє рух та надсилає сигнал тривоги на головний пульт системи безпеки. Деякі системи безпеки записують події на камеру при виявленні руху. Найпоширеніші застосування датчиків руху – це вуличні ліхтарі, системи зовнішнього освітлення, автоматичні двері, що активуються рухом. Нове покоління датчиків руху використовується в гарнітурах віртуальної реальності та ігрових контролерах, наприклад Kinect. Рух виявляється за допомогою інфрачервоного випромінювання у вигляді тепла, яке виділяють люди та тварини, за допомогою випромінювання і прийому віддзеркалених мікрохвильових імпульсів і ультразвукових хвиль, подібно до радарів, а також за допомогою виявлення вібрацій.



Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Робототехніка

Робототехніка не є новою галуззю технологій, але з останніми інженерними розробками та зростаючим використанням штучного інтелекту ро́боти стають чимось зовсім іншим, ніж просто автоматизовані маніпулятори, які виконують повторювані завдання на складальних лініях автомобільних заводів.

Сьогодні повністю автономні ро́боти, здатні розуміти своє оточення та взаємодіяти з ним, стали реальністю. Потужні процесори, удосконалені датчики та приводи, розпізнавання зображень і складні поведінкові алгоритми на базі штучного інтелекту – усе це дає змогу ро́ботам вийти за межі простих роботів-пилососів на кшталт Roomba.



Проблеми, які виникають з розвитком робототехніки

Соціальні
Чи зможуть ро́боти замінити людей на їх робочих місцях? Які професії під загрозою?

Фінансові
Чи справді роботизація знижує загальні витрати, чи просто замінює людей дешевшою робочою силою, яка витримує додаткове навантаження?

Якісні
Чи можуть ро́боти виконувати ту саму робо́ту так само точно та ефективно, як людина?



Суспільство негативно реагує на надмірне використання технологій штучного інтелекту. Якщо контроль над машинами опиниться не в тих руках, вони можуть стати причиною руйнувань або катастроф. Машини не думають, перш ніж діяти. Вони можуть бути запрограмовані на погані вчинки або навіть на масове знищення людства та руйнування.

Ро́боти можуть використовуватись для виконання повторюваних, складних або небезпечних завдань. Наприклад, робот може працювати в екстремальних умовах або входити до складу групи з розмінування.



Дрони

Дрони оснащені камерами та датчиками, їх можна використовувати для різних цілей: аерофотозйомка, спостереження, наукові дослідження чи доставка товарів. Дрони зазвичай асоціюються з безпілотними літальними апаратами(БПЛА), проте існують і наземні, водні і навіть підводні дрони. Вони можуть слідувати дуже точним курсом доти, доки їхня батарея може забезпечувати двигуни енергією. Якщо батарея розряджається, він може безпечно приземлитися без будь-яких пошкоджень.



Дрон (англ. Dynamic Remotely Operated Navigation Equipment) *– динамічне дистанційно кероване навігаційне обладнання), також відомий як безпілотний літальний апарат (БпЛА) – це тип літального апарату, яким можна керувати дистанційно або який може літати автономно за попередньо запрограмованим планом польоту.*

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Сфери застосування дронів

Цивільна: для реагування на надзвичайні ситуації, для рятувальних операцій, для охорони правопорядку

Соціальна: для ведення зйомок, оскільки вони мають камери з високою роздільною здатністю, та для журналістських цілей

Наукова: у наукових дослідженнях, моніторингу забруднення та інших наукових галузях, як-от геодезія, археологія тощо

Комерційна: для транспортування та доставки, оскільки вони суттєво розвантажують рух транспорту в місцях, прибираючи його з доріг і спрямовуючи в повітря

Військова: військові дрони використовуються, коли ситуація вважається занадто ризикованою чи складною, чи неможливою для виконання людиною



Автономне водіння

Автономне керування транспортними засобами базується на поступовому розвитку технологій, які класифікуються за ступенями автоматизації.

Існує шість рівнів автоматизації водіння за даними Товариства автомобільних інженерів (SAE):

Рівень 0
Без автоматизації водіння

Є базовим, що включає всі традиційні транспортні засоби, де вся відповідальність і виконання всіх оперативних завдань лежать на людині. На цьому рівні водій вручну керує транспортним засобом і контролює його

Рівень 1
Допомога водієві

На найнижчому рівні автоматизації водій стежить за системою. Для допомоги водієві доступна єдина автоматизована система, наприклад система допомоги при паркуванні – парктронік або утримання смуги

Рівень 2
Часткова автоматизація водіння

Автомобіль оснащено вдосконаленою системою допомоги водієві (ADAS), яка допомагає водію керувати автомобілем, а також регулює швидкість. Однак людина-водій має бути готова втрутитися

Рівень 3
Умовна автоматизація водіння

Автомобіль має здатність виявляти оточення й на основі цього приймати обґрунтовані рішення. Водій-людина має бути напоготові взяти під контроль керування автомобілем

Рівень 4
Висока автоматизація водіння

Взаємодія з людиною не потрібна, але доступна функція ручного керування. Транспортний засіб працює в автономному режимі

Рівень 5
Повна автоматизація водіння

Автомобіль працює повністю автономно й не потребує уваги людини взагалі

За допомогою систем часткової автоматизації автомобіль може автономно їздити на великі відстані в певних дорожніх ситуаціях, наприклад, на автомагістралях. Однак водій має можливість перебрати керування на себе протягом декількох секунд, наприклад, на ділянках дорожніх робіт.

Останні рівні – повністю автономні, оскільки транспортні засоби будуть незалежними в будь-яких ситуаціях та умовах водіння. Кожен у машині буде пасажиром, тому що автомобіль виконує всю роботу, пов'язану з водінням.

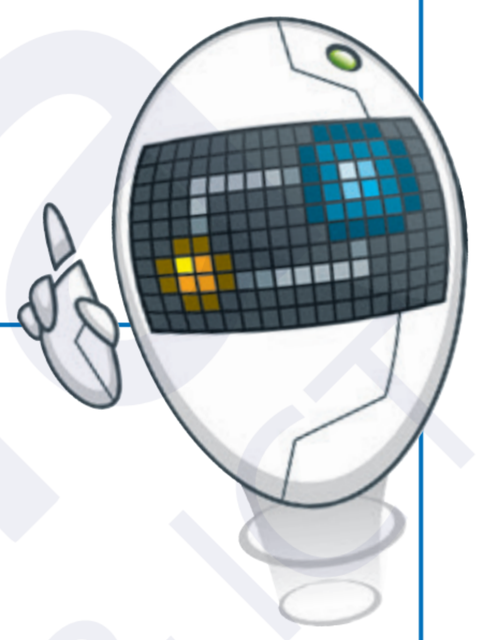
Кожен рівень визначається відповідним обсягом та ступенем автоматизації. На нижчих рівнях деякі функції автоматизовані, як-от гальмування, рульове керування та прискорення.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Автономне водіння – це складна технологія, яка пов'язана з різними викликами. Деякі з них охоплюють:

1. Технічні виклики
Автономне водіння вимагає передових датчиків і комп'ютерних систем для точного виявлення та інтерпретації навколишнього середовища.

2. Безпека і надійність
Забезпечення безпеки пасажирів та інших учасників дорожнього руху є найважливішим завданням автономного водіння. Система повинна бути надійною та відмовостійкою, а процес прийняття рішень – прозорим.



3. Регулювання
Автономне водіння є відносно новою технологією, тому бракує чітких правил і стандартів для її впровадження.

4. Схваленням громадськості
Формування довіри суспільства до технології автопілота є складним завданням. Люди повинні бути впевнені, що автономні транспортні засоби можуть безпечно пересуватися на дорогах і вони не становитимуть загрози їхній безпеці.

Автоматична гальмівна система

Датчики використовують дані від лазерів, радарів, ультразвукових хвиль, інфрачервоного випромінювання або відео з камер, щоб виявити інші транспортні засоби або перешкоди на шляху автомобіля.

GPS-система також може виявляти фіксовані (нерухомі) небезпеки, наприклад знаки зупинки, за допомогою бази даних про місце розташування.

Більшість цих систем призначена для зниження швидкості автомобіля перед тим, як він наблизиться до небезпеки. Ці системи можуть автоматично активувати гальма при виявленні перешкоди або надати водієві допомогу при гальмуванні у вигляді сигналу чи голосового повідомлення.

Технології автоматичного гальмування покладаються на інформацію від датчиків.



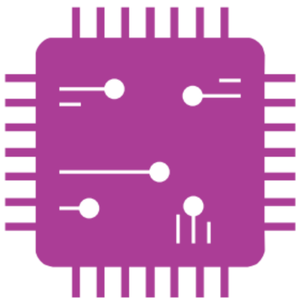
Практичне завдання

Завантажте файл-заготовку зі сторінки інтернет-підтримки. Виконайте запропоноване завдання.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Тема 2

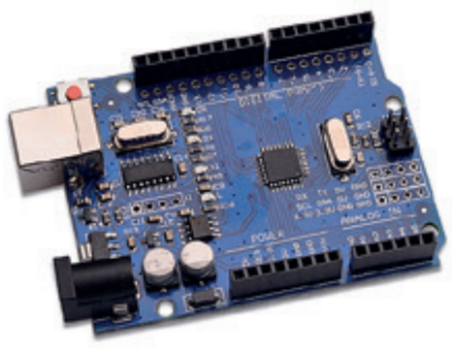
Мікроконтролери, взаємодія з ними



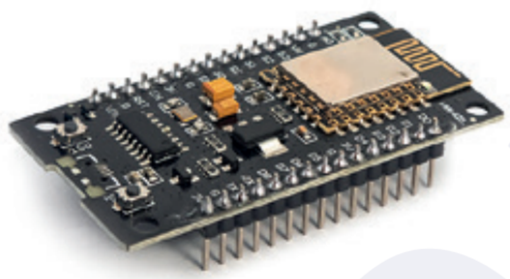
Головна складова систем моніторингу та керування, яка управляє всією системою – мікроконтролер.

Мікроконтролер – це невеликий комп'ютер на одній платі, який може керувати різними пристроями.

Найпоширеніші мікроконтролери



Arduino – мікроконтролер з відкритим кодом, простий у використанні. Програмується мовою, схожою на C / C++

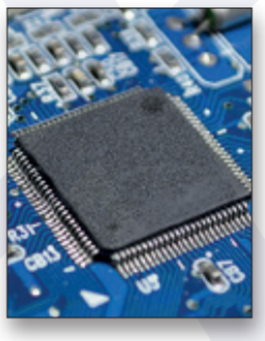


ESP32 – контролер із вбудованим Wi-Fi, використовують для підключення пристроїв до мережі

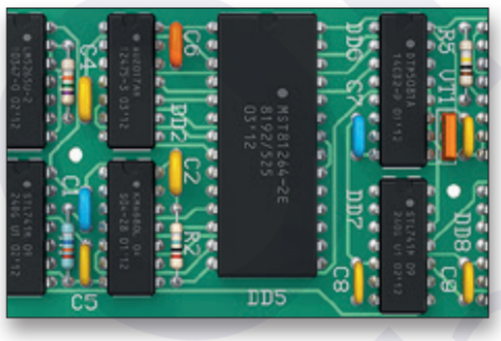


Raspberry Pi – повноцінний одноплатний комп'ютер, який можна використовувати для складніших IoT-проєктів. Програмується мовами Python, JavaScript тощо

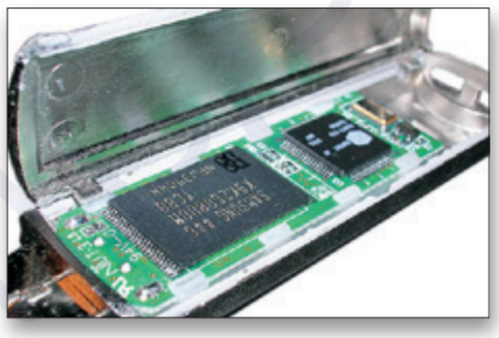
Основні компоненти типової плати



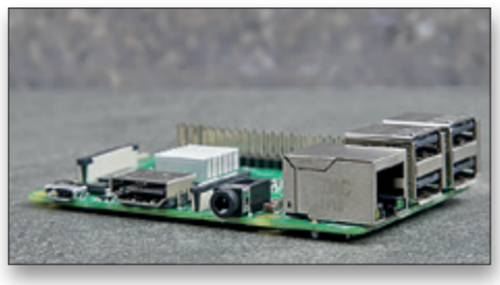
Процесор виконує команди



Оперативна пам'ять (RAM) для тимчасового зберігання даних



Постійна пам'ять (Flash) для збереження програми



Порти вводу / виводу для підключення датчиків, світлодіодів, двигунів тощо



Інтерфейси зв'язку –
USB, Wi-Fi, Bluetooth, UART, I2C



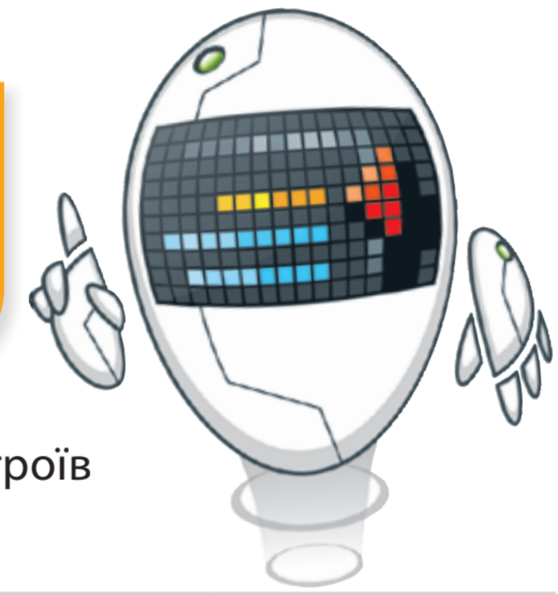
Завдяки таким платам можна створювати розумні пристрої: сигналізації, електронні системи пропуску, автоматичні системи освітлення, поливу, годування тварин тощо.

А завдяки відкритій архітектурі до плати Arduino, Raspberry Pi чи будь-якого іншого мікроконтролера можна під'єднати різні датчики, системи індикації, двигуни, сервоприводи, екрани для виводу інформації, тощо.

Саме до плат мікроконтролерів підключають різноманітні датчики та сенсори. Але щоб пристрій став «розумним», треба не тільки зібрати та передати дані, а й аналізувати їх, прийняти рішення, чи треба щось змінити в системі.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Відкрита архітектура *– це принцип побудови комп'ютерних або електронних систем, за яким вони створюються так, щоб їх можна було легко змінювати, розширювати та поєднувати з іншими пристроями або програмами, незалежно від виробника.*



Наведені приклади мікроконтролерів – це яскраві приклади пристроїв з відкритою архітектурою. Вони мають:

Докладну технічну документацію
Яку може переглянути будь-хто

Універсальні порти вводу / виводу
До яких можна підключати найрізноманітніші модулі

Сотні доступних бібліотек та прикладів коду
Для використання різних сенсорів, дисплеїв, двигунів тощо

Так, система з відкритою архітектурою не є «закритою коробкою», а навпаки – дає змогу користувачу або розробнику змінювати її склад, підключати нові компоненти, оновлювати програмне забезпечення або створювати власні доповнення.

Однією з найбільших переваг відкритої архітектури є те, що навколо неї формується активна спільнота користувачів і розробників. Це люди з усього світу – учні, вчителі, інженери, програмісти, хобісти – які створюють, діляться і вдосконалюють проєкти на основі відкритих архітектур. Спільнота допомагає навчатися, обмінюватися досвідом і розвивати технології разом. Завдяки підтримці спільноти навіть початківець може швидко почати створювати власні пристрої та долучитися до глобального технічного руху.

Переваги відкритої архітектури

Відкрита архітектура	Закрита архітектура
Запрограмовані правила	Обмежена у можливостях
Підтримує різні пристрої	Працює лише з «рідними» модулями
Підходить для навчання та експериментів	Можна лише користуватися
Розвивається спільнотою	Контролюється одним виробником

Ще одна складова таких систем – програма, яка буде керувати системою.

Залежно від пристрою чи платформи, для програмування використовують різні мови

C / C++ – найчастіше використовуються для програмування контролерів Arduino

Python – зручна для початківців, широко застосовується на Raspberry Pi

JavaScript (через **Node.js**) – для роботи з мережею та вебсервісами

MicroPython або **CircuitPython** – полегшені версії Python для мікроконтролерів

Вам уже відомо, що програми створюють команди розробників. Вони підтримують ці програми, випускають оновлення та додають нові функції. Таке програмне забезпечення називається ***пропрієтарним*** *– тобто закритим. Його не можна змінювати або копіювати без дозволу.*

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

На відміну від закритого, відкрите програмне забезпечення дає користувачам більше свободи: програму можна використовувати, змінювати, копіювати та розповсюджувати. Код такої програми відкритий – будь-хто може подивитись, як вона працює, і внести свої зміни.

API (Application Programming Interface) *– це набір правил і команд, процедур, функцій, структур і констант, які уможливлюють взаємодію програм між собою або з обладнанням.*



LibreOffice® – приклад відкритого програмного забезпечення. Наприклад, ви можете створити власну функцію, яка буде рахувати кількість учнів, у прізвищах яких є літера «А».

Пов'язаним з відкритим програмним забезпеченням можна вважати термін **API**.

Що означає відкритий код програми

- Користувачі можуть **вільно запускати** програму з будь-якою метою
- Користувачі мають **доступ до вихідного коду** та можуть вивчати, як програма працює, й адаптувати її під свої потреби
- Користувачі можуть **вдосконалювати програму** та публікувати свої поліпшення, щоб ними могла скористатися вся спільнота
- Користувачі можуть бути впевнені, що програмне забезпечення **не містить прихованих функцій**, які можуть загрожувати їхній конфіденційності або безпеці

Це правила «спілкування» між програмами, які дають їм змогу обмінюватися даними. Приклад використання API у повсякденному житті – застосунки перегляду погоди. Коли ви відкриваєте мобільний застосунок погоди, він не зберігає інформацію самостійно. Замість цього він використовує API, щоб отримати актуальні дані з метеорологічного серверу та відобразити їх на екрані.

За допомогою API можна написати програму, яка отримує дані з датчика температури та виводить їх на екрані.

Як різні програми та функції можуть передавати або виводити інформацію на ваш пристрій?

І тут можна згадати про таку властивість ПЗ, як **сумісність**.

Сумісність *– це здатність пристроїв або програм працювати разом без конфліктів.*

Сумісними можуть бути: флешка та ноутбук, Google Docs чи файл PDF і смартфон.

Більш широким терміном є **інтероперабельність**. Наприклад, ви можете підключити принтер виробника Canon до ноутбука чи комп'ютера від компанії Apple, попросити ваш розумний холодильник увімкнути телевізор.

Інтероперабельність *– здатність різних систем обмінюватися інформацією та використовувати її.*

Відкрита архітектура – це свобода. Вона дозволяє створювати, змінювати та вдосконалювати пристрої під свої потреби. Саме завдяки відкритій архітектурі мільйони людей у світі – від школярів до інженерів – можуть створювати власні розумні пристрої, досліджувати електроніку й робити техніку більш доступною та корисною.

У процесі створення нових пристроїв або програмних продуктів важливу роль відіграє **прототип**. Його головна мета – перевірити ідею, перш ніж витрачати ресурси на повномасштабну розробку.

Прототип не обов'язково повинен мати завершений вигляд. Іноді це лише базова модель з основними функціями або зовнішнім макетом. Він допомагає розробникам, інженерам або дизайнерам побачити, як працює задум на практиці.

Прототип – *перша, спрощена версія майбутнього виробу.*

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Розробка прототипу дає змогу

- **Перевірити функціональність:** чи працює пристрій або програма згідно із задумом
- **Оцінити зручність:** як користувач взаємодіє з продуктом
- **Виявити недоліки:** технічні або логічні помилки, складності у використанні
- **Приймати рішення:** чи варто продовжувати розробку, що потрібно змінити, додати або прибрати

Етапи створення прототипу

Формування ідеї	Планування структури	Побудова первинної версії	Налагодження
Розуміння, що саме має виконувати майбутній продукт.	Визначення основних компонентів: що потрібно створити, який це матиме вигляд, як взаємодіятиме з користувачем.	Створення базового пристрою або програмної оболонки.	Усунення початкових помилок і перевірка працездатності.

Залежно від типу продукту, прототип може бути фізичним (наприклад, електронний пристрій на основі мікропроцесора) або цифровим (наприклад, інтерфейс вебсайту чи мобільного застосунку).

Результати тестування зазвичай вказують на ті місця, які потребують модифікації – тобто змін, покращень або навіть повного перероблення певних частин.

Іноді після кількох етапів модифікації створюється нова версія прототипу – ближча до фінального продукту.

*У багатьох галузях – від техніки до дизайну інтерфейсів – **створення прототипу** є обов'язковим етапом роботи.*



Модифікація може стосуватися:

- **апаратної частини** (наприклад, заміна датчика, покращення живлення)
- **програмного коду** (оптимізація, додавання нових функцій, виправлення помилок)
- **дизайну інтерфейсу** (зміна кольорів, розміщення кнопок, підказок тощо)

***Мікроконтролери та сенсори** – це основа сучасних розумних пристроїв. Завдяки відкритій архітектурі, API та сумісності створення власних проєктів стає доступним навіть школярам і школяркам.*

Практичне завдання

Завантажте файл-заготовку зі сторінки інтернет-підтримки. Виконайте запропоноване завдання.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Тема 3
Штучний інтелект

Протягом багатьох десятиліть учені, які займаються комп'ютерними науками, намагаються створити комп'ютери, які можуть мислити, як люди.

Експертна система – це технологія ШІ, яка імітує прийняття рішень експертом у певній галузі: медицина, фінанси, технічна діагностика, управління проєктами. Вони почали з'являтися в 1960–70-х роках. Ці системи використовують бази знань для розв'язання складних проблем. Вона працює, використовуючи базу даних експертних знань для прийняття рішень і надання відповідей. Наприклад, медична експертна система може використовувати знання про хвороби, симптоми та методи лікування для встановлення діагнозу пацієнту. На відміну від сучасних інструментів ШІ, які можуть генерувати помилкову або вигадану інформацію (так звані «галюцинації»), експертні системи надають обґрунтовані відповіді та надають пояснення щодо запропонованого результату.

Штучний інтелект (ШІ) *– це галузь комп'ютерних наук, яка займається створенням систем і програм, здатних виконувати завдання, що зазвичай вимагають людського інтелекту: розпізнавання закономірностей, мови, зображень, розв'язання проблем, прийняття рішень.*

З розвитком розпізнавання образів і теорії комп'ютерного навчання з'явилася нова технологія – машинне навчання. **Машинне навчання** – напрям штучного інтелекту, який передбачає навчання комп'ютерів на основі даних. Це передбачає створення алгоритмів, які можуть виявляти закономірності та робити прогнози на основі даних, на яких вони були навчені. Наприклад, алгоритм машинного навчання може бути використано для прогнозування того, який фільм сподобається людині на основі її історії переглядів.

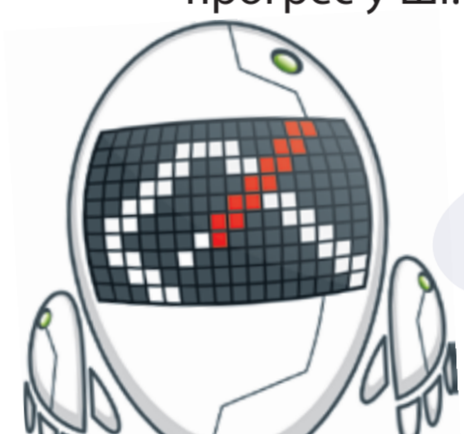
Важливі події в історії розвитку штучного інтелекту

Дартмутська конференція 1956 року вважається місцем народження штучного інтелекту як галузі досліджень, а Джон Маккарті вперше ввів цей термін у 1955 році.

У 1966 році в Стенфордському дослідницькому інституті був створений робот Shakey. Це був перший автономний мобільний робот.

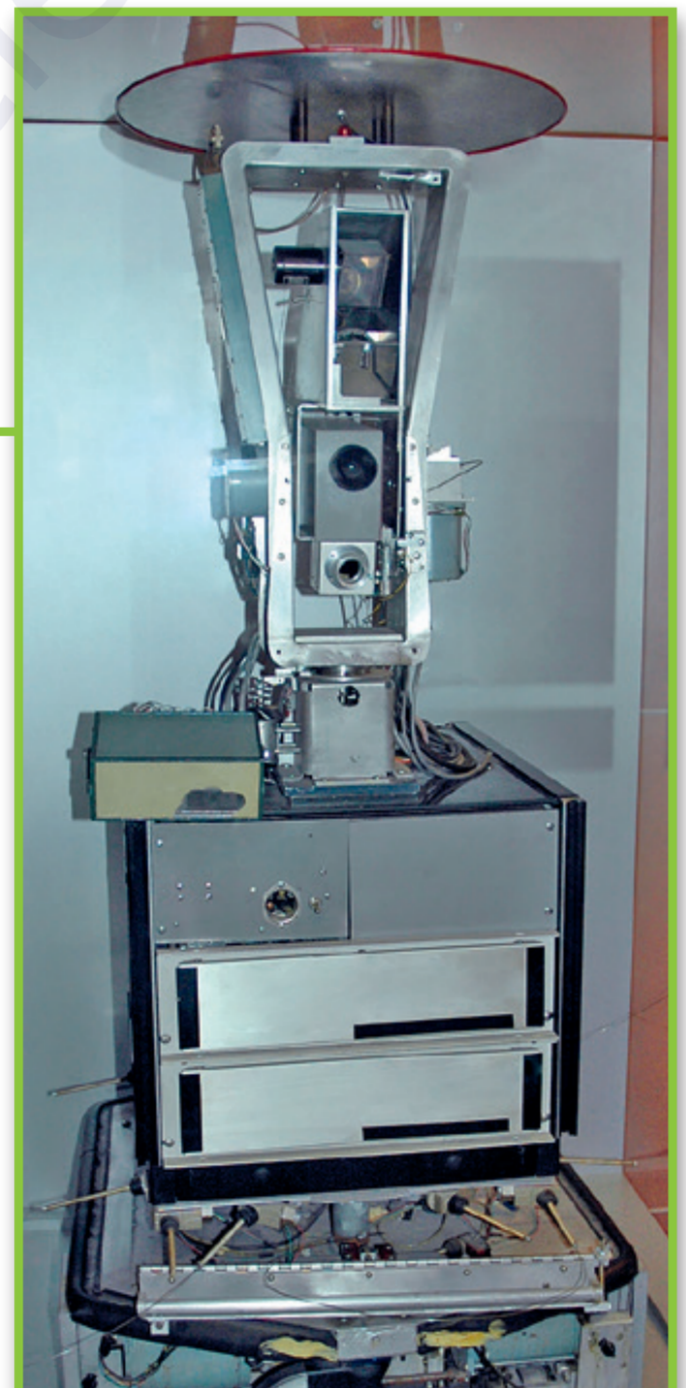
Комп'ютерна програма ELIZA була розроблена в середині 1960-х років Джозефом Вейценбаумом. Вона імітувала розмову терапевта з пацієнтом. Це був один з перших прикладів обробки природної мови.

Експертні системи у 1970-х роках ознаменували значний прогрес у ШІ. Ці системи використовували знання та логіку для вирішення проблем і прийняття рішень.



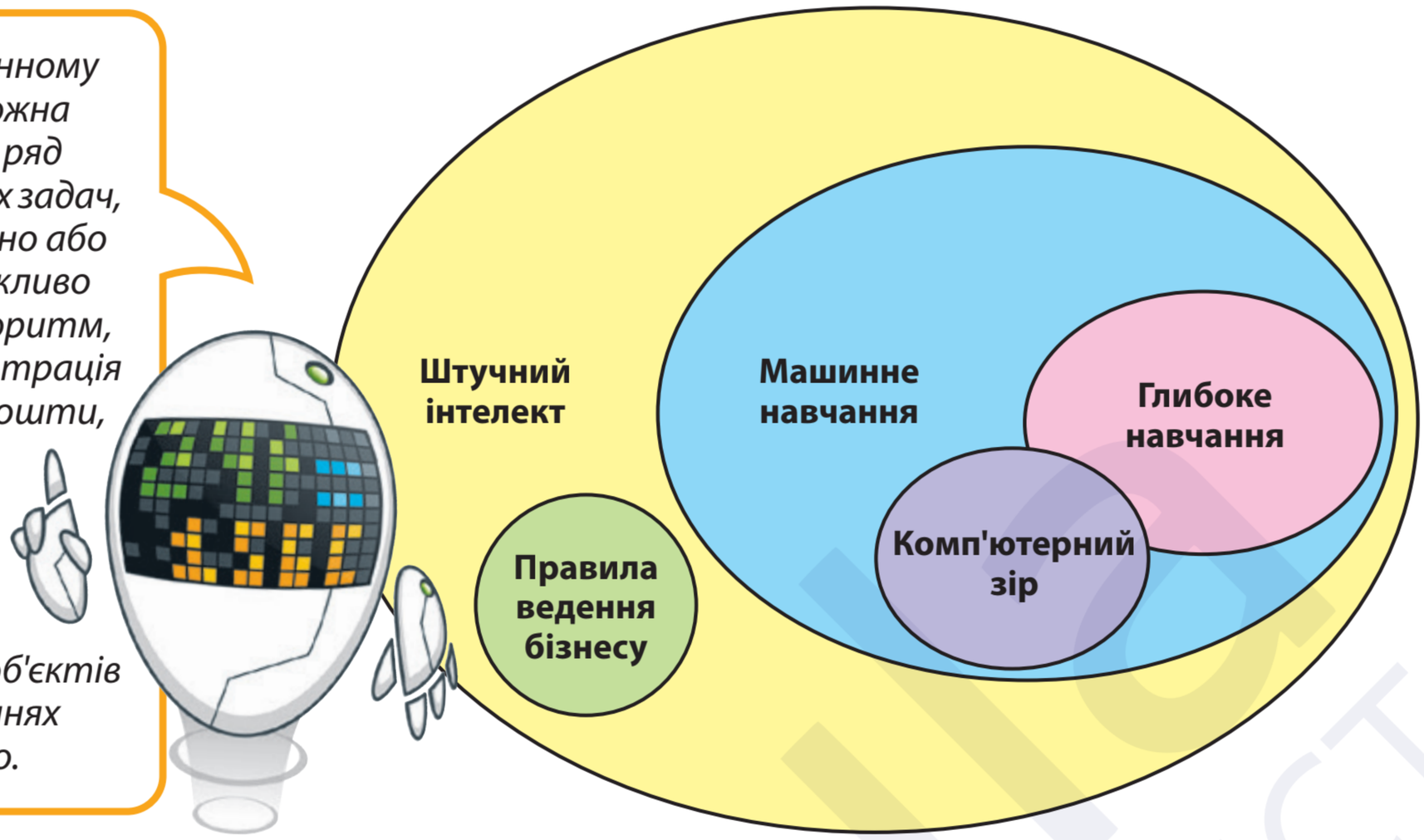
Deep Blue – шаховий комп'ютер, розроблений компанією IBM. У 1997 році він переміг чемпіона світу із шахів Гаррі Каспарова.

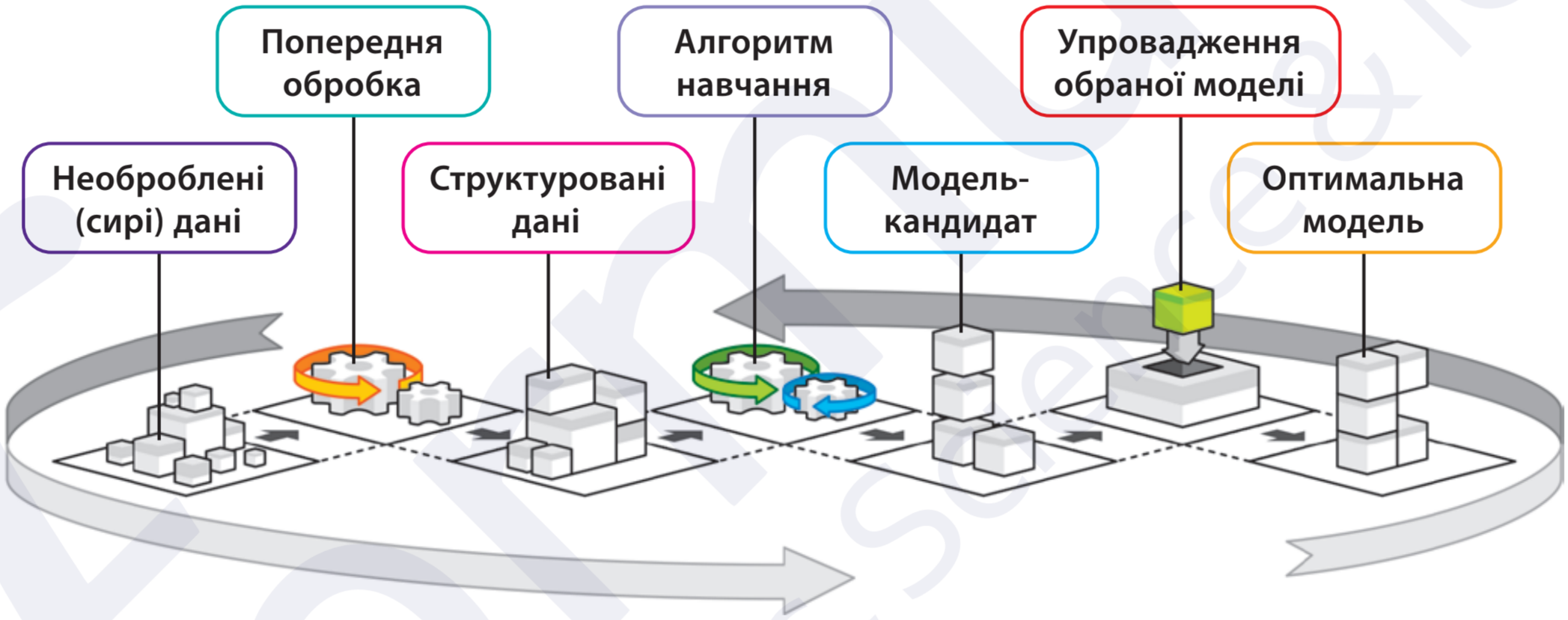
У 2023 році OpenAI випустила GTP-4, найдосконалішу на той час велику мовну модель, яка використовує методи глибокого навчання для генерації тексту, схожого на людський.



Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Завдяки машинному навчанню можна покращити ряд обчислювальних задач, для яких складно або просто неможливо створити алгоритм, наприклад, фільтрація електронної пошти, виявлення мережевих зловмисників, розпізнавання тексту, розпізнавання об'єктів на зображеннях або у відео.





Основні характеристики ШІ

Збір даних	Збір та зберігання великих обсягів даних, які використовуються як вхідні дані для системи ШІ
Запрограмовані правила	Визначення конкретних правил і алгоритмів, які система ШІ буде використовувати для прийняття рішень і виконання певних завдань
Здатність до міркування	Системи ШІ мають здатність міркувати й робити логічні висновки на основі наданих їм даних і правил
Здатність до навчання	Системи ШІ можуть постійно вдосконалюватися завдяки навчанню на основі зібраних даних і отриманого зворотного зв'язку. Це досягається за допомогою алгоритмів машинного навчання, які дають змогу системам ШІ автоматично оновлювати свої моделі та підвищувати їхню точність із часом

ДОВІДКА

Для процесу машинного навчання потрібно постійно додавати нові дані та через певні проміжки часу заново перенавчати модель. Це необхідно для того, щоб модель завжди була актуальною і точною та відповідала змінним умовам і новим сценаріям.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Застосування машинного навчання

Ймовірно, ви вже мали нагоду скористатися застосунками чи програмами, де використовуються технології машинного навчання. Розглянемо кілька прикладів, як вони працюють у дії.

Комп'ютерний переклад стикається зі складнощами в точності, плавності та повноти перекладу.

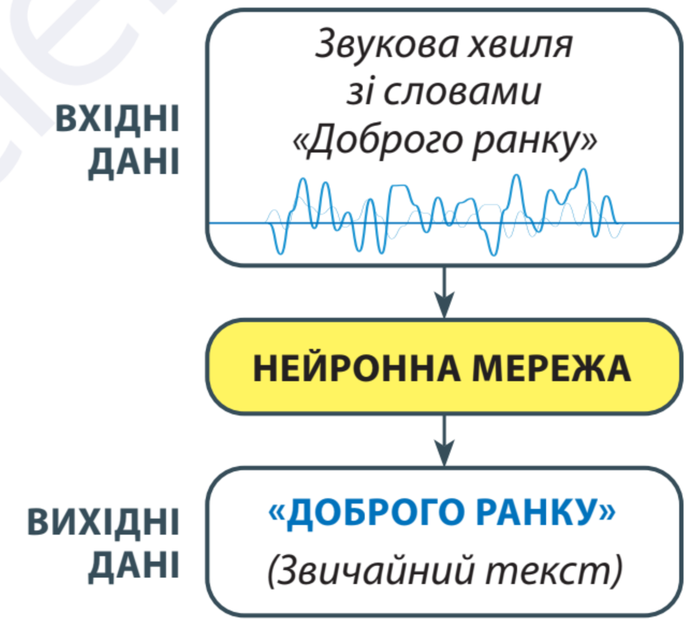
Комп'ютерний переклад

Існують системи автоматичного машинного перекладу, які перекладають текст із вебсайтів, документів і навіть зображень, а також мовлення в режимі реального часу за допомогою інструментів відеоконференцій, наприклад Microsoft Teams, чи дивлячись відео на YouTube.

Неоднозначність	Словниковий запас	Контекст	Граматика та синтаксис	Культурні відмінності
Однією з найбільших проблем є те, що природні мови часто мають кілька значень для одного й того ж слова чи фрази, що ускладнює вибір правильного значення варіанта для систем машинного перекладу.	Системам машинного перекладу може бути складно з рідкісними чи незвичними словами та фразами, що призводить до помилок у перекладі.	Перекладачі не завжди можуть розуміти контекст речення, що призводить до помилок у перекладі, які змінюють сенс оригінального тексту.	Багато природних мов мають складні правила граматики та синтаксису, які важко врахувати для точного перекладу речень.	Системи машинного перекладу не враховують культурних відмінностей, що призводить до перекладів, які стають недоречними чи навіть образливими в певних культурах.

Розпізнавання мовлення

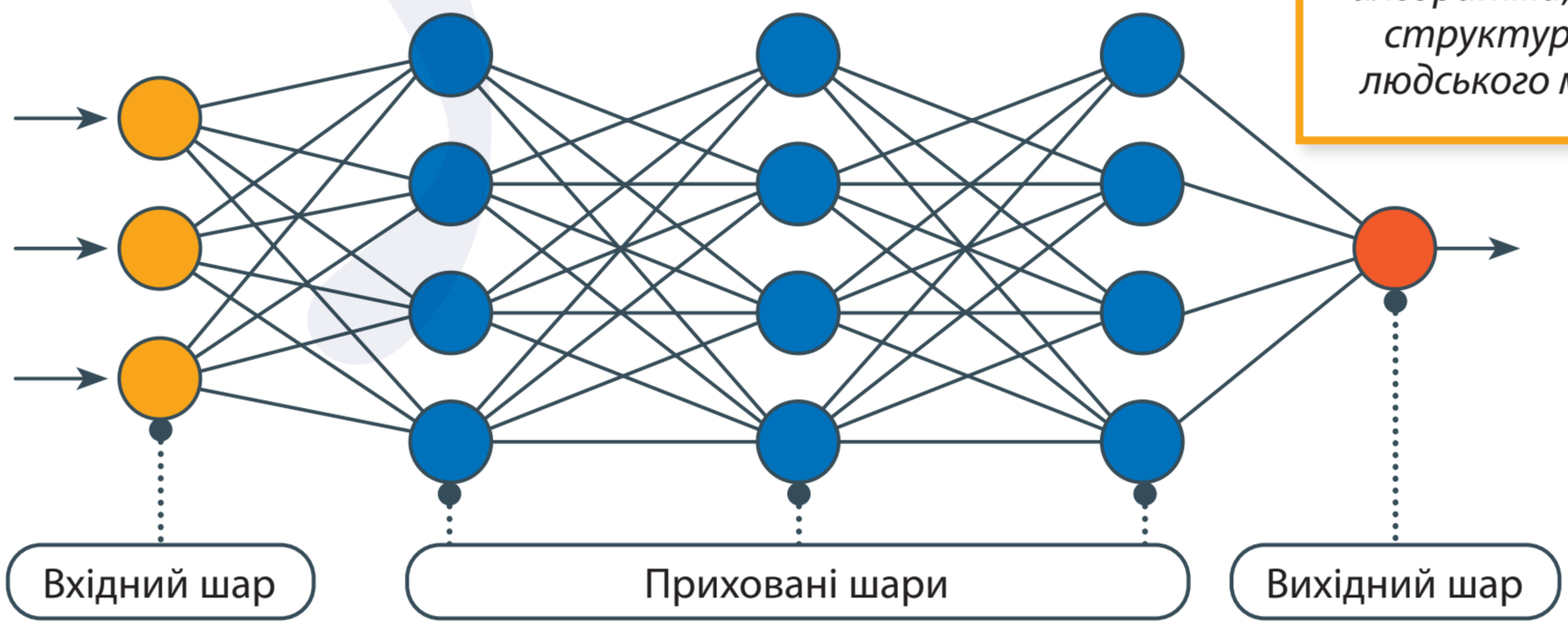
Розпізнавання мовлення, також відоме як **розпізнавання голосу**, – це процес перетворення усного мовлення в текст. Його вбудовують у різні пристрої: смартфони, ігрові консолі, розумні годинники чи смартхолодильники тощо. Завдяки машинному навчанню пристрій може розпізнавати та «розуміти» вимовлені слова, оцифровуючи звуки і порівнюючи їх зі збереженими зразками.



Нейронні мережі

Нейронні мережі складаються із взаємопов'язаних вузлів «нейронів», які призначені для обробки інформації та прогнозування. Наприклад, нейронна мережа може бути використана для ідентифікації об'єктів на зображенні за допомогою розпізнавання шаблонів у пікселях.

Нейронні мережі *– це тип машинного навчання, яке використовує алгоритми, що імітують структуру та функції людського мозку.*





Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Розпізнавання образів

Розпізнавання образів – це галузь комп'ютерних наук, яка займається ідентифікацією та інтерпретацією шаблонів і об'єктів на цифрових зображеннях. Ефективність цієї технології залежить від здатності класифікувати зображення. Класифікація – зіставлення шаблону з даними. Один із поширених прикладів – гарнітура, яка постійно фіксує зображення навколо людини з порушенням зору чи незрячої людини, яка допомагає їй зрозуміти, що відбувається, через опис того, що «бачить» і розпізнає система.

Незважаючи на досягнення в цій галузі, все ще існують проблеми, які потрібно вирішити для підвищення точності та надійності розпізнавання зображень.



Варіабельність зображень	Шум на зображенні	Перевантаження фону	Обмеження в реальному часі
Різні зображення можуть відрізнятися за освітленням, орієнтацією, масштабом тощо. Це впливає на точність алгоритмів розпізнавання зображень.	Шум може включати артефакти, пил, подряпини та інші спотворення, які можуть призвести до хибних результатів.	Алгоритми розпізнавання зображень повинні вміти розрізняти цільовий об'єкт і фон, який часто може бути захаращений або містити сторонні об'єкти.	Алгоритми повинні бути швидкими та ефективними, щоб обробляти зображення в режимі реального часу, і бути корисними для практичного застосування.

Віртуальний персональний асистент

Віртуальний персональний асистент (VPA) – це програма, яка імітує розмову для надання користувачеві голосової чи текстової інформації через веб- або мобільний інтерфейс. Наприклад, Siri, Cortana чи Google Assistant можуть керувати календарем, нагадуваннями та пропонувати кафе, ресторани на основі наших уподобань і місця перебування.

Попри переваги VPA, їхнє впровадження та використання пов'язане з низкою проблем.



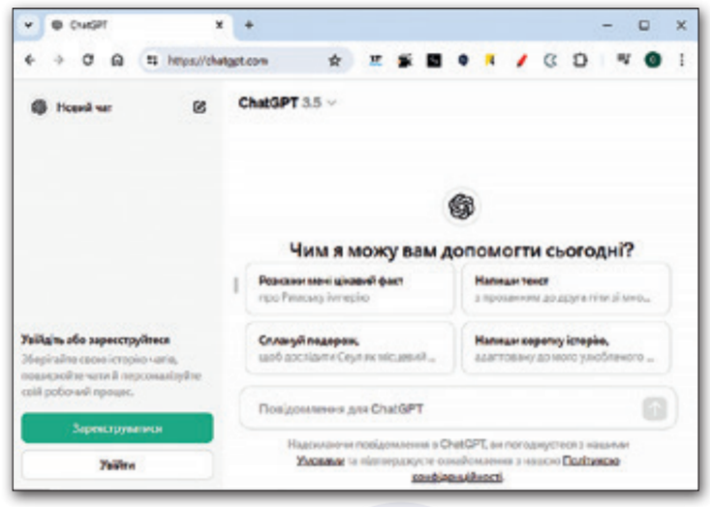
Виклики обробки природної мови (NLP)	Конфіденційність і безпека	Обмежений функціонал	Інтеграція з іншими системами
Віртуальні асистенти покладаються на технології NLP, щоб розуміти запити користувача. Але технологія все ще перебуває на ранніх стадіях розвитку й може неправильно тлумачити нюанси людської мови та контексту, що призводить до непорозумінь.	Віртуальні помічники збирають і зберігають великі обсяги персональних даних, що викликає занепокоєння з приводу їх захисту: які дані збираються, як вони використовуються та кому передаються.	Віртуальні асистенти хороші настільки, наскільки хороші технології, на яких вони побудовані. Вони можуть виконувати лише ті завдання, які не виходять за межі їхніх запрограмованих можливостей. Вони ще не здатні вирішувати складні та різноманітні завдання.	VPA не завжди добре інтегруються з іншими системами, як-от календарі, контакти та інші застосунки, що ускладнює їх безперебійне використання. Але постійно вдосконалюються технології, які дають змогу покращити сумісність віртуальних помічників з різними платформами.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Генерація тексту

Генерація тексту – це процес, під час якого комп'ютерна система за допомогою штучного інтелекту створює текст, який на вигляд наче створений людиною. Такі системи навчаються на великих обсягах текстових даних і вчаться розуміти структуру мови, граматику, контекст і навіть стиль.

Сучасні програми, як-от чат-боти або перекладачі, можуть писати речення, відповіді й навіть цілі тексти, ґрунтуючись на тому, як люди зазвичай говорять і пишуть.



Приклади застосування

Чат-боти – відповідають на запитання, можуть вести діалог

Copilot

Gemini

ChatGPT

Машинний переклад – перекладають з багатьох мов

Google Translate

DeepL

grammarly

Автоматичне написання текстів – для новин, описів товарів, резюме тощо

Claude

copy.ai

Copymate

Підказки при наборі тексту – як у смартфонах або в Gmail, Word, Google Документи

Але потрібно пам'ятати, що ШІ не «бачить» і не «розуміє», як людина, – він просто аналізує дані.



Генерація зображень та відео

Генерація зображень і відео – це процес створення нових зображень або відео за допомогою ШІ. Такі системи навчаються на великій кількості зображень чи відео й вчаться відтворювати подібні.

За допомогою ШІ можна створити зображення, відео чи анімацію за текстовим описом або просто змінити фон на світлині. Ця технологія використовується в іграх, кіноіндустрії, дизайні та навчанні. ШІ не «малює» самостійно, як людина, але комбінує те, що «вивчив». Такі моделі «вивчають», який вигляд мають об'єкти, кольори, форми, а коли ви вводите опис (наприклад: «Казковий ліс із гігантськими грибами замість дерев, світлячки осявають шлях, магічна атмосфера, детальний цифровий арт»), ШІ перетворює цей текст на картинку.

Програма використовує складні математичні обчислення, щоб створити нове зображення, яке раніше не існувало: DALL·E, Midjourney, Adobe Firefly.

Для створення відео ШІ спочатку створює кадри – окремі картинки.

Потім ці кадри об'єднуються в рухоме відео. Деякі моделі можуть навіть додавати звук, музику або голос: Runway ML, Pika Labs, Sora від OpenAI.

У мультфільмах та іграх ШІ використовується для створення персонажів і сцен. У рекламі – для швидкого створення яскравих відео.

У навчанні – для візуалізації складних тем. У медицині – для створення 3D-моделей органів.



Fi

OpenAI
Sora

ДОВІДКА

Важливо перевіряти джерела інформації та не плутати згенероване штучним інтелектом із реальністю. Зображення, створені за допомогою ШІ, можуть бути нереалістичними чи містити неточності.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Генерація музики

ШІ може створювати музику, використовуючи різні методи: алгоритми машинного навчання, генеративні моделі та інші обчислювальні методи. Система штучного інтелекту аналізує наявні музичні патерни, вивчає, як побудовані мелодії, акорди, ритми, та генерує нові композиції на їх основі. Ви задаєте стиль або настрій, наприклад: «весела електронна музика». ШІ створює нову мелодію. Таку музику вже використовують у відеоіграх – для створення фонової музики, у кіно та рекламі – для швидкого створення саундтреків, у застосунках для медитації – для генерації спокійних звуків, у музичних експериментах – коли люди створюють пісні разом із ШІ.

Згенерована музика може бути несподіваною або незвичною.

ШІ не має емоцій, але може імітувати настрій у музиці.

Дотримуйтесь авторських прав, якщо використовуєте музику в публічних проєктах.

FlexClip

Riffusion



Якість та оригінальність музики, створеної штучним інтелектом, все ще залишається предметом дискусій, багато експертів вважають, що людська творчість і музичне самовираження не можуть бути повністю відтворені машинами.

Ігри стають «розумнішими»

ШІ вже багато років використовується в іграх для покращення ігрового досвіду та вдосконалення ігрового дизайну. Сьогодні еволюція систем машинного навчання докорінно змінює ігрову індустрію. ШІ покращив візуальну якість, і тепер ігрові середовища стали більш реалістичними, ніж будь-коли раніше. Персонажі в іграх рухаються та поводяться природно.

Приклади застосування ШІ в іграх

Стратегія в реальному часі
ШІ дає змогу створювати більш інтелектуальні та динамічні ігрові середовища, наприклад, для інтелектуального управління ресурсами та адаптивного рівня складності

Процедурна генерація контенту
Алгоритми ШІ можуть створювати ігровий контент, як-от рівні, ландшафти або випадкові зустрічі, на основі попередньо заданих правил і поведінки гравця

Персоналізація
ШІ може адаптувати ігровий досвід до конкретних гравців, з урахуванням їхнього стилю гри, рівня навичок і вподобань

Поведінка неігрових персонажів (NPC)
ШІ часто використовується для створення більш реалістичної поведінки NPC, наприклад більш інтелектуального супротивника, що ускладнює гру для гравця

Обробка природної мови
ШІ можна використовувати для розпізнавання голосу та обробки природної мови, що дає змогу гравцям взаємодіяти з NPC та ігровим світом у більш захопливий і реалістичний спосіб

Прогнозування погоди

ШІ може надавати більш точну та актуальну інформацію про погоду, ніж традиційні методи, аналізуючи великі обсяги даних з різних джерел, як-от супутники, радари та метеостанції. Алгоритми ШІ здатні створювати високодеталізовані та локалізовані прогнози погоди, враховуючи конкретні географічні та екологічні фактори. Більш швидкі та часті оновлення можуть допомогти заздалегідь попередити про складні погодні умови та покращити реагування і планування в надзвичайній ситуації. Системи прогнозування погоди на основі ШІ можуть автоматизувати багато ручних завдань, надаючи можливість метеорологам зосередитися на аналізі та інтерпретації.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.



Вплив штучного інтелекту на майбутній ринок праці

Використання ШІ має потенціал для автоматизації певних завдань, але малоймовірно, що ШІ повністю замінить людей на робочому місці. Хоча ШІ може виконувати завдання швидше, ефективніше та без втоми, йому бракує креативності, емоційного інтелекту та навичок міжособистісного спілкування, притаманних людині.

Зокрема, багато робіт вимагають людського дотику та особистої взаємодії, яку ШІ поки не може забезпечити. Тож імовірніше, що ШІ буде доповнювати та покращувати людську роботу, а не замінить її повністю. Проте важливо підготуватися до змін на ринку праці, розвиваючи навички, які доповнюють можливості ШІ: критичне мислення, розв'язання проблем і комунікативні навички.

Етика штучного інтелекту

Разом із користю виникають і етичні питання – питання про те, що є правильним або неправильним у використанні ШІ.

Це охоплює такі теми, як захист приватності, запобігання упередженості в алгоритмах, відповідальність за рішення, які приймає ШІ, дотримання прав людини. Важливо не лише розвивати технології, а й установлювати чіткі моральні й правові межі їх застосування.

Приватність

ШІ збирає та накопичує велику кількість інформації про користувачів. Наприклад, історія місць перебування, пошукові запити в інтернеті або вміст ваших повідомлень. Це може порушувати ваше право на особисте життя.

Упередженість

Якщо ШІ навчається на неправильних або упереджених даних, він може приймати несправедливі рішення. Наприклад, нечесно оцінювати людей за кольором шкіри або статтю.

Безпека

Якщо ШІ працює неправильно або його зламають хакери, це може бути небезпечно. Наприклад, у самокерованих автомобілях або в лікарнях.

Відповідальність

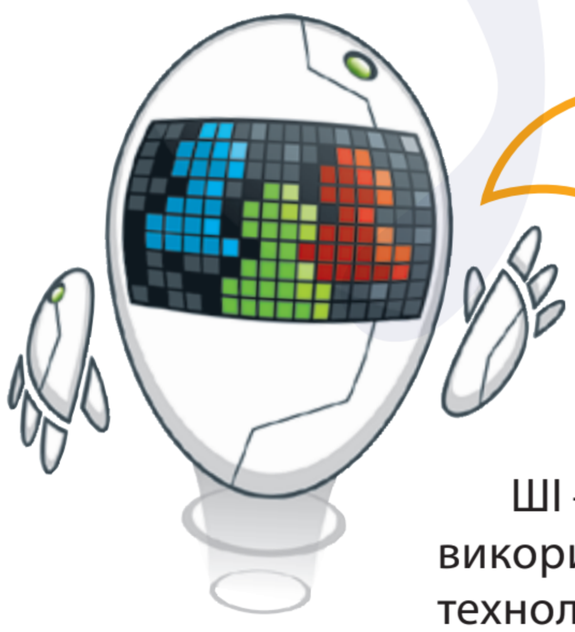
Якщо ШІ зробив помилку – хто винен? Програміст чи компанія? Це складне питання, яке ще обговорюють у світі.

Витіснення з роботи

З розвитком і вдосконаленням ШІ зростає занепокоєння, що багато професій можуть зникнути, а це призведе до втрати роботи для великої кількості людей та економічної нестабільності.

Права людини

Застосування ШІ у військовій сфері та правоохоронних органах може порушувати основні права та свободи громадян, особливо якщо не дотримуватись чітких етичних правил і контролю.



Що ви можете робити?

- *Створювати ШІ, який поважає права людини.*
- *Використовувати правдиві та різноманітні дані для навчання.*
- *Забезпечувати прозорість – щоб люди розуміли, як працює ШІ.*
- *Завжди залишати контроль за людиною.*

ШІ – потужний інструмент, який може допомагати людям. Але дуже важливо використовувати його відповідально. Етика ШІ націлена на те, щоб зробити технології безпечними, справедливими та корисними для всіх.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Діпфейк

Діпфейк (від англ. deepfake) – це відео, фото або аудіо, які на вигляд як справжні, але насправді створені за допомогою інструментів ШІ. Наприклад, можна зробити відео, де відома людина каже щось, чого вона ніколи не говорила, створити розмову, якої ніколи не існувало. Це підриває довіру суспільства.

Діпфейки можуть мати дуже реалістичний вигляд. Люди здатні повірити у фальшиві новини, неправдиві заяви чи вигадані події.

Діпфейки можуть використовувати для шантажу, булінгу та навіть впливати на вибори в країнах. На жаль, тому існує багато прикладів.

Якщо ми не можемо відрізнити правду від фейку, то починаємо менше довіряти відео, новинам і навіть одне одному.

Діпфейки – це приклад того, як технології можуть бути небезпечними, якщо їх використовують неправильно. Тому важливо вміти розпізнавати фейки, щоб не стати жертвою обману.

Перевіряйте джерела інформації! Не вірте одразу всьому, що бачите в інтернеті. Використовуйте фактчекінг. Є сайти, які перевіряють, чи правда це. Будьте обережними в соцмережах! Не поширюйте підозрілий контент.



Майбутнє ШІ

Існує багато потенційних можливостей застосувань ШІ, які нині досліджуються, щоб стати реальністю в найближчому майбутньому.

Застосування штучного інтелекту в майбутньому

Охорона здоров'я	ШІ має потенціал для революції у сфері охорони здоров'я, покращуючи діагностику, пошук та відкриття нових ліків, персоналізацію планів лікування
Транспорт	Очікується, що ШІ відіграватиме ключову роль у майбутньому транспорту, наприклад, в автономних транспортних засобах, системах керування дорожнім рухом та оптимізації маршрутів перевезень
Освіта	ШІ може допомогти у створенні індивідуальних навчальних програм для учнів / учениць і студентів / студенток, забезпечити роботу віртуальних репетиторів
Фінанси	ШІ можна використовувати для автоматизації фінансових процесів, виявлення шахрайства та надання інвестиційних рекомендацій
Довкілля	ШІ може допомогти відстежувати й аналізувати зміни клімату, а також розробляти нові стратегії для зменшення їх наслідків
Енергетика	ШІ має потенціал оптимізувати споживання енергії та навіть сприяти появі нових, більш ефективних джерел енергії
Виробництво	ШІ можна використовувати для оптимізації виробничих процесів, автоматизації рутинних завдань і покращення контролю якості
Розваги	ШІ можна застосовувати для створення персоналізованих розваг і розробки нових форм інтерактивних медіа

Практичне завдання

Завантажте файл-заготовку зі сторінки інтернет-підтримки. Виконайте запропоноване завдання.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Тема 4
Цифрові пристрої та здоров'я

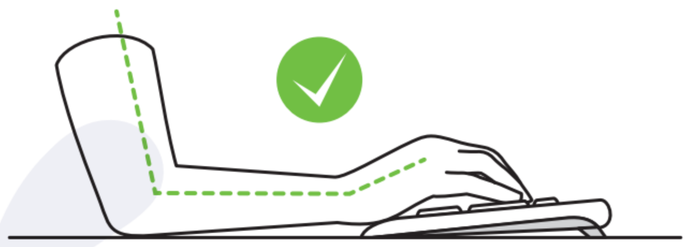
Сьогодні ми щодня користуємося комп’ютерами, планшетами, телефонами та іншими цифровими пристроями. Вони допомагають навчатися, спілкуватися, грати в ігри, дізнаватися нове. Але важливо пам’ятати: надмірне використання техніки може впливати на наше здоров’я.

Кількість часу, який люди проводять перед моніторами, постійно зростає. Це, своєю чергою, спричиняє нові проблеми зі здоров’ям. Тривале користування комп’ютером, неправильна поза під час сидіння, хибно встановлений рівень яскравості на частоти оновлення екрана, а також неправильне положення рук на клавіатурі та мишці можуть викликати різноманітні захворювання.

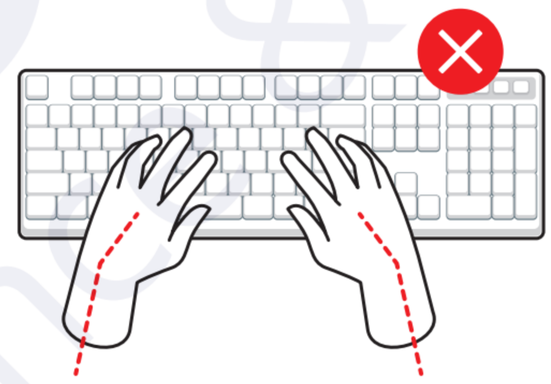
Серед найпоширеніших симптомів – біль у м’язах і суглобах, проблеми із зором і синдром зап’ястного каналу. Ці недуги часто виникають у повсякденному житті через неправильне та надмірне використання комп’ютерів.











Синдром комп'ютерного зору

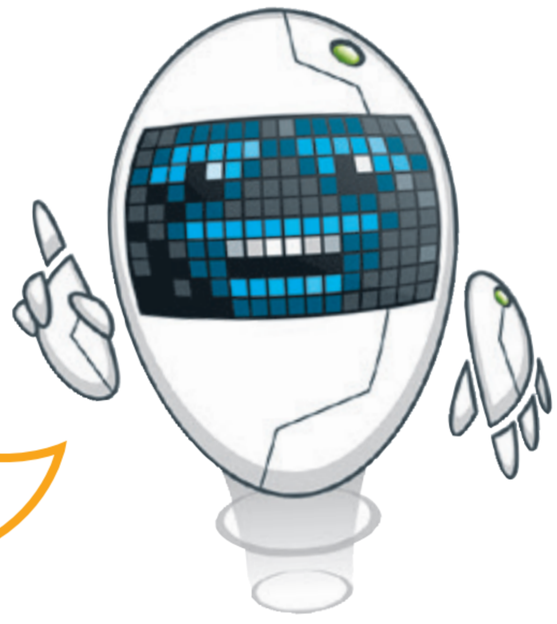
Читання з екрана зменшує частоту моргання, що втомлює очі, викликає сухість, а в окремих випадках – розмиття зору. Людське око за своєю природою краще фокусується на об'єктах, розташованим на відстані понад шість метрів, тому робóта, що виконується зблизька, створює додаткове навантаження на очні м'язи. Використання комп’ютерів та інших пристроїв вимагає короткозорого або проміжного зору, що також може спричинити погіршення зору. Усі ці симптоми об’єднуються під назвою **зоровий комп’ютерний синдром (ЗКС).**

ЗКС – це стан, пов'язаний з розвитком сучасних технологій. Найпоширеніші симптоми: головний біль, біль або сухість в очах, нудота. Симптоми особливо виражені у людей з короткозорістю, астигматизмом або пресбіопією.

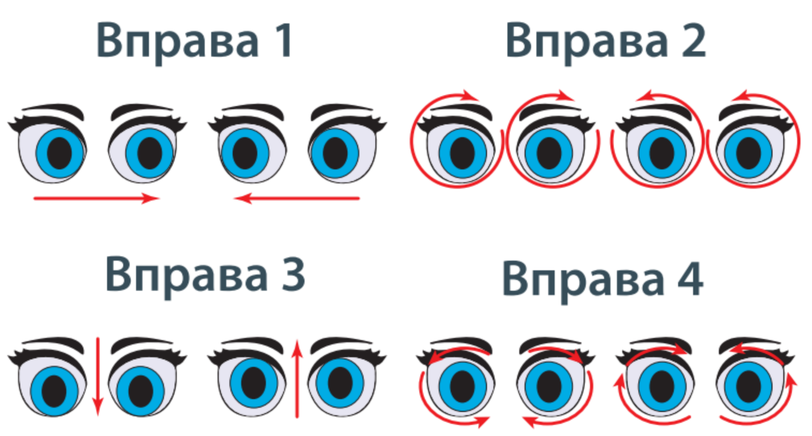
Як зменшити вплив ЗКС або запобігти його виникненню:

- *робіть часті перерви під час роботи за комп’ютером;*
- *використовуйте правило 20-20-20: кожні 20 хвилин дивіться на об’єкт на відстані 20 кроків протягом 20 секунд;*
- *освітлення в кімнаті має бути втричі яскравішим за екран;*
- *уникайте відбиття світла на екрані від вікон або ламп;*
- *розміщуйте екран трохи нижче від рівня очей.*

Кожні 20 хвилин робіть 20 секундний відпочинок і дивіться на щось на відстані більш ніж 20 кроків.



ГІМНАСТИКА ДЛЯ ОЧЕЙ



Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Проблеми, пов'язані з порушенням постави

Тривале використання комп'ютера може збільшити ризик виникнення травм. Неправильна постава під час сидіння за комп'ютером може спричинити біль у м'язах і суглобах, біль у спині, оніміння в плечах, руках, зап'ястях або кистях, а також перенапруження очей, як уже згадувалося вище.

Сидіння в неправильній або одній і тій самій позі протягом тривалого часу викликає фізичну втому через зменшення кровообігу в м'язах. Підвищена напруга м'язів, біль у шиї та потилиці може викликати головний біль.

Щоб уникнути травм, робоче місце має бути обладнане належним чином:

- *стілець повинен підтримувати спину;*
- *ступні мають стояти на підлозі або на підставці;*
- *монітор має бути розташований у правильному положенні – трохи нижче від рівня очей.*



Правильне використання смартфонів і планшетів

Надмірне використання планшетів і смартфонів негативно впливає на поставу та біомеханіку тіла. Багато людей тримають ці пристрої на рівні грудей і нахиляють голову вперед і вниз, щоб дивитися на екран.

Це створює втричі більше навантаження на м'язи шиї, ніж настільні комп'ютери. Це, у свою чергу, призводить до болю в шиї, верхній частині спини, плечах і руках.

Зверніть увагу, що використання планшетів, смартфонів та інших світлових екранів у вечірній час збільшує ризик безсоння.

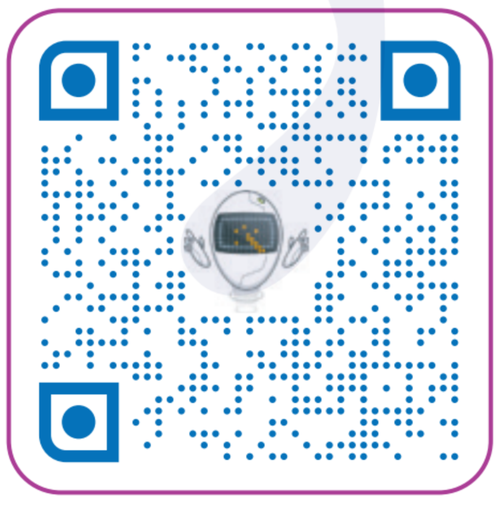
Правильний спосіб тримати смартфони та планшети *– піднімати пристрій досить високо, щоб голова не була нахилена вперед, а дивилася вгору.*



Практичне завдання

Завантажте файл-заготовку зі сторінки інтернет-підтримки. Виконайте запропоноване завдання.

https://formula.education/online-ict-9-klas/





Інші платформи



Проєкт



Підсумки



Глосарій

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

2. Дані, інформація, безпека



У цьому розділі ви ознайомитеся з поняттями даних та інформації.

Ви дізнаєтесь, як кодуються дані, та отримаєте уявлення про методи шифрування.

Також ви дослідите атрибути, які визначають якість інформації, і навчитеся перевіряти достовірність даних.

У результаті ви станете більш обізнаними у світі інформаційних технологій та зможете ефективно використовувати дані у своїй навчальній діяльності.

У цьому розділі ви навчитеся:

- відрізняти дані від інформації;
- присвоювати коди даним;
- розрізняти типи даних;
- перевіряти атрибути якості;
- кодувати та декодувати інформацію;
- більш глибоко розуміти роль інформації в нашому житті.

У цьому розділі ви дізнаєтеся:

- у чому різниця між даними, інформацією та знаннями;
- які існують види даних;
- як кодують дані;
- які бувають методи шифрування;
- що таке якість інформації;
- які атрибути визначають якість інформації;
- які бувають типи даних;
- що таке великі дані;
- які існують сучасні технології.

Програмне забезпечення:

- Microsoft Windows;
- Google Chrome;
- Microsoft Edge.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Тема 1
Дані, інформація, знання

Дані та інформація

Щодня нас оточують дані.

Ці дані надходять із телебачення, газет, книг, інтернету. Але чим дані відрізняються від інформації?

Дані *– це факти або елементи, які були зібрані, але ще не оброблені чи не організовані.*

Наприклад, список чисел або імен – це дані. Нижче наведено приклад даних – набір персональних даних учня.

Оленка
вул. Ясна, 56
212 500 2020
olenka@digital-kids.com
05.06.2011

Інформація *– це дані, які впорядковані або оброблені так, щоб вони набули змісту чи стали корисними.*

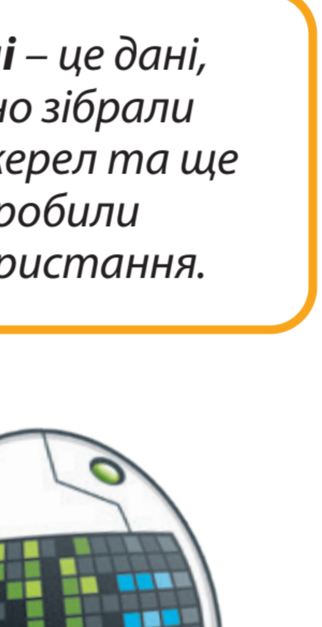
Наприклад, коли дані впорядковані або пояснені, вони перетворюються на інформацію.

Так, студентський квиток містить упорядковану інформацію про студента.

На цьому студентському квитку ви можете побачити таку інформацію: ім'я, домашню адресу, телефон, електронну пошту, дату народження.

УЧНІВСЬКИЙ КВИТОК №214

Ім'я	Оленка
Домашня адреса	вул. Ясна, 56
Телефон	212 500 2020
Email	olenka@digital-kids.com
Дата народження	05.06.2011



EDUCATION SCHOOL

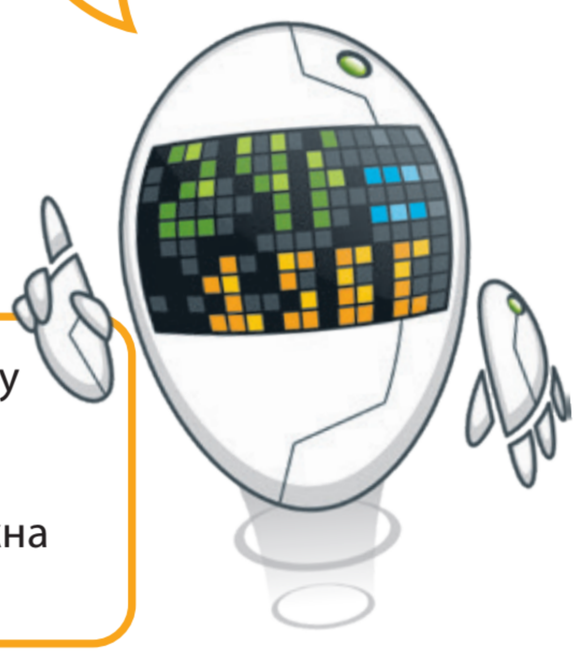
Сирі дані та інформація

Після того, як сирі дані проаналізовані, вони перетворюються на інформацію. Розглянемо декілька прикладів.

Число «**1752008**» вважається необробленими, сирими даними, оскільки це число не має жодного контекстного значення. Якщо ж це число представити як «17/5/2008 – ваша дата народження», то це вже інформація, оскільки вона надає знання про певний факт.

Результати проходження тесту кожного учня чи учениці – це окремі дані. Середній бал класу або всієї школи – це інформація, яка отримана із цих даних.

Сирі дані *– це дані, які щойно зібрали з різних джерел та ще не обробили для використання.*



ІСТОРІЯ

Термін «дані» походить від однини латинського слова datum, яке спочатку означало «щось дане». У латині datum – це однина, а data – множина. Його перше використання датується 1600-ми роками. Згодом «datum» в однині вийшло з ужитку, і зараз вживається лише слово «data», яке можна трактувати що в однині, так і в множині.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Інформація для подальшої обробки

Інформація з'являється в результаті аналізу даних. Коли ми пояснюємо або організовуємо необроблені дані, вони перетворюються на інформацію. Іноді інформація з одного аналізу може бути використана як дані для інших досліджень або обробки.

Наприклад, збір значень температури може надати впорядковану інформацію про історію температурних показників по всьому світу за останні 100 років. Подумайте, ці дані можуть бути організовані та проаналізовані, щоб довести – середня глобальна температура зростає.

Різниця між даними та інформацією

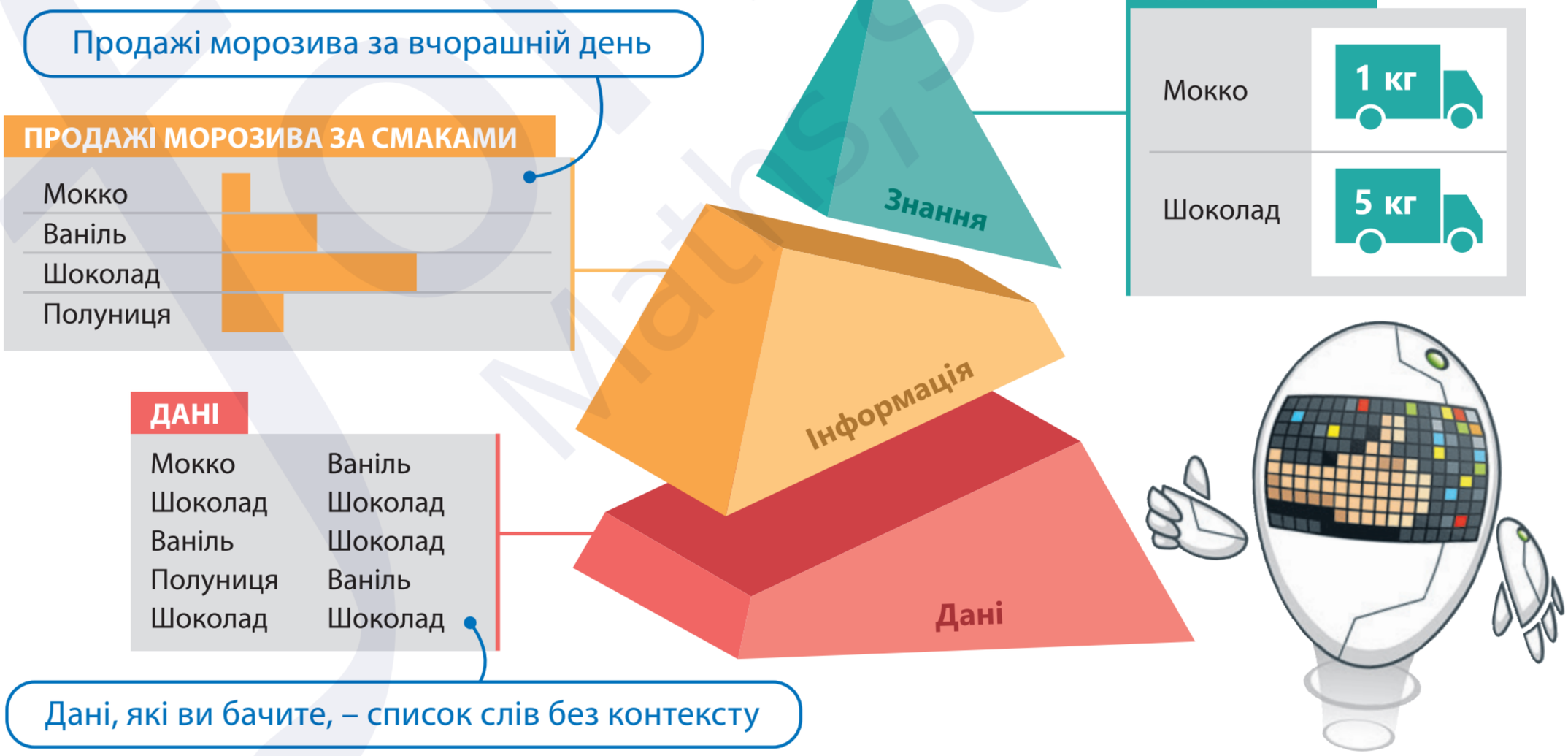
Дані	Інформація
Не мають ніякого сенсу	Має логічне значення
Неструктуровані	Має логічну структуру
Подані у вигляді чисел, фігур або статистичних даних	Подано у вигляді звітів, графіків або діаграм
Не мають залежності ні від чого	Залежить від даних
Введені користувачем або зібрані комп'ютерною системою	Отримана в результаті обробки даних

Знання

Можна сказати, що коли людина розуміє певну інформацію про щось, вона має знання з цієї теми. Інформація перетворюється на знання, коли застосовується критичне мислення, оцінка, структура або організація.

Розгляньмо приклад.

Знання *– це наше розуміння світу. Іншими словами, це відповідний збір інформації таким способом, що робить її корисною.*

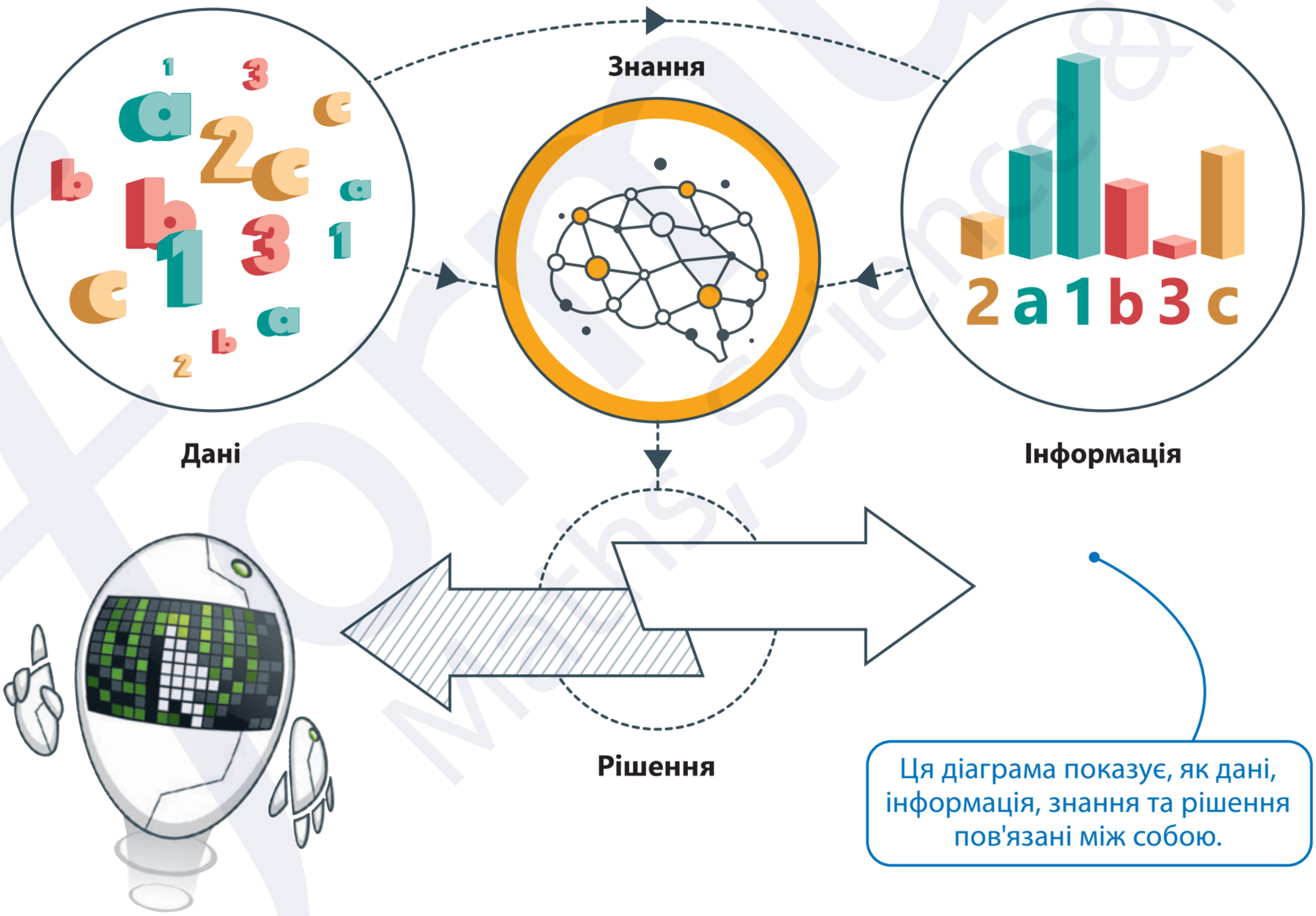


Тепер, коли ми організували ці дані, можна отримати інформацію: список продажів смаків морозива за вчорашній день. Щоб отримати більше інформації, варто провести невеликий аналіз. Наприклад, найпопулярнішим смаком морозива, проданого вчора, було шоколадне. Знання полягає в тому, що менеджер магазину побачить, що шоколад – найпопулярніший смак морозива. Наступного разу, коли він робитиме замовлення, то замовить у п'ять разів більше шоколадного морозива, ніж морозива зі смаком мокко.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Відмінності між інформацією та знаннями

	Інформація	Знання
Значення	Вдосконалена форма оброблених даних	Релевантна інформація, яка спонукає до висновків
Прогнозованість	Недостатньо для здійснення прогнозів	Уможливлює прогнозування або прийняття рішення
Передача	Може бути легко передана усно, письмово або за допомогою електронних засобів	Вимагає вивчення предмета
Результат	Результатом є розуміння	Результатом є осмислення
Мета	Відповіді на запитання: хто, коли, що і де	Відповіді на запитання: як і чому



Практичне завдання

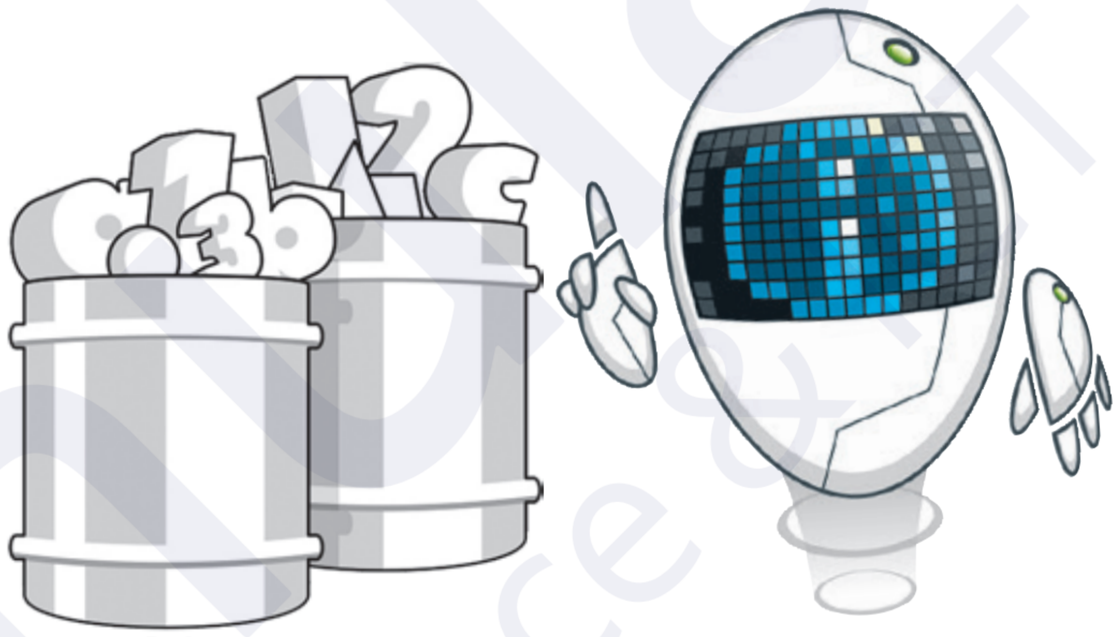
Завантажте файл-заготовку зі сторінки інтернет-підтримки. Виконайте запропоноване завдання.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Тема 2
Збір даних

Форми даних

Вам уже відомо, що дані складаються з необроблених фактів і цифр. Більш того, дані надходять у різних формах. Наприклад, даними, які можна зібрати, може бути кількість фільмів, які подивилася людина, колір очей, як людина ставиться до певної теми й багато іншого. Хоча дані, як правило, буквено-цифрові (текст, цифри та символи), вони також можуть складатися із зображень, відео чи звуку.



Отримання даних

Найважливішим етапом будь-якого проєкта є етап збору даних, тобто процес збору фактів, цифр і слів про цільові змінні. Збір даних може здійснюватися за допомогою різних пристроїв, як-от датчики, реєстратори даних. Цей процес вимагає глибокого розуміння параметрів дослідження, а також планування і старанної робóти для отримання якісних даних. Вони сприяють проведенню належного аналізу й подальшого отримання значущої інформації щодо досліджуваного явища.

Методи збору даних залежать від типу даних, але завжди важливо перевіряти етапи збору даних, щоб забезпечити їхню точність та достовірність.

Джерела даних

Джерела даних можуть бути первинними чи вторинними.

Джерела первинних даних

Первинне джерело даних містить дані, які були зібрані безпосередньо вручну. Первинним джерелом даних можуть бути дані, зібрані з датчиків, реєстраторів даних або навіть з анкет. Наприклад, датчик температури, який збирає дані про температуру повітря, вважається первинним джерелом даних. Інший приклад – датчик швидкості вітру (анемометр), який вимірює швидкість вітру. Анкета, яку роздають клієнтам щодо того, якій погоді вони віддають перевагу під час закордонних подорожей, також є первинним джерелом даних.

Анемометр – датчик швидкості вітру, є первинним джерелом даних. Вітер приводить в дію електричний генератор. Вихідний сигнал генератора надходить до електричного лічильника, відкаліброваного для вимірювання швидкості вітру.



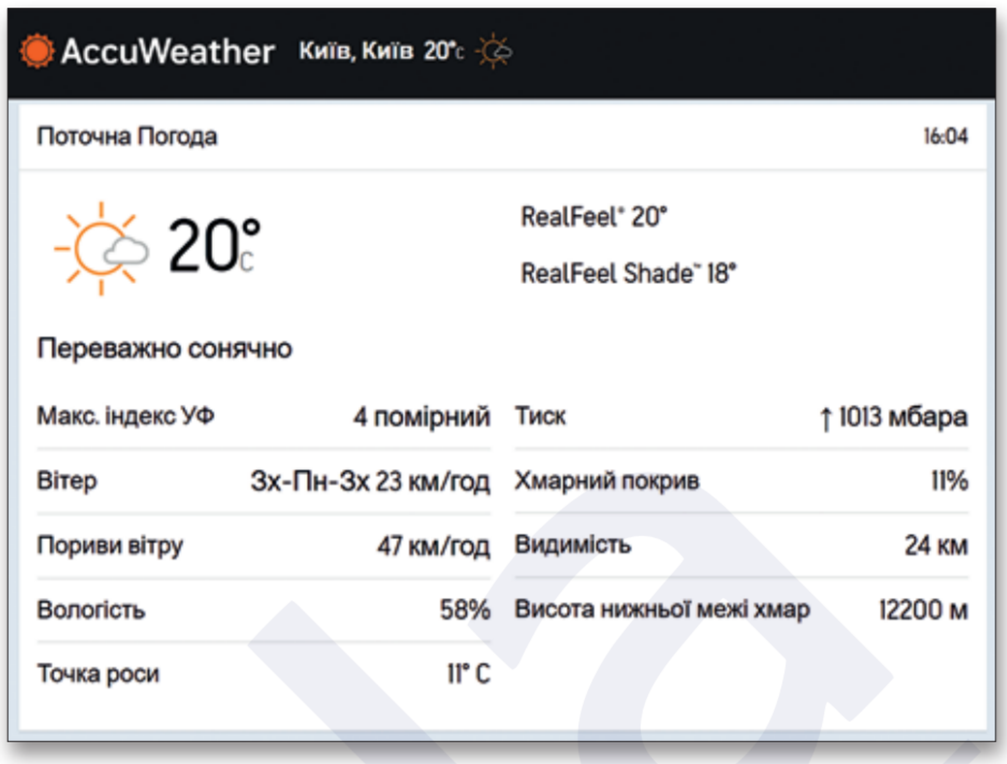
ДОВІДКА

Перевага первинних даних полягає в тому, що вони спеціально адаптовані до потреб аналізу. Іноді буває неможливо зібрати всі дані, тому беруть випадкову вибірку, яка представляє всі дані.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Вторинні джерела даних

Вторинні дані створюються, коли ми використовуємо первинне джерело даних для отримання інших даних. Наприклад, можна використовувати дані про температуру повітря та швидкість вітру з двох різних датчиків, щоб отримати дані для іншого параметра, який називається вітро-холодовий індекс. Спочатку використовують датчики температури і вітру як первинні джерела даних для збору даних про температуру та швидкість вітру, а потім дослідник може використати формулу вітро-холодового індексу як вторинне джерело даних для отримання цієї інформації.



Відмінності між первинними та вторинними джерелами даних

	Первинне джерело даних	Вторинне джерело даних
Оригінальність	Зібрані безпосередньо з першоджерел	Не оригінальні дані, тому що хтось їх уже зібрав
Форма	Перебувають у сирому та неорганізованому вигляді	Перебувають у впорядкованій та обробленій формі
Точність	Точніші, оскільки є актуальними	Менш точні, оскільки відображають минулі події
Джерело	Збирані за допомогою датчиків, опитувань, інтерв'ю, експериментів тощо	Отримані з книг, журналів, документів, вебсайтів, блогів тощо
Економічна ефективність	Дорога й трудомістка процедура	Менш витратна процедура і менш трудомістка за часом

Внутрішні та зовнішні джерела даних

Джерела даних можна поділити на внутрішні та зовнішні. Внутрішні джерела даних відображають ті дані, які перебувають під контролем установи, тоді як зовнішні дані – це будь-які дані, створені за межами організації. Наприклад, дані, зібрані з датчика, який належить університету або науковій установі, вважаються внутрішніми даними, тоді як дані, зібрані від інших установ, осіб або з джерел за межами цього університету, вважаються зовнішніми даними стосовно цього університету.



Вторинні джерела

Внутрішні джерела	Зовнішні джерела
Попередні маркетингові дослідження	Урядові джерела
Фінансові звіти	Інтернет
Дані про обсяги продаж	Корпоративна звітність
Інвентаризація запасів	Університети

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Якість інформації

Коли дані обробляються, упорядковуються, структуруються або представляються в певному контексті так, щоб зробити їх корисними, вони називаються **інформацією**. Цінність інформації для певного використання визначається терміном **якість інформації**. Це важливий фактор, який показує, наскільки інформацію можна використовувати для прийняття рішень. Зі зростанням обсягів збору та збереження даних якість інформації, отриманої в результаті їх обробки, набуває великого значення. Забезпечення високої якості інформації допомагає точно визначити реальні потреби для реалізації проєктів, ефективно надавати послуги та підвищити продуктивність кожного робочого дня. Натомість неточна інформація може спричинити порушення процесів, знизити ефективність і призвести до затримок у виконанні проєктів. Якість інформації можна перевірити за певними критеріями, які називаються **атрибутами якості**.

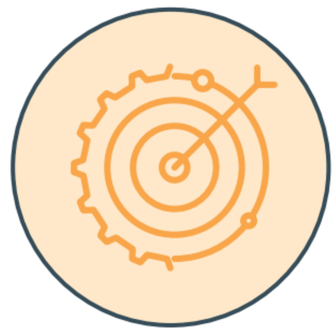


Перевірка атрибутів якості

Перш ніж почати збирати інформацію через інтернет-джерела та перед тим як почати обробляти її, треба переконатися в надійності інформації, яку пропонують нам на вебсайті. Важливість перевірки п'яти атрибутів якості інформації полягає в тому, що вони допомагають перевірити надійність інформації, яку збираємося отримати.

Якщо інформація не є надійною, ми не зможемо здобути якісні знання.

Точність



Інформація повинна бути точною, щоб вважатися якісною. Що вища точність інформації, то краща її якість. Арифметичні та граматичні помилки є найпоширенішими причинами неточної інформації.

Ось деякі запитання, які допоможуть перевірити, чи є інформація точною

Чи можна перевірити факти, статистику або іншу інформацію з інших джерел?
Чи можна повторити експеримент і чи дасть він такі ж результати?
Чи відомо, що автор інформації є експертом у цій сфері?
Чи зазначив автор свої джерела?
З огляду на ваші знання, чи здається інформація точною?
Чи не містить інформація неправильно написаних слів, пропущених чи неправильно розміщених символів та чи правильно оформлені цитати?

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.



Доцільність

Інформація повинна бути пов'язана з вашою темою або питанням дослідження. Якщо є додаткова чи невідповідна інформація, користувач буде змушений витратити час на пошук даних, щоб знайти те, що потрібно. Що більше інформація відповідає потребам одержувача, то вища її якість.

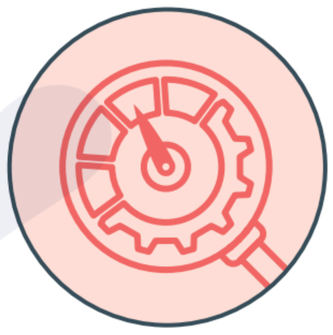
Актуальність



Дата публікації або створення інформації свідчить про те, наскільки вона актуальна й наскільки сучасна тема, яку ви досліджуєте. Завжди треба перевіряти, що у вас є найновіша інформація і вона отримана з оригінального дослідження.

Важливі кроки для хронології інформації

Перевірте дати використаних джерел
Перевірте історію ключових слів стосовно інтелектуальних прав, як-от зареєстровані торгові марки, авторські права, патенти та комерційні таємниці
Перевірте історію переглядів або редагування інформації
Перевірте дату публікації



Стислість

Якість інформації також визначається рівнем деталізації. Якщо ви надаєте занадто мало або занадто багато інформації, ви знижуєте її якість. Надання дуже багатьох деталей робить її перевантаженою і може ускладнити пошук потрібної інформації, тоді як недостатня кількість даних ускладнює її розуміння. Правильний обсяг інформації є ключовим атрибутом якості.

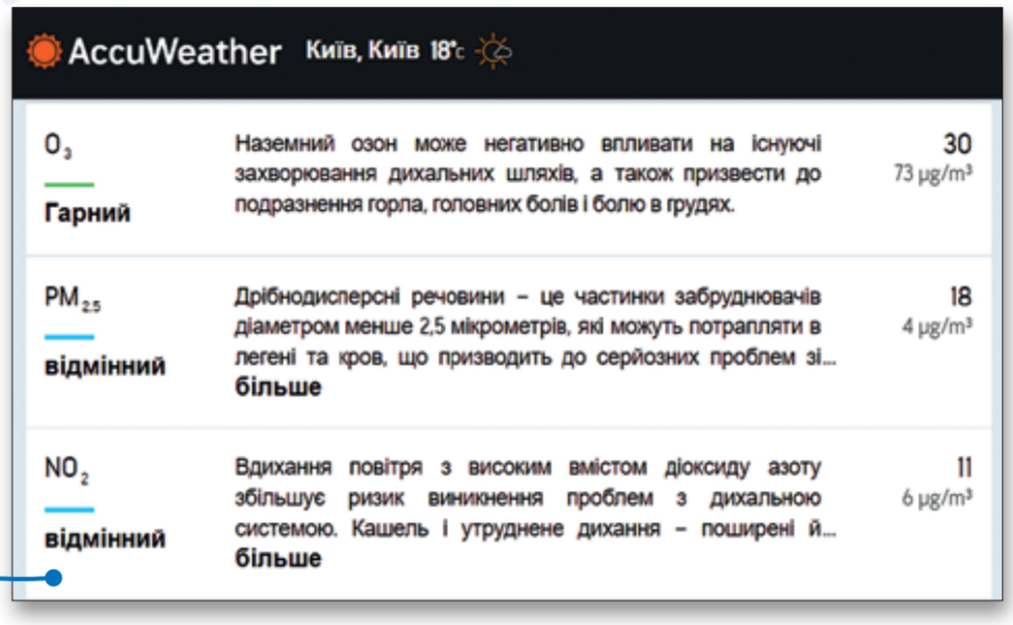
Повнота



Це міра повноти, яка необхідна для того, щоб надана інформація давала повну картину реальності, а не лише її частину. Відсутність усієї потрібної інформації означає, що ви не зможете приймати правильні рішення, а інформація не може бути використана належним чином, що також означає її низьку якість.



Цей вебсайт є прикладом джерела інформації, яка відповідає п'яти специфічним атрибутам якості.



Практичне завдання

Завантажте файл-заготовку зі сторінки інтернет-підтримки. Виконайте запропоноване завдання.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Тема 3
Типи даних і кодування

Ви вже з'ясували, що таке дані, розглянемо їх типи. Дані можуть мати різні форми, наприклад, кількість відвідувачів заходу, тривалість цього заходу тощо. У науковому дослідженні використовуються переважно два типи даних: **числа** та **категорії**.



Числові дані

Числові дані складаються з вимірюваних фактів. Кількість подій, що відбулися в місті, – приклад числових даних. Вони можуть бути **дискретними** та **безперервними**.

Дискретні дані

Дискретні дані представляють вимірювані елементи й можуть приймати тільки певні значення – це кількісні натуральні числа (0, 1, 2, 3...), наприклад кількість учнів у класі.

Безперервні дані

Безперервні дані являють собою вимірювання даних і можуть приймати будь-яке значення на основі дійсних чисел, наприклад зріст людини.

Дані за категоріями

Це тип даних, який має нечислові значення. Такі дані належать до типу даних, які можуть бути збережені та ідентифіковані на основі імен або міток, наданих їм. Деякі приклади таких даних включають колір очей, стать, модель комп'ютера, місто. Ці дані можуть бути **номінальними** чи **порядковими**.

Номінальні дані

Номінальні дані – це дані, які використовуються для представлення категорій або маркування змінних без кількісного значення та без упорядкування. Наприклад, результати тесту можуть класифікуватися як «здав» або «не здав», дані про колір волосся членів сім'ї тощо.

Порядкові дані

Порядкові дані – це тип даних з певним порядком. Порядкові дані групують дані відповідно до якоїсь системи ранжування. Наприклад, результати тесту можна згрупувати за оцінками: A, B, C, D та E. Розмір футболки – S, M, L, XL.

Графічні, відео- та аудіодані

Хоча дані зазвичай мають буквено-цифрову форму (текст, цифри та символи), вони можуть складатися із зображень, аудіозаписів або відеокліпів.



Графічні дані
Графічні дані складаються з діаграм, графіків тощо. Наприклад, набір зображень визначних пам'яток певного регіону або графік кількості відвідувачів туристичного напрямку в країні.

Зображення та відеодані
Цифровим зображенням може бути фотографія чи ілюстрація, представлена пікселями або векторами. Відеодані складаються із серії рухомих зображень та аудіо, як-от телевізійна реклама туристичної кампанії, відео про Італійську Рив'єру, стрім з Таймс-сквер у Нью-Йорку чи онлайн-екскурсія Львовом тощо.

Аудіодані
Аудіодані складаються з різних звуків і звукових ефектів: іаудіокниги, записи подкастів або аудіогіди туристичними місцями Мукачева, Рима чи Парижа.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Статичні та динамічні дані

Інколи дані змінюються після того, як їх записали, а інколи – ні. Завдяки цьому дані можна охарактеризувати як **статичні** або **динамічні**.

Статичні дані

Статичні дані – це дані, які залишаються незмінними після запису.

Прикладами є номери рахунків-фактур, код товарів або дати народження. Ще один приклад – паперова газета: після друку інформація в ній більше не змінюється.

Динамічні дані

Динамічні дані – це дані, які можуть змінюватися з часом і потребують регулярного оновлення. Прикладом динамічних даних є інформація на вебсайті, оскільки вона може змінюватися, коли вебсайт оновлюють.

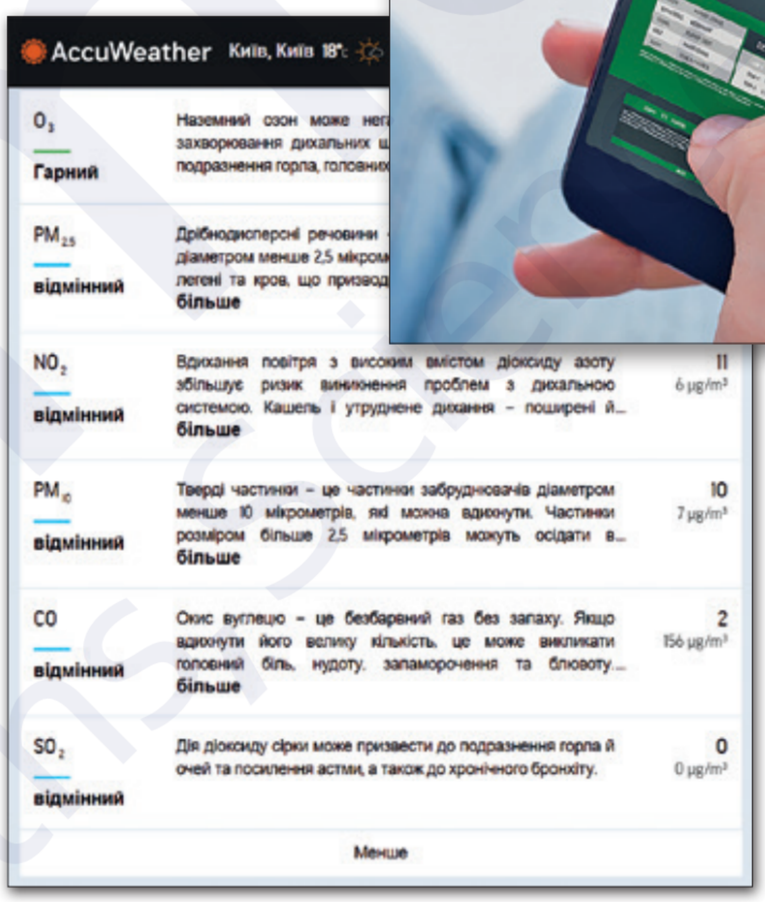
Інші приклади – результати спортивних змагань у реальному часі, розклади авіарейсів або прогнози погоди.

Вебсайти – динамічні джерела даних, адже користувачі можуть взаємодіяти з ним. Якщо ви хочете подивитися фільм у кіно, можна придбати квитки онлайн, але спочатку потрібно дізнатися про їх наявність.

Сайти належать до динамічних джерел даних, оскільки інформація на них може постійно оновлюватися. Наприклад, якщо спортивний вебсайт відображає рахунок футбольного матчу, він оновлюється після кожного забитого голу, забезпечуючи актуальну інформацію.







Кодування даних

Кодування даних передбачає використання різних кодів: чисел, літер або скорочень – для опису змісту й контексту речень, фраз або абзаців. Отже, дані можуть бути впорядковані певним способом, що полегшує їх аналіз і допомагає робити висновки. Розгляньмо деякі приклади з повсякденного життя, які використовуються для представлення даних.

Під час проведення експериментів або опитувань зібрана інформація часто має «сирий» вигляд і потребує упорядкування. Цей процес називають ***кодуванням даних.***

Коди аеропортів

Міжнародна асоціація повітряного транспорту (IATA) встановила трилітерний код, що визначає аеропорти по всьому світу. За допомогою цих кодів можна шукати авіаквитки онлайн. Ці ж коди відображаються на багажних бирках, прикріплених на стійках реєстрації в аеропортах, що допомагає уникнути втрати багажу.

Код аеропорту

Код	Назва аеропорту
JFK	Міжнародний аеропорт імені Джона Кеннеді
LHR	Лондонський аеропорт «Хітроу»
KBP	Міжнародний аеропорт «Бориспіль»

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.



Коди валют

Кожна країна світу має власну валюту, і для фінансових операцій використовують символи або коди валют замість повної назви валюти.

Код валюти

Код	Валюта
USD	Долар США
EUR	Євро
GBP	Фунт стерлінгів
UAH	Українська гривня



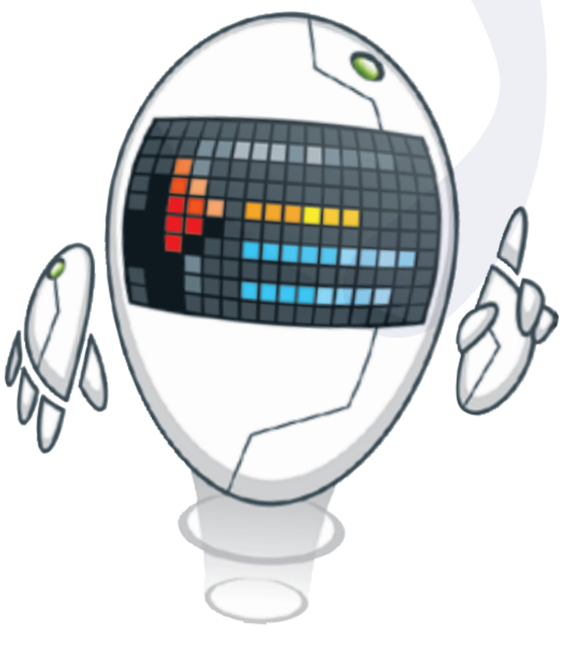
Переваги кодування даних

Введення даних відбувається швидше	Легше набрати «JFK», ніж повну назву «John F. Kennedy International Airport»
Займає менше місця	Складно написати повну назву країни на номерному знаку автомобіля чи на громадському транспорті (таксі, автобусах), але завдяки міжнародним кодам транспортних засобів це більше не проблема
Прискорює пошук даних	Кожен регіон має свій код. Його використовують для пошуку адреси за поштовим індексом, номером будинку. Ці ж коди застосовує пошта для полегшення доставки



Недоліки кодування даних

Неоднозначність значення даних	Подібні коди важко розрізнити
Кодування може бути складним для зрозуміння	Код може бути важко інтерпретувати або запам'ятати його значення
Коди, що використовуються, можуть бути надто складними	Наприклад, коли треба закодувати багато інформації, набору літер може бути недостатньо. Тоді додають числа або довгі номери, що ускладнює процес кодування – наприклад, при кодуванні товарів у магазинах





Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Штрихкоди



Ми бачимо штрихкоди щодня, наприклад, на електронних квитках, посилках, товарах у супермаркетах тощо.

Штрихкод – це етикетка з тонкими чорними лініями різної товщини та з різними комбінаціями цифр. Вони використовуються для впорядкування та індексації інформації або позначення цін на товари.

Перші шість цифр ідентифікують компанію, яка виробляє або продає товар.

Наступні цифри описують сам продукт.

Контрольна цифра перевіряє, що всі інші цифри правильно зчитуються сканером.





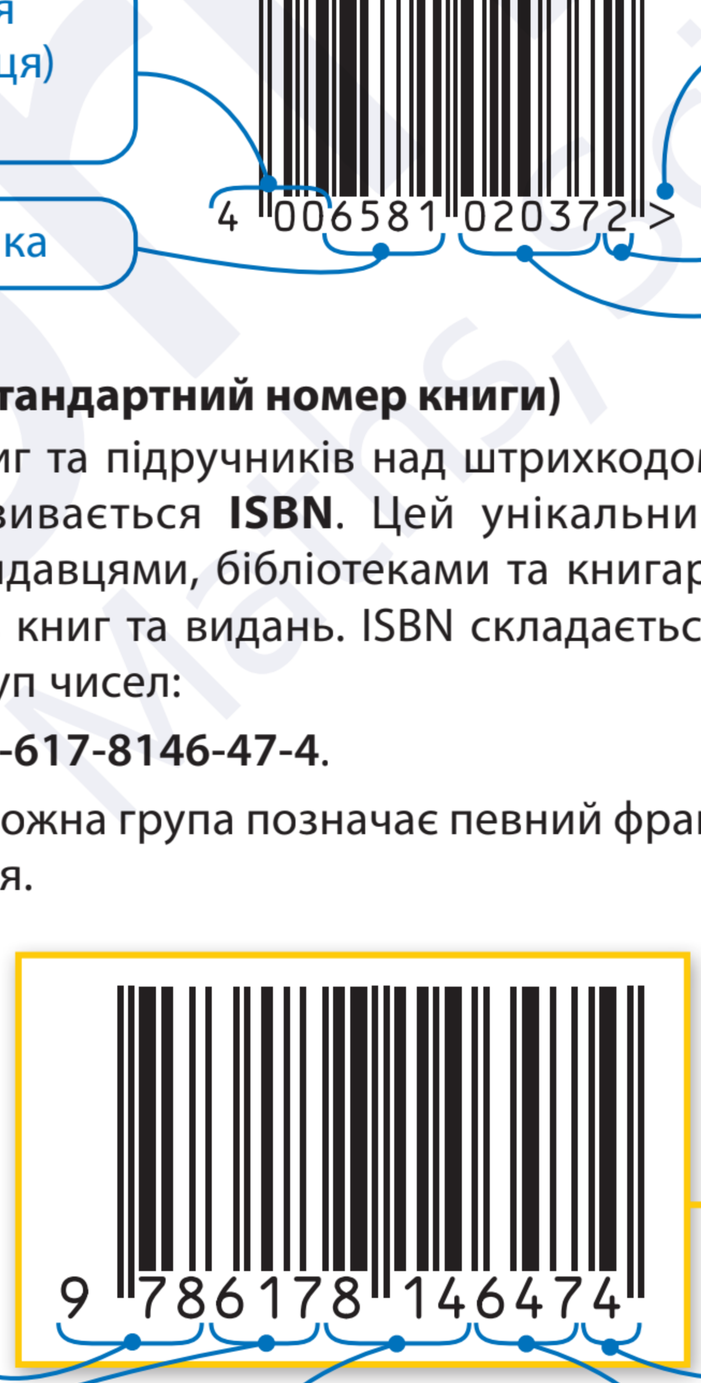
ISBN (Міжнародний стандартний номер книги)

На звороті більшості книг та підручників над штрихкодом указаний номер, який називається **ISBN**. Цей унікальний номер використовується видавцями, бібліотеками та книгарнями для ідентифікації назв книг та видань. ISBN складається із 13 цифр і ділиться на 5 груп чисел:

ISBN: 978-617-8146-47-4.

Ось приклад коду ISBN. Кожна група позначає певний фрагмент інформації про видання.





Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.



Друга група цифр ISBN містить номер, який указує на мову та країну, в якій була опублікована книга. Наприклад, у наведеному вище ISBN номер «617» показує, що книга була опублікована в Україні. Якщо книга опублікована в США, цей номер буде «0». Якщо в інших англомовних країнах – буде «1».

Коди для 2-ї групи елементів ISBN

Код	Країна / мова
0	США / Мова англійська
1	Мова англійська
2	Мова французька
3	Мова німецька
4	Японія
960	Греція
969	Пакистан
99969	Оман

Цей QR-код веде на сайт https://www.accuweather.com/



Дизайнери створюють QR-коди у вигляді витворів мистецтва, додаючи кольори, малюнки та навіть анімацію, зберігаючи при цьому функціональність.

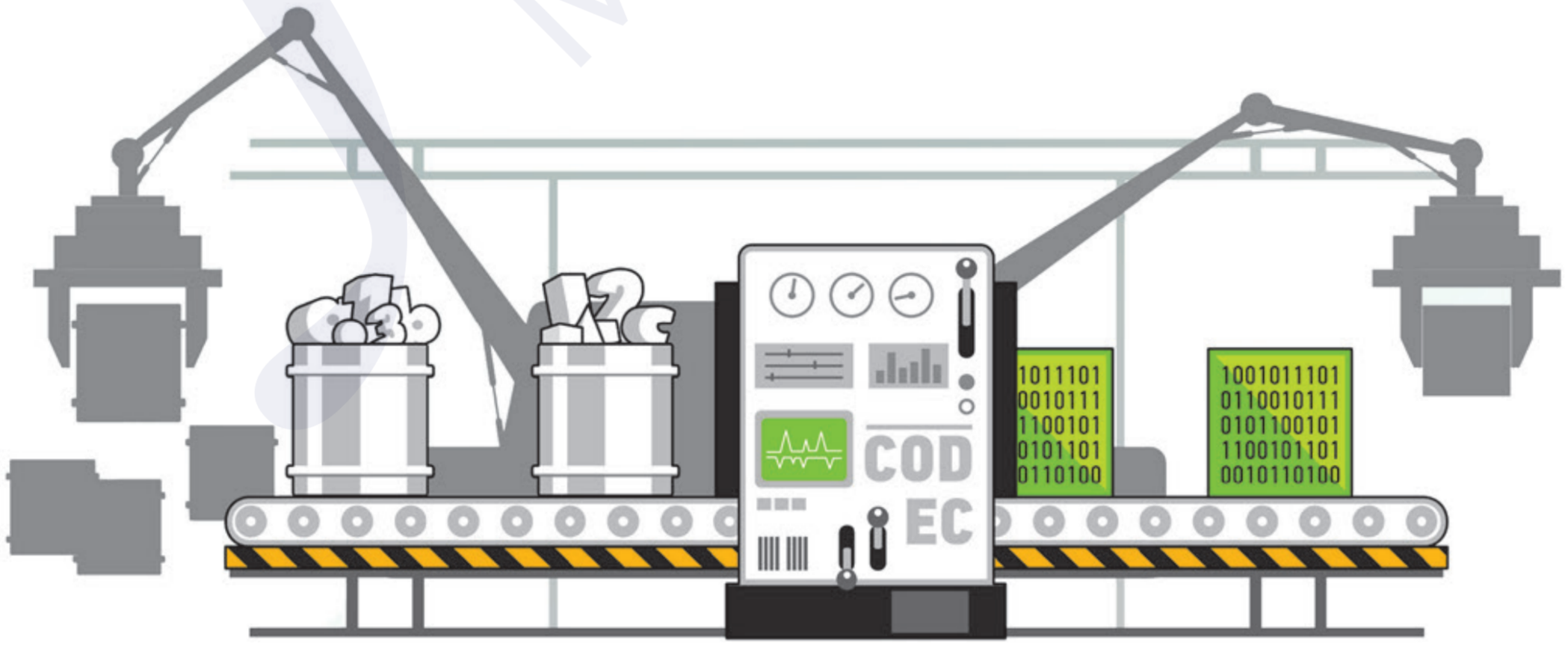
QR-коди

QR-код (Quick Response Code) – сучасна форма штрихкоду. Він складається із чорних квадратів, розташованих у сітці, і містить набагато більше інформації. QR-коди можуть мати посилання на вебсайти, відео чи файли. Достатньо просканувати ці коди за допомогою камери смартфона, щоб отримати доступ до інформації, яку вони містять.

Деякі штрихкоди друкуються з використанням ультрафіолетового чорнила, їх видно лише під UV-освітленням – це застосовується для безпеки та логістики.

Двійкове кодування даних

Кодування – це процес перетворення даних з однієї форми в іншу. Оскільки комп'ютери здатні зберігати та працювати лише з двійковими цифрами, нам потрібні способи перетворення різних типів даних – чисел, тексту, зображень, відео – у двійкову форму. Якщо дані не перетворені в двійкову форму, комп'ютер просто не зрозуміє їх і не зможе їх обробити. Програми, які використовуються для кодування даних, називаються **кодеками**.



Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Кодування тексту

Кодування тексту – це тип кодування, який перетворює текст у байти. Це досягається шляхом перетворення кожного символу (літери, цифри, знаки та пробіли) у двійковий код.

Набір символів – це просто список символів і двійковий код, який використовується для представлення кожного з них.

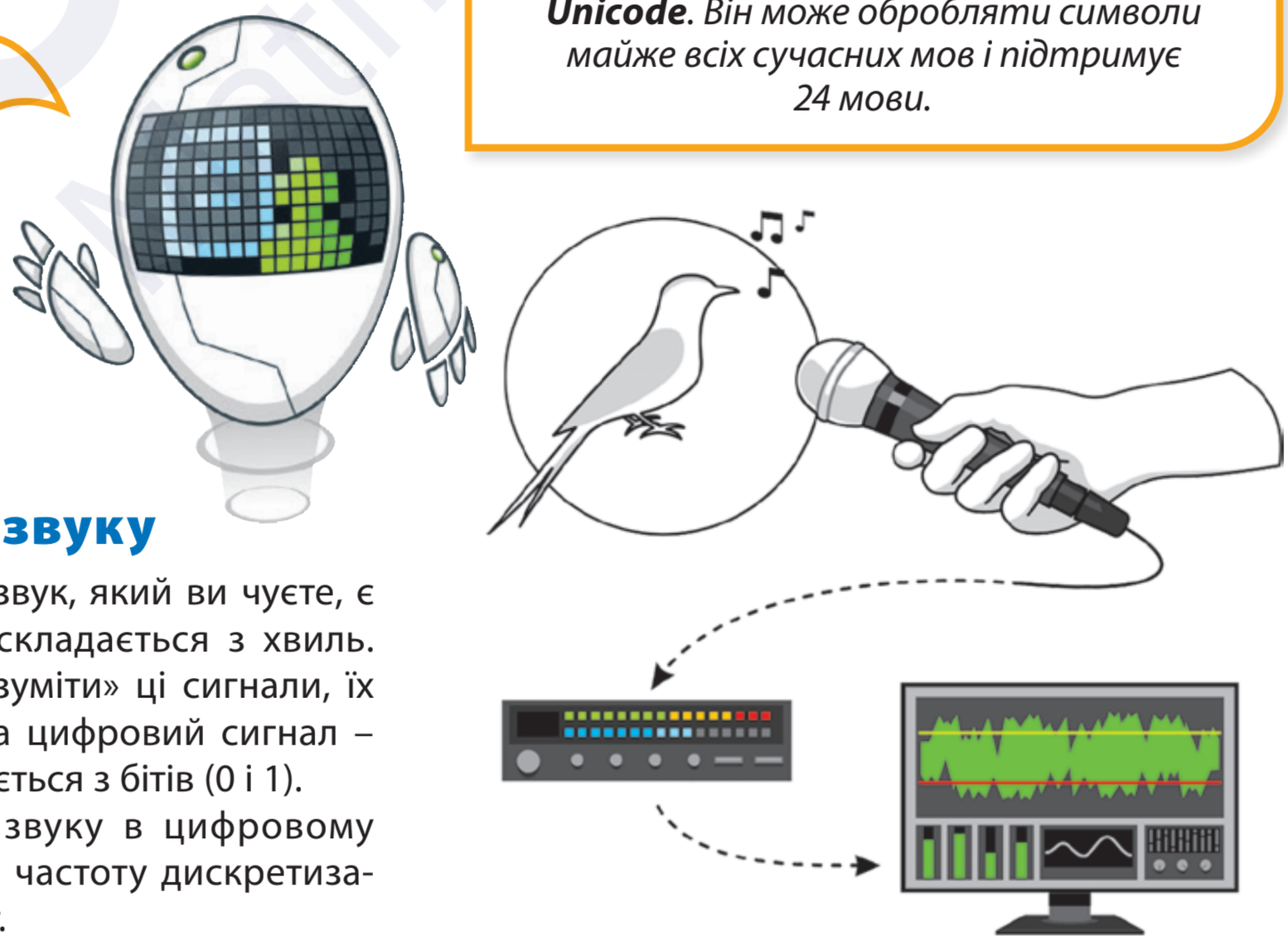
Нижче наведено набір символів ASCII.

*Поширений метод кодування тексту – це **ASCII** (**American Standard Code for Information Interchange**) – американський стандартний код для обміну інформацією.*

	_0	_1	_2	_3	_4	_5	_6	_7	_8	_9	_A	_B	_C	_D	_E	_F
0_	**NUL** 0000 0	**SOH** 0001 1	**STX** 0002 2	**ETX** 0003 3	**EOT** 0004 4	**ENQ** 0005 5	**ACK** 0006 6	**BEL** 0007 7	**BS** 0008 8	**HT** 0009 9	**LF** 000A 10	**VT** 000B 11	**FF** 000C 12	**CR** 000D 13	**SO** 000E 14	**SI** 000F 15
1_	**DLE** 0010 16	**DC1** 0011 17	**DC2** 0012 18	**DC3** 0013 19	**DC4** 0014 20	**NAK** 0015 21	**SYN** 0016 22	**ETB** 0017 23	**CAN** 0018 24	**EM** 0019 25	**SUB** 001A 26	**ESC** 001B 27	**FS** 001C 28	**GS** 001D 29	**RS** 001E 30	**US** 001F 31
2_	**SP** 0020 32	**!** 0021 33	**"** 0022 34	**#** 0023 35	**$** 0024 36	**%** 0025 37	**&** 0026 38	**'** 0027 39	**(** 0028 40	**)** 0029 41	***** 002A 42	**+** 002B 43	**,** 002C 44	**-** 002D 45	**.** 002E 46	**/** 002F 47
3_	**0** 0030 48	**1** 0031 49	**2** 0032 50	**3** 0033 51	**4** 0034 52	**5** 0035 53	**6** 0036 54	**7** 0037 55	**8** 0038 56	**9** 0039 57	**:** 003A 58	**;** 003B 59	**<** 003C 60	**=** 003D 61	**>** 003E 62	**?** 003F 63
4_	**@** 0040 64	**A** 0041 65	**B** 0042 66	**C** 0043 67	**D** 0044 68	**E** 0045 69	**F** 0046 70	**G** 0047 71	**H** 0048 72	**I** 0049 73	**J** 004A 74	**K** 004B 75	**L** 004C 76	**M** 004D 77	**N** 004E 78	**O** 004F 79
5_	**P** 0050 80	**Q** 0051 81	**R** 0052 82	**S** 0053 83	**T** 0054 84	**U** 0055 85	**V** 0056 86	**W** 0057 87	**X** 0058 88	**Y** 0059 89	**Z** 005A 90	**[** 005B 91	**** 005C 92	**]** 005D 93	**^** 005E 94	**_** 005F 95
6_	**`** 0060 96	**a** 0061 97	**b** 0062 98	**c** 0063 99	**d** 0064 100	**e** 0065 101	**f** 0066 102	**g** 0067 103	**h** 0068 104	**i** 0069 105	**j** 006A 106	**k** 006B 107	**l** 006C 108	**m** 006D 109	**n** 006E 110	**o** 006F 111
7_	**p** 0070 112	**q** 0071 113	**r** 0072 114	**s** 0073 115	**t** 0074 116	**u** 0075 117	**v** 0076 118	**w** 0077 119	**x** 0078 120	**y** 0079 121	**z** 007A 122	**{** 007B 123	**|** 007C 124	**}** 007D 125	**~** 007E 126	**DEL** 007F 127

*Інший метод кодування тексту – **Unicode**. Він може обробляти символи майже всіх сучасних мов і підтримує 24 мови.*

***Кодування звуку** – це процес перетворення звукових сигналів у цифрові.*



Кодування звуку

У природі будь-який звук, який ви чуєте, є аналоговим сигналом і складається з хвиль. Щоб комп'ютер міг «зрозуміти» ці сигнали, їх потрібно перетворити на цифровий сигнал – тобто у форму, що складається з бітів (0 і 1).

Для представлення звуку в цифровому форматі використовують частоту дискретизації, глибину бітів і бітрейт.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Частота дискретизації

Частота дискретизації – це кількість зразків звуку, які беруться кожну секунду. Що більше зразків взяти, то вища якість звуку, але потрібно більше місця для зберігання. Частота дискретизації вимірюється в герцах (Гц). Стандартна частота для музики становить 44 100 зразків на секунду, 44 100 зразків на секунду = 44 100 герц, або 44,1 кілогерца (кГц).

Бітрейт

Бітрейт – це кількість бітів даних, які обробляються кожну секунду.

Бітрейт зазвичай вимірюється в кілобітах на секунду (кбіт/с).

Бітрейт розраховується за формулою:

Бітрейт =
= Частота дискретизації × Розрядність × Кількість каналів

Стиснення звуку

Для зменшення розміру файлу застосовують **стиснення**. Дані видаляються для зменшення розміру файлу, що також корисно під час потокового передавання та завантаження файлів.

Найпопулярнішими форматами стисненого аудіо є **MP3** (MPEG-1 Audio Layer III) і **WMA** (Windows Media Audio). MP3 є найбільш часто використовуваним цифровим аудіоформатом. Аудіо, збережені як MP3-файли, мають розширення файлу «.mp3», вимагають значно менше місця для зберігання, ніж файли в інших аудіоформатах. Така ефективність у розмірі файлу робить MP3 кращим вибором для зберігання та обміну музикою в цифровому вигляді.

Існує два типи стиснення:
стиснення без втрат
і ***стиснення з втратами****.*

При **стисненні без втрат** даних ви отримуєте менший файл без жодних спотворень або втрат інформації і можете використовувати його в будь-якому типі аудіо.

Стиснення з втратами призводить до втрати частини даних, які не відновлюються, що може впливати на якість звуку, але дає значно більший ступінь стиснення. При цьому типі стиснення видаляється непомітна для слухача інформація, що призводить до зменшення розміру вихідного файлу до десяти разів. Наприклад, WAV-файл, стиснутий до MP3, буде мати стиснення з втратами.

Розрядність

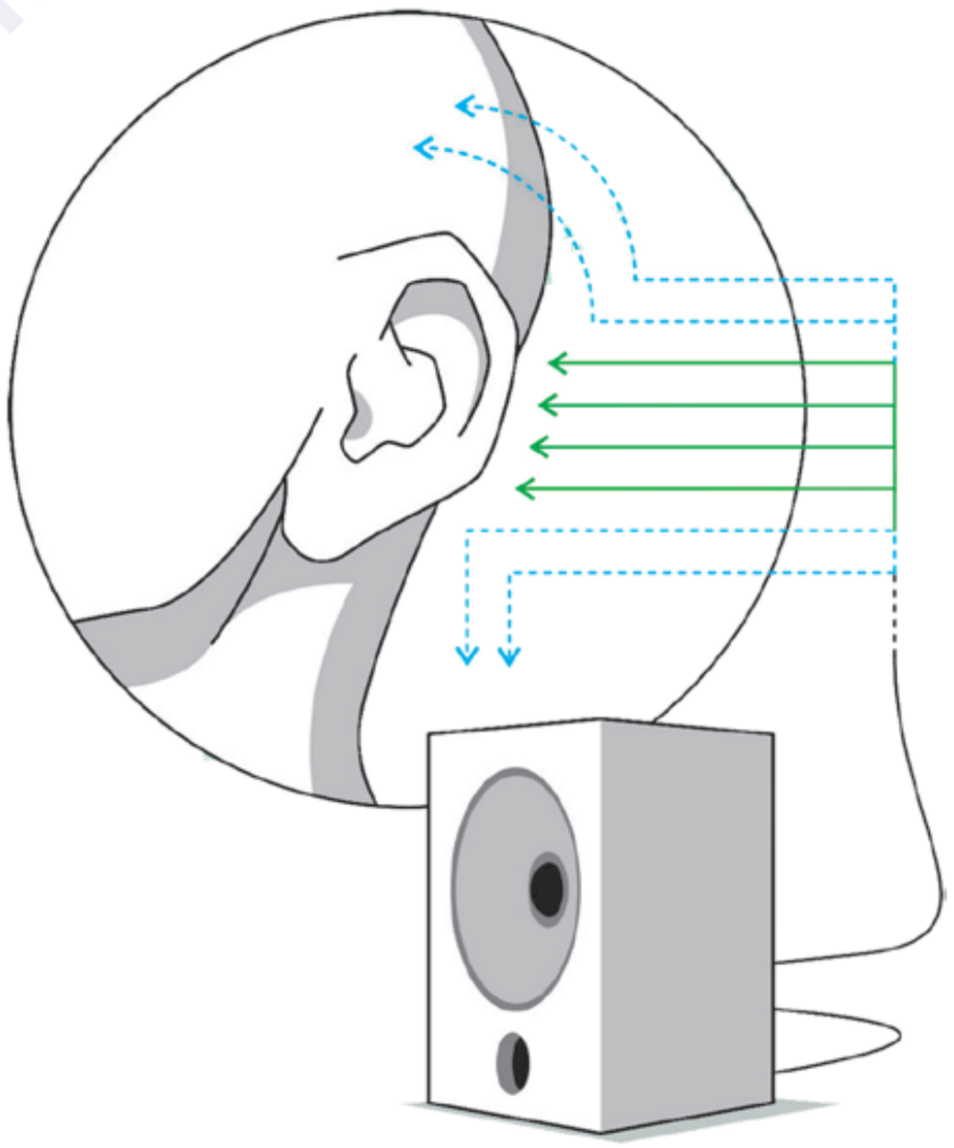
Розрядність – це кількість бітів, які використовують для кодування для кожного зразка звуку.

Що вища розрядність, то вища точність цифрового аудіо й тим краща якість звуку. Для запису музики використовують 24-бітну розрядність, тоді як для голосового аудіо використовують 8 бітів розрядності.

MP3 *видаляє звукові компоненти, які людське вухо зазвичай не чує. Це дає змогу ефективно стиснути аудіо з мінімально помітним зниженням якості.*



Типовий формат зберігання звукового файлу – ***WAV*** *(формат аудіофайлу Waveform), який використовується для комп'ютерів, містить стиснені аудіодані.*



Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Кодування зображень

Якщо ви хочете зберегти зображення, його спочатку треба закодувати. Зображення на комп'ютері представлені растровими зображеннями. **Растрове зображення** – це двовимірний масив у просторі, що складається з незалежних елементів зображення, які називають пікселями. **Піксель** є найменшим елементом зображення. Кожен піксель має числове значення, яке називається **шириною**. Кількість бітів, доступних для кодування пікселя, називається **глибиною**. Прикладами типової глибини пікселів є 1 (для чорно-білих або монохромних зображень), 2, 4, 8, 16 або 24 біти. Числові значення можуть представляти або білу, або чорну точку в двокольорових ілюстраціях, або відтінок сірого в суміжному чорно-білому зображенні, або кольорову особливість елемента зображення для кольорових зображень.

Растрові зображення – це тип зображення, який використовується в цифрових камерах, смартфонах і на вебсайтах. Деякі поширені растрові формати – JPEG, GIF, PNG і WebP. Такі формати зображень зазвичай потребують більше місця для зберігання, ніж текст або прості малюнки. Якщо у вас є два зображення однакового розміру, вони будуть використовувати однаковий обсяг пам'яті на вашому пристрої, навіть якщо одна картина дуже деталізована, а інша – простий малюнок лише з кількома формами. WebP – це новий тип растрового зображення, який допомагає зробити файли меншими за розміром без втрати якості, що чудово підходить для економії місця та швидкого обміну зображеннями в мережі.

Кількість бітів	Кількість кольорів
1	Монохром
4	16
8	256
16	65536
24	16 мільйонів

Розмір зображення важливий. Що більше розмір зображення, то більше місця для зберігання потрібно.



Стиснення зображення

Основним недоліком растрового формату зображення є розмір файлу. Великий розмір файлу є не тільки недоліком для зберігання, але це також викликає затримку передавання зображення в мережі. На щастя, існують методи стиснення зображень, які можуть призвести до зменшення розміру в 25–50 разів.

Залежно від методу стиснення стиснене зображення або строго ідентичне оригіналу, або зазнало деяких змін.

Стиснення файлу з втратами переважно пов'язане з файлами зображень, де певна втрата інформації не буде виявлена більшістю користувачів.

Кодування відео

Кодування відео – це процес перетворення відео у формат, який працює в інтернеті та на смартфонах і планшетах. Це схоже на створення фільму з багатьох фотографій, які швидко рухаються одна за одною.

Коли кодується відео, зображення зберігаються як «кадри». Фільми на екранах показують певну кількість кадрів у секунду, що називається **частотою кадрів**. Зазвичай це близько 24 кадри у секунду, але для більш якісних відео частота може досягати 100 кадрів у секунду. Наприклад, деякі відео високої роздільної здатності (HD) використовують 50 або 60 кадрів у секунду. Частоту кадрів позначають як «fps».

Ще одне важливе поняття – **бітрейт**. Це показник того, скільки даних відтворюється в секунду відео. Якщо бітрейт високий, відео на вигляд якісніше, але й файл більший. Якщо бітрейт низький, то файл менший, але якість гірша.

Розмір зображення (ширина в пікселях) × × (висота в пікселях)	Роздільна здатність
1920 × 1080	HD
3840 × 2160	4K UHD
7680 × 4320	8K UHD
15360 × 8640	16K UHD
1024 × 768	XGA
1280 × 720	WXGA

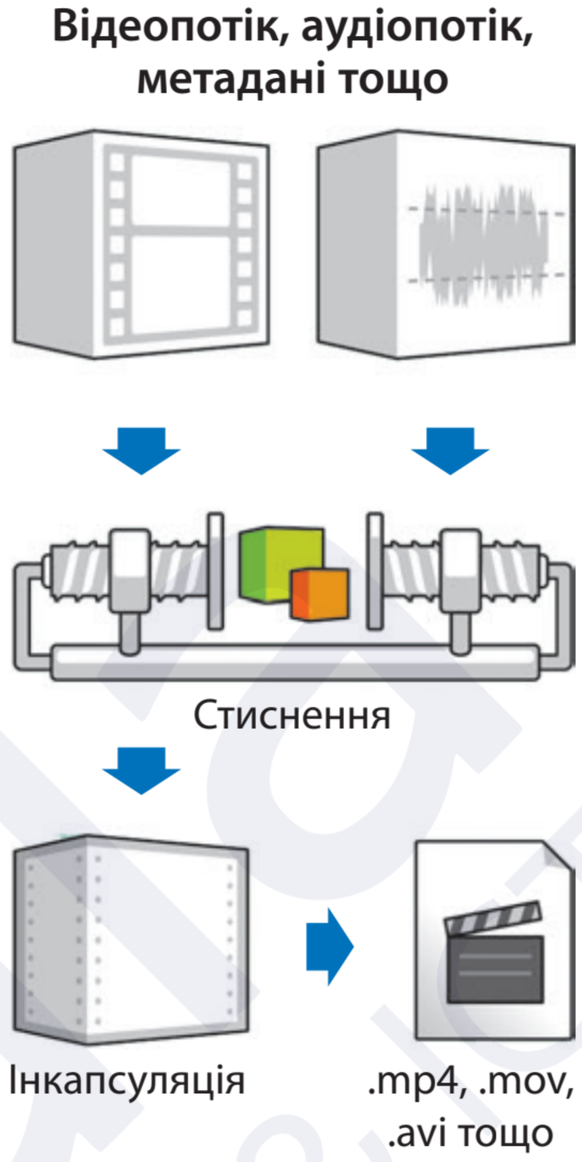
Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Стиснення відео

Стиснення відео призводить до зменшення трьох параметрів: роздільної здатності, розмірів та бітрейту. Наприклад, 1,5-годинне восьмибітове відео у форматі 24 кадри на секунду вимагає приблизно 334 ГБ пам'яті, тому стиснення має важливе значення.

Як уже згадувалося, у процесі стиснення деякі дані можуть бути втрачені, що викликає появу «артефактів» – низької якості зображення або навіть випадкових кольорових блоків, які з'являються й зникають на екрані. Ці блоки ще називають артефактами стиснення. Найпоширенішими форматами стиснення відео є MP4 та MOV, які використовують кодеки, наприклад MPEG, а прикладом методу стиснення без втрат є DV (Digital Video).

Прикладом алгоритму стиснення є Run length encoding (RLE). Він перетворює і замінює послідовно однакові значення в код за допомогою числа повторів і самого значення. Це один з найпростіших способів стиснення, дуже легкий у реалізації і не потребує багато обчислювальних ресурсів, але ефективний з файлами, які мають багато повторюваних даних.



Шифрування

Шифрування – це метод захисту даних від перегляду небажаними особами. Для цього дані потрібно закодувати так, щоб їх не можна було розшифрувати, якщо немає спеціального ключа. У криптографії термін «ключ» означає інструмент, наприклад пароль, який використовується для шифрування і розшифрування інформації. Шифрування використовується всюди. Наприклад, коли ви використовуєте банківську картку для купівлі чогось онлайн, ваш комп'ютер шифрує цю інформацію, щоб інші не могли викрасти ваші особисті дані під час передачі.

Симетричне шифрування

Це тип шифрування, у якому один і той самий ключ використовується для шифрування і розшифрування повідомлення. Секретний ключ, який може бути числом, словом або фразою, застосовується до даних для їх перетворення у вигляд, що не піддається розшифруванню. Повідомлення та ключ повинні бути передані у певний спосіб. Відправник й отримувач мають знати секретний ключ, який використовується для шифрування і розшифрування всіх повідомлень.



Організації та приватні особи зазвичай використовують шифрування для захисту конфіденційних даних, що зберігаються на комп'ютерах, серверах і мобільних пристроях, як-от смартфони чи планшети.



ДОВІДКА

Файл графічного обміну (GIF) і Portable Network Graphics (PNG) забезпечують стиснення без втрат. Файл зображення Joint Photographic Experts Group (JPEG) – це зображення, яке має стиснення з втратами і пропонує більш високий ступінь стиснення, ніж PNG, з компромісом у якості.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Асиметричне шифрування

Проблема симетричного шифрування полягає в тому, що якщо хтось дізнається секретний ключ, він зможе розшифрувати повідомлення. В асиметричному шифруванні, яке також називають криптографією з відкритим ключем, цю проблему вирішено. Такий тип шифрування використовує пару ключів: відкритий і приватний. Відкритий ключ можна зробити доступним для всіх, хто хоче надіслати вам повідомлення. Приватний ключ зберігається в секреті, щоб лише ви могли ним користуватися. Повідомлення, зашифроване відкритим ключем, розшифровується приватним ключем, і навпаки: повідомлення, зашифроване приватним ключем, можна розшифрувати відкритим ключем.

Асиметричне шифрування повільніше за симетричне, оскільки для шифрування та розшифрування повідомлення потрібно більше обчислювальних ресурсів.

Щоб надіслати зашифровані дані комусь, потрібно зашифрувати їх за допомогою відкритого ключа цієї особи.

Особа, яка отримує дані, розшифровує їх відповідним приватним ключем.



Шифрування електронних листів до їх надсилання важливе, щоб у разі перехоплення повідомлення хакером або іншою особою, крім одержувача, ваші повідомлення були нечитабельними та незрозумілими.



Шифрування жорсткого диска

Шифрування диска призначене для захисту всього пристрою зберігання даних на вашому комп'ютері, на відміну від захисту окремих файлів. Таке шифрування використовує програмне забезпечення для шифрування всіх даних, які потрапляють на диск. Цей тип шифрування використовується не лише для жорстких дисків, а й для інших носіїв, наприклад USB-накопичувачів або резервних копій.

Шифрування електронної пошти

Шифрування електронної пошти ґрунтується на захисті потенційно конфіденційної інформації під час її надсилання електронною поштою. Зокрема, шифрується з'єднання між відправником і сервером, а також самі повідомлення. Коли ваші електронні листи зашифровані, навіть якщо їх перехоплять, то не зможуть прочитати.

ІСТОРІЯ

У 700 році до н. е. стародавні греки та спартанці використовували скиталу як криптографічний метод. Це був інструмент для створення секретних повідомлень: вони обгортали смужку шкіри або пергаменту навколо палиці та писали повідомлення на ній.

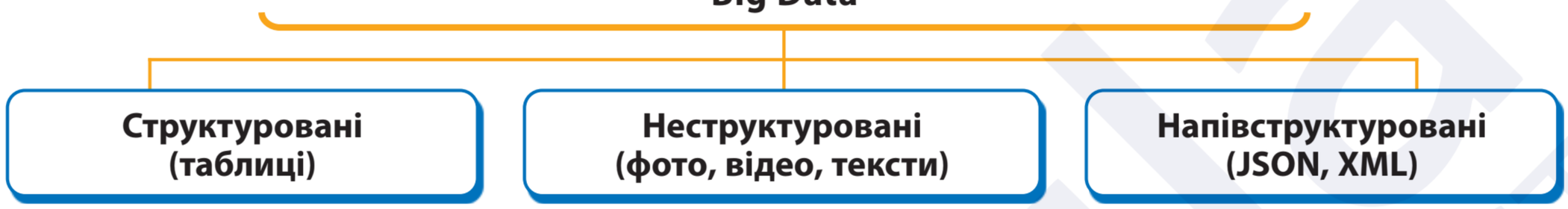
Практичне завдання

Завантажте файл-заготовку зі сторінки інтернет-підтримки. Виконайте запропоноване завдання.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Тема 4
Великі дані

Big Data, або великі дані – цей термін використовують для опису надзвичайно великих обсягів інформації, які важко або неможливо обробити за допомогою звичайних програм чи баз даних.



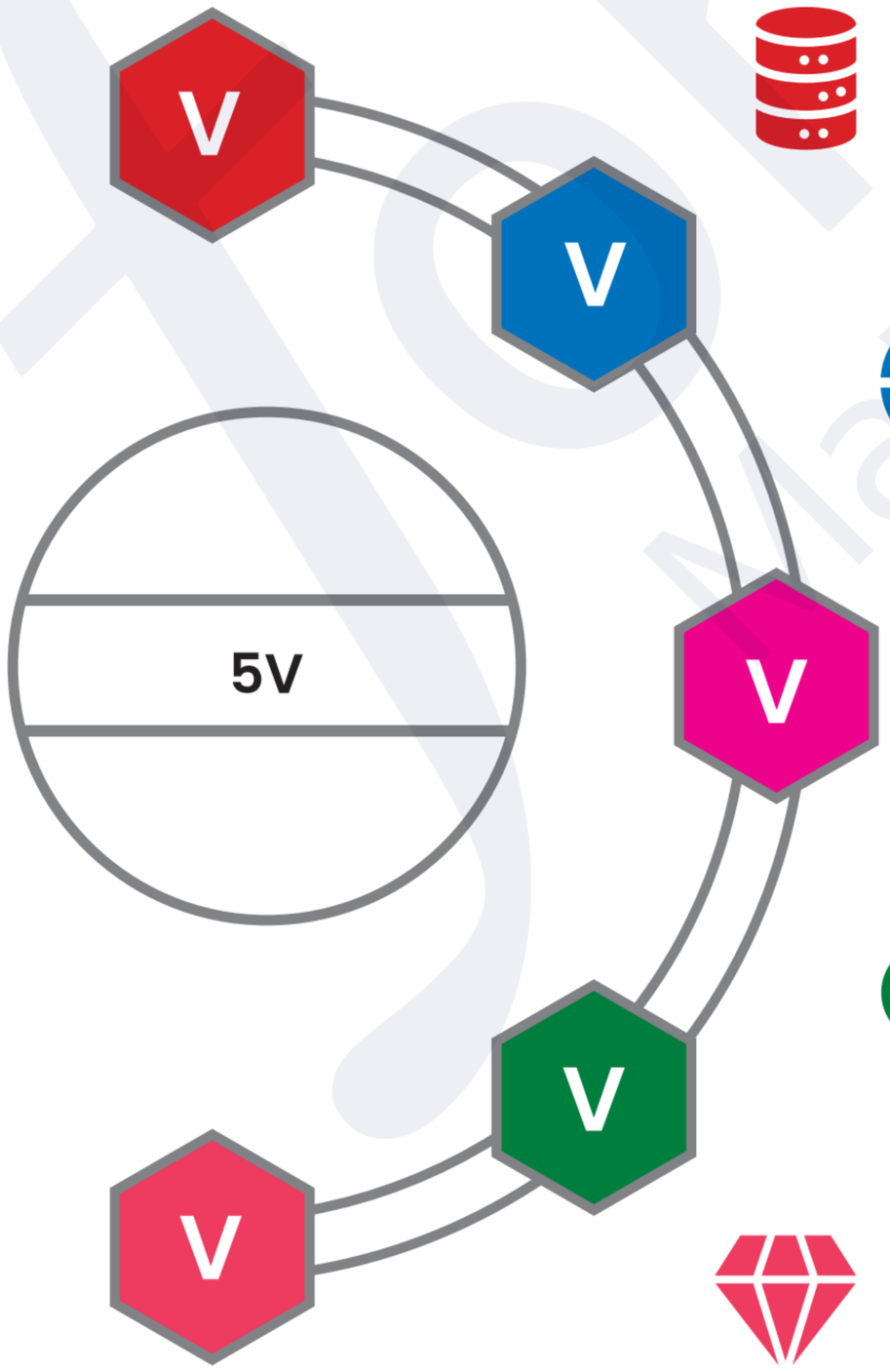
Обсяг великих даних вимірюється в терабайтах, петабайтах і навіть екзабайтах.

Джерела великих даних:

- Соціальні мережі – повідомлення, фото, відео, коментарі
- Сенсори та IoT-пристрої – клімат, тиск, рух, енергоспоживання
- Системи відеоспостереження
- Датчики у смартфонах, авто, виробництві
- Електронна комерція – покупки, уподобання, історія транзакцій
- Медицина – електронні карти пацієнтів, результати аналізів
- Цифрові ЗМІ – новини, блоги, коментарі

Властивості

Великі дані мають особливі властивості, які позначаються абревіатурою 5V.



Volume (Обсяг)
Величезна кількість даних, що генеруються і зберігаються

Velocity (Швидкість)
Дані надходять і змінюються дуже швидко

Variety (Різноманітність)
Багато різних типів і форматів даних

Розширена модель 5V додає ще два критерії.

Veracity (Достовірність)
Якість, точність і надійність даних

Value (Цінність)
Наскільки дані корисні для аналізу і прийняття рішень

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Як працює технологія Big Data

Збір даних

Дані надходять із багатьох джерел одночасно: соціальні мережі, сенсорні дані, логи серверів, транзакції та багато інших. Системи повинні мати можливість об'єднувати дані різних типів, обробляти інформацію в реальному часі, готувати її для подальшого аналізу.



Зберігання

Через великі обсяги інформації для зберігання великих даних використовують хмарні сервіси (Google Cloud, Amazon S3), розподілені системи (Hadoop, NoSQL), які можуть обробляти дані одночасно на багатьох серверах.

Аналіз

Аналіз великих даних – це процес обробки та інтерпретації величезних обсягів інформації з метою отримання корисних висновків. Він включає пошук шаблонів і закономірностей у даних, прогнозування майбутніх подій (так звану предиктивну аналітику), створення рекомендацій, наприклад у сфері маркетингу, а також аналіз текстів, зображень і відео.

Для цього часто застосовуються сучасні цифрові технології.

Сучасні цифрові технології для аналізу великих даних

Машинне навчання дає змогу системам самостійно вчитися на основі даних

Штучний інтелект імітує людське мислення та допомагає автоматизувати рутинні завдання

Візуалізація даних допомагає зрозуміти результати аналізу завдяки графікам, діаграмам та іншим наочним формам подання інформації

Ці інструменти спрямовані на глибше розуміння даних, точніші прогнози й ухвалення обґрунтованих рішень.

Big Data та персональні дані



Великі дані часто містять особисту інформацію користувачів: історію покупок, геолокацію, вподобання, пошукові запити.

Хоча ці дані важливі для компаній (персоналізована реклама, покращення сервісів), вони можуть нести ризики для приватності. Наприклад, витік даних, несанкціонований доступ, порушення конфіденційності.

У Європі діє закон GDPR, який зобов'язує компанії інформувати користувачів про збір даних, надавати право контролю над персональною інформацією, забезпечувати її захист.

GDPR (General Data Protection Regulation) *– це загальний регламент Європейського Союзу про захист даних, який встановлює правила обробки та захисту персональних даних фізичних осіб у межах ЄС.*



В Україні діє **Закон** «**Про захист персональних даних**», а також інші нормативні акти, які регулюють використання ІТ.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Використання Big Data

У транспорті великі дані використовують для оптимізації маршрутів, управління дорожнім рухом і зменшення заторів. Завдяки аналізу інформації з навігаторів, камер і датчиків вдається забезпечити ефективне пересування як приватного, так і громадського транспорту.

У медицині такі дані допомагають у персоналізованому лікуванні, враховуючи індивідуальні особливості пацієнтів. Також відбувається аналіз історій хвороб для прогнозування ризиків та покращення діагностики.

В освітній сфері вони дають змогу відстежувати прогрес здобувачів і здобувачок освіти, виявляти слабкі місця та будувати адаптивне навчання, яке підлаштовується під потреби кожного учня й учениці.

Основні проблеми Big Data

Однією з основних проблем, пов'язаних із використанням великих даних, є питання конфіденційності та безпеки.

По-перше, існує ризик витоку персональної інформації, особливо якщо дані зберігаються без належного захисту. Такі дані можуть використовуватися без згоди користувачів, що є порушенням прав на приватність. Подібні ситуації можуть призвести до незаконного доступу або зловживання інформацією, яка стосується особистого життя людини.

Майбутній розвиток

У майбутньому великі дані матимуть ще більший вплив на різні сфери життя. Зокрема, у медицині вони сприятимуть швидшій та точнішій діагностиці, а також підвищенню ефективності лікування завдяки аналізу великої кількості клінічних даних.

У бізнесі технології великих даних допоможуть точніше прогнозувати попит, оптимізувати витрати та підвищити прибутковість компаній.

Міста розвиватимуться як «розумні» – зі зручнішою інфраструктурою, ефективнішим транспортом, енергоспоживанням і безпекою завдяки аналітиці в реальному часі.

*Технології аналізу **великих даних** активно застосовують в різних сферах життя та діяльності.*

У науці великі дані використовуються для обробки величезних масивів інформації, наприклад, під час дослідження геному людини, структури Всесвіту або змін клімату.

У маркетингу великі дані дають змогу компаніям аналізувати поведінку клієнтів, створювати персоналізовані рекламні кампанії, прогнозувати попит та інтереси покупців.

У сільському господарстві аналіз даних допомагає прогнозувати погоду, планувати роботи та автоматизувати зрошення, що підвищує ефективність вирощування культур.

Для ефективного використання великих даних необхідно приділяти особливу увагу їх перевірці, очищенню та оновленню.

Ще одна важлива проблема – якість і достовірність даних. Інформація, яка надходить із різних джерел, може бути неточною, неповною або застарілою. Якщо такі «брудні» або неперевірені дані потрапляють в аналітичні системи, це може призвести до помилкових висновків і неправильних рішень.

У сфері науки вони стануть основою для нових відкриттів в астрономії, фізиці, біології та інших галузях, адже допоможуть обробляти гігантські обсяги інформації, недоступні для аналізу звичайними методами.

Із розвитком штучного інтелекту аналіз даних стає ще потужнішим, що допомагає людині приймати складні рішення швидко і точно.



Практичне завдання

Завантажте файл-заготовку зі сторінки інтернет-підтримки. Виконайте запропоноване завдання.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Тема 5
Сучасні технології

Інтернет речей (IoT)

Інтернет речей вже став буденністю.

Наприклад, навігатор, який допомагає не заблукати, зекономити час, побудувавши оптимальний маршрут, з урахуванням заторів, ремонтів доріг, місць, де сталися аварії, та запропонувавши альтернативні варіанти.

Розумний годинник – вимірює пульс, кроки, відстежує сон і передає дані на смартфон.

Розумний холодильник – повідомляє, коли закінчуються продукти.

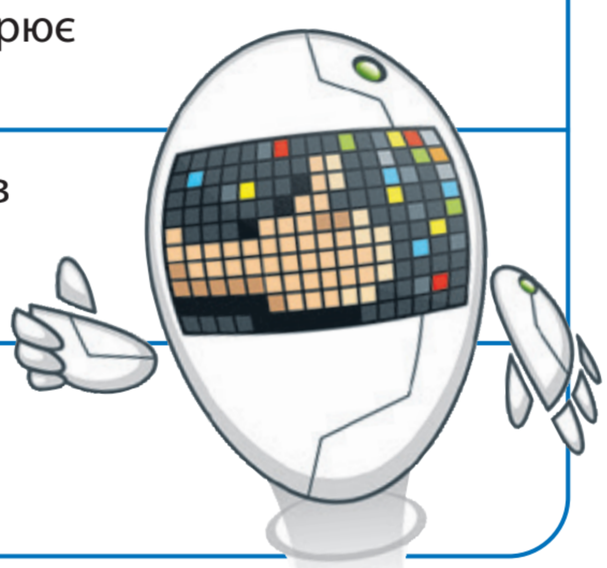
Таких прикладів стає все більше, а наше повсякденне життя стає комфортнішим. Вони мають на меті вивільнити нам час та сприяють його раціональному використанню.

Однак, як і завжди, крім корисностей, існують і серйозні проблеми, пов'язані з Інтернетом речей.



Інтернет речей *(англ. Internet of Things, IoT) – це мережа з'єднаних між собою розумних пристроїв, які можуть обмінюватися даними через інтернет.*

Ненадійне зберігання даних	IoT-пристрої часто зберігають конфіденційні дані, наприклад особисту інформацію, яка може стати вразливою до зламів або витоків
Слабкі паролі	Багато IoT-пристроїв постачаються із заводськими паролями, які легко зламати. Це робить їх легкою мішенню для хакерів
Уразливість до шкідливого ПЗ	Пристрої IoT можуть бути заражені вірусами, які поширюються на інші пристрої в мережі й створюють серйозні загрози безпеці
Відсутність оновлень	Багато IoT-пристроїв не оновлюються регулярно, тому залишаються вразливими до вже відомих загроз
Недостатнє шифрування	Деякі IoT-пристрої не використовують надійного шифрування даних, що уможливлює легке перехоплення інформації зловмисниками
Порушення конфіденційності	IoT-пристрої можуть збирати й передавати великі обсяги персональних даних, що викликає занепокоєння щодо конфіденційності
Несумісність	Пристрої від різних виробників можуть не мати спільних стандартів і не «розумітися» між собою, що створює ризики та ускладнює безпечне підключення
Недостатня автентифікація	Деякі IoT-пристрої не мають надійних механізмів автентифікації, що дозволяє стороннім особам отримати доступ до них
Недостатній контроль авторизації	Багато IoT-пристроїв не мають належного обмеження доступу, що дає змогу стороннім маніпулювати пристроєм або керувати ним



Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Мобільні платежі

Мобільні платежі – це спосіб оплати товарів і послуг за допомогою смартфона або іншого гаджета: браслета чи каблучки. Тобто замість готівки ви просто використовуєте свій пристрій.

Це відбувається завдяки **технології NFC** (англ. Near Field Communication) – технології бездротового зв'язку на дуже короткій відстані (до 10 см), яка дає змогу пристроям обмінюватися даними, просто торкнувшись один одного або перебуваючи поруч.

Технології NFC використовуються під час оплати за допомогою смартфона (Google Pay, Apple Pay), у безконтактних картках (банківські, проїзні, студентські, учнівські), під час передачі файлів між телефонами, в електронних квитках (на концерт, у транспорті), у системах доступу (наприклад, вхід у школу чи офіс).

Просто встановіть мобільний застосунок, наприклад Google Pay чи Apple Pay, залежно від операційної системи вашого смартфона. Відкрийте його та внесіть дані вашої банківської картки. Вони зберігаються зашифрованими, тобто захищеними.



Правила безпечного користування NFC

Використовуйте пароль або біометрію (відбиток пальця, розпізнавання обличчя).

Не передавайте смартфон стороннім людям.

Завжди вимикайте NFC після користування та блокуйте екран.



Переваги мобільних платежів

Не потрібно носити готівку.

Не треба хвилюватися про забуту чи загублену банківську картку.

Швидко – оплата займає кілька секунд.

Безконтактно – не потрібно торкатися термінала.

Блокчейн

Блокчейн – це технологія, яка забезпечує безпечну й прозору передачу інформації та активів. Уявіть собі цифрову книгу (реєстр), у якій зберігається запис усіх транзакцій між сторонами. Кожна сторінка цієї книги – «блок», що містить багато транзакцій. Кожен новий блок пов'язаний з попереднім, утворюючи «ланцюжок» блоків, – блокчейн (англ. block – блок, chain – ланцюг). Однією з ключових особливостей блокчейну є його децентралізованість, тобто його не контролює жодний суб'єкт, наприклад уряд або корпорація. Натомість кожен учасник мережі має копію реєстру й може перевіряти та підтверджувати транзакції. Це робить майже неможливим зміну або підробку даних.



Ще однією важливою особливістю цієї технології є високий рівень захисту, оскільки блокчейн використовує криптографію для захисту чутливої інформації, як-от фінансові транзакції чи персональні дані. Тому блокчейн часто використовують для надійного зберігання інформації.

ДОВІДКА

На виставці CES 2025 (Лас-Вегас, США) компанії представили голографічні дисплеї нового покоління – це прозорі 3D-дисплеї, які не потребують окулярів.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Криптовалюта

Криптовалюта – це цифрова чи віртуальна валюта, яка використовує криптографію для захисту та функціонує незалежно від центрального банку. Транзакції з криптовалютою побудовані на технології блокчейн, яка дозволяє здійснювати безпечні, прозорі та захищені від підробки грошові перекази. Біткоїн є найвідомішим прикладом криптовалюти, але існує багато інших, як-от Ethereum, Tether, Binance Coin. Ключова риса криптовалют – децентралізованість, тобто вона не контролюється жодним урядом чи фінансовою установою. Саме тому це уможливлює швидкі та дешеві транзакції: криптовалюти мають низькі комісії й дають змогу здійснювати швидкі міжнародні



платежі. Однак вартість криптовалют може різко коливатись, і вони не завжди приймаються як засіб оплати, тому важливо розуміти ризики, пов'язані з їх використанням.



Телемедицина

Телемедицина – це використання цифрових технологій: відеоконференції, дистанційний моніторинг, електронні медичні картки для надання медичних послуг та інформації на відстані. Це важливо, оскільки дає змогу медичним працівникам охопити пацієнтів, які інакше не мали б доступу до медичної допомоги або через те, що живуть у сільській чи віддаленій місцевості, або через труднощі з мобільністю чи транспортуванням. Зокрема, телемедицина може покращити доступ до медичної допомоги завдяки скороченню часу очікування та уникненню потреби в поїздках, а також через надання більш зручних та ефективних послуг.

Допоміжні технології

Допоміжні технології – це пристрої, інструменти та продукти, які допомагають особам з інвалідністю досягти більшої незалежності в повсякденному житті. Ці технології можуть бути використані для покращення доступу до комунікацій, мобільності, освіти та можливостей працевлаштування серед іншого.

Приклади допоміжних технологій:

- *програмне забезпечення для читання з екрана;*
- *програмне забезпечення для розпізнавання голосу та зображень;*
- *альтернативні та додаткові комунікаційні пристрої;*
- *пристрої, що генерують мовлення.*



ДОВІДКА

Рукавиці для перекладу жестової мови перетворюють жести на текст або голос у реальному часі, що значно полегшує спілкування між людьми з порушеннями слуху та тими, хто не знає мови жестів.

Нейроінтерфейси для керування пристроями на кшталт Neuralink або Synchron дають змогу людям з обмеженою рухливістю керувати комп'ютерами, смартфонами або навіть ро́ботами силою думки.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Біометрія

Біометрія – це сучасна технологія, яка використовує особливості людини для автентифікації. Це можуть бути що зовнішні ознаки (наприклад, обличчя чи відбитки пальців), то й поведінкові риси (наприклад, голос або почерк). У кожної людини ці характеристики неповторні, тому вони добре підходять для ідентифікації. У нових паспортах чи ID-картках застосовуються біометричні дані для автентифікації власників. **Біометричні дані** – це унікальні вимірювані фізичні характеристики, які використовуються для підтвердження особи. Наприклад, колір очей, зріст, відбитки пальців, обличчя, райдужка ока, голос, форма руки, почерк, хода. У деяких випадках біометричні системи вже повністю замінили PIN-коди. Наприклад, для розблокування смартфона чи комп'ютера можна використовувати відбиток пальця, обличчя (Windows Hello, Apple Face ID) або голос. В установах з високим рівнем безпеки – центри обробки даних, дослідницькі центри, різноманітні лабораторії, приміщення з обмеженим доступом тощо – використовують сканери очей, які сканують райдужну оболонку чи сітківку вашого ока.

Переваги такої автентифікації безперечні: не потрібно запам'ятовувати паролі, можна здійснити швидкий доступ, дуже складно підробити дані.

Але й недоліки досить очевидні. Можливі помилки розпізнавання – наприклад, через шрами, вікові зміни чи технічні обмеження пристрою.

Біометрія – це сучасний і надійний спосіб ідентифікації, але потребує обережного ставлення до персональних даних. Існують певні проблеми, пов’язані з конфіденційністю та безпекою. Біометрична інформація – чутливі персональні дані, тому її потрібно збирати лише за згодою та з відома тієї людини, якій вона належить.

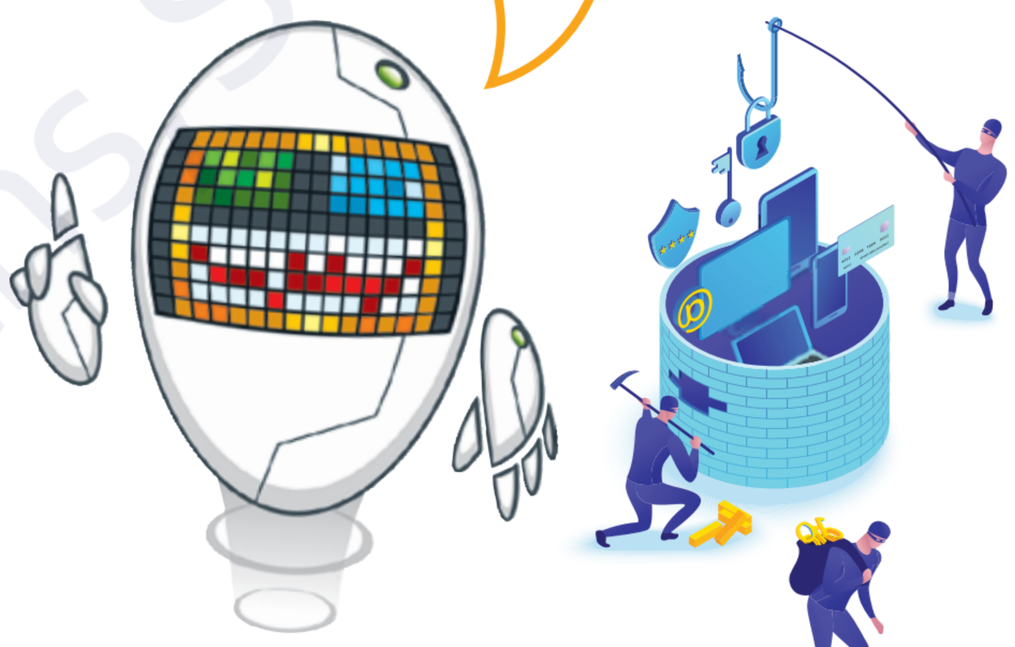


Навіть якщо дані не вкрадені, виникають питання:

- Хто їх збирає, зберігає та контролює?
- Як вони використовуються поліцією чи державними службами?
- Чи не порушуються права людини?

Щоб використання біометричних технологій для безпеки та правоохоронної діяльності було законним та етичним, потрібні чіткі правові норми. Особисту інформацію варто збирати лише за потреби та в обсягах, які забезпечать роботу системи ідентифікації, а також обмежити до неї доступ.

Біометричні системи можуть бути зламані для отримання чужої біометричної інформації. Це може призвести до крадіжки особистості – коли хтось видає себе за іншу людину.



Високошвидкісний мобільний зв'язок

У наш час найпоширеніший стандарт у мобільних мережах – LTE (Long-Term Evolution) і LTE Advanced. Він належить до покоління 4G і забезпечує стабільне з’єднання навіть для об’єктів, що рухаються на високій швидкості – до 350–500 км/год. Технологія 5G забезпечує ще вищу пропускну здатність – понад 1 Гбіт/с, а також значно меншу затримку сигналу. Це критично важливо для застосунків, що працюють у реальному часі, зокрема для розумних пристроїв та Інтернету речей (IoT).



На Всесвітній конференції з 5G Чжан Юнтао, заступник директора з технологій компанії Ericsson у Китаї, в інтерв’ю зазначив, що мережа 6G буде не просто черговим етапом підвищення швидкості. За його словами, 6G забезпечить злиття віртуального та реального світів. Він прогнозує, що голографічна комунікація перестане бути елементом наукової фантастики й у майбутньому почне активно розвиватися як реальна технологія.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Зберігання даних

Нині у світі генеруються величезні обсяги даних. З кожним роком даних стає більше, тож для зберігання фото, відео, ігри, програми, документи – нам потрібно більше місця. Місця для їх зберігання ніколи не буде достатньо.

Існує три основні типи зберігання даних:



Сховище файлів упорядковує дані у файли й теки та використовується в персональних комп'ютерах і серверах для зручного керування даними.

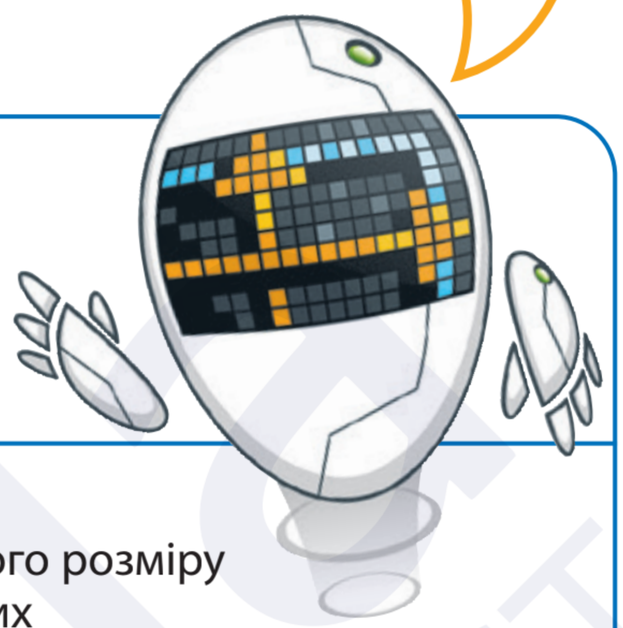


Блочне сховище розбиває дані на блоки фіксованого розміру та є поширеним явищем у системах зберігання даних корпоративного рівня.



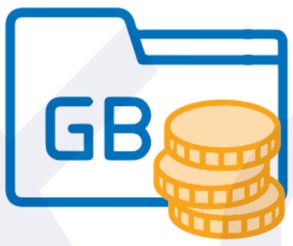
Об'єктне сховище зберігає дані як об'єкти з унікальними ідентифікаторами та ідеально підходить для хмарних сховищ та вебзастосунків.

Найменший у світі жорсткий диск – розміром з монету.



Проблеми, які потрібно враховувати при зберіганні даних

Ціна за гігабайт



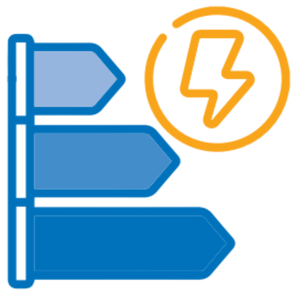
Швидкість доступу до даних



Термін зберігання



Споживання енергії – особливо від процесорів і накопичувачів у серверних



3D-друк

3D-друк і лазерне різання відкрили нові можливості.

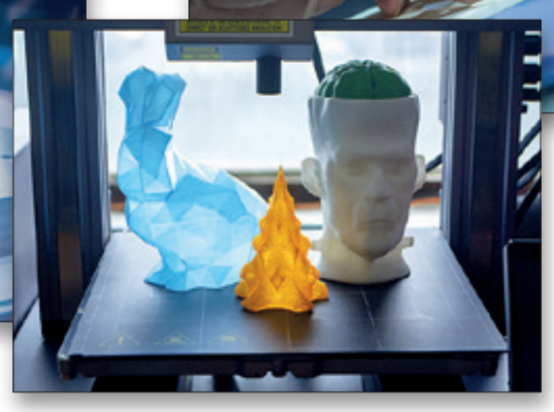
3D-друк широко використовується в мистецтві, інженерії та освітніх проєктах. Наприклад, технічні фахівці можуть виготовити запасну частину для обладнання або навіть покращити його. У медицині ця технологія дає змогу створювати індивідуальні імпланти та протези, адаптовані до потреб конкретного пацієнта.

3D-друк вже використовується для зведення цілих споруд, як-от будинків, шкіл. Серед переваг: зменшення потреби в робочій силі, пришвидшення будівництва, підвищення точності та зменшення кількості будівельних відходів.









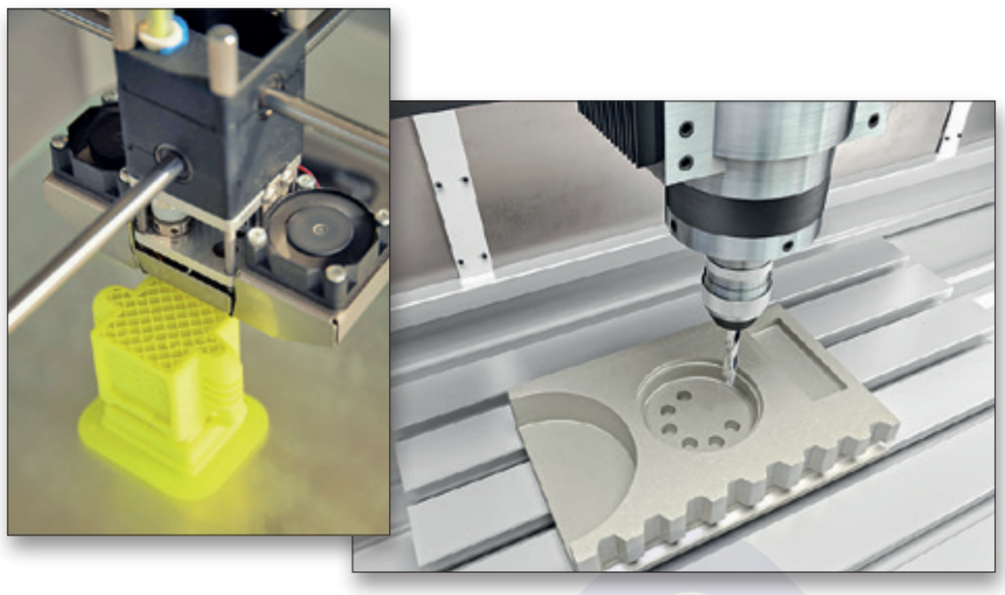
Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Адитивний vs субтрактивний друк

Адитивний і субтрактивний друк – це два різні методи, які використовуються в 3D-друці.

Адитивний друк – це процес створення тривимірного об'єкта шляхом поступового додавання шарів матеріалу.

Субтрактивний друк, навпаки, полягає у видаленні матеріалу з цільного блоку для формування бажаної форми за допомогою фрезерних, лазерних або інших інструментів.



Фрезерування	Лазерне різання	Водний струмінь
Видалення матеріалу за допомогою фрезерних інструментів.	Видалення матеріалу за допомогою лазера.	Видалення матеріалу за допомогою високошвидкісного струменя води.

Обробка з числовим програмним керуванням (ЧПК) є прикладом субтрактивного друку. У цьому методі комп'ютер керує інструментом: свердлом або фрезерним верстатом, щоб вирізати об'єкт із блоку матеріалу – зазвичай металу або дерева. Цей метод широко застосовується в таких галузях, як аерокосмічна, автомобільна та інженерна промисловість, де потрібна висока точність і ретельне дотримання розмірів.



Вибір методу 3D-друку здебільшого залежить від типу використовуваного матеріалу та необхідної точності.



Біодрук

Біодрук – це технологія, яка використовує методи 3D-друку для створення біологічних структур: тканин й органів. Вона передбачає використання принтера для розміщення живих клітин за певною схемою разом із допоміжним матеріалом, щоб сформувати 3D-структуру. Метою біодруку є створення замінників тканин і органів для медичних цілей, наприклад для трансплантації. Ця технологія вже застосовується для друку шкіри, хрящів, кісток і навіть складних органів: печінки та серця. Біодрук вже дає можливість застосовувати нові методи лікування хвороб і станів, які раніше не піддавалися лікуванню.

ДОВІДКА

3D UTU – українська компанія, яка виробляє будівельні 3D-принтери.
Процес згортання / розгортання пристрою займає до трьох годин.
Для будівництва потрібні лише електроенергія, спеціальна бетонна суміш і вода.

Практичне завдання

Завантажте файл-заготовку зі сторінки інтернет-підтримки. Виконайте запропоноване завдання.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Тема 6
Навчання в цифрову епоху

Ми живемо в епоху стрімких технологічних змін, коли цифрові технології кардинально змінюють усі сфери людського життя. Освіта не стала винятком – вона переживає справжню революцію. Традиційна модель навчання, де учень / учениця сидить за партою і слухає вчителя / вчительку, поступово доповнюється та трансформується завдяки цифровим інструментам, які відкривають нові можливості для ефективного та цікавого навчання.

Цифрова освіта – це використання комп'ютерів, планшетів, інтерактивних дошок, освітніх програм, онлайн-платформ та інших технологічних рішень для створення більш ефективного, доступного та персоналізованого навчального досвіду.

Цифрова освіта *– це комплексний підхід до навчання, який інтегрує сучасні технології в освітній процес для покращення якості знань, розвитку навичок XXI століття та підготовки учнів до життя в цифровому суспільстві.*

Основні компоненти цифрової освіти

- **Цифрові інструменти та платформи для навчання**
- **Онлайн-ресурси та електронні підручники**
- **Інтерактивні методи викладання**
- **Індивідуалізоване навчання з урахуванням індивідуальних потреб**
- **Дистанційне навчання та змішані форми освіти**

Переваги цифрової освіти

Доступність та гнучкість

Цифрова освіта руйнує географічні та часові бар'єри. Студенти можуть навчатися у будь-який час і в будь-якому місці, маючи доступ до якісних освітніх ресурсів. Наприклад, учениця із невеликого села може прослухати лекцію провідного професора з київського університету або взяти участь у віртуальній екскурсії до Лувру.

Індивідуалізоване навчання

Цифрові технології дають змогу адаптувати навчальний процес до індивідуальних потреб кожного здобувача чи здобувачки освіти. Штучний інтелект може аналізувати прогрес студента / студентки та пропонувати персоналізовані завдання, які відповідають його / її рівню знань та стилю навчання.

Інтерактивність та залученість

Сучасні цифрові інструменти роблять навчання більш цікавим та інтерактивним. Замість пасивного сприйняття інформації учні й учениці активно взаємодіють з навчальним матеріалом, що покращує засвоєння знань.

Розвиток цифрових навичок

Використання технологій у навчанні природно розвиває цифрові навички, які є критично важливими для успішної кар'єри в сучасному світі.

Онлайн-платформи для навчання

Khan Academy – безкоштовна платформа, яка пропонує відеоуроки з математики, природничих наук, історії та інших предметів. Діти можуть навчатися у власному темпі, отримуючи миттєвий зворотний зв'язок.

Coursera та **edX** – платформи, які пропонують курси від провідних університетів світу. Наприклад, студент може пройти курс «Введення в штучний інтелект» від Стенфордського університету, не виходячи з дому.

Duolingo – додаток для вивчення іноземних мов, який використовує ігрову діяльність для мотивації учнів та учениць. Завдяки адаптивному алгоритму, програма підлаштовується під рівень кожного користувача.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Віртуальна та доповнена реальність

Google Expeditions уможливлює віртуальні екскурсії до найвіддаленіших куточків світу. Учні й учениці можуть «відвідати» стародавній Рим, дослідити дно океану або навіть подорожувати всередині людського тіла.

Anatomyou VR – застосунок для вивчення анатомії людини у віртуальній реальності. Студенти-медики можуть детально розглянути органи та системи організму в 3D-форматі.

Штучний інтелект в освіті

Carnegie Learning використовує ШІ для створення персоналізованих програм з математики. Система аналізує помилки учнів / учениць та пропонує додаткові вправи для опрацювання складних для них тем.

Grammarly – ШІ-помічник для перевірки граматики та стилю письма, який допомагає учням та ученицям покращувати свої навички письма англійською мовою.

Геймифікація в навчанні

Minecraft Education Edition – освітня версія популярної гри, яка використовується для навчання історії, математики, програмування та інших предметів. Діти можуть будувати історичні споруди, вивчати хімічні реакції або програмувати роботів.

Kahoot! – платформа для створення інтерактивних квізів та ігор, які роблять повторення матеріалу веселим та захопливим.

Цифрові лабораторії

PhET Interactive Simulations від Колорадського університету пропонує інтерактивні симуляції з фізики, хімії, біології та математики. Учні й учениці можуть проводити віртуальні експерименти без ризику та витрат на обладнання.

Labster – платформа віртуальних лабораторій, яка дає змогу студентам проводити складні наукові експерименти у віртуальному середовищі.

Мобільні застосунки для навчання

GeoGebra – це безкоштовний інтерактивний програмний комплекс для вивчення математики, який об'єднує динамічну геометрію, алгебру, аналіз, статистику та електронні таблиці.

Quizlet – платформа для створення та використання флеш-карток для запам'ятовування інформації.

Виклики цифрової освіти

Цифровий розрив

Не всі учні й учениці мають рівний доступ до технологій. Різниця в доступі до інтернету, сучасних пристроїв та технічної підтримки може поглиблювати освітню нерівність.

Підготовка освітян

Успішне впровадження цифрової освіти потребує відповідної підготовки педагогів. Вони мають не лише вміти користуватися технологіями, а й розуміти, як ефективно інтегрувати їх у навчальний процес.

Залежність від технологій

Надмірна залежність від цифрових інструментів може призвести до втрати важливих традиційних навичок, таких як рукописне письмо або здатність до глибокого читання.

Безпека та конфіденційність

Використання цифрових платформ порушує питання захисту персональних даних дітей та забезпечення безпечного онлайн-середовища. Для цього важливо впроваджувати надійні системи шифрування, що зможуть захистити інформацію.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Цифрова освіта в Україні

В Україні активно впроваджуються ініціативи цифрової освіти.

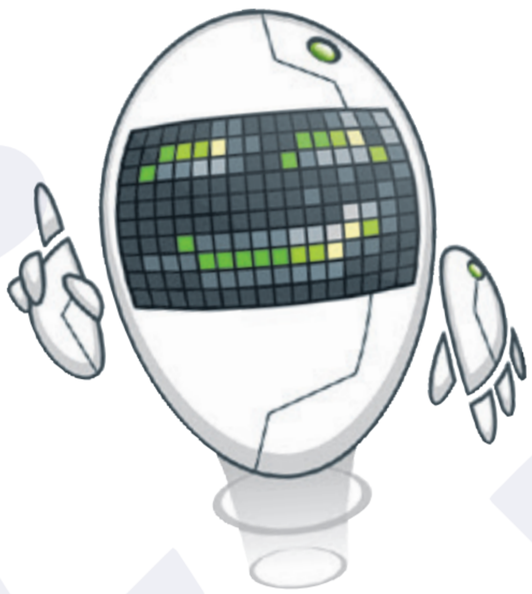
Всеукраїнська школа онлайн – проєкт, який забезпечує дистанційне навчання для учнів та учениць закладів середньої освіти.

Prometheus – українська платформа масових відкритих онлайн-курсів (MOOC), яка пропонує безкоштовні курси з різних галузей знань.

EdEra – платформа інтерактивних онлайн-курсів для шкільної та вищої освіти в Україні.

Дія.Цифрова освіта – державна ініціатива з розвитку цифрових навичок громадян і громадянок України.

Цифрова освіта – це не просто тренд, а вимога сучасного світу. Вона відкриває нові можливості для ефективного навчання, робить освіту більш доступною та цікавою. Однак успішне впровадження цифрової освіти потребує комплексного підходу, який включає технічну підтримку, підготовку вчителів, забезпечення рівного доступу та розв'язання питань безпеки.



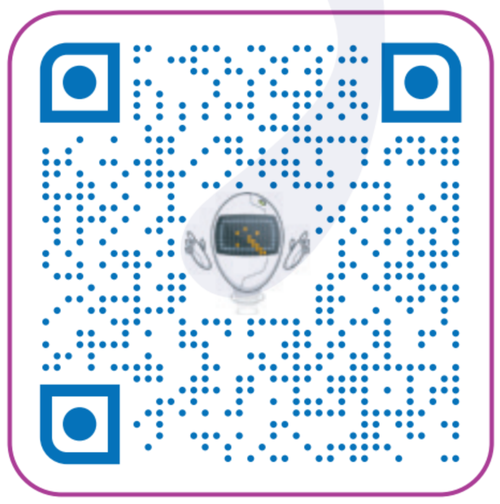
Майбутнє освіти вбачається у гармонійному поєднанні традиційних методів навчання з інноваційними цифровими технологіями. Це дасть змогу створити освітнє середовище, яке не лише передає знання, а й розвиває критичне мислення, креативність та навички, необхідні для успіху в цифровому суспільстві.

Цифрова освіта вже змінює способи навчання мільйонів людей по всьому світу, і ця трансформація лише прискорюється. Тому важливо, щоб здобувачі і здобувачки освіти не лише вміли користуватися цифровими інструментами, а й навчилися критично мислити та оцінювати їхній вплив на суспільство та власне майбутнє.

Практичне завдання

Завантажте файл-заготовку зі сторінки інтернет-підтримки. Виконайте запропоноване завдання.

https://formula.education/online-ict-9-klas/





Інші платформи



Проєкт



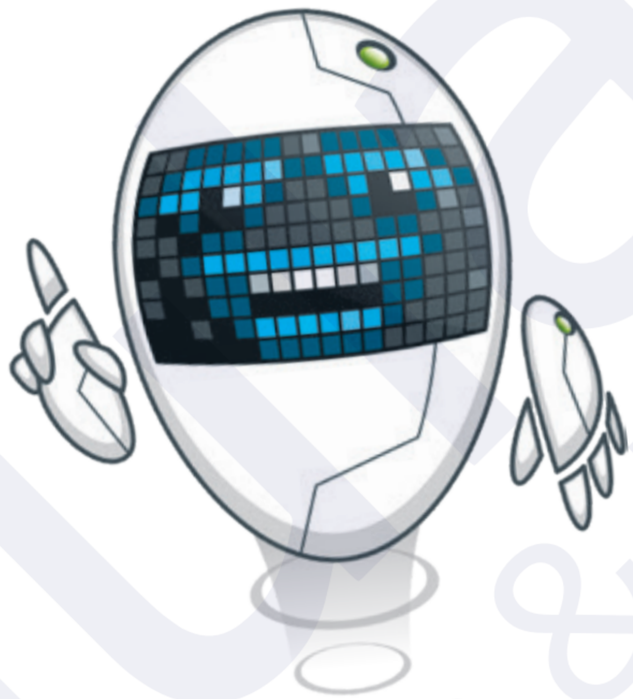
Підсумки



Глосарій

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

3. Упорядкування даних



У цьому розділі ви крок за кроком навчитеся створювати структуровані дані. Дізнаєтеся, як використовувати таблиці для організації даних, як створювати форми та звіти для введення й подальшого збереження та друку даних. Навчитеся отримувати конкретні дані та «витягувати» з них корисну інформацію.

У цьому розділі ви навчитеся:

- організовувати та створювати бази даних;
- систематизувати свою інформацію в таблицях;
- створювати зв'язки між ними;
- встановлювати первинний ключ;
- створювати форми та звіти;
- створювати прості та складні запити;
- імпортувати та експортувати інформацію.

У цьому розділі ви дізнаєтеся:

- що таке база даних;
- як створити структуру бази даних;
- як імпортувати дані до бази даних;
- як зв'язати інформацію в базі даних;
- як зробити поле унікальним;
- як роздрукувати дані;
- як отримати певну інформацію з бази даних.

Програмне забезпечення та обладнання:

- Microsoft Access;
- LibreOffice Base.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Тема 1
Структурування інформації

Чи складаєте списки покупок перед тим, як іти до магазину? Такі списки – простий приклад структурованої інформації. Термін «структурований» означає організований, упорядкований, такий, що має певну структуру або систему. Зокрема, простий список зазвичай розділений на рядки та стовпці. У цьому прикладі кожен рядок матиме стовпці:

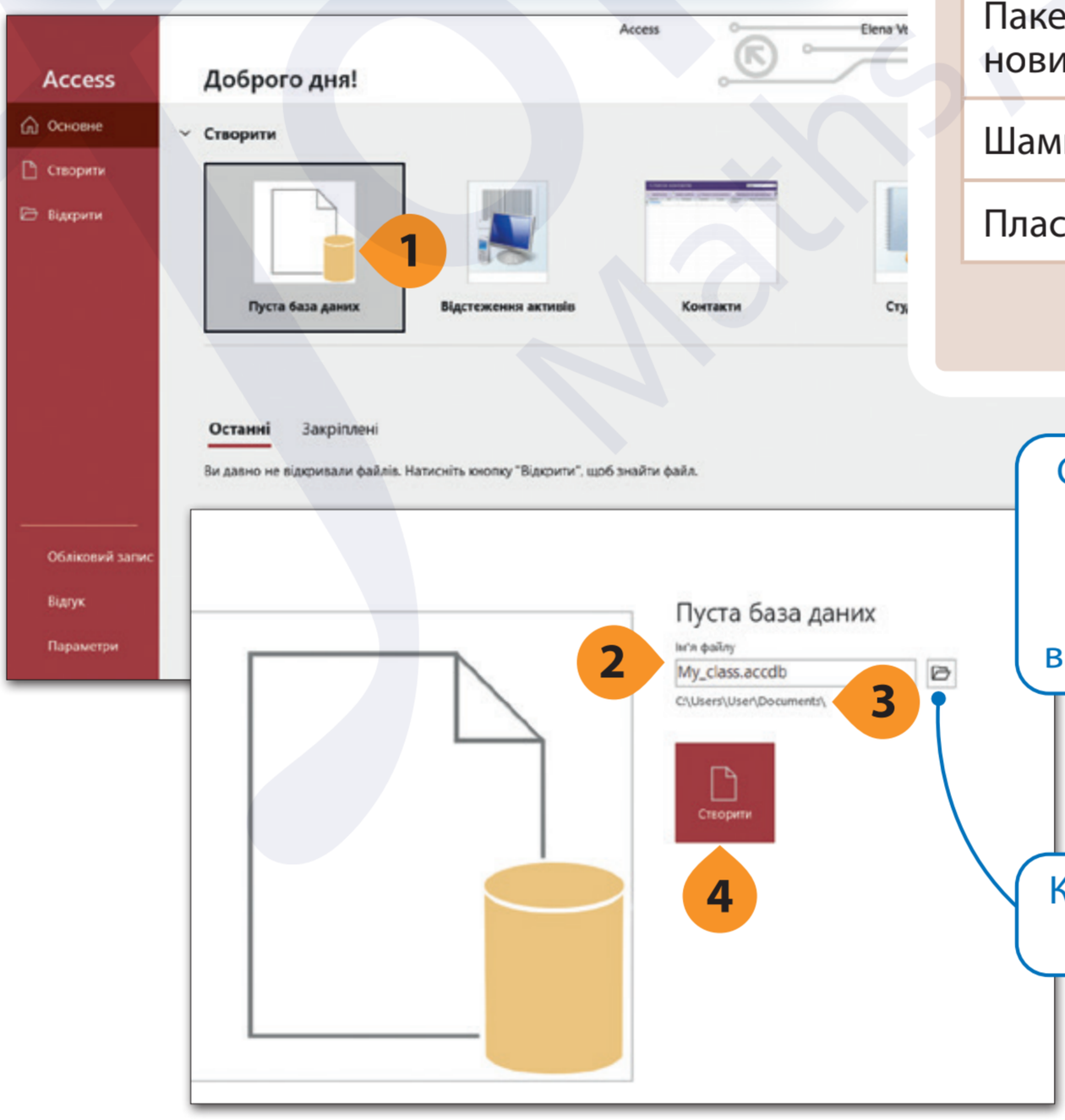
1. Назва продукту
2. Кількість
3. Орієнтовна ціна (допомагає оцінити загальну вартість усіх покупок)

Створення бази даних

Створіть базу даних з оцінками учнів та учениць у вашій школі. Скористайтеся Microsoft Access.

Щоб створити базу даних:

- відкрийте **Microsoft Access**;
- серед доступних шаблонів виберіть **Пуста база даних** (1);
- у наступному вікні введіть **Ім'я файлу** для нової бази даних (2). Сам файл буде збережено у вказаному місці (3);
- натисніть **Створити** (4).



Цей список є інформаційною таблицею.

Список покупок

Назва продукту	Кількість	Орієнтовна ціна
Шоколадні батончики	5	13,50 грн
Йогурт	3	32,99 грн
Пакет кукурудзяних чипсів	2	65,00 грн
Упаковка молока	4	43,70 грн
Пакет з апельсиновим соком	1	63,70 грн
Шампунь	1	79,90 грн
Пластівці	1	43,60 грн
	Разом	342,39 грн

Структурована інформація подається у вигляді таблиці, що складається з рядків (записів) і стовпців (полів). Кожен стовпець має заголовок, що вказує, які саме дані в ньому містяться.

Клацніть **Огляд** та оберіть інший шлях для збереження бази даних.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

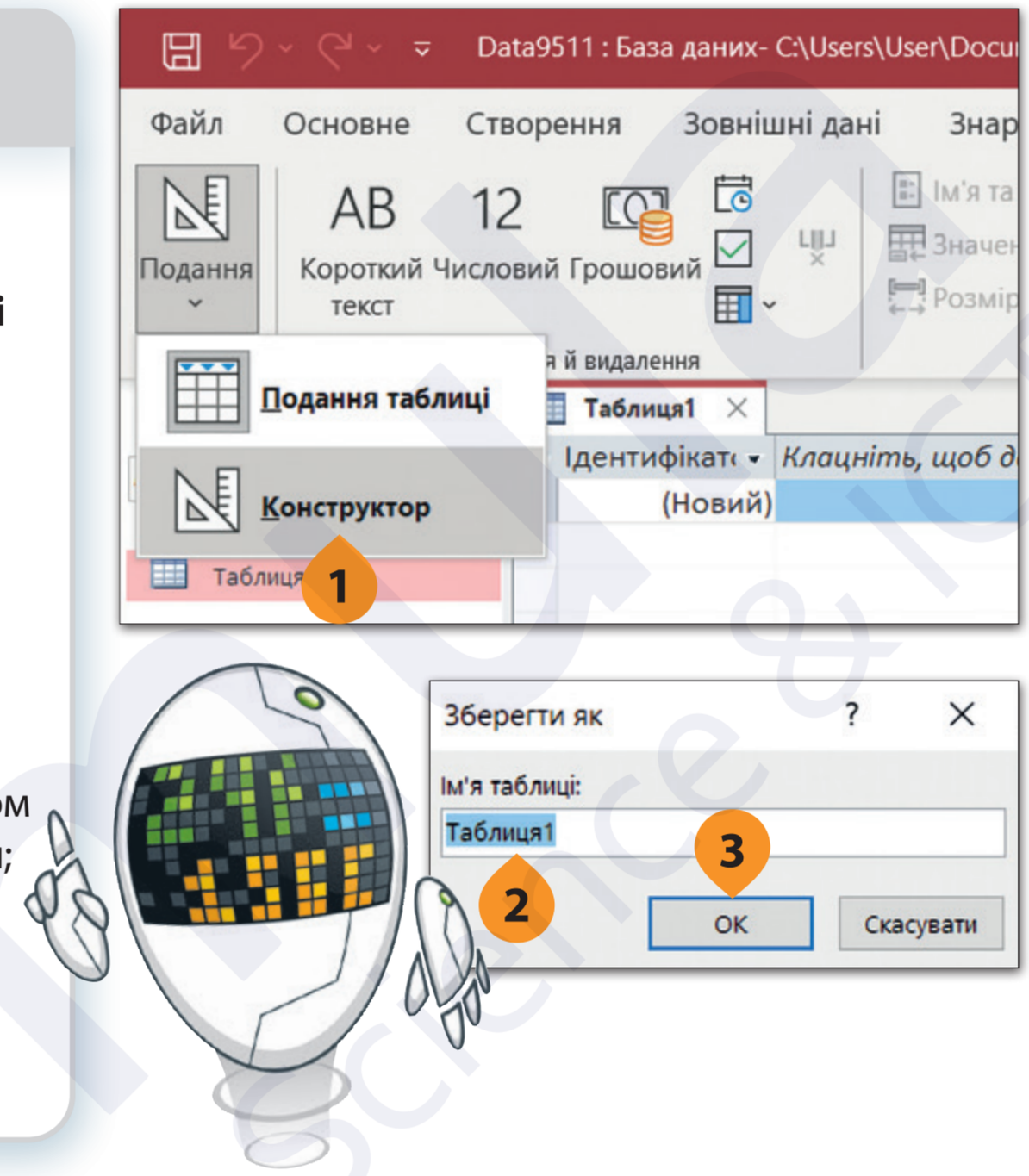
Файл нової бази даних створений. Продовжіть побудову таблиці зі структурованою інформацією у вашій базі даних.

Коли створюється порожня база даних, **Microsoft Access** відкриває вікно **Подання таблиці** й створює порожню таблицю з назвою Таблиця1. У цьому режимі ви можете одразу вводити дані в таблицю, а програма автоматично побудує її структуру.

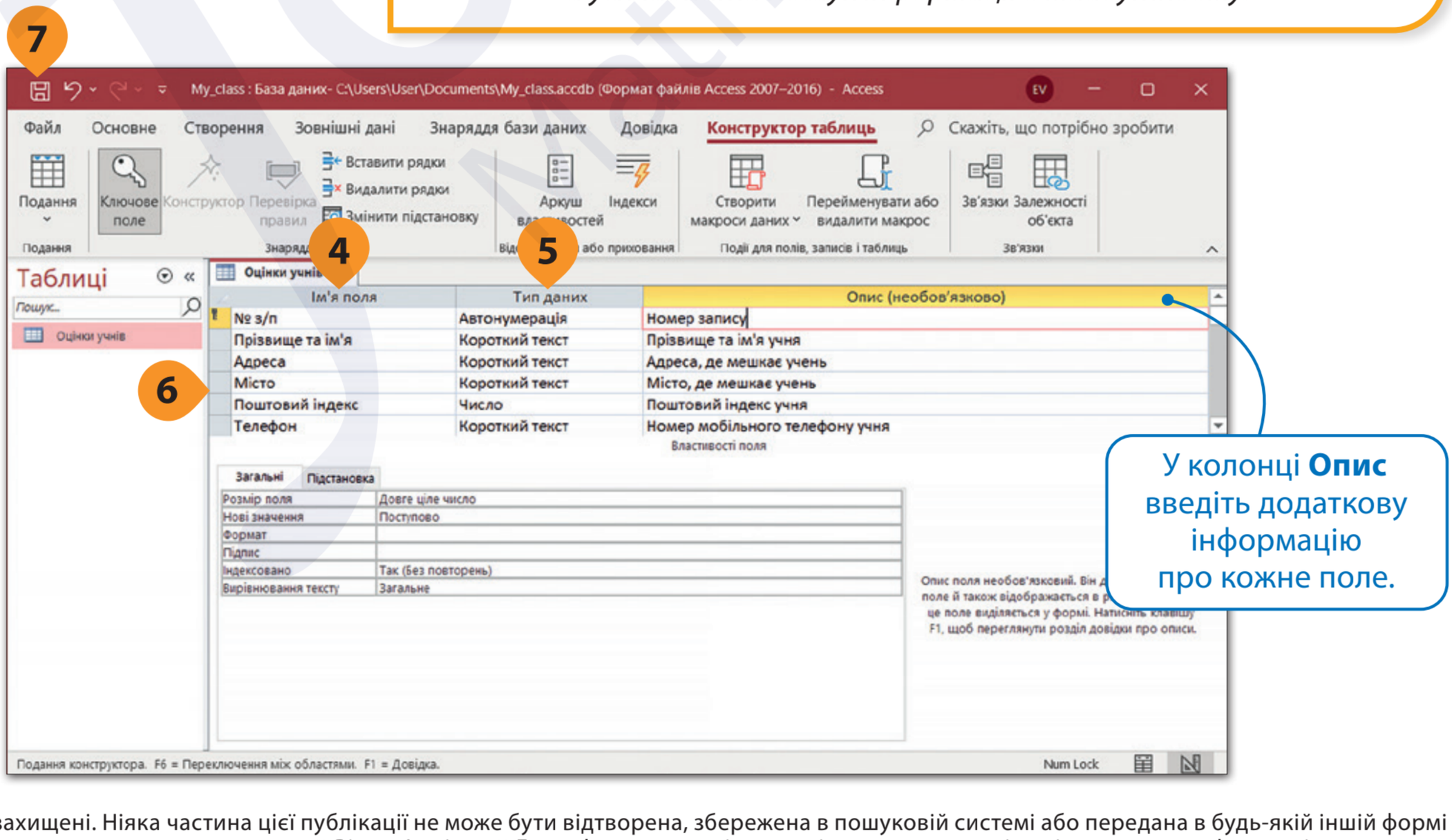
Використайте режим **Конструктор**, щоб самостійно вирішити, якою буде структура таблиці. Потім або переходьте до **Режиму таблиці** для введення даних, або введіть дані за допомогою інших способів, наприклад форм.

Щоб створити таблицю у режимі Конструктора:

- на вкладці **Поля таблиці** у групі **Подання** оберіть **Конструктор** (1);
- у вікні **Зберегти як** у полі **Ім'я таблиці** введіть назву таблиці, наприклад **Оцінки учнів та учениць** (2), і натисніть **ОК** (3);
- у стовпці **Ім'я поля** (4) введіть назви кожного заголовка вашої таблиці, а у списку **Тип даних** (5) оберіть відповідний тип, викликавши спадне меню;
- назвіть перше поле **Номер** за порядком і встановіть тип даних **Автонумерація**;
- уведіть решту назв полів (6) і виберіть відповідні типи даних;
- натисніть **Зберегти** (7) або натисніть Ctrl + S, щоб зберегти дані.



Опис полів не є обов'язковим. Але якщо таблиця велика з великою кількістю полів, відсутність описів може призвести до плутанини й ускладнити пошук інформації в майбутньому.



У колонці **Опис** введіть додаткову інформацію про кожне поле.

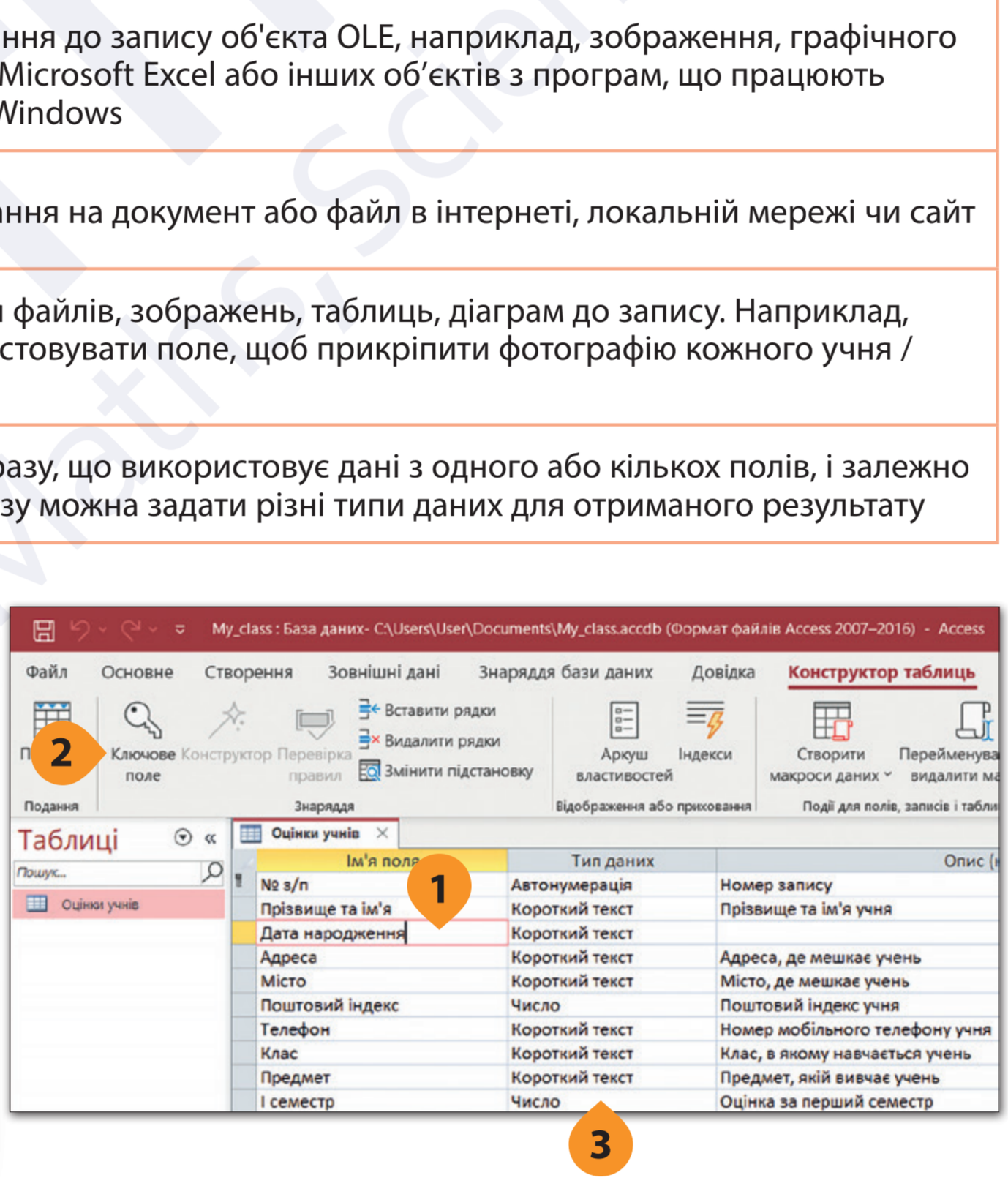
Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Типи даних, які можна використовувати для полів таблиці

Тип даних	Призначення
Короткий текст	Текст може мати до 255 символів
Довгий текст	Блок тексту довжиною понад 255 символів, який також може містити відформатований текст
Число	Числові дані, які можна використовувати в розрахунках
Велике число	Використовуйте цей тип даних для ефективного обчислення великих чисел. «Велике число» має діапазон від -2^{63} до $2^{63}-1$
Дата й час	Для дати та часу
Дата й час (розширені)	Дати й години в діапазоні від 0001–01–01 00:00:00 до 9999–12–31 23:59:59.9999999
Грошова одиниця	Грошові дані, які завжди містять десяткові дроби, 4 знаки після коми
Автонумерація	Унікальне значення, яке програма Access створює для кожного нового запису. Найчастіше це значення використовується як первинний ключ таблиці
Так / Ні	Логічні значення – **Істина** або **Хибність**. Для булевих значень: Yes / No, True / False, On / Off, 1 / 0
Об'єкт OLE	Для прикріплення до запису об'єкта OLE, наприклад, зображення, графічного об’єкту, книги Microsoft Excel або інших об’єктів з програм, що працюють у середовищі Windows
Гіпер-посилання	Адреса посилання на документ або файл в інтернеті, локальній мережі чи сайт
Вкладення	Для вкладання файлів, зображень, таблиць, діаграм до запису. Наприклад, можна використовувати поле, щоб прикріпити фотографію кожного учня / учениці
Обчислю-вальний	Створення виразу, що використовує дані з одного або кількох полів, і залежно від цього виразу можна задати різні типи даних для отриманого результату

Властивості поля

У режимі **Конструктор таблиць**, якщо натиснути на поле (**Ім'я поля**, **Тип даних** або **Опис**), внизу вікна з'являються його властивості (1), які можна змінювати. На вкладці **Конструктор таблиць** є **Ключове поле** (2) – воно дає змогу зробити обране поле первинним ключем.



Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Властивості полів

Властивість	Використання
Розмір поля	Максимальний розмір даних у полі. Наприклад, поле типу **Короткий текст** може містити від 1 до 255 символів. Для кращої продуктивності завжди вказуйте найменший достатній розмір. Деякі типи даних мають фіксований розмір поля, наприклад дата / час або валюта
Маска вводу	Дає можливість установити шаблон, який допомагає вводити дані. Наприклад, маска введення для поля **Короткий текст** 3 може відображати дати у довгому, середньому чи короткому форматі та дозволяти вводити лише цифри
Підпис	Це текст підпису, який за стандартними налаштуваннями відображається для цього поля у формах, звітах і запитах. Якщо ця властивість порожня, використовується назва поля. Щоб підпис був ефективним, він повинен бути коротким і зрозумілим
Значення стандартне	Автоматично призначає вказане значення цьому полю при додаванні нового запису
Правило перевірки	Умова, яка має бути істинною щоразу, коли додаєте чи змінюєте значення в цьому полі
Текст перевірки	Повідомлення, що відображається, коли введене значення не відповідає правилу перевірки
Обов'язково	Ця властивість вимагає, щоб дані були обов’язково введені в поле. Використовується для важливих полів, наприклад ім’я
Дозволити нульову довжину	Коли встановлено параметр Так, то дає можливість вводити рядок нульової довжини («») у полях типу **Гіперпосилання**, **Короткий текст** або **Довгий текст**
Індексовано	Визначає, чи має поле індекс. Користуйтеся індексами для прискорення операцій з полями, які часто використовуються, щоб пришвидшити роботу з ними

Зв'язки між таблицями

Зв'язки між таблицями дуже важливі, оскільки вони об'єднують дані з різних таблиць, щоб надати можливість отримувати потрібну інформацію.

Щоб побачити, як працюють зв'язки між таблицями, створіть декілька таблиць в базі даних.

Розділіть уже створену таблицю **Оцінки учнів та учениць** на три окремі таблиці, як наведено далі.



Учні й учениці

ID_учня / учениці, Прізвище_та_ім'я, Дата_народження, Адреса, Місто, Поштовий_індекс, Номер_телефону, Клас

Предмети

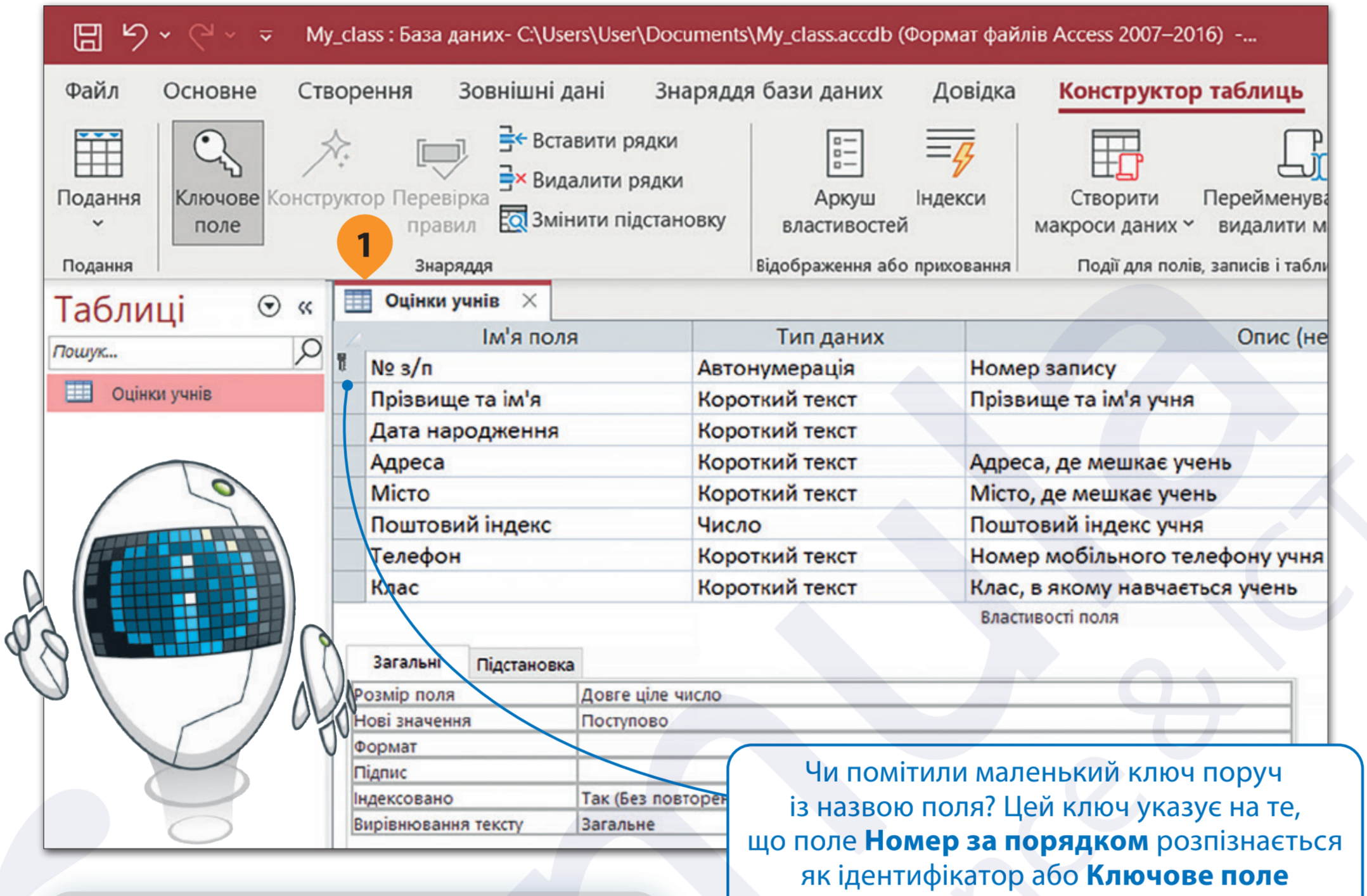
ID_предмета, Предмет, Клас, Учитель / учителька

Оцінки

ID_школяра / школярки, ID_курсу, Оцінка_I_семестр, Оцінка_II_семестр, Річна_оцінка

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

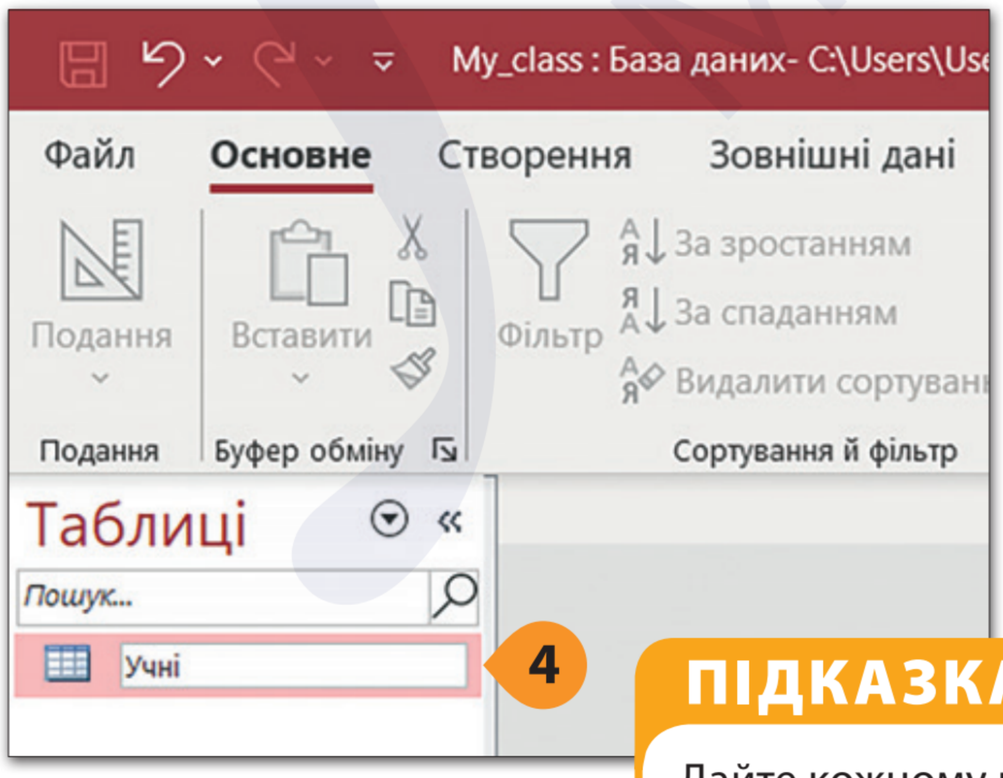
Внесіть зміни до бази даних та зв'яжіть таблиці між собою.

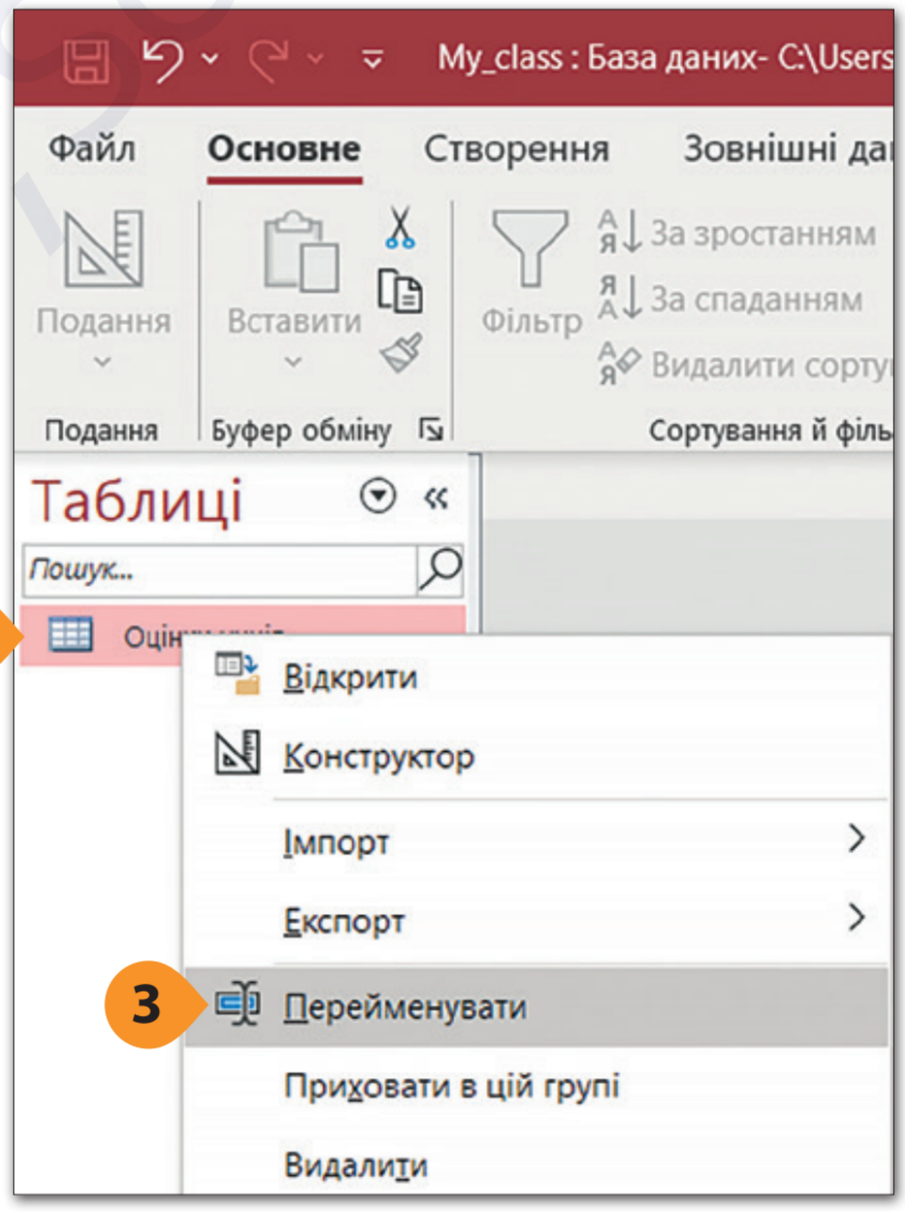


Чи помітили маленький ключ поруч із назвою поля? Цей ключ указує на те, що поле **Номер за порядком** розпізнається як ідентифікатор або **Ключове поле** для кожного запису, який додаєте в таблицю. Ідентифікатор запису унікальний і не може бути однаковим у двох записах.

Щоб перейменувати таблицю:

- натисніть (1), щоб закрити таблицю **Оцінки учнів**;
- на панелі **Таблиці** клацніть правою кнопкою миші по назві таблиці **Оцінки учнів** (2) й оберіть **Перейменувати** (3);
- введіть нову назву таблиці **Учні** (4) та натисніть **Enter ↵**.





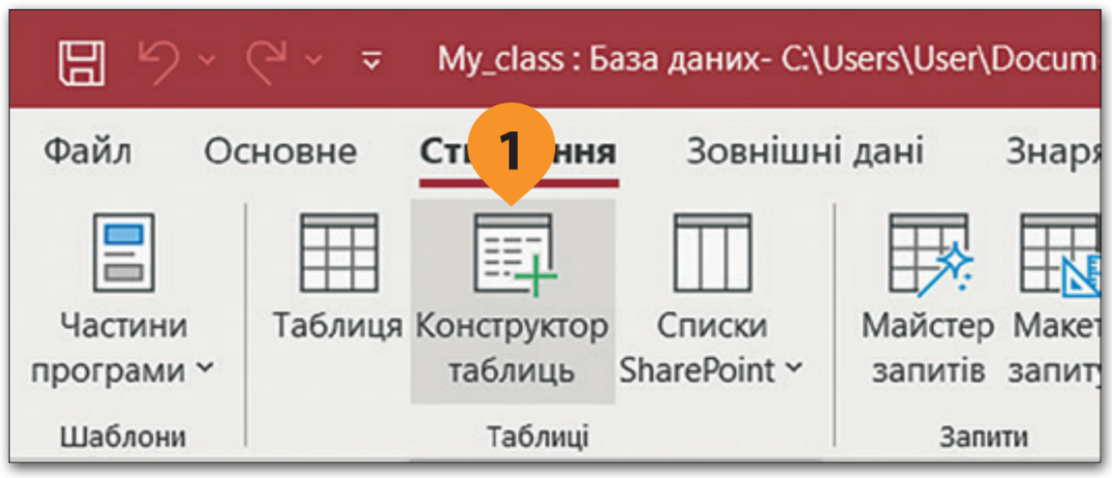
ПІДКАЗКА

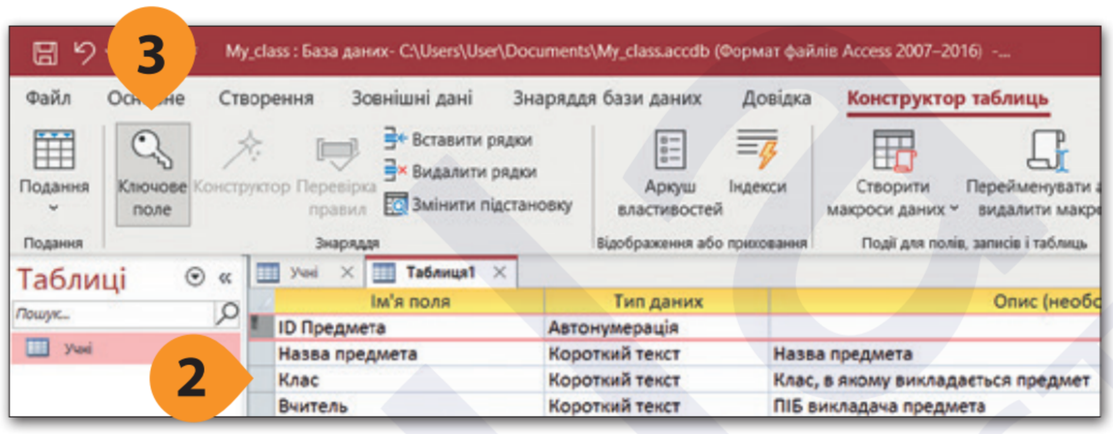
Дайте кожному полю просту та інформативну назву, щоб його зміст був зрозумілий без потреби переглядати дані всередині.

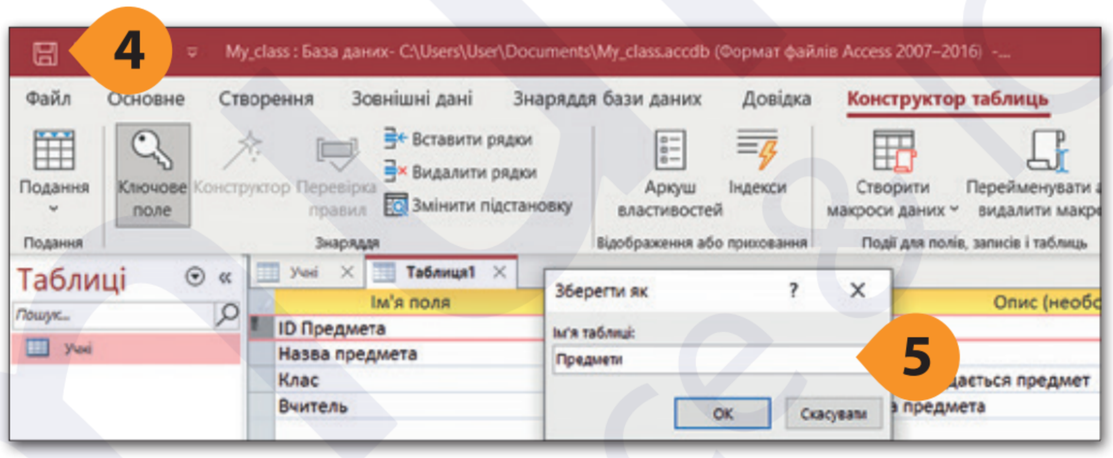
Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

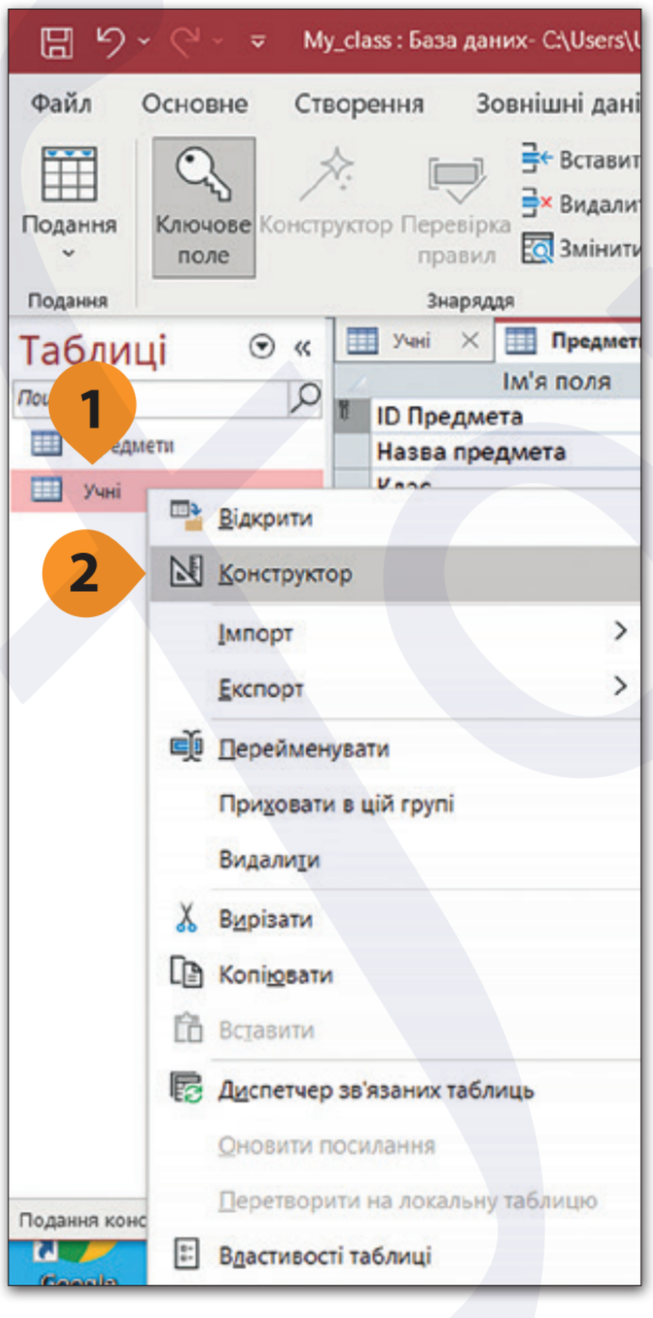
Щоб створити таблицю Предмети:

- на вкладці **Створення** у групі **Таблиці** натисніть **Конструктор таблиць** (1);
- уведіть відповідні **Імена полів** і **Типи даних** (2). Обов'язково встановіть поле **ID_предмета** як **Ключове поле** (3);
- клацніть **Зберегти** (4). Програма запитає назву створеної таблиці. Надрукуйте **Предмети** та натисніть **ОК** (5).



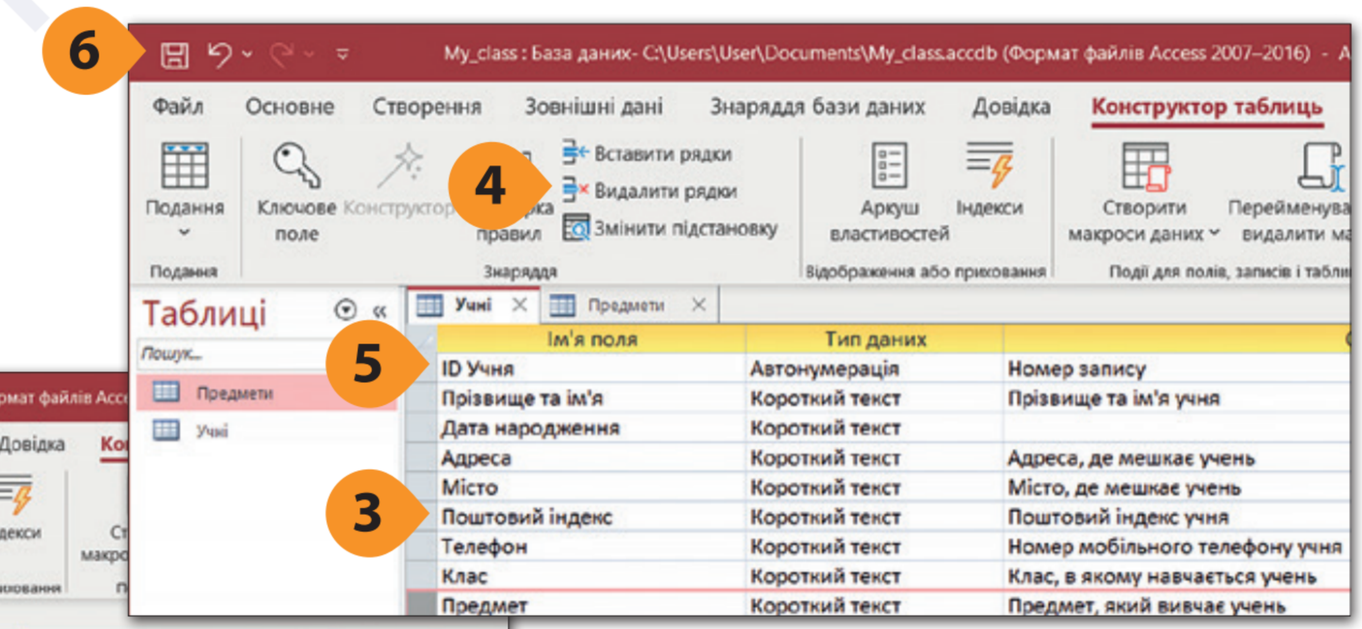


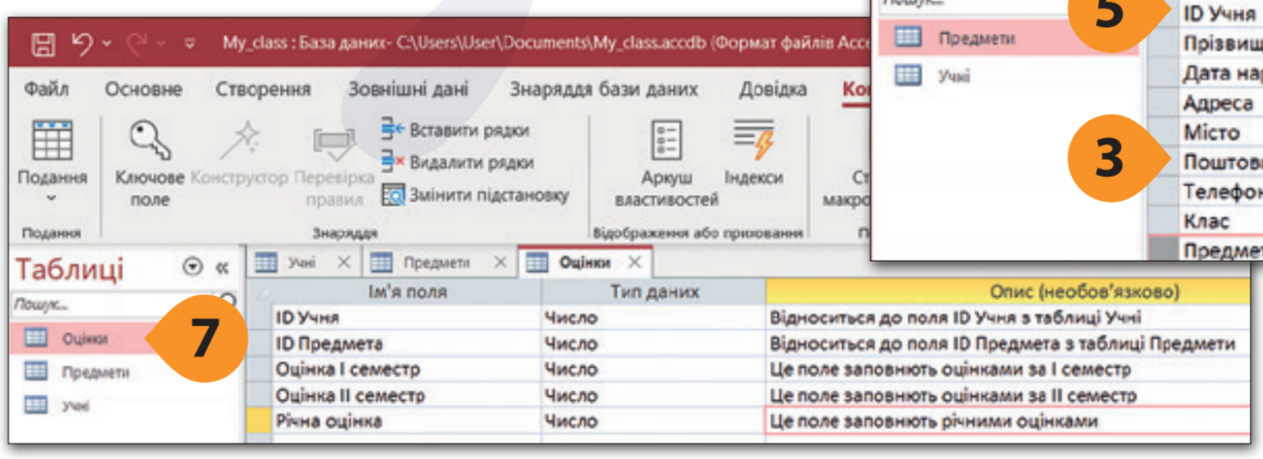




Щоб змінити таблицю Учні й учениці:

- на панелі **Таблиці** клацніть правою кнопкою миші по таблиці **Учні** (1) та оберіть **Конструктор** (2);
- виділіть усі зайві поля (3), які потрібно видалити, й натисніть **Видалити рядки** (4). Перейменуйте перше поле з **№ з/п** на **ID_учня** (5), оскільки це більш відповідна назва;
- натисніть **Зберегти** (6);
- створіть таблицю **Оцінки** (7) самостійно та введіть необхідні поля. **Ключове поле** тут не потрібне.

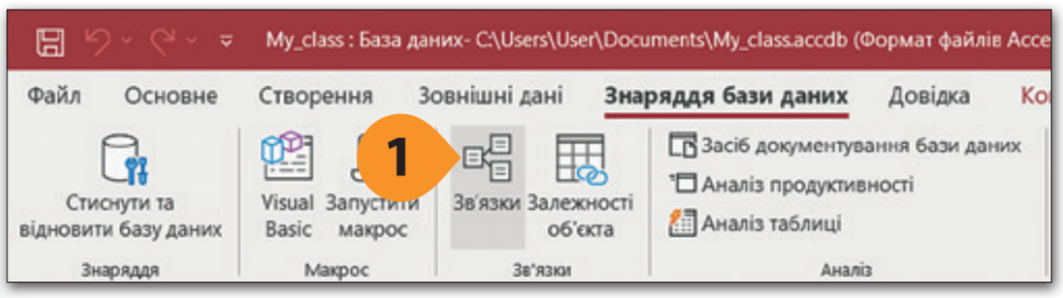




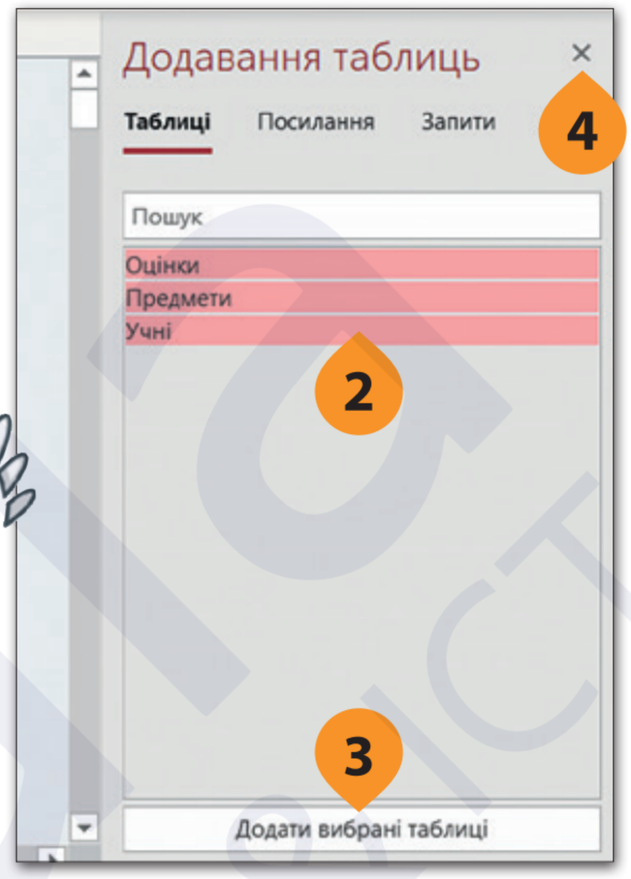
Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Щоб створити зв'язки між таблицями:

- на вкладці **Знаряддя бази даних** у групі **Зв'язки** клацніть **Зв'язки** (1);
- на панелі **Додавання таблиць** виберіть усі таблиці (2), натисніть **Додати вибрані таблиці** (3), клацніть **Закрити панель** (4);
- перетягніть поле **ID_учня** з таблиці **Учні** на поле **ID_учня** таблиці **Оцінки** (5);
- з'явиться вікно **Редагування зв'язків**. Натисніть **Створити** (6);
- у такий самий спосіб перетягніть поле **ID_курсу** з таблиці **Предмети** в поле **ID_курсу** таблиці **Оцінки** (7);
- клацніть **Закрити** (8). Якщо з'явиться відповідний запит, збережіть зміни. Ваші таблиці зв'язані.







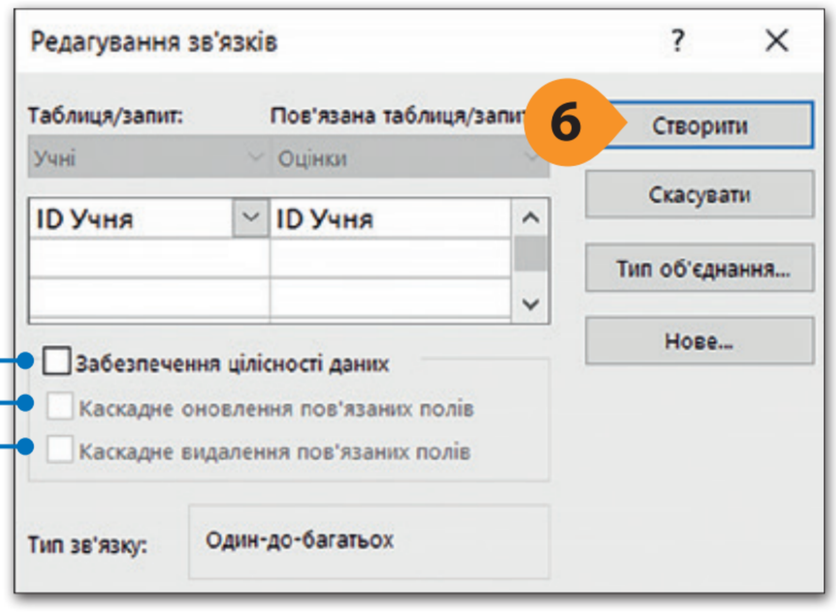


*Щоб зв'язати два поля, вони повинні мати сумісні типи даних. Найпоширеніший випадок – зв'язування поля **Автонумерація** з полем **Число**, оскільки вони є сумісними.*

Забезпечення цілісності даних запобігає появі порожніх записів. Це потрібно для того, щоб не виникало записів, які посилаються на інші записи, яких більше не існує.

Каскадне оновлення пов'язаних полів оновлює всі зв'язані записи, якщо в таблиці змінюється ключове поле.

Каскадне видалення пов'язаних полів видаляє всі зв'язані з ним записи, якщо запис видаляється. Це допомагає уникнути порожніх записів, ні з чим не пов'язаних у базі даних.



Практичне завдання

Завантажте файл-заготовку зі сторінки інтернет-підтримки. Існує багато типів даних, спробуйте не заплутатися в них.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

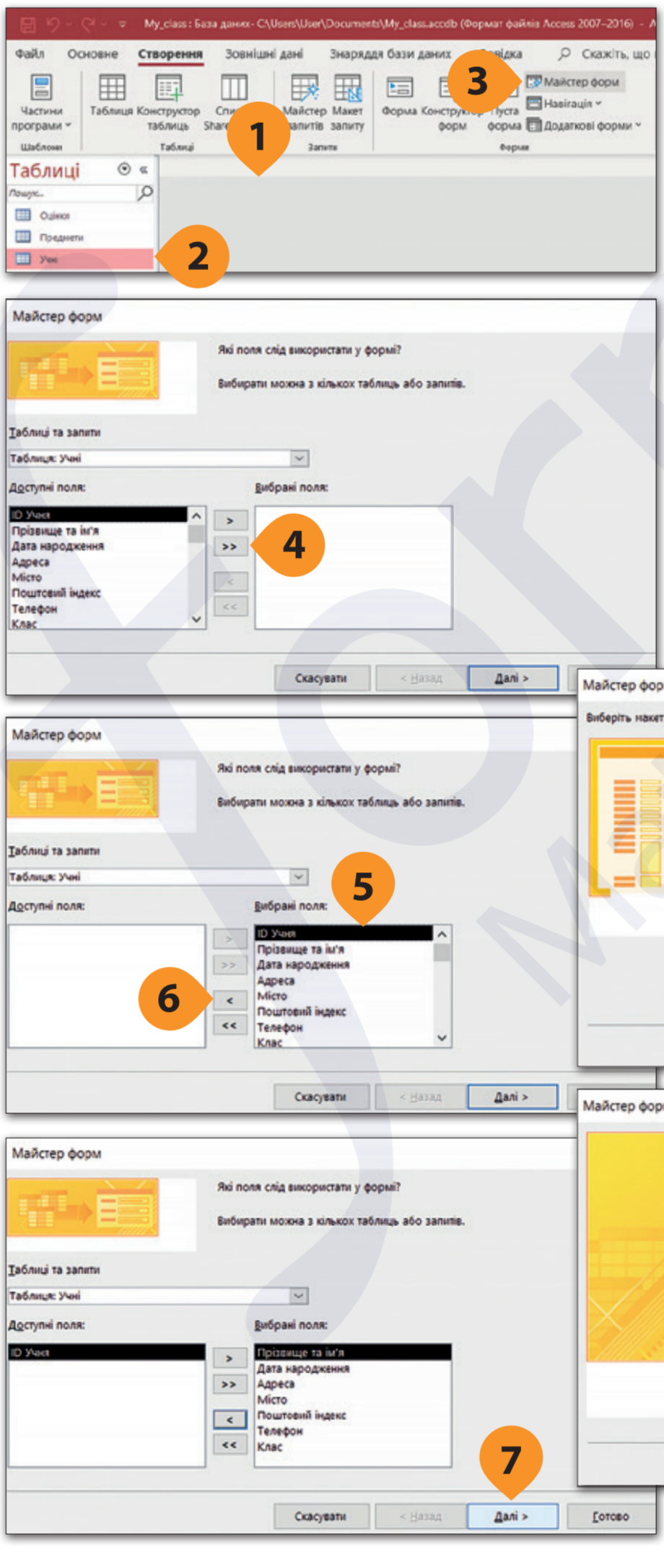
Тема 2

Форми для введення даних

Тепер, коли ви навчилися створювати таблиці в базі даних, можна побудувати форму для введення даних. Вона допоможе додавати інформацію або редагувати вже внесені дані. Форми – це інтерфейс, який ви створюєте для користувачів. Ви завжди можете додати дані безпосередньо в таблиці. Але краще не дозволяти редагувати таблиці тим, у кого не достатньо знань про бази даних. Додайте інформацію у таблицю **Учні** за допомогою форми.

Щоб створити форму за допомогою Майстра форм:

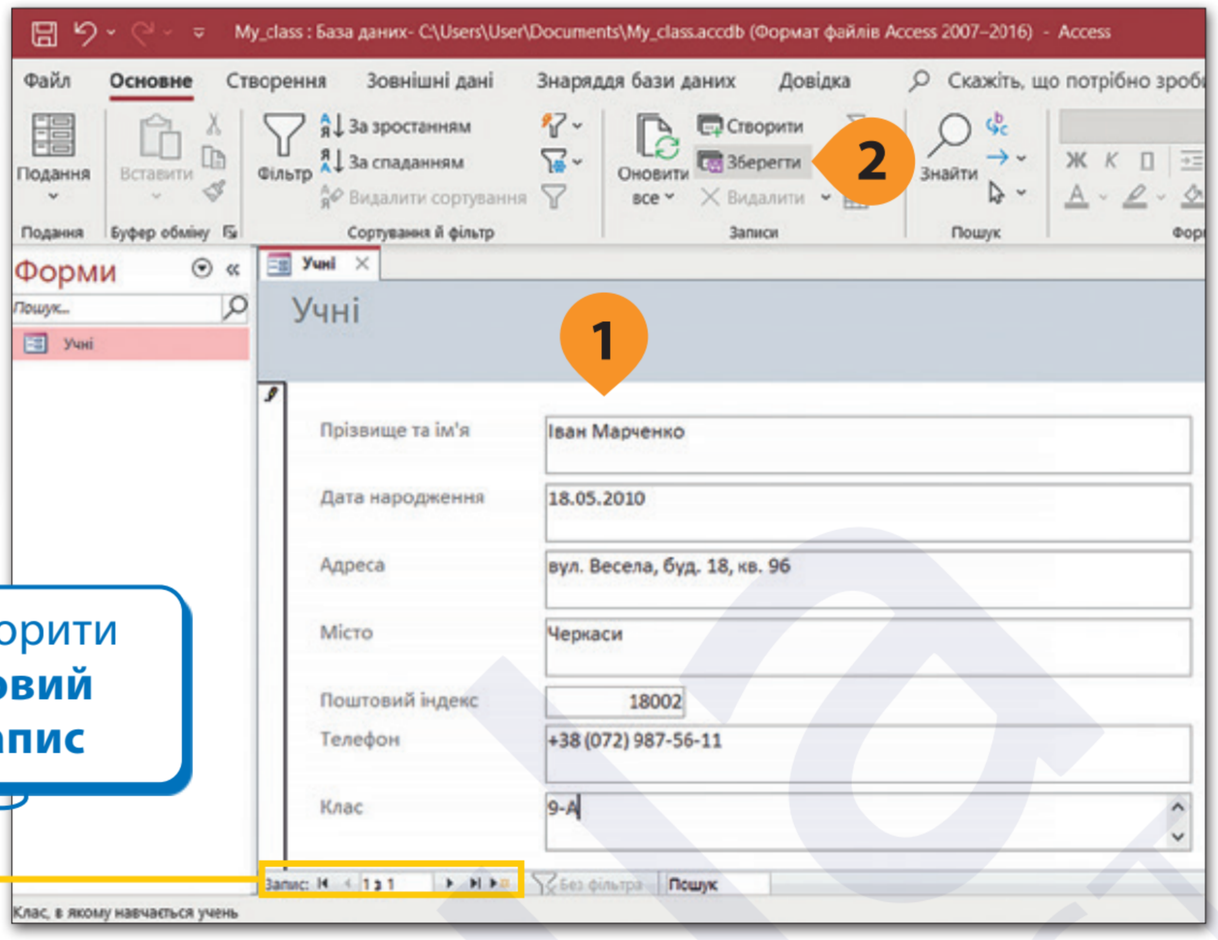
- переконайтеся, що всі об'єкти бази даних закрито (1), і на панелі **Таблиці** оберіть таблицю **Учні** (2);
- на вкладці **Створення** в групі **Форми** натисніть **Майстер форм** (3);
- у вікні **Майстра форм** натисніть **>>** (4), щоб включити всі поля у форму;
- щоб у формі не з'являлося поле **ID_учня** (воно не містить важливих даних й оновлюється автоматично), видаліть його, обравши у списку **Вибрані поля** (5) та клацніть **<** (6);
- натисніть **Далі** (7);
- у наступному вікні залиште макет **Стовпці** та натисніть **Далі** (8);
- клацніть **Готово** (9).

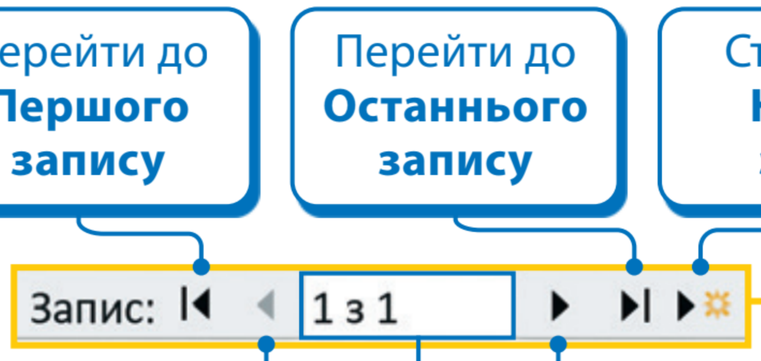


Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Щоб додати записи у форму:

- заповніть форму, що створили (1);
- коли ви закінчите, на вкладці **Основне** в групі **Записи** натисніть **Зберегти** (2).





Перейти до **Попереднього запису**

Відображає номер **Поточного запису**

Перейти до **Наступного запису**

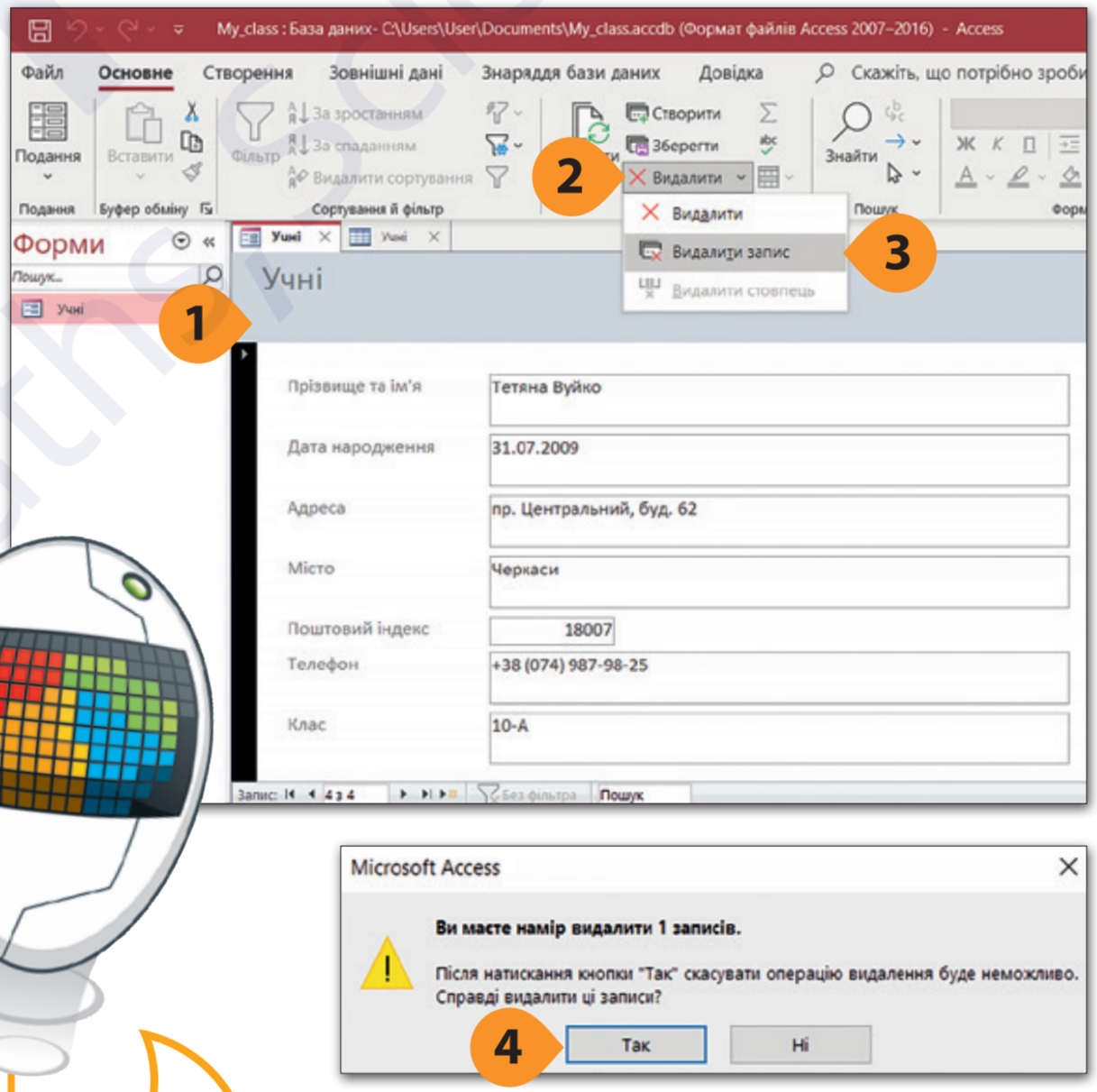
Учні

№ з/г	Прізвище та ім'я	Дата народ	Адреса	Місто	Поштовий інде	Телефон	Клас
1	Іван Марченко	18.05.2010	вул. Весела, буд. 18, кв. 96	Черкаси	18002	+38 (072) 987-56-11	9-А
2	Маричка Васько	20.12.2009	вул. Маркарова, буд.3	Черкаси	18004	+38 (052) 561-78-99	9-Б
3	Павло Іванченко	01.02.2010	вул. Соборна, буд. 9б, кв. 22	Черкаси	18005	+38 (072) 456-98-98	9-А
4	Тетяна Вуйко	31.07.2009	пр. Центральний, буд. 62	Черкаси	18007	+38 (074) 987-98-25	10-А
(Новий)					0		

Так само як ви вводите дані безпосередньо у форму, можна внести дані в таблицю. Просто відкрийте таблицю **Учні** та надрукуйте ці записи.

Щоб видалити запис:

- перейдіть до запису, який будете видаляти, наприклад четвертий, і натисніть селектор запису поруч із ним (1);
- на вкладці **Основне** в групі **Записи** клацніть стрілку вниз поруч із пунктом **Видалити** (2), а потім натисніть **Видалити запис** (3);
- у вікні з попередженням, що з'явиться, натисніть **Так** (4).



ПІДКАЗКА

Для швидкого заповнення форми можна натискати **Tab** після введення кожного поля, щоб перейти до наступного. Якщо натиснути **Tab** під час роботи з останнім полем форми, програма перейде до наступного рядка, щоб додати ще один запис.

*Оскільки поле **ID_учня / учениці** має автоматичну нумерацію, коли ви видаляєте запис, його номер не замінюється. Якщо ви видалите останній запис, наприклад **4**, наступним буде **5**.*

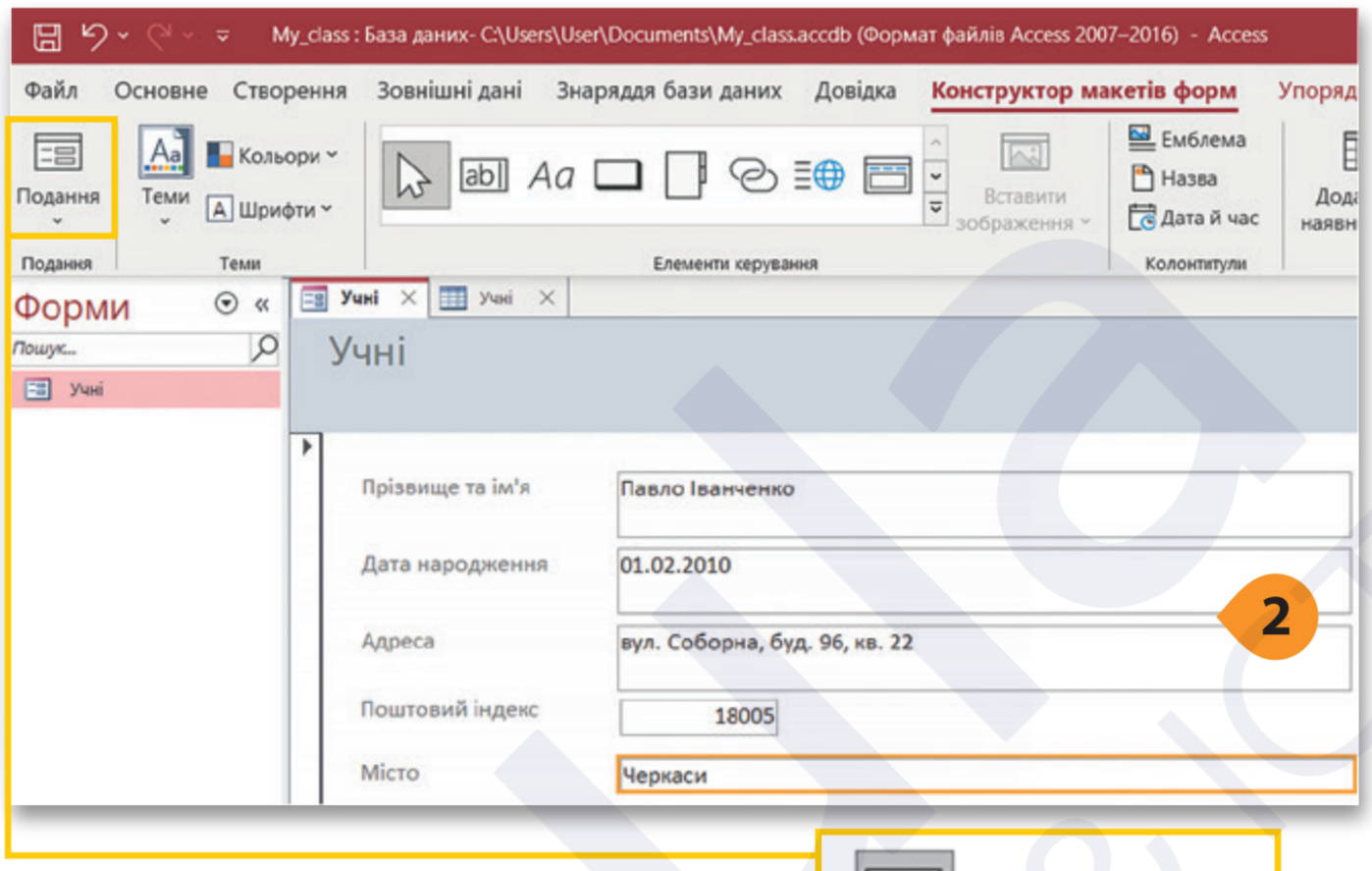
Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Персоналізуйте свої форми

Налаштувати власну форму можна різними способами. Наприклад, додавати чи змінювати поля та підписи.

Щоб змінити макет вашої форми:

- на вкладці **Основне** в групі **Подання** клацніть **Режим розмітки** (1);
- у **Режимі розмітки** можна змінювати розмір і перетягувати всі поля, а також розташовувати їх у потрібному вам порядку (2).

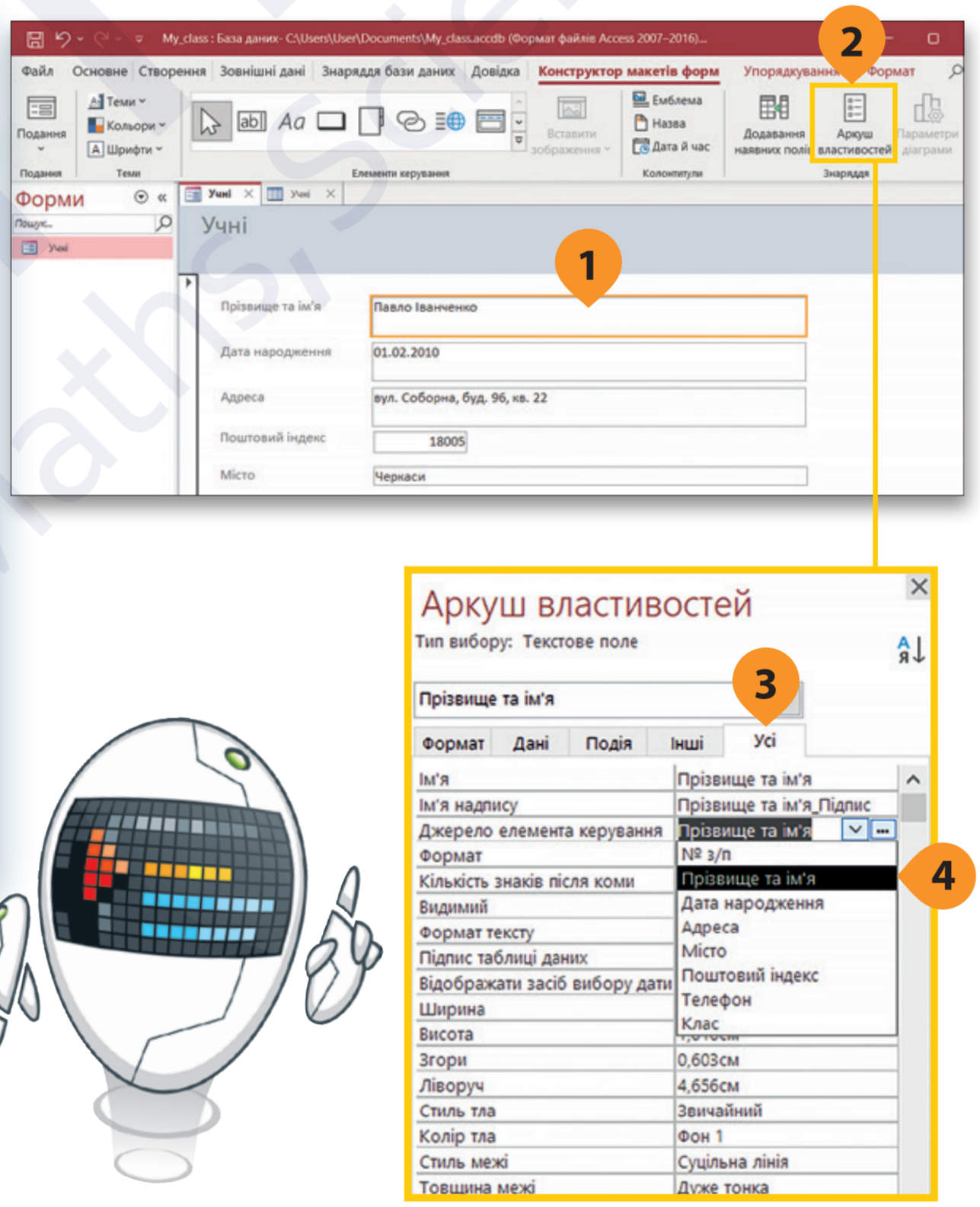


Змініть джерело даних

Ви також можете змінити **Джерело даних** для **текстових полів**. Так можна обрати певне поле в таблиці, куди зберігатиметься текст, який буде введений у **текстовому полі** форми.

Щоб змінити джерело даних для текстового поля:

- клацніть текстове поле, щоб вибрати його, наприклад текстове поле, що містить **Павло Іванченко** (1);
- на вкладці **Конструктор макетів форм** у групі **Знаряддя** натисніть **Аркуш властивостей** (2);
- на панелі **Аркуш властивостей** на вкладці **Усі** (3) клацніть на список **Джерело елемента керування** та оберіть поле, яке потрібно зв'язати з цим текстовим полем, наприклад **Ім'я** (4).

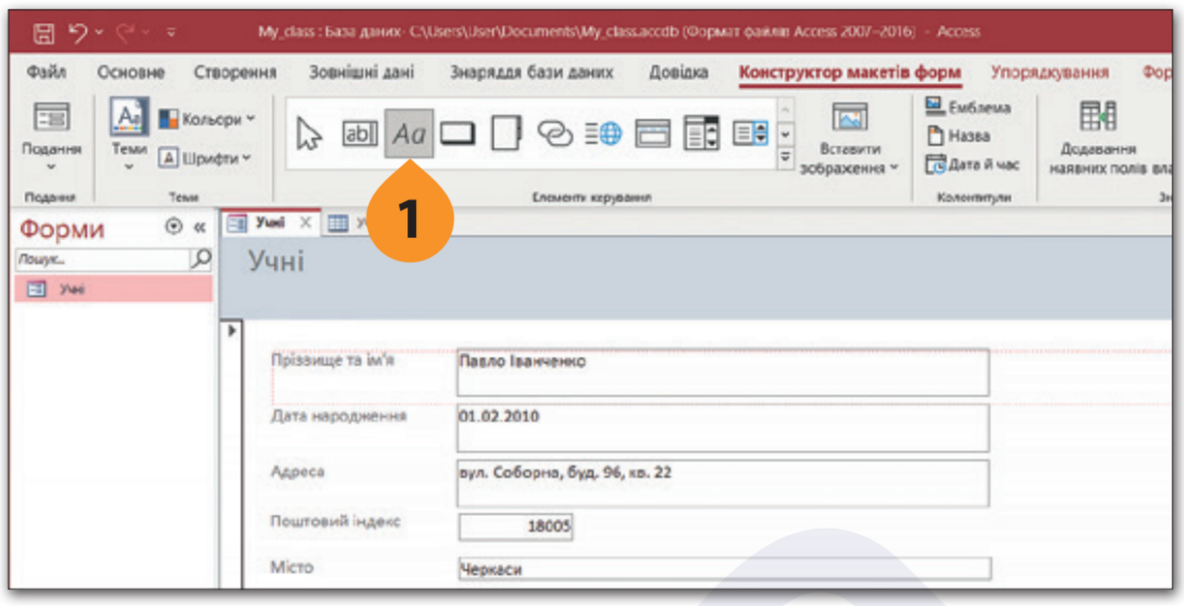


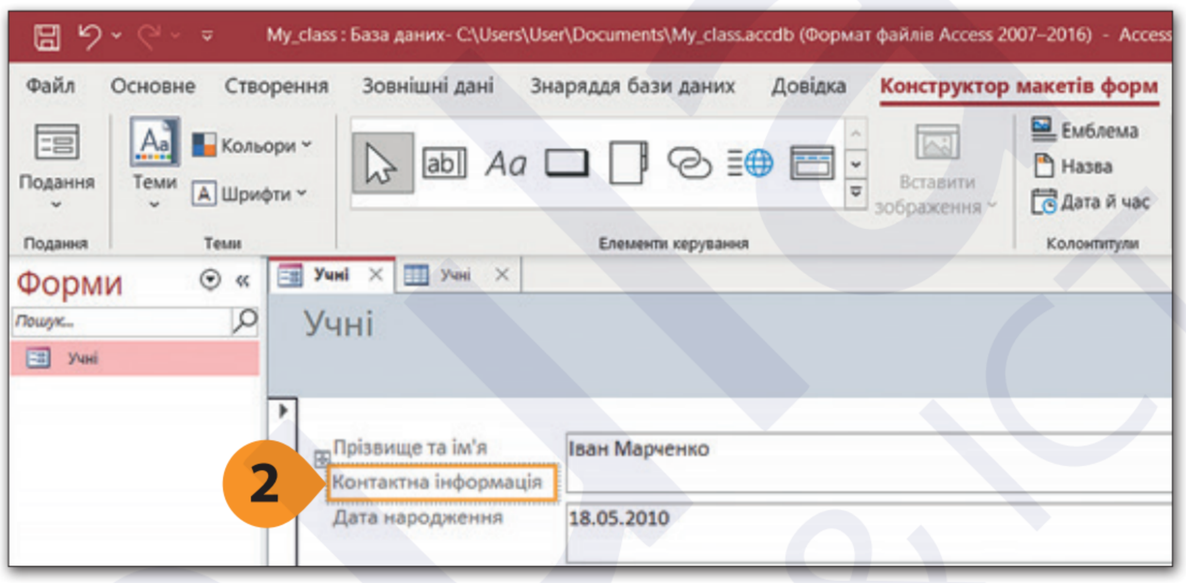
Тепер усе, що буде написано в цьому текстовому полі, буде збережено у вибраному полі таблиці бази даних.

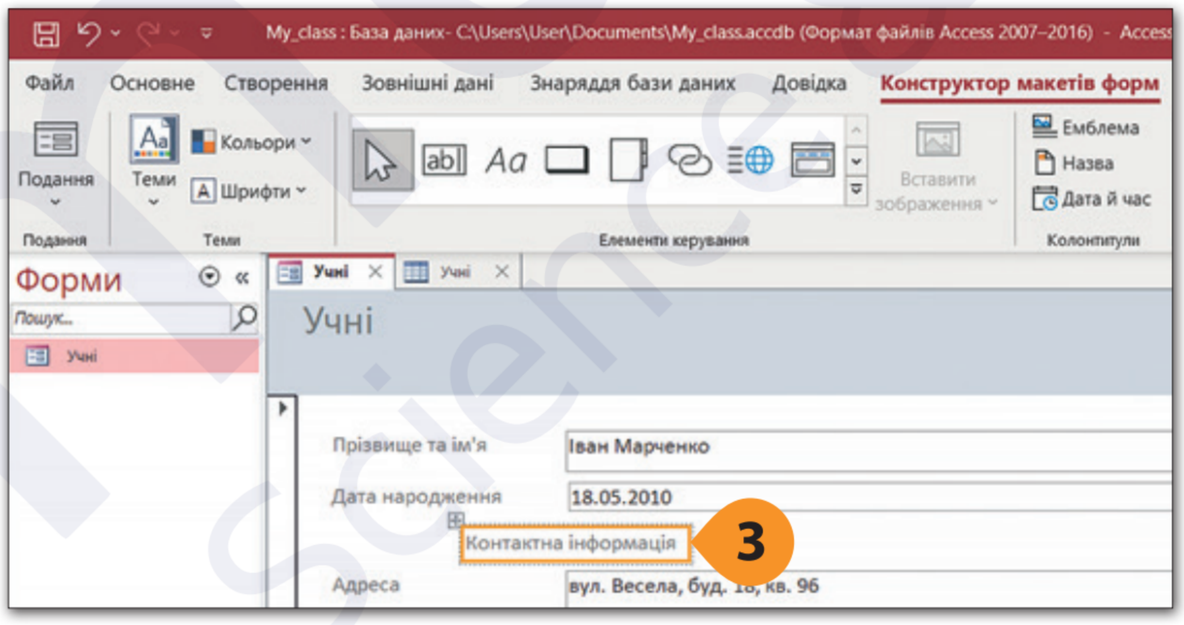
Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Щоб додати підпис:

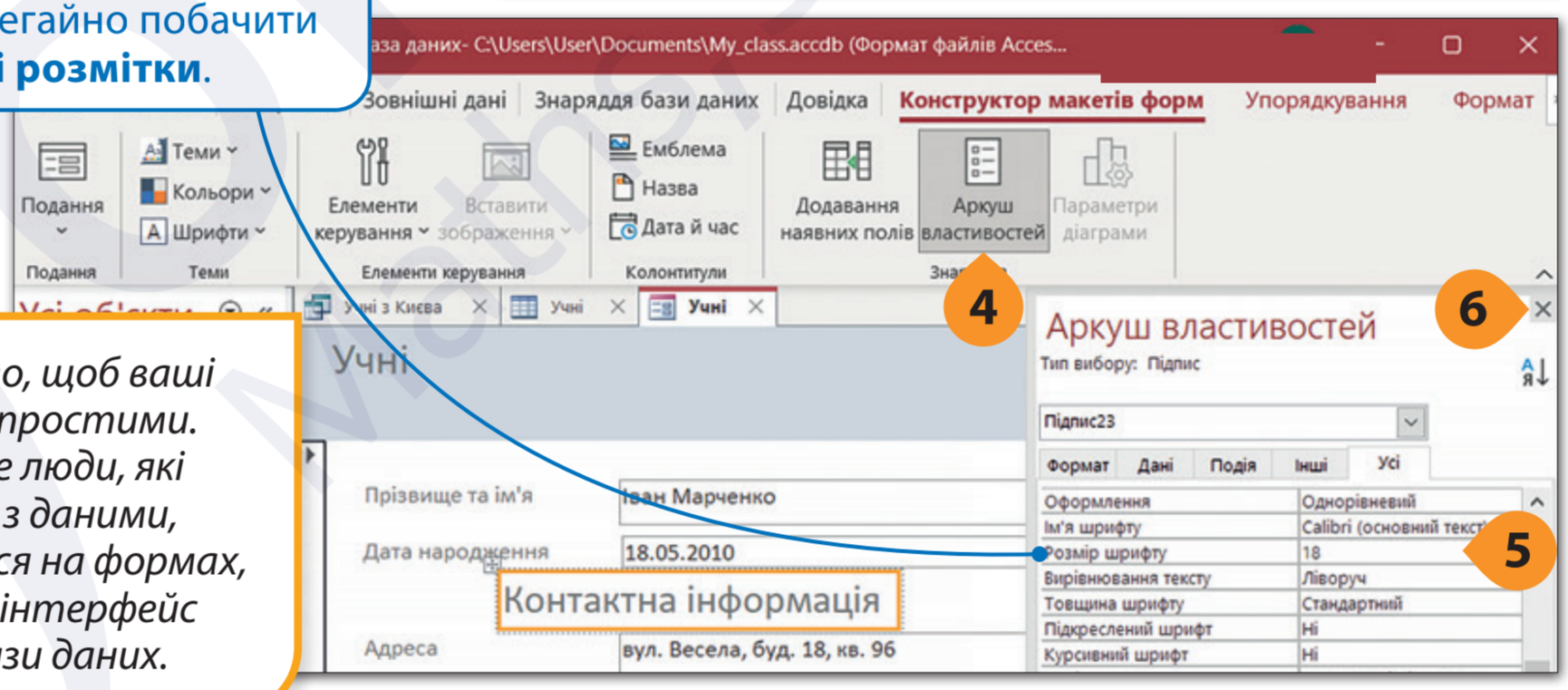
- на вкладці **Конструктор макетів форм** у групі **Елементи керування** натисніть **Підпис** (1);
- клацніть будь-де у формі. З'явиться нове поле підпису. Введіть **Контактна інформація** (2);
- перетягніть **Поле підпису** (3) над даними контактної інформації;
- вибравши **Поле підпису**, на вкладці **Конструктор макетів форм** у групі **Знаряддя** клацніть **Аркуш властивостей** (4);
- на панелі **Аркуш властивостей** змініть розмір шрифту (5) на **18**;
- закрийте **Аркуш властивостей** (6) та збережіть форму.







Можна поекспериментувати та змінити й інші властивості та подивитися, що станеться. Зміни можна негайно побачити у **Режимі розмітки**.



Дуже важливо, щоб ваші форми були простими. Найчастіше люди, які працюють з даними, зосереджуються на формах, тому що це інтерфейс будь-якої бази даних.

Практичне завдання

Завантажте файл-заготовку зі сторінки інтернет-підтримки. Створіть спортивну таблицю, подібну до наведеної у файлі, і заповніть дані за допомогою власної форми введення даних.

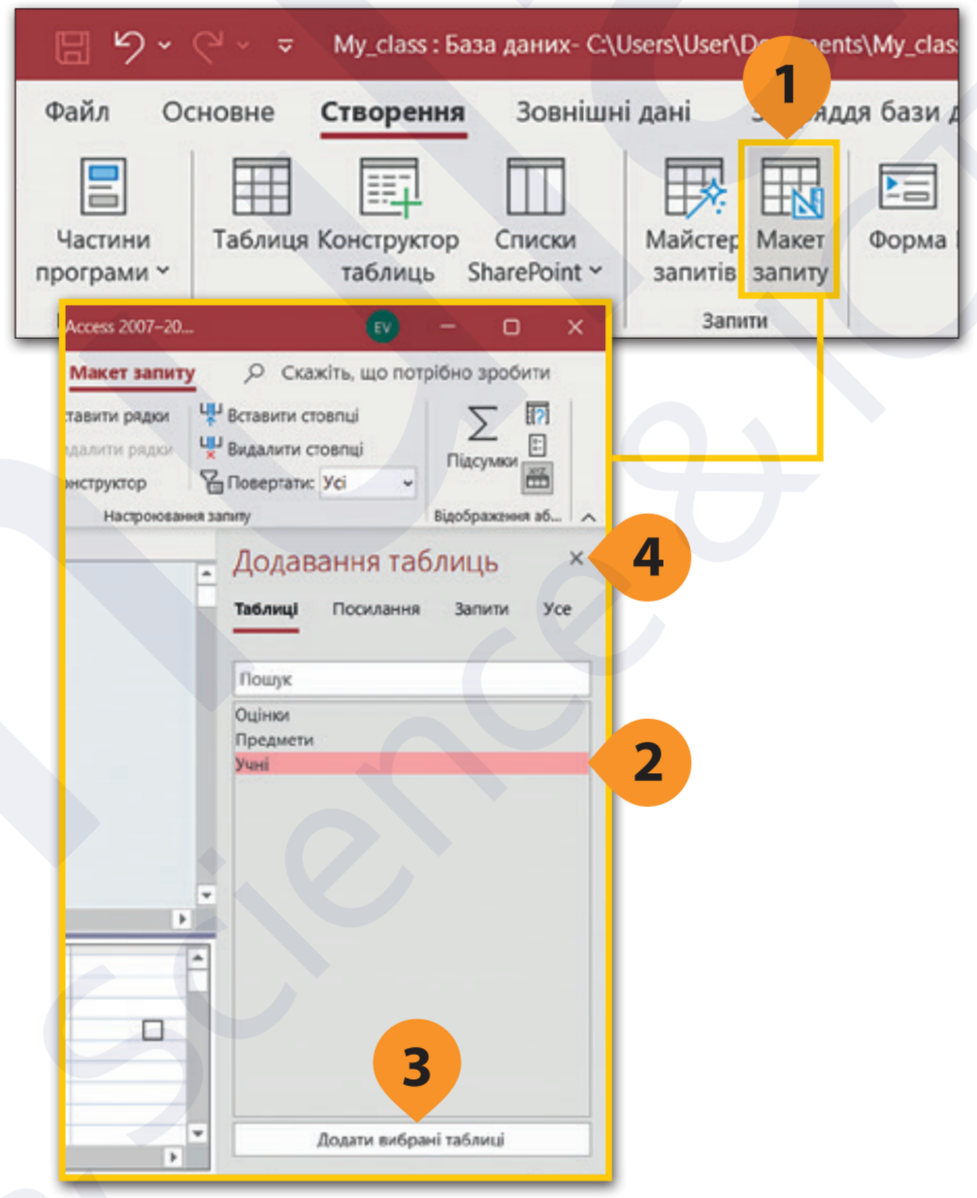
Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

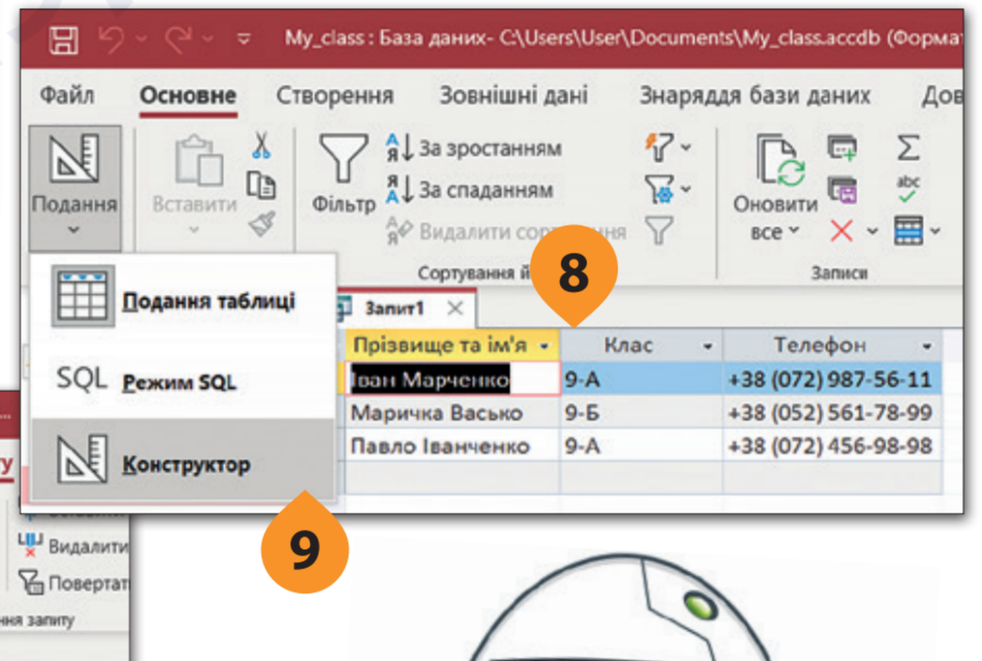
Тема 3
Запити

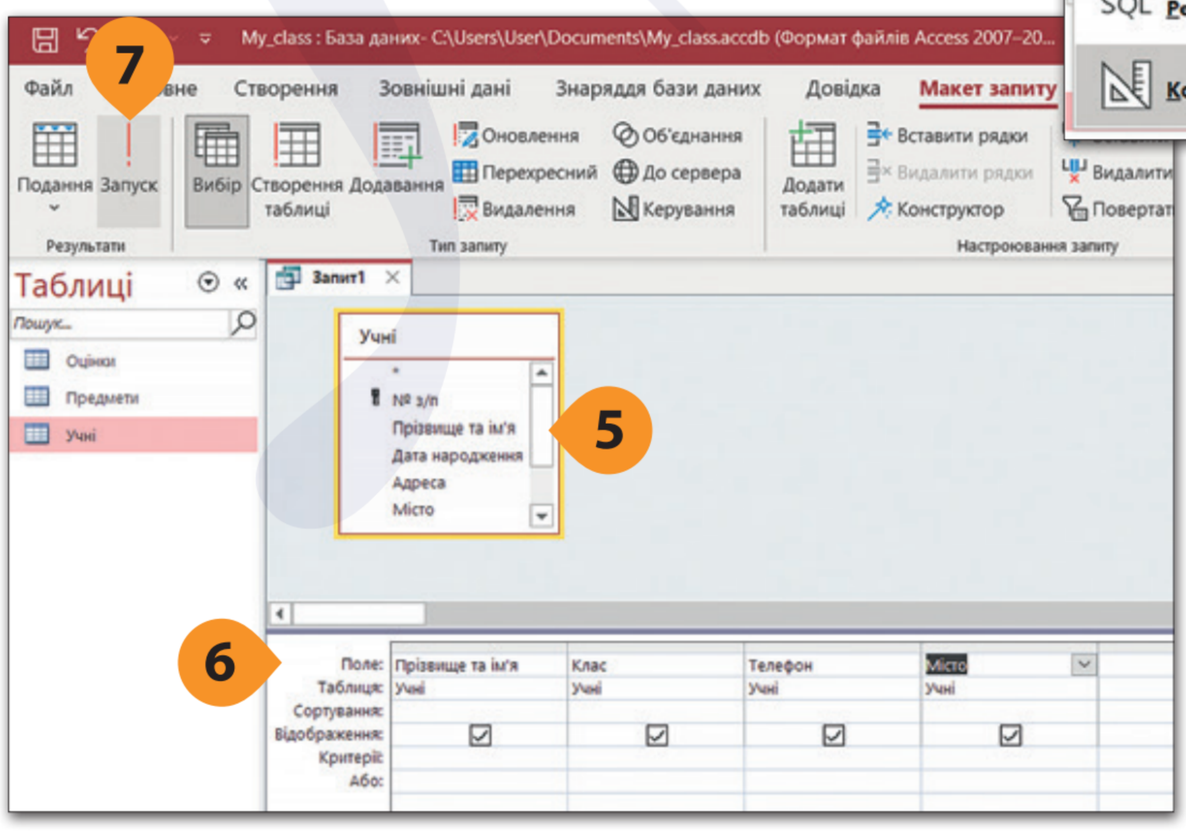
Таблиці бази даних можуть містити багато записів, у деяких випадках мільйони чи навіть мільярди! Щоб знайти певну інформацію, потрібно відфільтрувати записи й вибрати ті, які слід неодмінно відобразити. Для цього варто створити запит. **Запит** – це спеціальне «питання», з яким звертаються до бази даних, щоб знайти певні дані та отримати потрібну інформацію. Для виконання цього завдання продовжіть роботу з таблицею **Учні й учениці**.

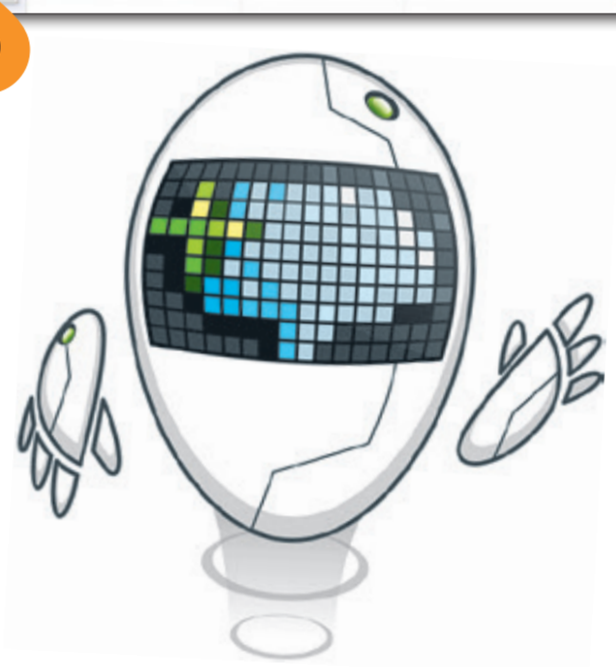
Щоб створити запит:

- на вкладці **Створення** в групі **Запити** клацніть **Макет запиту** (1);
- на панелі **Додавання таблиць** клацніть таблицю **Учні** (2) та натисніть **Додати вибрані таблиці** (3). Закрийте панель (4);
- відкриється вкладка **Конструктор запитів** разом з обраною таблицею. Двічі клацніть по полях таблиці **Учні**, які треба відобразити (у цьому прикладі **Прізвище, Клас, Номер_телефону** та **Місто**) (5). Вибрані поля з'являться в нижній частині вікна (6);
- на вкладці **Макет запиту** в групі **Результати** натисніть **Запуск** (7). У вікні **Перегляд таблиці** (8) з'являться результати вашого запиту;
- на вкладці **Основне** в групі **Подання** клацніть **Конструктор** (9), щоб повернутися до подання **Конструктора запиту**. У ньому можна продовжити налаштування запиту іншими способами.









Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

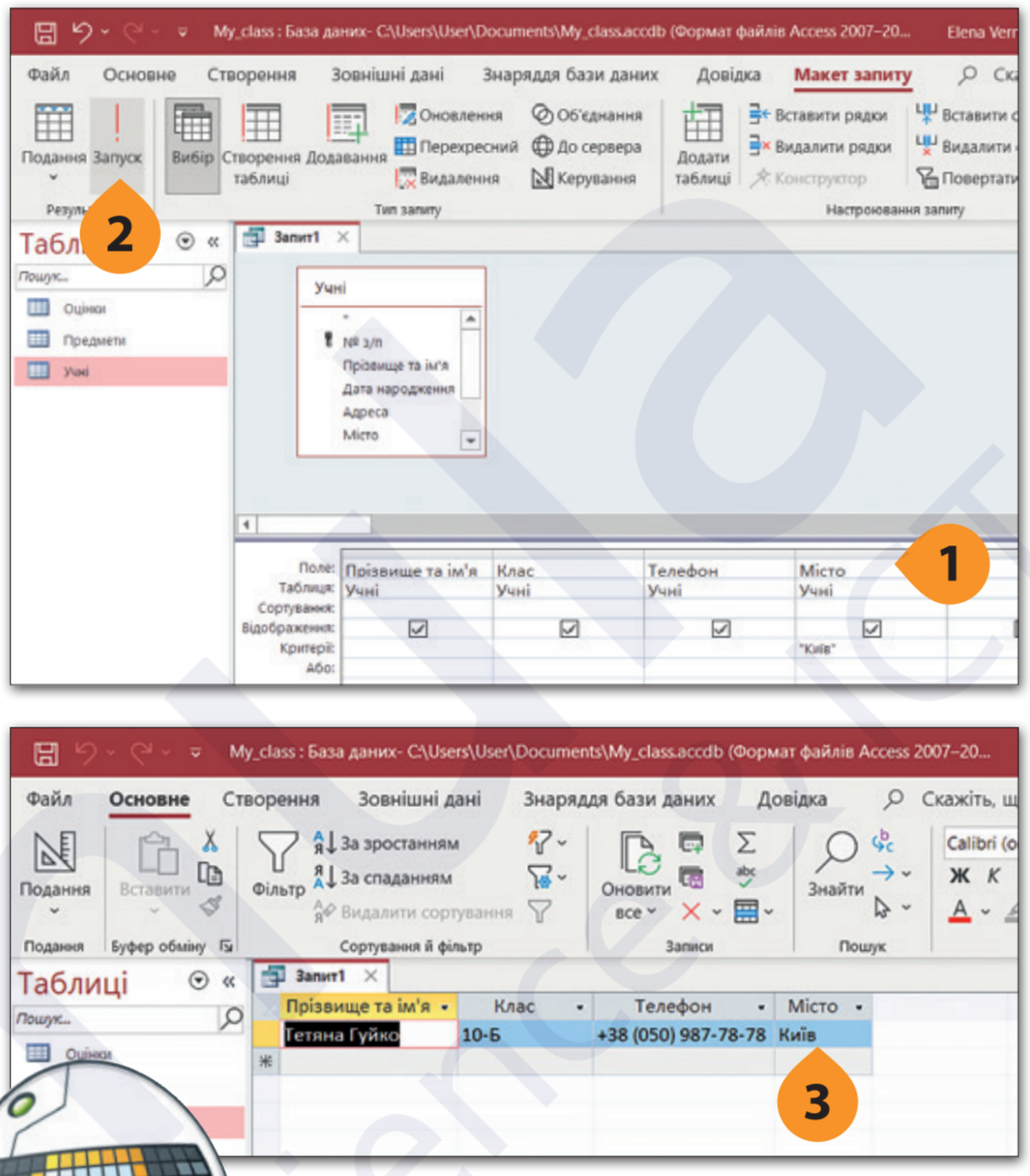
Додаємо критерії

Виберіть деякі з ваших даних для відображення в результатах запиту. Сформулюйте запитання до бази даних.

Мета завдання – запитати програму: «Дізнатися імена, класи та номери телефонів усіх учнів та учениць, які живуть у Києві».

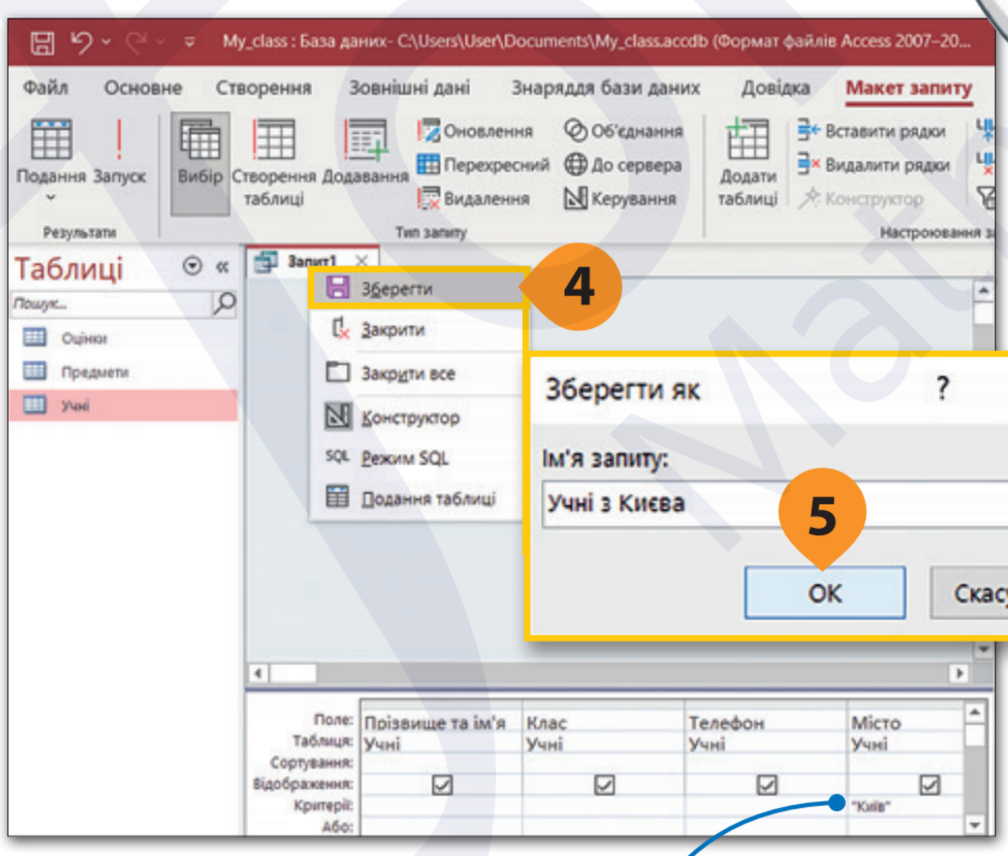
Щоб вибрати деякі дані:

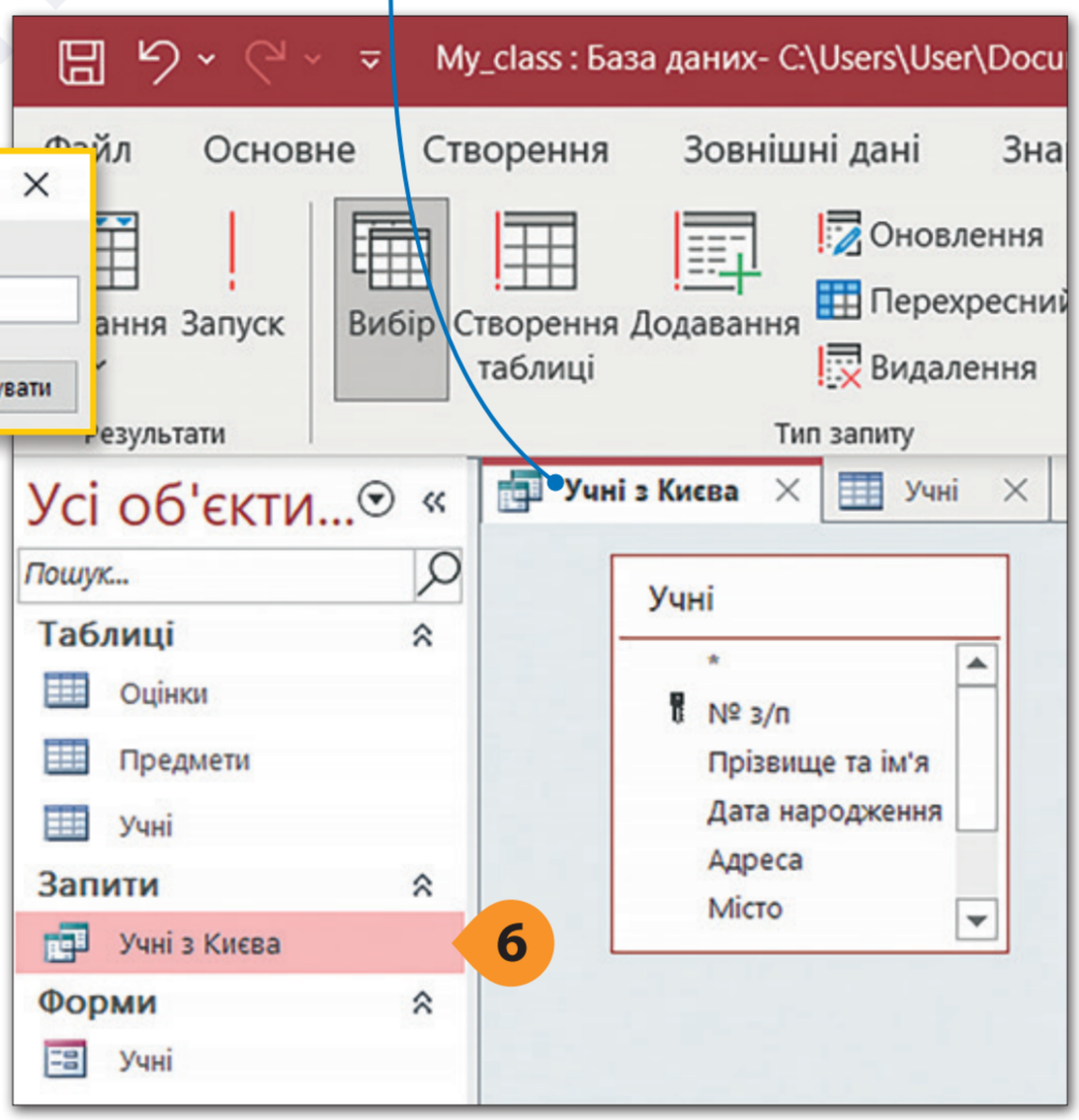
- у поданні **Конструктор** знайдіть стовпчик поля **Місто**, а в рядку **Критерії** введіть **«Київ»** (1);
- на вкладці **Макет запиту** у групі **Результати** натисніть **Запуск** (2);
- як видно з результатів (3), створений запит наказав базі даних вибрати та відобразити лише ті записи, які містять слово **«Київ»** у полі **Місто**;
- збережіть його, клацнувши правою кнопкою миші на вкладці запиту та обравши **Зберегти** (4);
- уведіть назву файлу, наприклад **Учні**, та натисніть **ОК** (5);
- збережений запит з'явиться в **Області переходів** (6).





Запит можна зберегти в будь-який момент, бажано після того, як виконаєте його та переконаєтеся, що він відображає потрібні результати.





Зверніть увагу, що лапки тут не обов'язкові, але рекомендується брати в лапки будь-який текст у рядку **Критерії**. Так програма зрозуміє, що ви маєте на увазі саме слово «Київ», а не щось інше, наприклад, поле з назвою Київ.

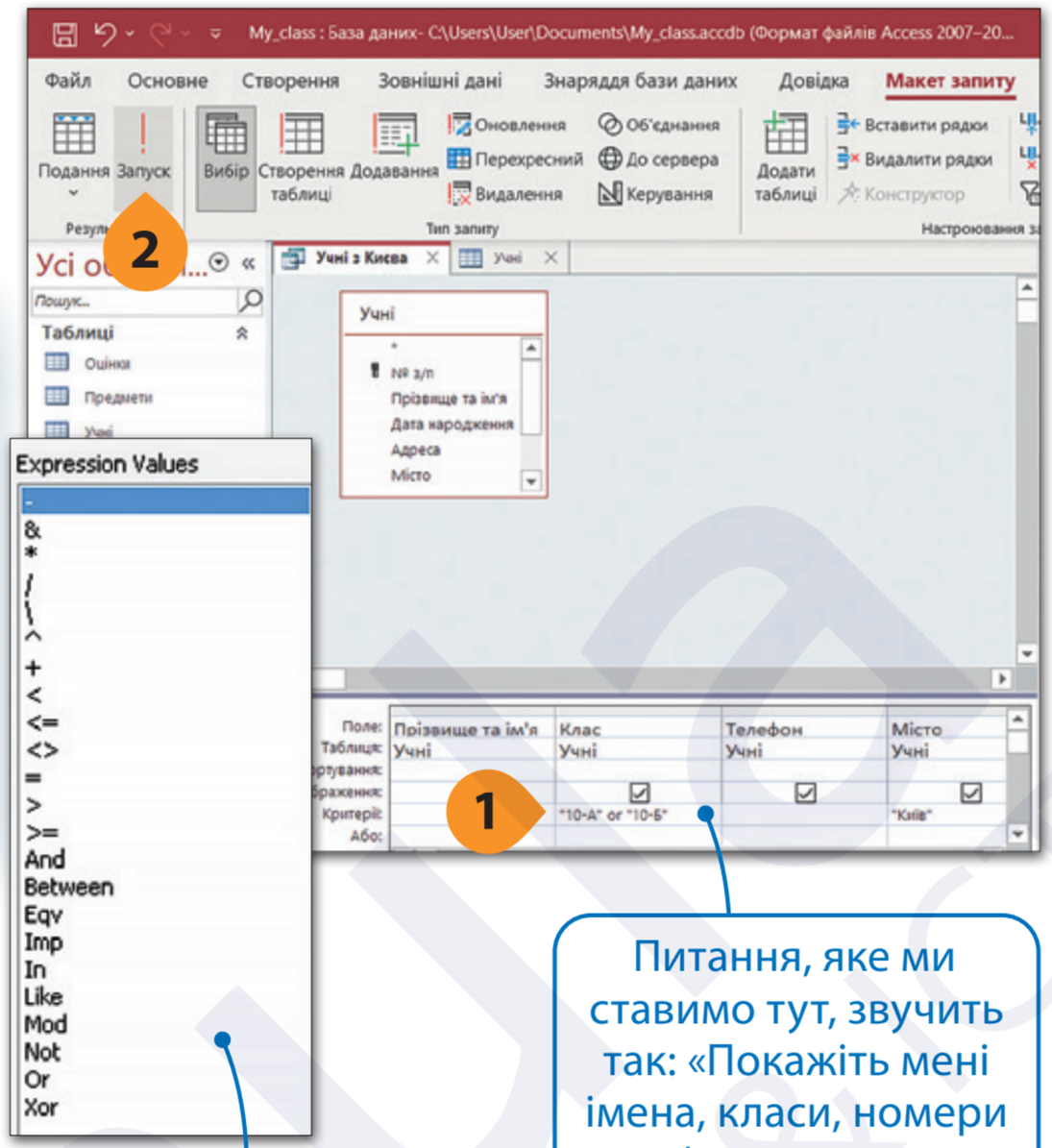
Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

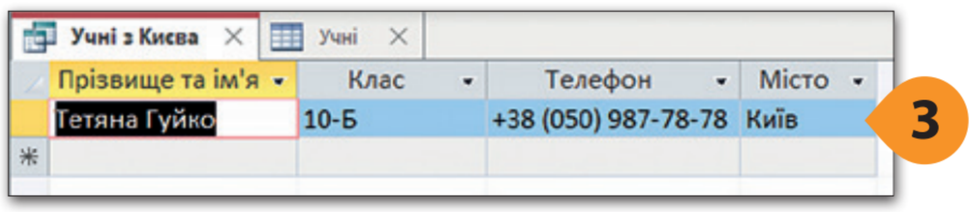
Складні критерії

Можна ще більше звузити результати запиту, застосувавши більш складні критерії до кількох полів.

Щоб застосувати складні критерії:

- у режимі **Конструктора запиту** знайдіть стовпчик поля **Клас** і в рядку **Критерії** введіть **10-А or 10-Б** (1);
- на вкладці **Макет запиту** у групі **Результати** натисніть **Запуск** (2);
- перегляньте результати (3).





Тут наведено оператори (наприклад, **Or**, який уже було використано), якими можна користуватися у виразах критеріїв. Комбінуючи їх, можна створити будь-які критерії – від простих до дуже складних.

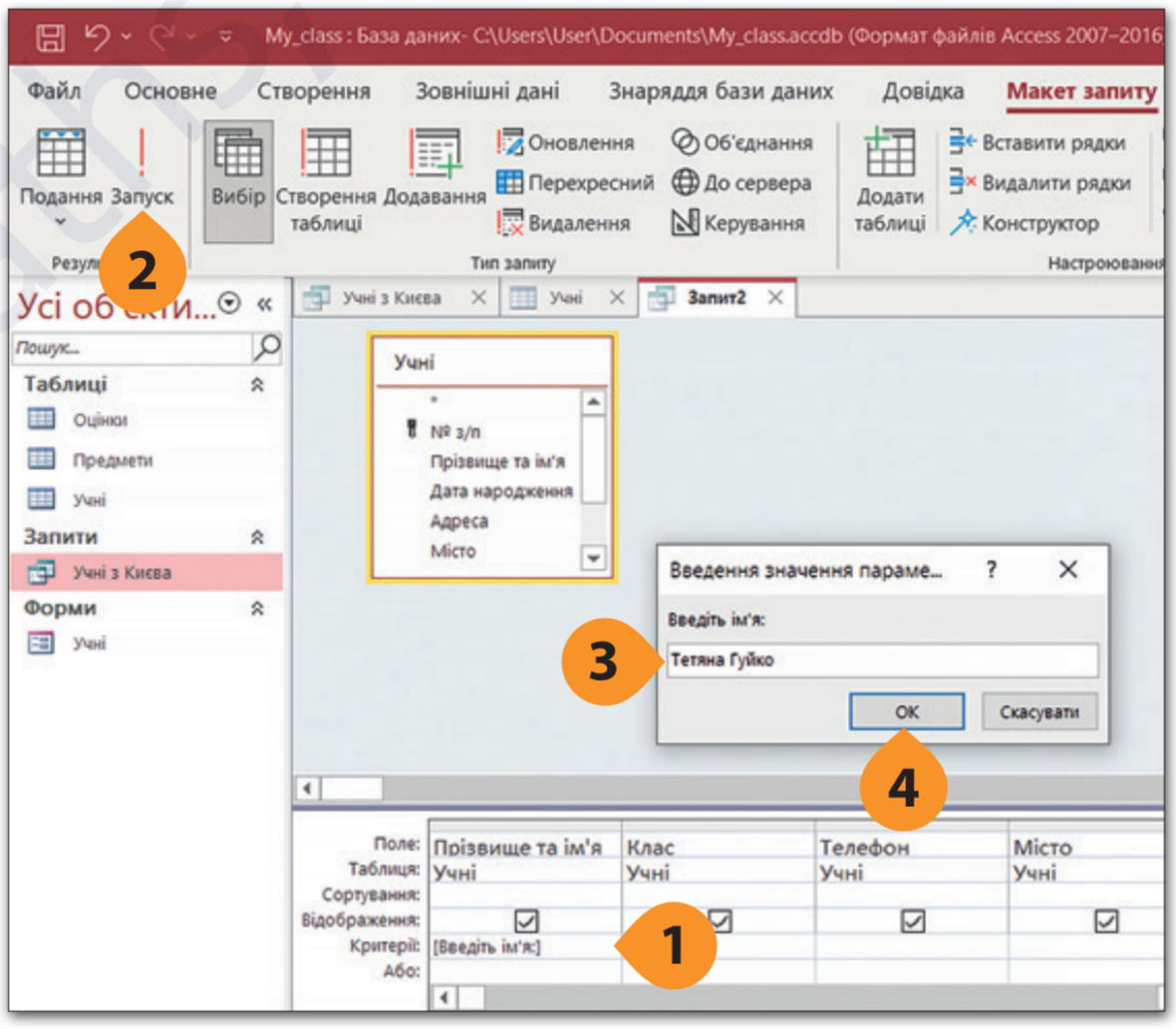
Питання, яке ми ставимо тут, звучить так: «Покажіть мені імена, класи, номери телефонів і міста всіх учнів та учениць, які живуть у Києві та навчаються в 10-А або 10-Б».

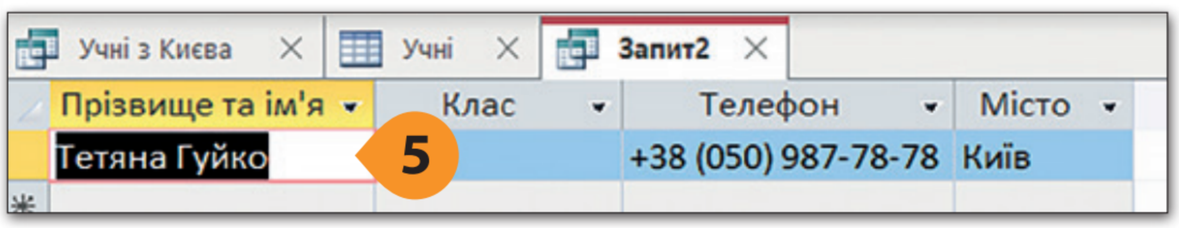
Параметри запитів

Зробимо щось цікавіше. Припустимо, треба створити запит, який надасть контактну інформацію конкретного учня або учениці. Це простіше, ніж можна подумати. Просто введіть ім'я учня або учениці в рядку **Критерії** поля **Ім'я.** Але коли база даних велика й налічує сотні дітей? Непрактично створювати новий запит щоразу, коли треба знайти іншого учня або учениці. У цьому випадку краще використовувати **Запит із параметрами**.

Щоб створити запит із параметрами:

- у поданні **Структура запиту** знайдіть стовпчик поле **Прізвище та Ім'я** і в рядку **Критерії** введіть **[Введіть ім'я:]** (1). Очистіть інші **Критерії**, якщо вони є;
- на вкладці **Макет запиту** у групі **Результати** натисніть **Запуск** (2);
- з'явиться вікно (3) з повідомленням і запитом для введення імені учня або учениці. Введіть ім'я, наприклад **Тетяна Гуйко**, натисніть **ОК** (4);
- можна використовувати той самий запит і вводити різні імена дітей (5), нічого не змінюючи та не створюючи нового запиту.





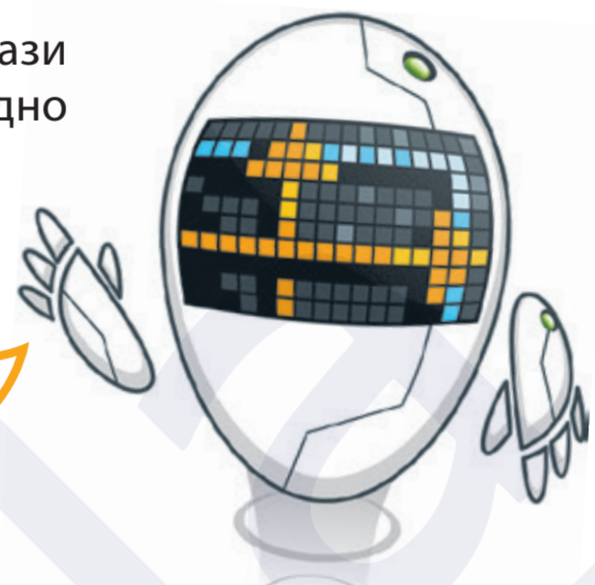
Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

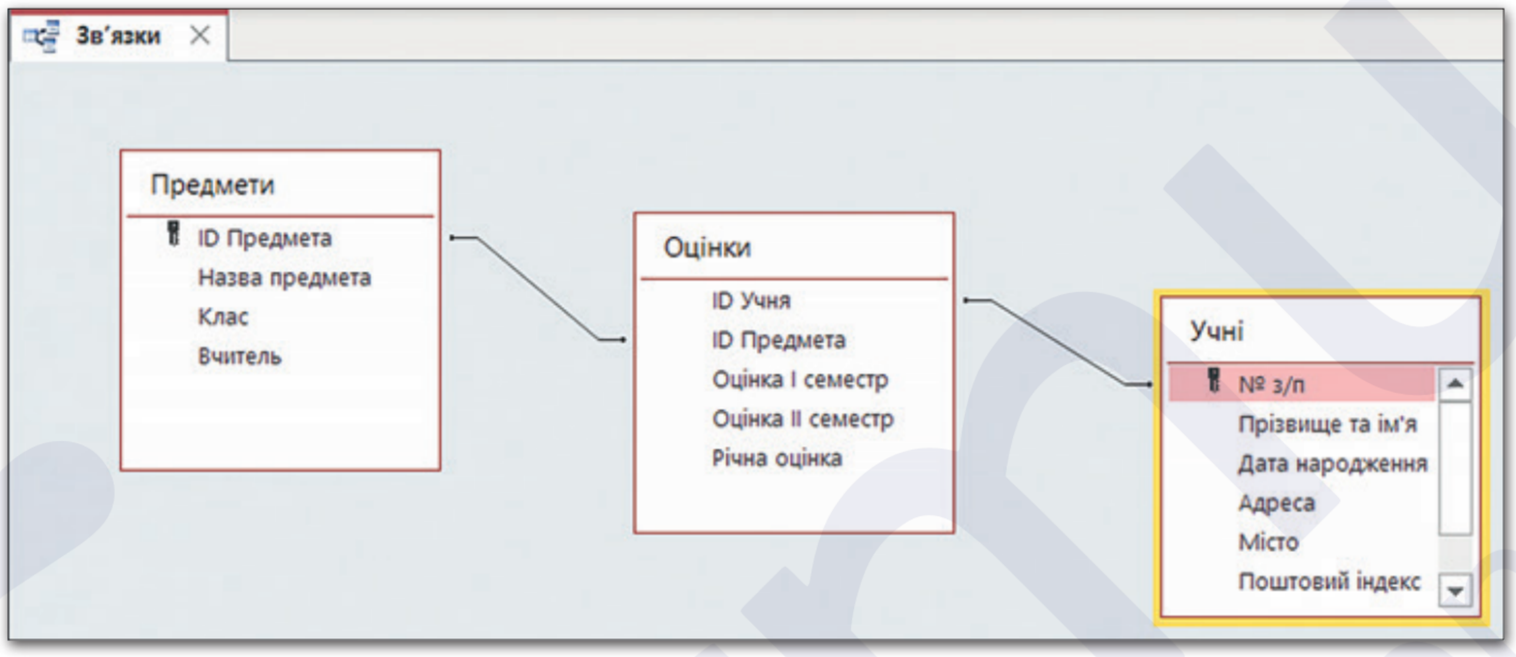
Запити до зв'язаних таблиць

До цього часу ви створювали запити, щоб отримати інформацію з однієї таблиці бази даних. Але в більшості баз даних таблиць багато й вони пов'язані між собою певним чином. Створимо запити для отримання інформації з декількох таблиць одночасно.

Скористайтеся всіма трьома пов’язаними таблицями з наявної бази даних: **Предмети**, **Оцінки** та **Учні**. Заповніть інші таблиці відповідно до поданих нижче даних.

Якщо наявні зв’язки між таблицями не підходять, їх можна видалити та створити нові. Зв’язки між таблицями також можна створювати лише для конкретного запиту.





Учні

№ з/г	Прізвище та ім'я	Дата народ	Адреса	Місто	Поштовий індек	Телефон	Клас
1	Іван Марченко	18.05.2010	вул. Весела, буд. 18, кв. 96	Черкаси	18002	+38 (072) 987-56-11	9-А
2	Маричка Васько	20.12.2009	вул. Маркарова, буд.3	Черкаси	18004	+38 (052) 561-78-99	9-Б
3	Павло Іванченко	01.02.2010	вул. Соборна, буд. 96, кв. 22	Черкаси	18005	+38 (072) 456-98-98	9-А
6	Тетяна Гуйко	25.08.2008	вул. Центральна, буд. 1, кв.46	Київ	00001	+38 (050) 987-78-78	10-Б
7	Дмитро Івченко	18.12.2008	вул. Авіаторна, буд. 347	Львів	79012	+38 (063) 777-25-52	9-Б
8	Світлана Мокряк	03.03.2009	вул. Будівельна, буд. 450	Київ	03005	+38 (097) 123-12-34	9-А
(Новий)							

Реляційна база даних *– це тип бази даних, у якій дані зберігаються у вигляді таблиць. Вони пов’язані між собою за допомогою ключів – первинних та зовнішніх. Це дає змогу ефективно організовувати, зберігати та обробляти великі обсяги взаємопов’язаної інформації.*

Предмети

ID Предмета	Назва предм	Клас	Вчитель
1	Історія	10-А	Сірко
2	Математика	10-А	Степаненко
3	Біологія	10-А	Фліпко
4	Історія	10-Б	Сірко
5	Математика	10-Б	Степаненко
6	Біологія	10-Б	Фліпко
7	Історія	9-А	Сірко
8	Біологія	9-А	Степаненко
9	Математика	9-А	Фліпко
10	Історія	9-Б	Сірко
11	Біологія	9-Б	Степаненко
12	Математика	9-Б	Фліпко
(Новий)			

Оцінки

ID Учня	ID Предмета	Оцінка I сем	Оцінка II сем	Річна оцінка
1	7	9	9	9
1	8	11	12	12
1	9	5	6	6
2	10	7	7	7
2	11	8	7	8
2	12	8	8	8
3	7	8	9	9
3	8	8	10	9
3	9	5	5	5
6	4	10	10	10
6	5	10	11	11
6	6	10	11	11
7	10	4	5	5
7	11	5	6	6
7	12	5	5	5
8	7	10	11	11
8	8	10	11	11
8	9	11	11	11
0	0	0	0	0

ІСТОРІЯ

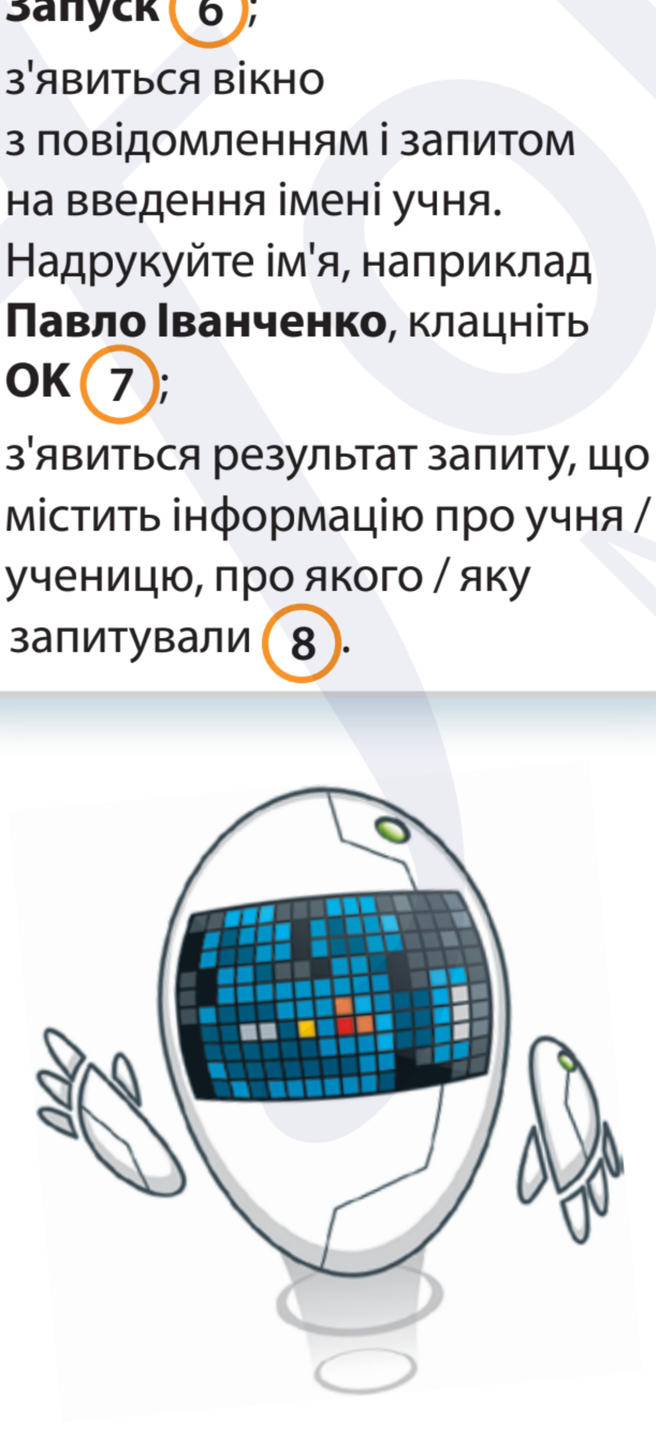
IBM розпочала дослідження реляційних баз даних у 1970 році. В 1974 році вона створила прототип реляційної СУБД, що став основою для SQL, – System R. У 1981 році її першим продуктом став SQL (Structured query language – мова структурованих запитів)/ DS. Однак, компанія Relational Software першою випустила комерційно доступну реляційну систему управління базами даних Oracle у 1979 році.

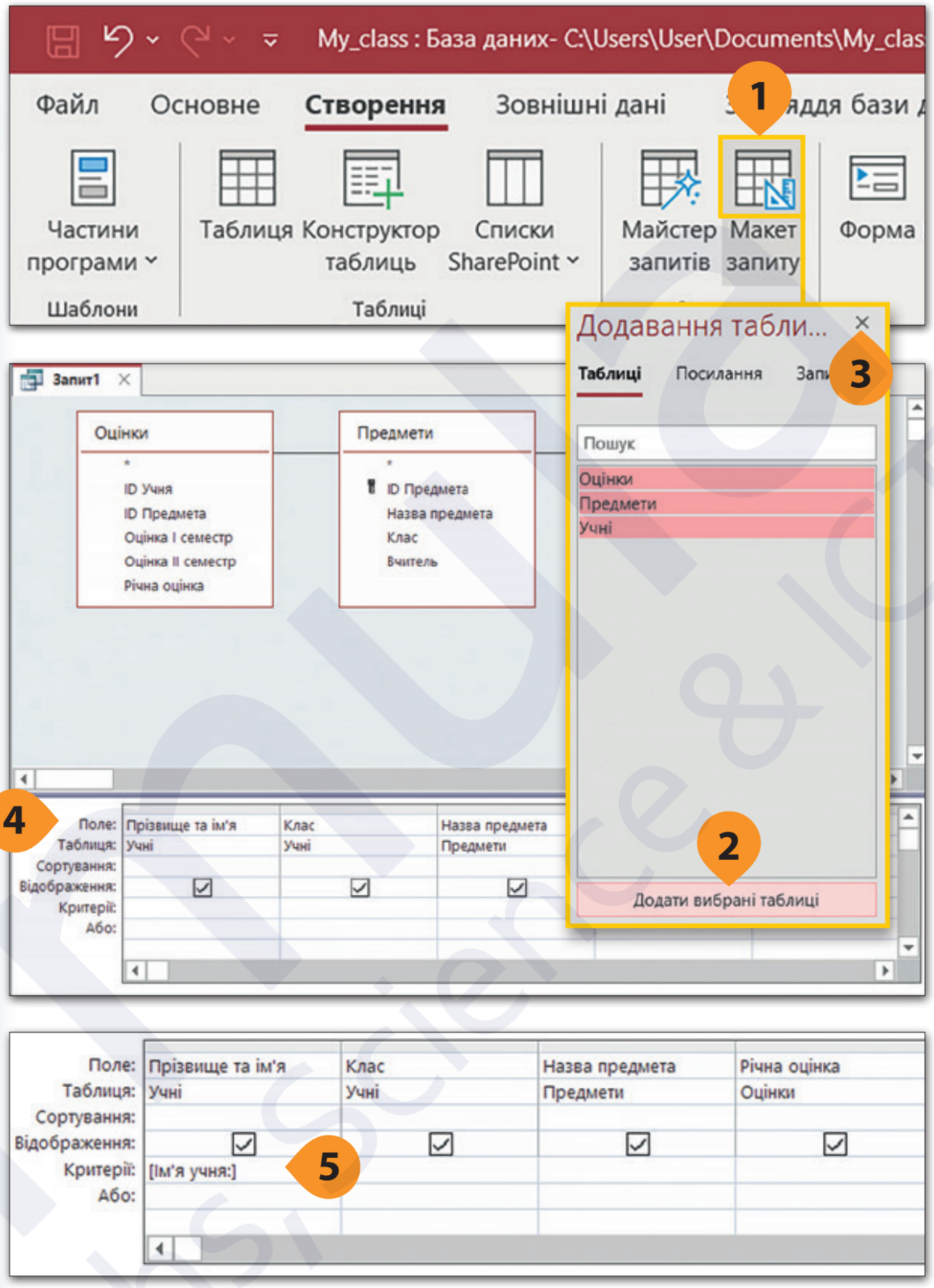
Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

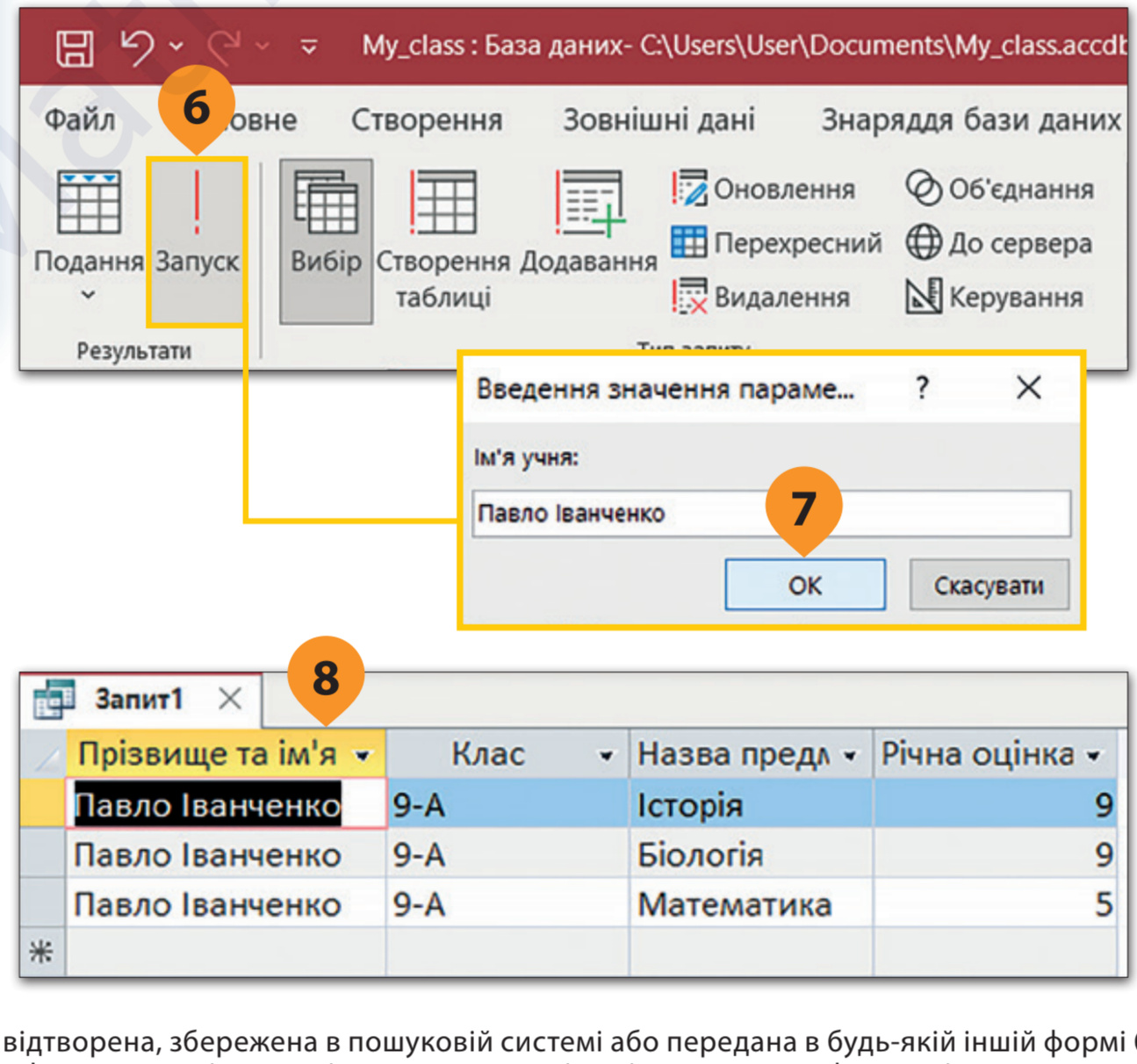
Створіть запит, який визначить усі предмети, пройдені певним учнем чи ученицею, разом із підсумковою оцінкою за кожен із них. У результаті слід вивести ім'я та клас учня / учениці, назву предмета та підсумкову оцінку з цього предмету.

Щоб створити запит до кількох таблиць:

- на вкладці **Створення** в групі **Запити** натисніть **Макет запиту** (1);
- на панелі **Додавання таблиць**, що з'явиться, оберіть усі таблиці та клацніть **Додати вибрані таблиці** (2);
- закрийте панель **Додавання таблиць** (3);
- на вкладці **Інструменти запитів** двічі клацніть по полях з таблиць, які потрібно відобразити (у цьому прикладі **Прізвище, Клас, Предмет та Річна_оцінка**) (4);
- додайте потрібні параметри. У полі **Критерії** введіть **[Ім'я учня:]** (5);
- на вкладці **Конструктор** у групі **Результати** натисніть **Запуск** (6);
- з'явиться вікно з повідомленням і запитом на введення імені учня. Надрукуйте ім'я, наприклад **Павло Іванченко**, клацніть **ОК** (7);
- з'явиться результат запиту, що містить інформацію про учня / ученицю, про якого / яку запитували (8).







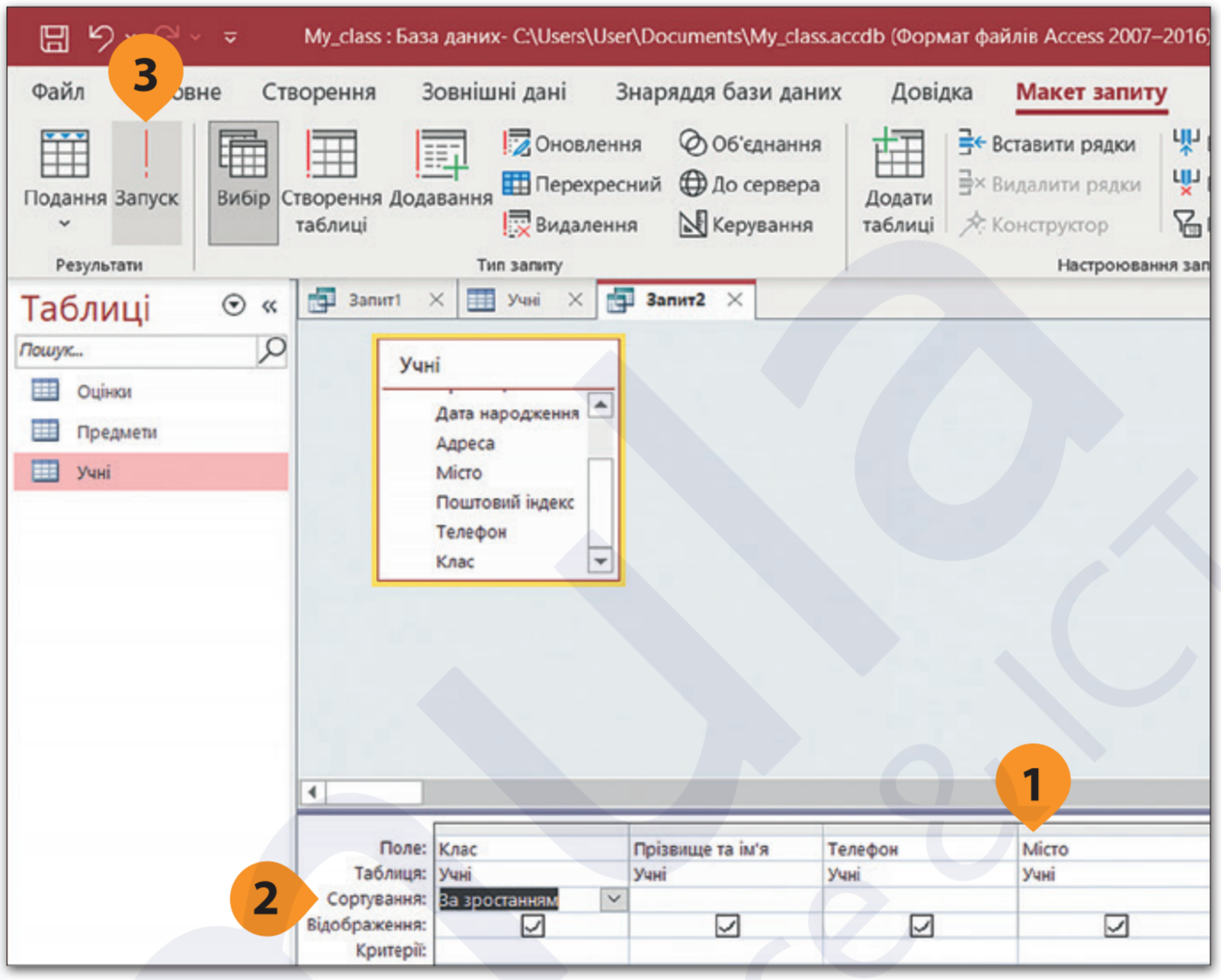
Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

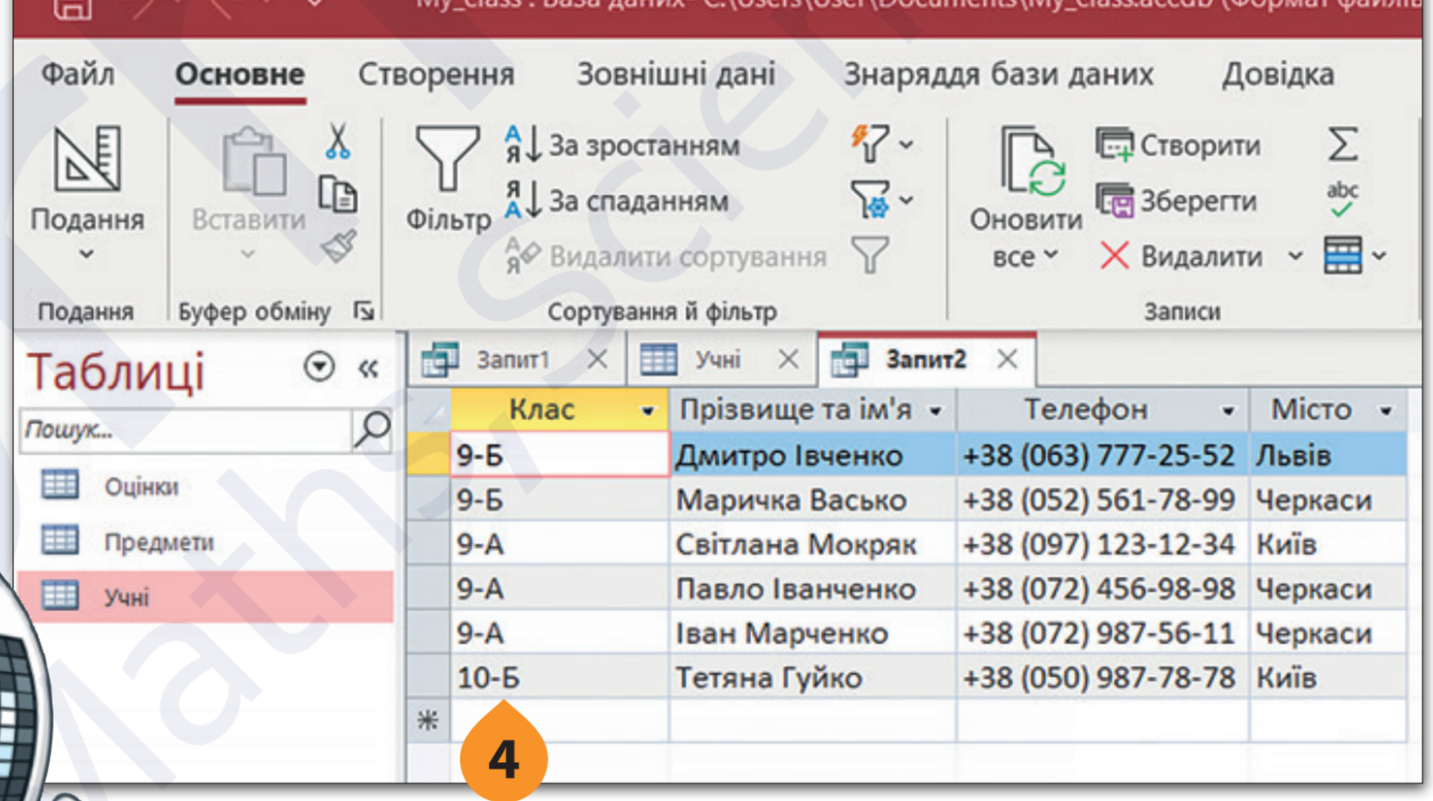
Сортування

Результати запиту можна розташувати в порядку зростання чи спадання.

Щоб відсортувати результати запиту за одним полем:

- створіть новий запит у поданні Конструктор, використовуючи таблицю **Учні** та поля: **Клас, Прізвище та ім'я**, **Телефон** і **Місто** (1);
- у поданні **Структура запиту** знайдіть стовпчик поля **Клас** і в рядку **Сортування** виберіть у розкривному списку значення **За зростанням** (2);
- на вкладці **Макет запиту** у групі **Результати** натисніть **Запуск** (3);
- записи запиту з'являться, починаючи з учнів та учениць **9-А** класу і закінчуючи учнями й ученицями **10-Б** класу (4).







Практичне завдання

Завантажте файл-заготовку зі сторінки інтернет-підтримки. Створіть складний запит. Запишіть відповідні критерії для кожного поля, щоб отримати результати.

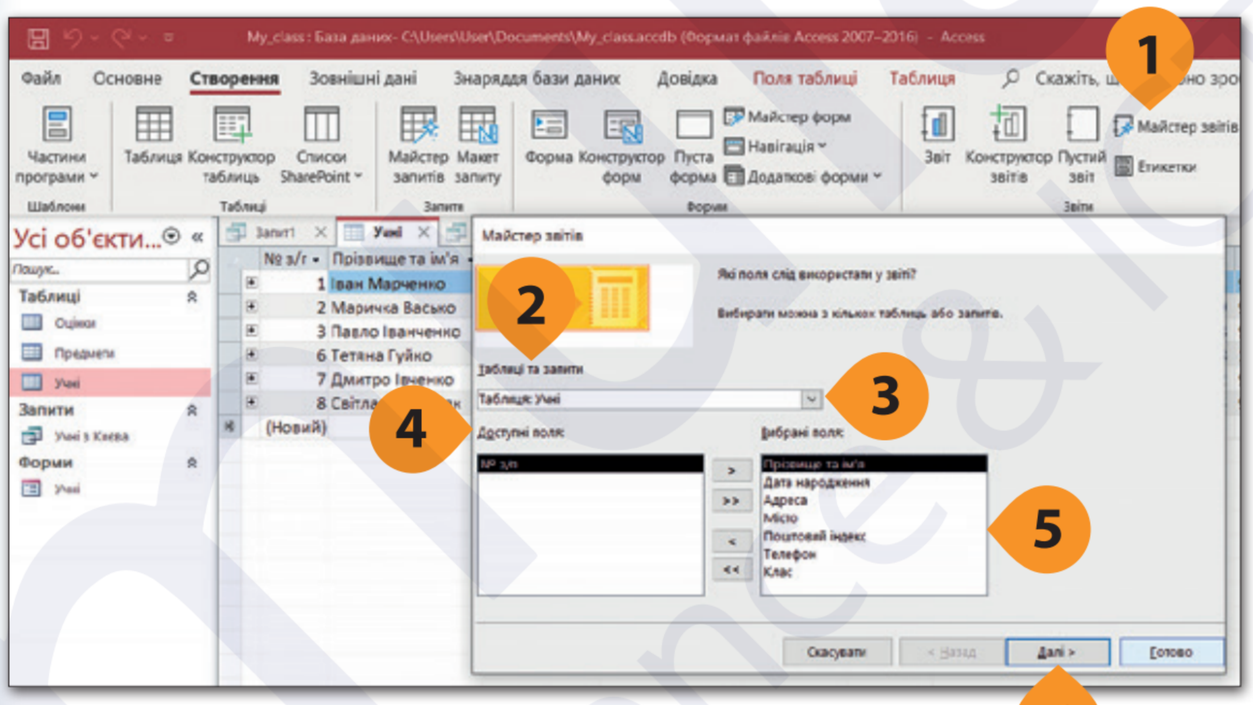
Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Тема 4
Звіти

Ви можете вводити, змінювати та класифікувати дані за допомогою програми, яка працює з базами даних. Якщо треба представити цю інформацію в зручному для читання вигляді для себе чи інших – потрібно створити звіт. Звіт бази даних ілюструє інформацію з таблиці або запиту в заздалегідь відформатованому, привабливому вигляді.

Щоб створити звіт:

- на вкладці **Створення** в групі **Звіти** натисніть **Майстер звітів** (1);
- коли відкриється вікно **Майстра звітів**, у спадному списку **Таблиці та запити** (2) оберіть **Таблиця: Учні** (3);
- перемістіть усі поля, крім **ID_учня**, зі списку **Доступні поля** (4) до списку **Вибрані поля** (5);
- натисніть **Далі** (6).

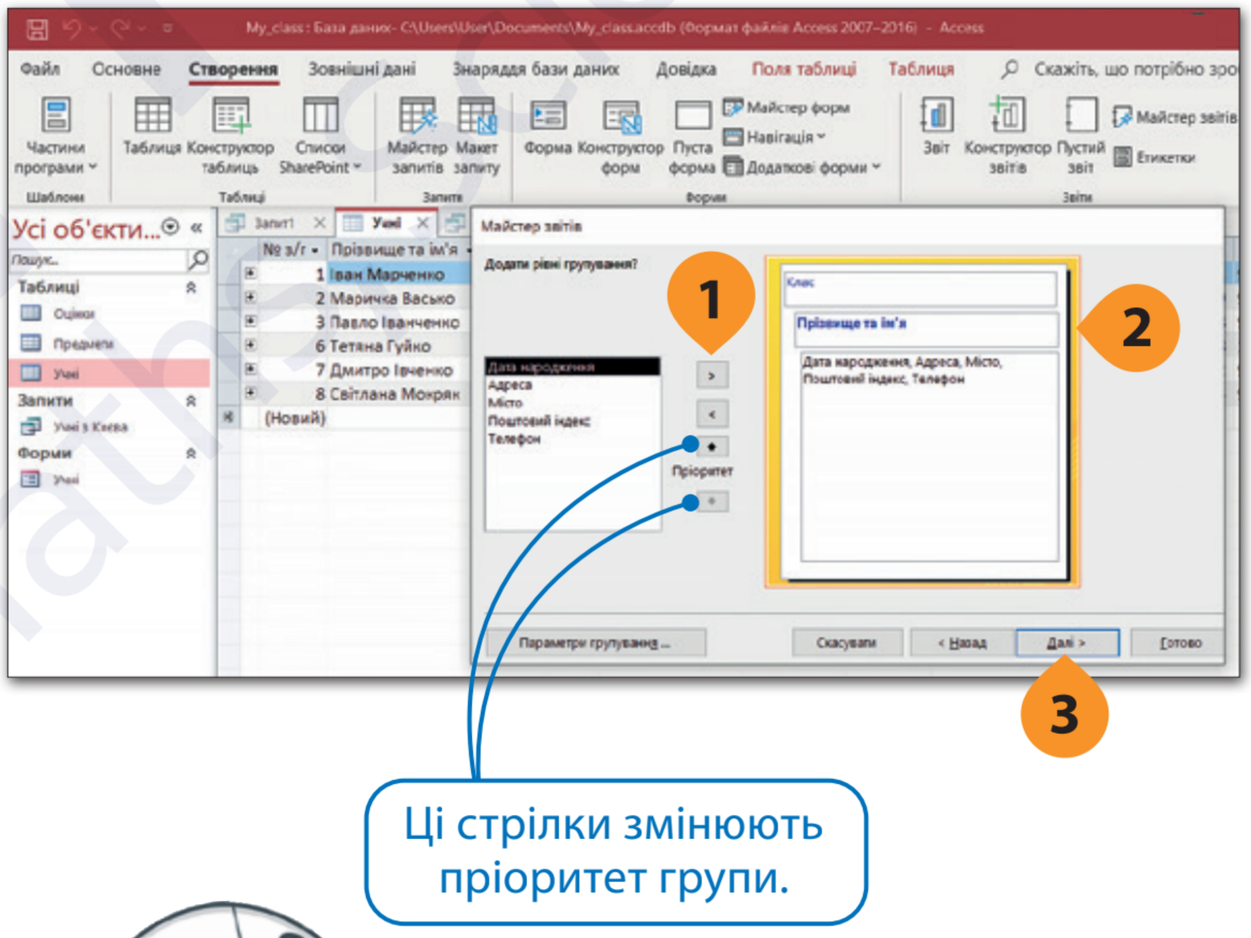


Групування даних

На наступному кроці можна створювати групи та підгрупи ваших даних.

Щоб додати групування:

- перемістіть усі поля, з яких ви хочете створити групи, праворуч (1). Верхнє поле має пріоритет над наступними (2);
- клацніть **Далі** (3).



Ці стрілки змінюють пріоритет групи.

Групування дає змогу переглянути всі пов'язані дані, наприклад, усю інформацію про одну особу. Так на одній сторінці можна побачити всі деталі про конкретну людину.



Зверніть увагу, що порядок, у якому ви переміщуєте поля, має значення.

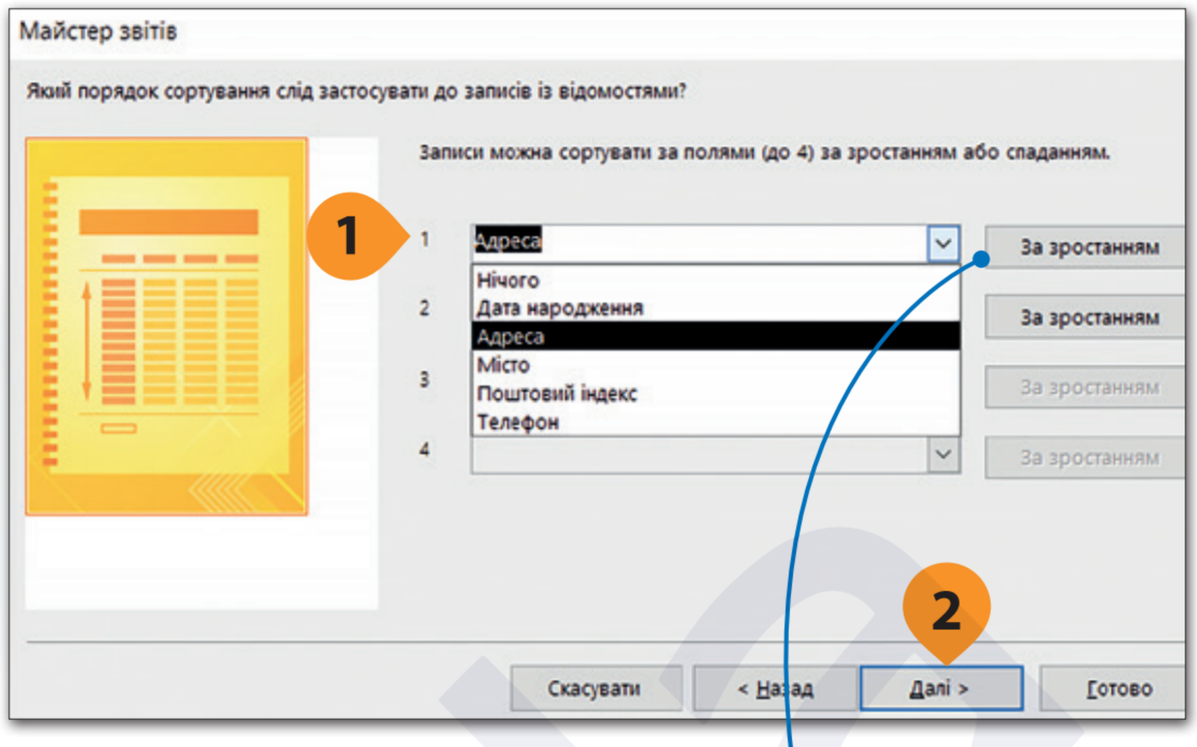
Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Сортування даних

На цьому кроці можна додати більше полів і визначити порядок їх сортування.

Щоб додати сортування:

- оберіть поле з першого списку, який випадає, наприклад **Адреса** (1);
- клацніть **Далі** (2).



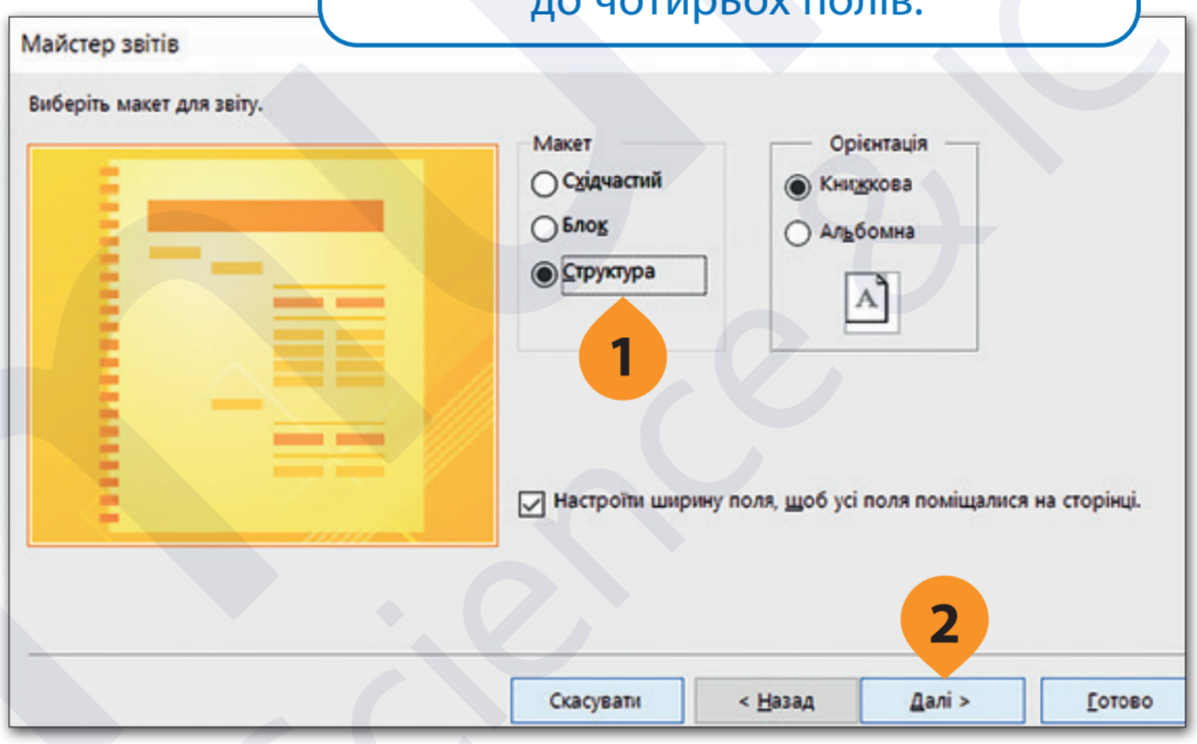
Для зміни порядку сортування просто натисніть кнопку. До списку сортування можна додати до чотирьох полів.

Макет

На наступному кроці можна обрати, який вигляд матиме ваш звіт.

Щоб вибрати вигляд макету:

- у розділі **Макет** натисніть **Структура** (1). Пізніше ви зможете поекспериментувати з іншими макетами, якщо захочете.
- натисніть **Далі** (2).





Поекспериментуйте з іншими макетами.

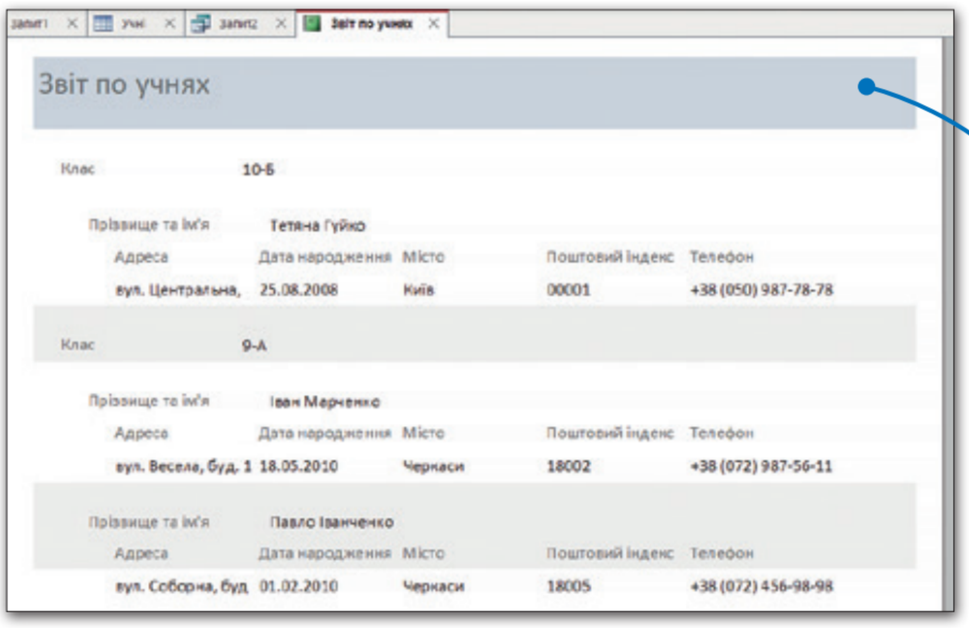
Назва звіту

На останньому кроці дайте назву вашому звіту.

Щоб дати назву вашому звіту:

- уведіть назву звіту в текстовому полі, наприклад **Звіт по учнях** (1);
- натисніть **Готово** (2).



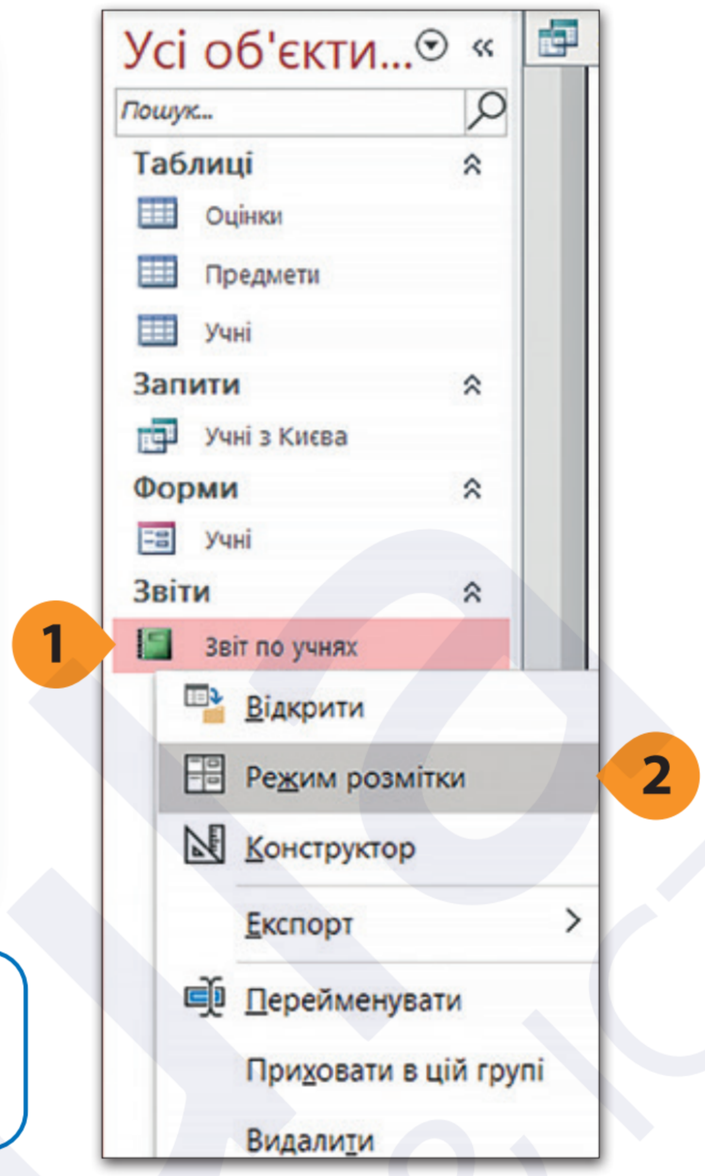


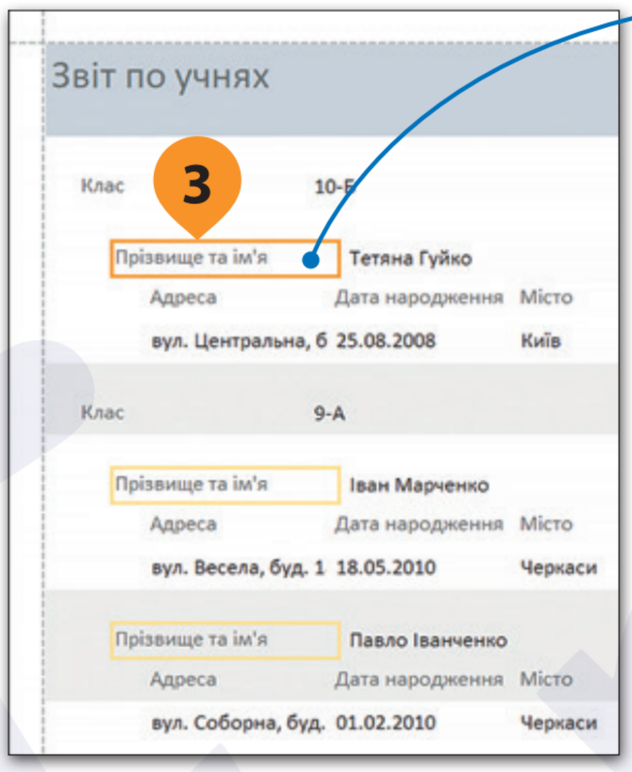
Ваш перший звіт готовий. Дані згруповані за класами, а в середині класу відсортовані за прізвищами.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

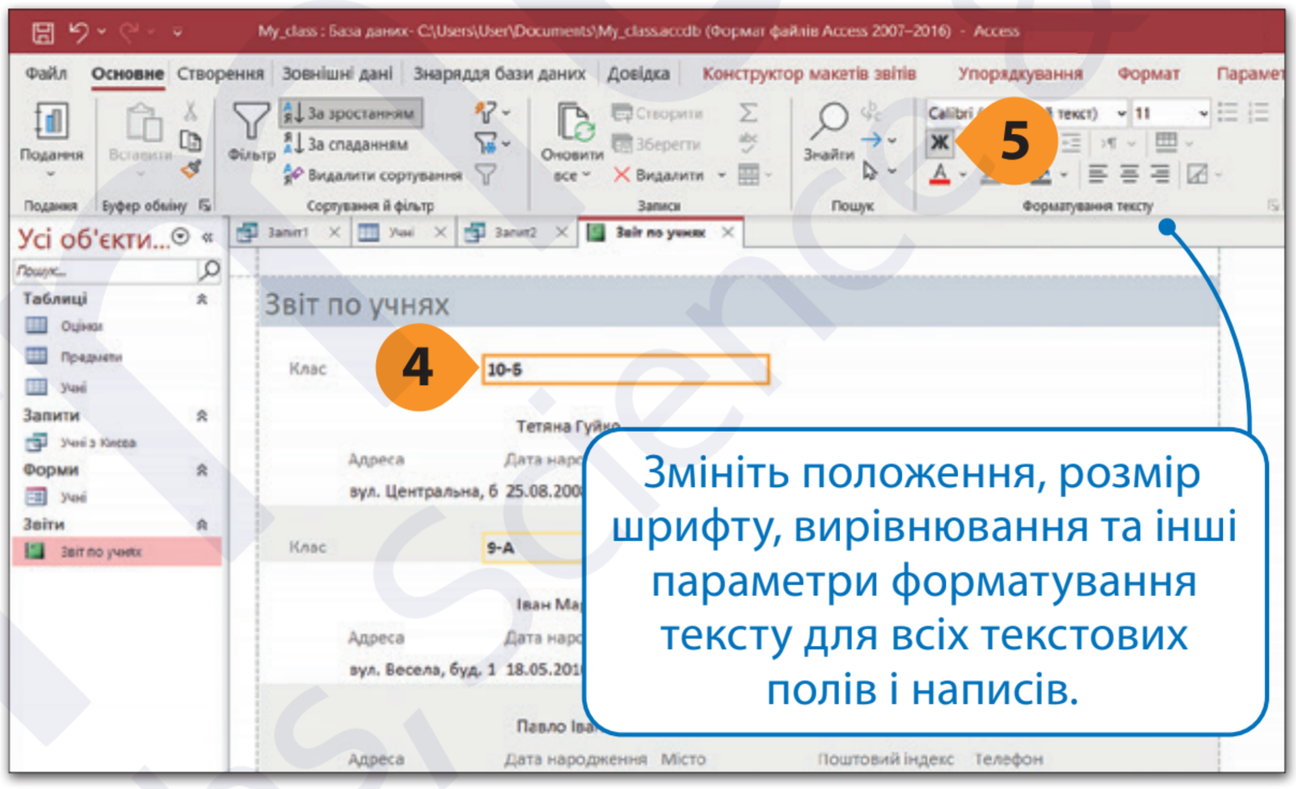
Щоб відредагувати звіт:

- клацніть правою кнопкою миші по створеному вами звіту в **Області переходів** (1);
- оберіть **Режим розмітки** (2);
- у вікні, що відкриється, можна побачити звіт у такому вигляді, який він матиме при відкритті;
- видаліть напис **Прізвище та ім'я** (3) або змініть положення цього текстового поля;
- клацніть по текстовому полю, яке є більш важливим, наприклад поле **Клас** (4), і на вкладці **Основне** в групі **Форматування тексту** зробіть його напівжирним (5);
- ваш звіт відредагований.





У цьому режимі видаліть елементи, які вважаєте непотрібними.



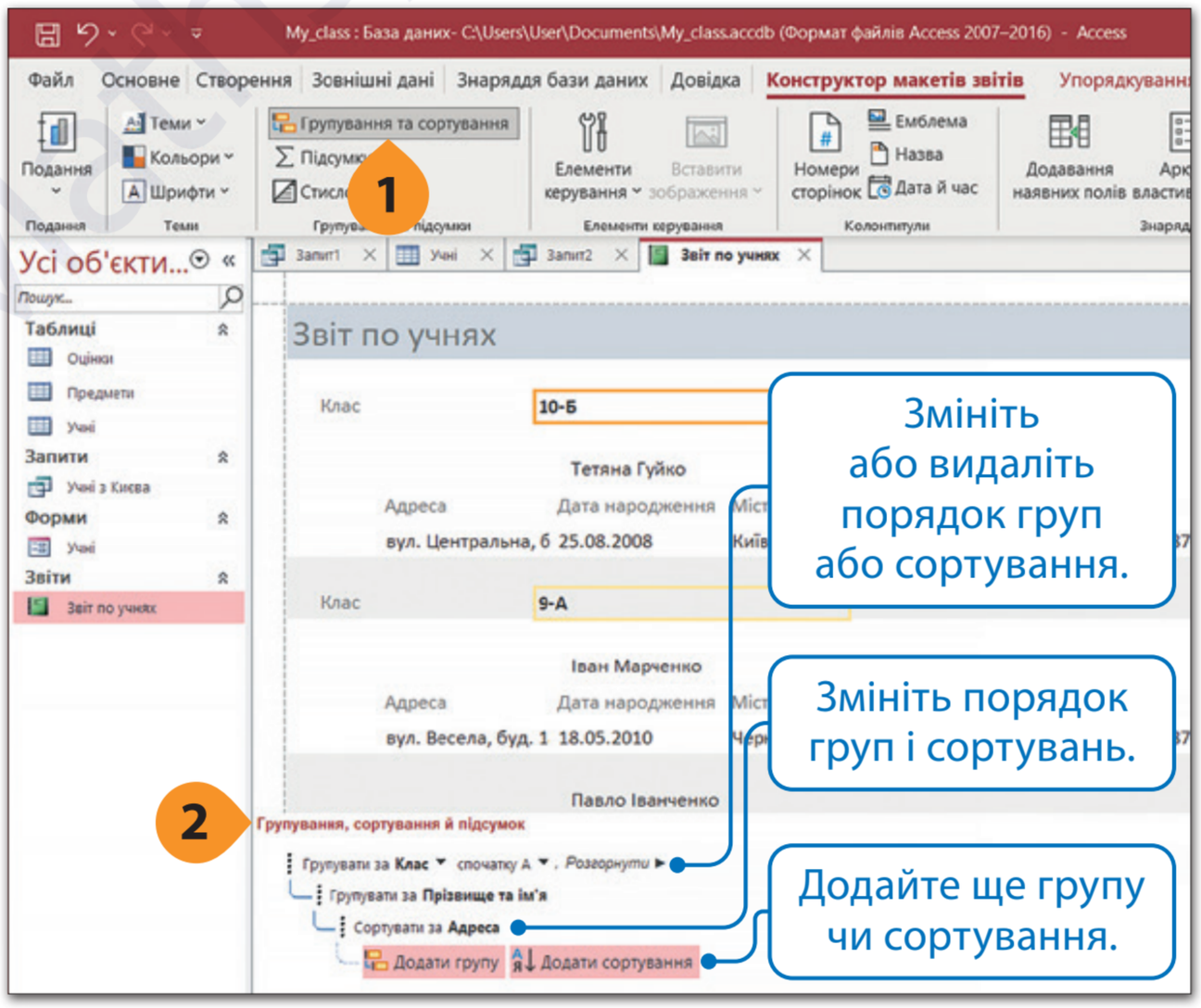
Змініть положення, розмір шрифту, вирівнювання та інші параметри форматування тексту для всіх текстових полів і написів.

Групування та сортування

Access дає змогу вдосконалити звіт, додати підрахунки, додаткове групування чи сортування поза межами, які надає **Майстер звітів.**

Щоб змінити параметри групування та сортування:

- на вкладці **Макетів звітів** у групі **Групування та підсумки** натисніть **Групування та сортування** (1);
- внизу з'явиться панель **Групування, сортування й підсумок** (2).



Змініть або видаліть порядок груп або сортування.

Змініть порядок груп і сортувань.

Додайте ще групу чи сортування.

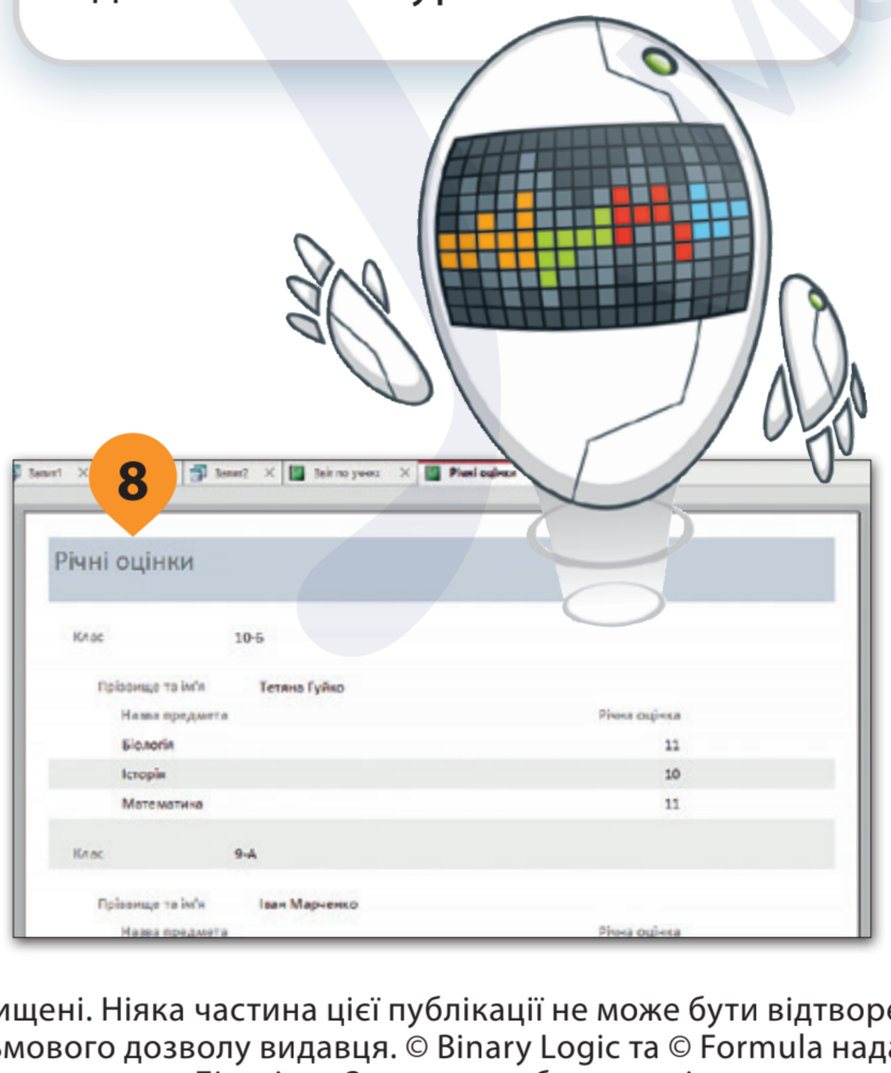
Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

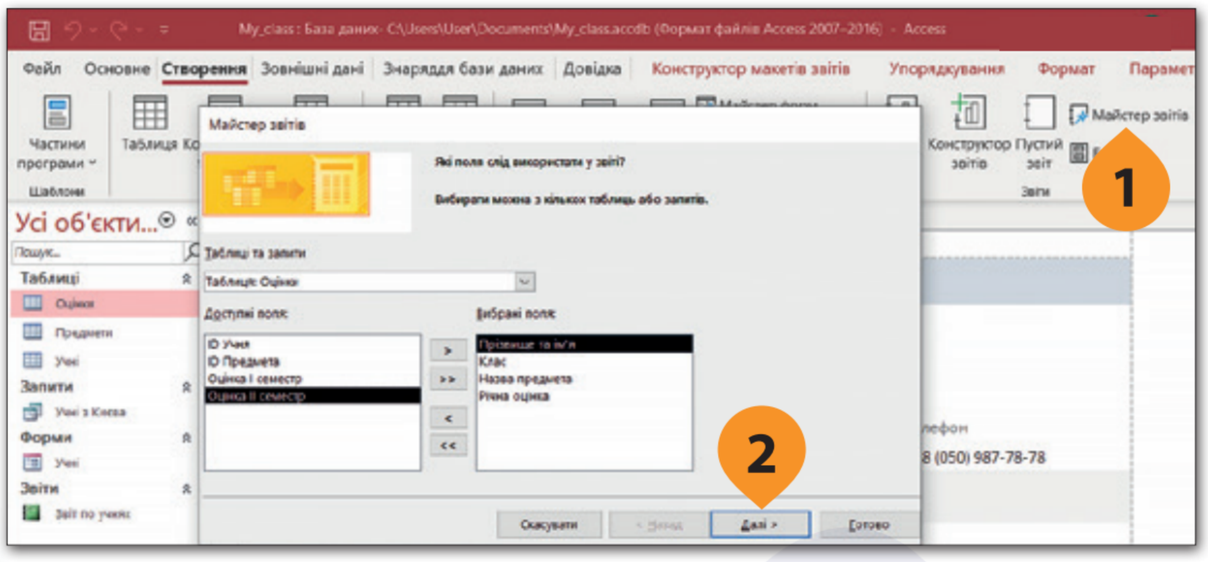
Підсумки

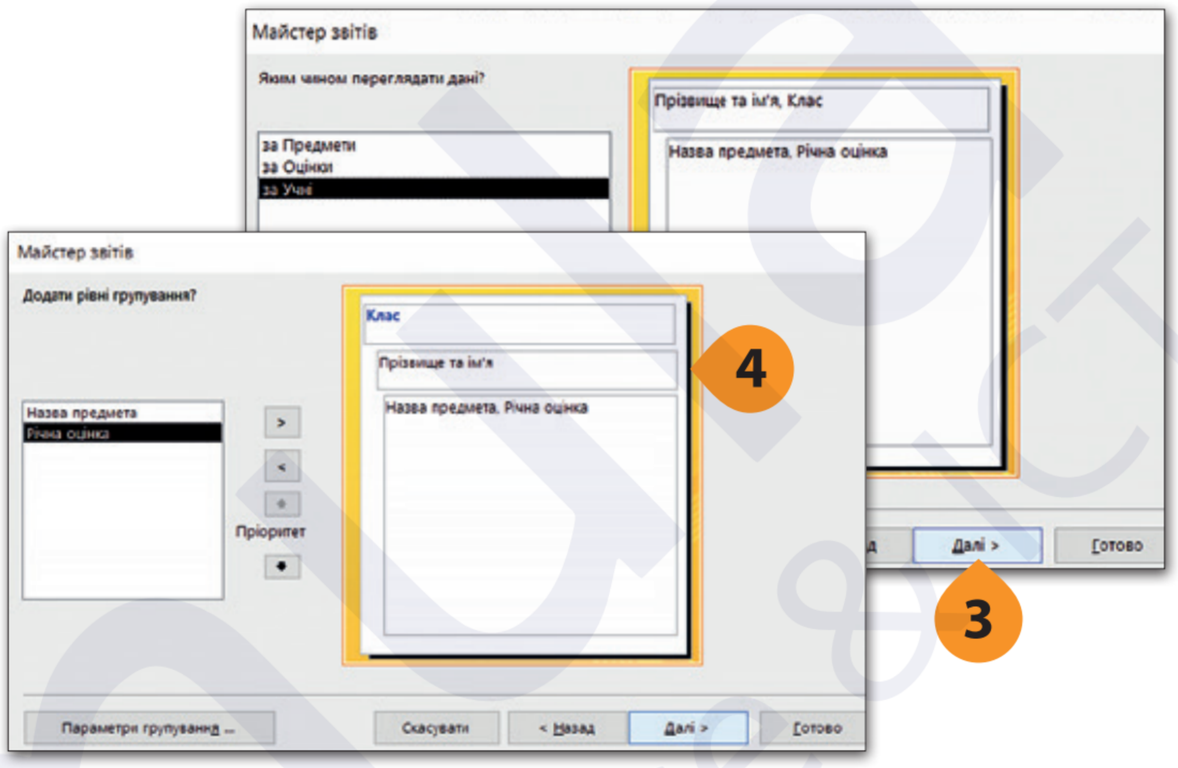
Інколи до звітів доцільно додавати обчислення, які виконуються щоразу під час перегляду звіту. Розгляньмо, як можна розрахувати та відобразити у звіті середню підсумкову оцінку для кожного учня / учениці.

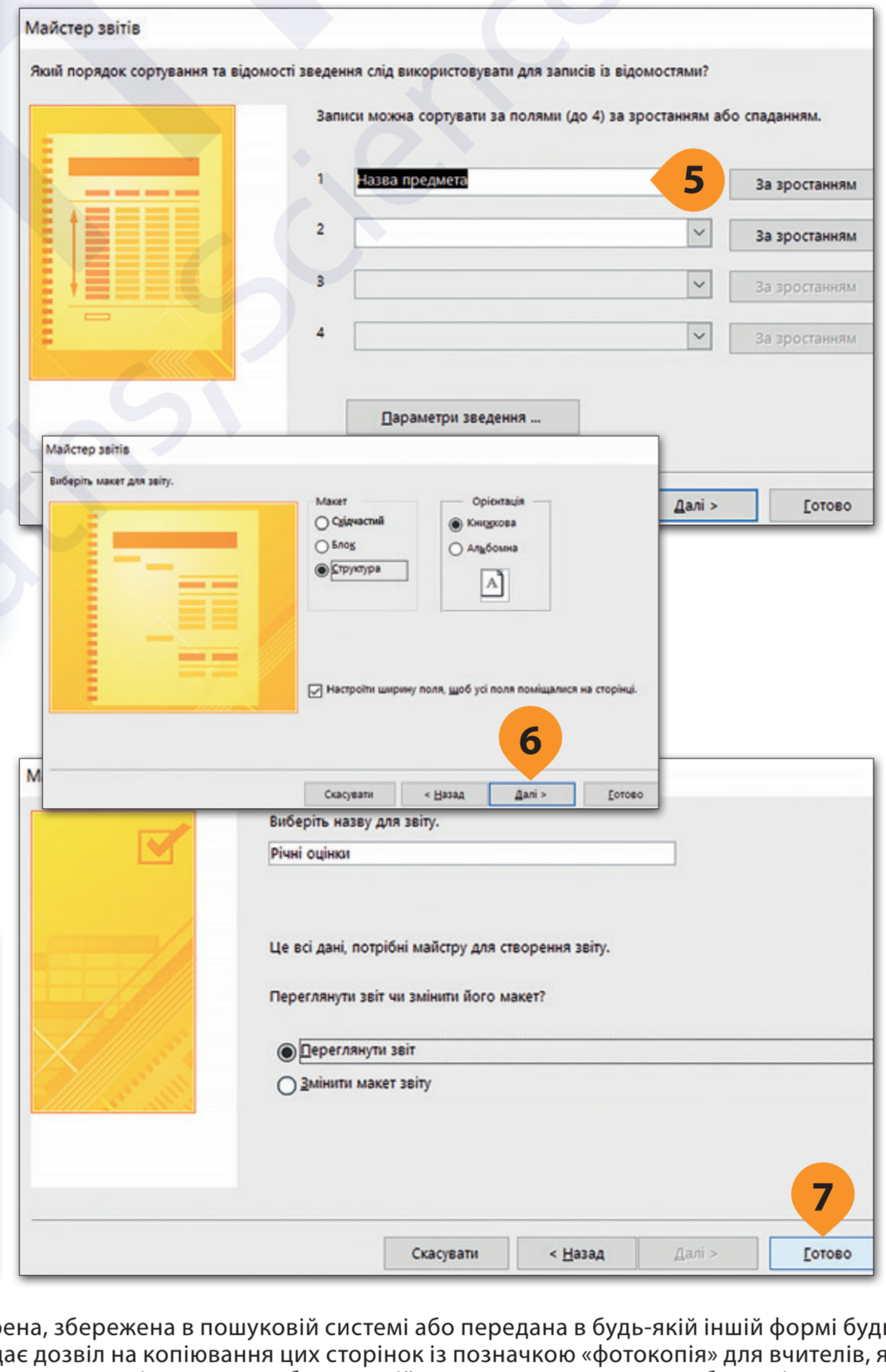
Щоб створити звіт з підсумками:

- на вкладці **Створення** в групі **Звіти** натисніть **Майстер звітів** (1);
- у вікні **Майстра звітів** у списку **Таблиці та запити** виберіть **Прізвище та ім'я, Клас, Назва предмета** та **Річна оцінка** з відповідних таблиць і натисніть **Далі** (2);
- на наступному кроці клацніть **Учні** у списку й натисніть **Далі** (3);
- у наступних кроках згрупуйте за полем **Клас** (4) та відсортуйте за полем **Предмет** (5);
- у розділі **Макет** натисніть **Структура** та натисніть **Далі** (6);
- дайте назву вашому звіту, наприклад **Річні оцінки**, і клацніть **Готово** (7);
- у результаті з'явиться звіт (8) з підсумковими оцінками всіх дітей за всіма курсами.





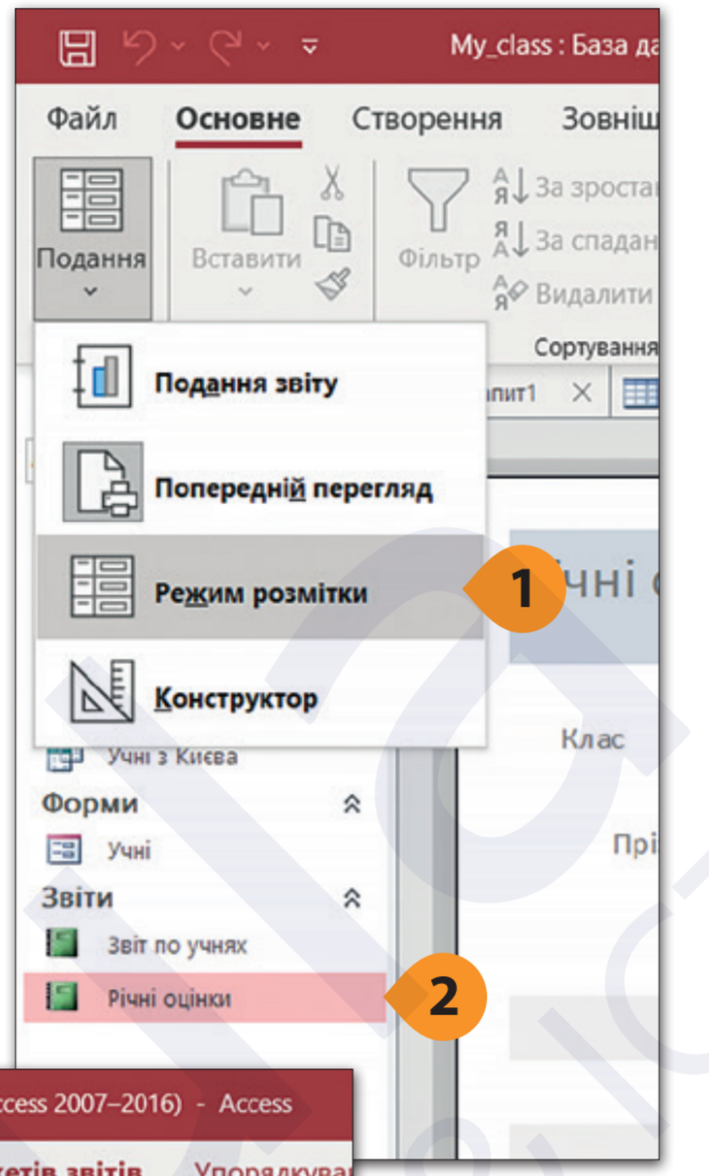


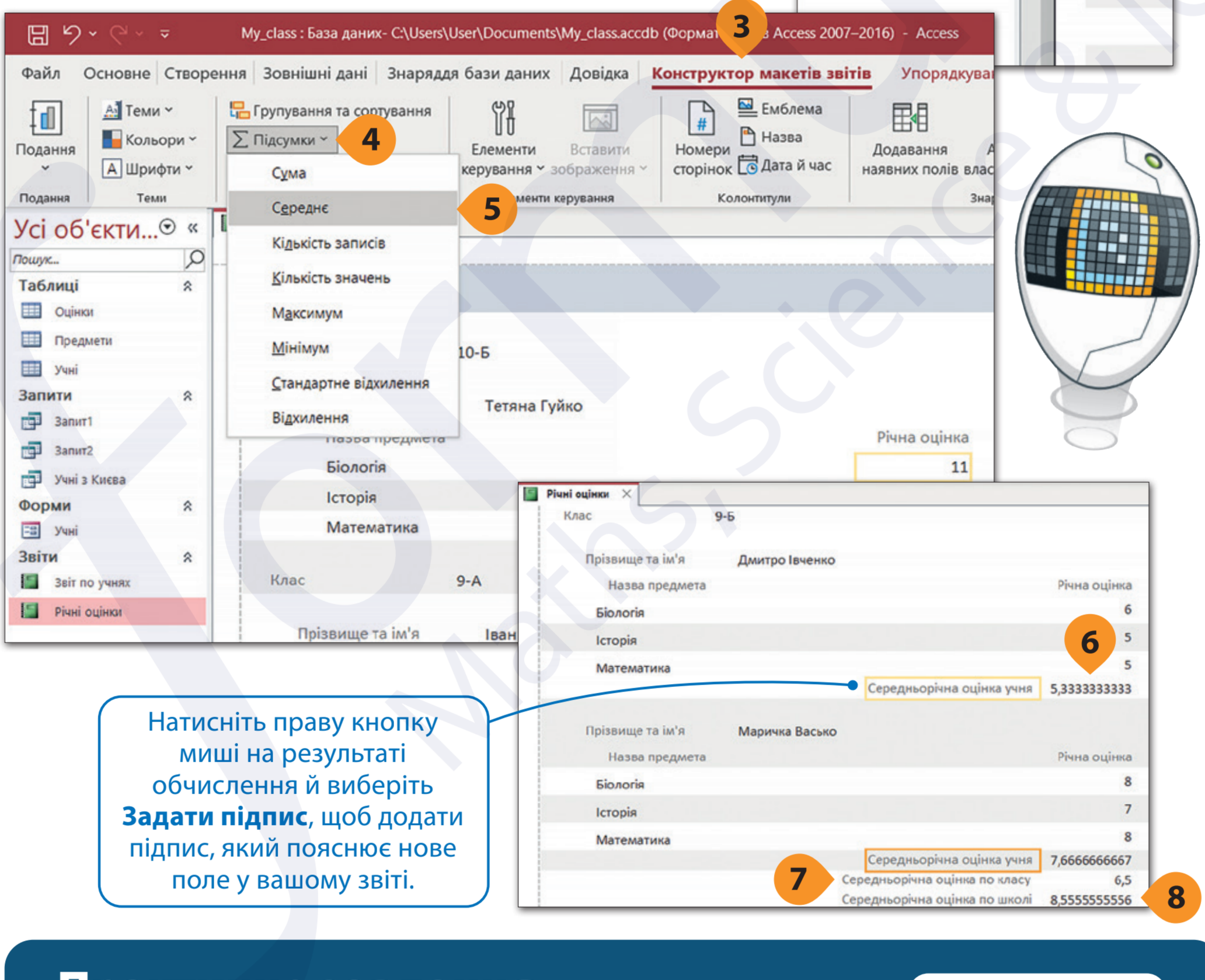


Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Щоб додати Підсумки:

- відкрийте звіт **Річні оцінки** у поданні **Режим розмітки** (1);
- клацніть по полю **Річні оцінки** (2);
- на вкладці **Конструктор макетів звітів** (3) у групі **Групування та підсумки** натисніть **Підсумки** (4), а потім оберіть **Середнє** (5);
- для кожного учня й учениці з'явиться нове поле (6), що містить середню оцінку з усіх предметів;
- до нового поля для кожного учня й учениці в кінці класу з'явиться ще одне поле, що містить середній бал класу (7). У кінці звіту з'являється загальний середній бал для всіх учнів та учениць усіх класів (8).





Натисніть праву кнопку миші на результаті обчислення й виберіть **Задати підпис**, щоб додати підпис, який пояснює нове поле у вашому звіті.

Практичне завдання

Завантажте файл-заготовку зі сторінки інтернет-підтримки. Створіть звіт та внесіть зміни у його зовнішній вигляд.

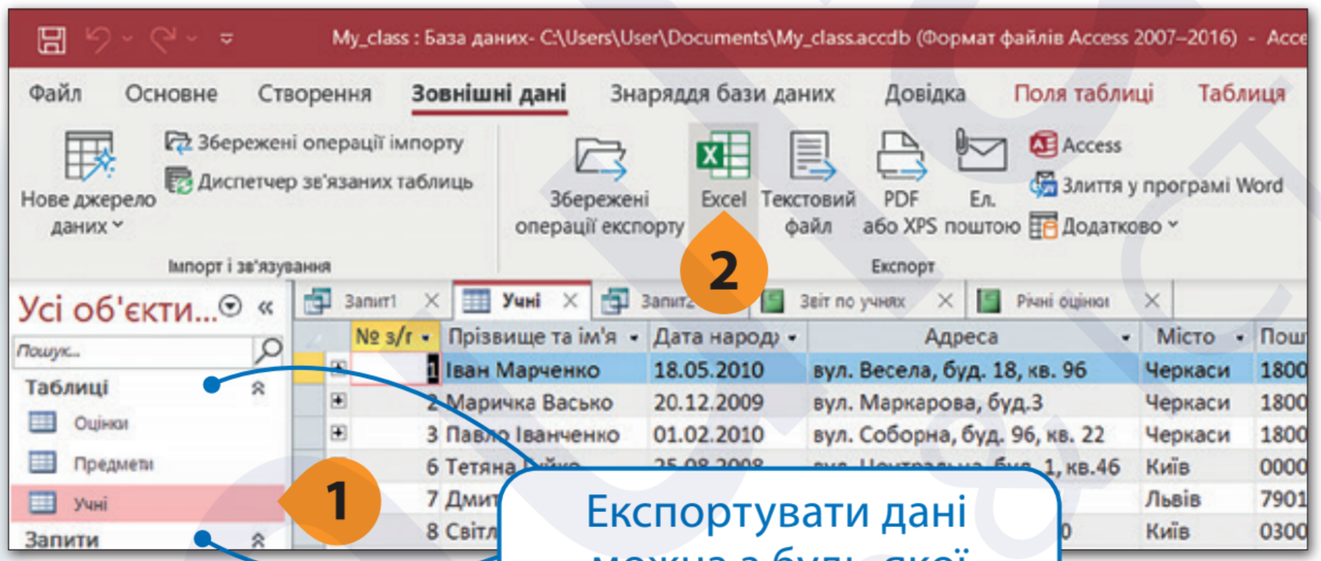
Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

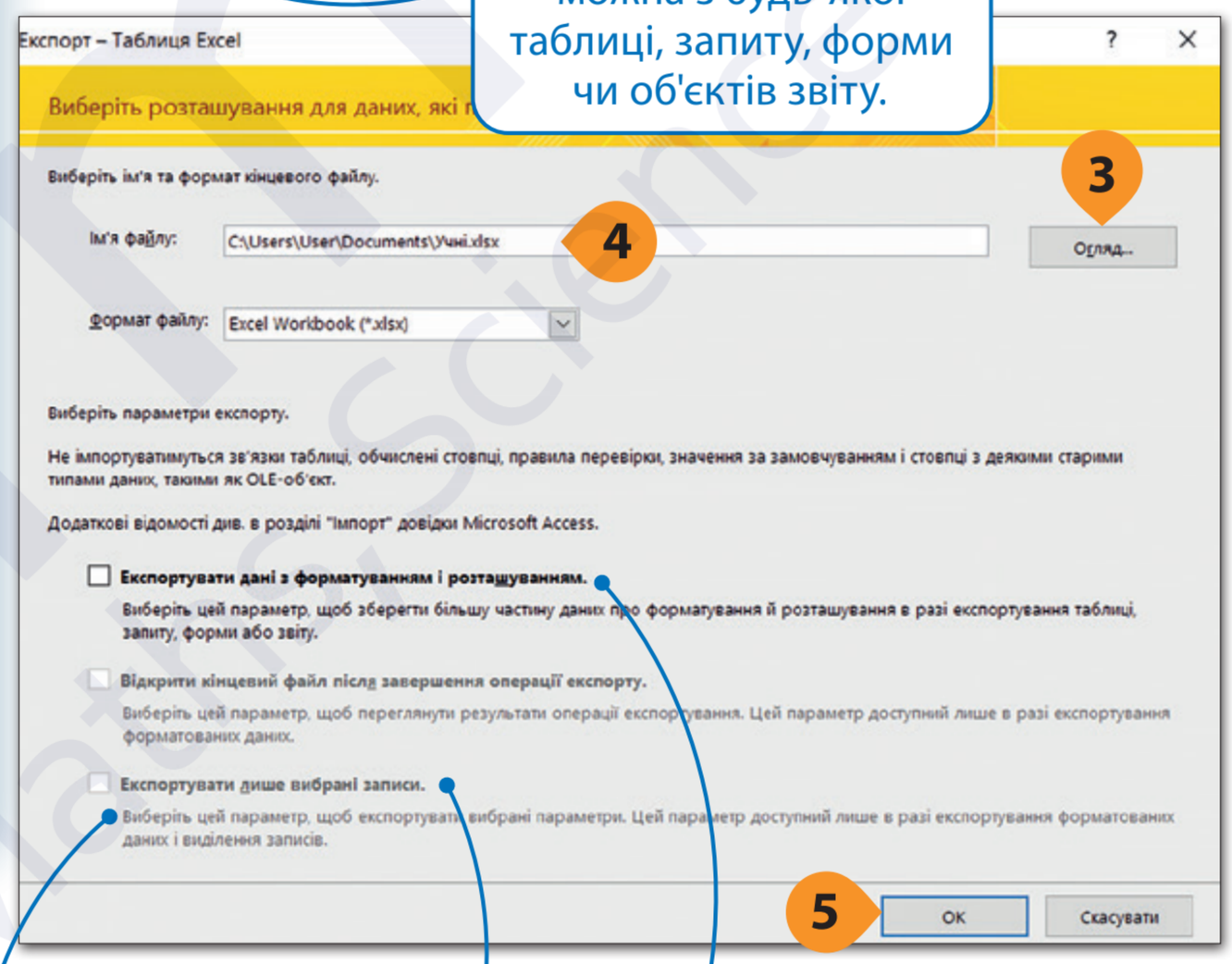
Тема 5
Імпорт та експорт даних

Імпорт і **експорт** даних – ключові операції для будь-якого програмного забезпечення, особливо для баз даних. Ці функції дають можливість переносити дані з однієї програми в іншу для подальшого використання. Наприклад, можна експортувати дані з бази даних в електронну таблицю для аналізу або, навпаки, імпортувати наявну таблицю з даними в базу даних, щоб спростити її оновлення та підтримку.

Щоб експортувати дані до Microsoft Excel:

- в **Області переходів** оберіть об'єкт, з якого треба експортувати дані, наприклад таблицю **Учні** (1);
- на вкладці **Зовнішні дані** в групі **Експорт** оберіть **Excel** (2);
- у вікні **Експорт – Таблиця Excel** натисніть **Огляд** (3), якщо треба обрати інше місце розташування файлу, введіть ім'я файлу Microsoft Excel, наприклад **Учні.xlsx** (4);
- натисніть **ОК** (5);
- у наступному вікні клацніть **Закрити** (6);
- ваші дані експортовано до **Microsoft Excel** (7).

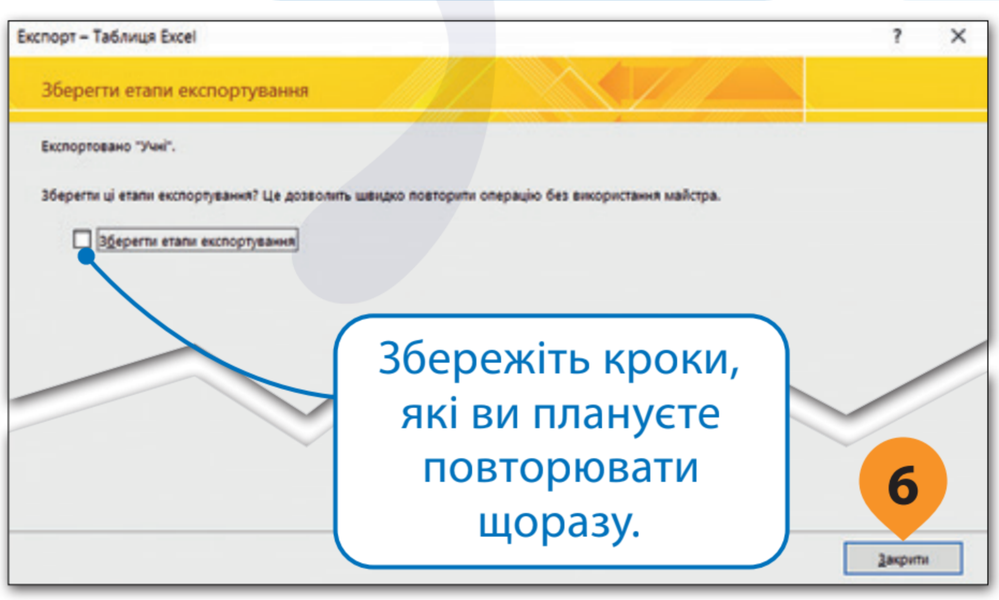


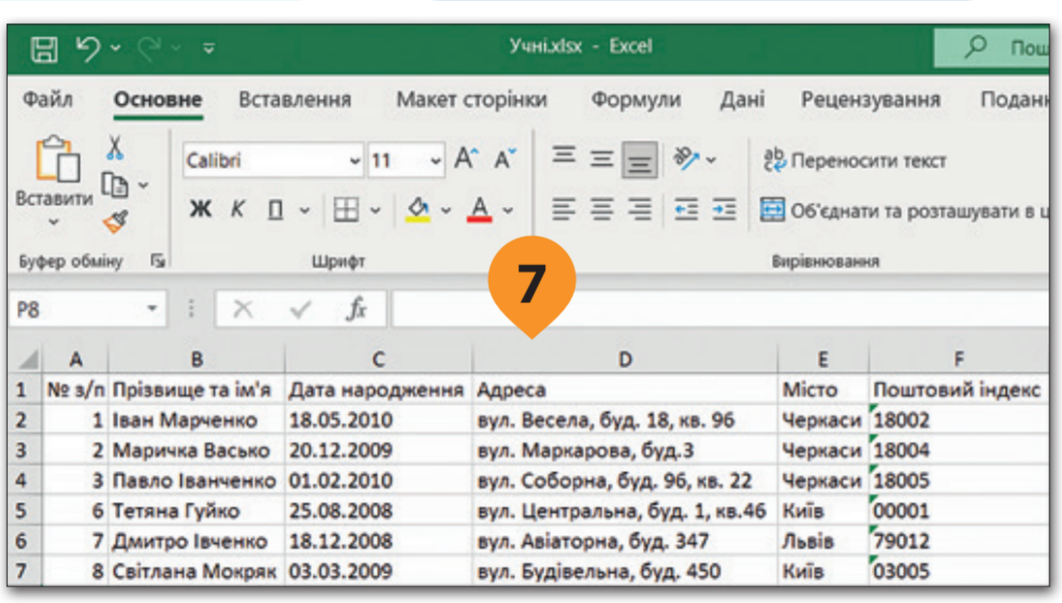


Відкрийте фінальний файл, щоб побачити результати.

Ця опції зберігає більшу частину форматування.

Виберіть і експортуйте певні записи до вашої таблиці.



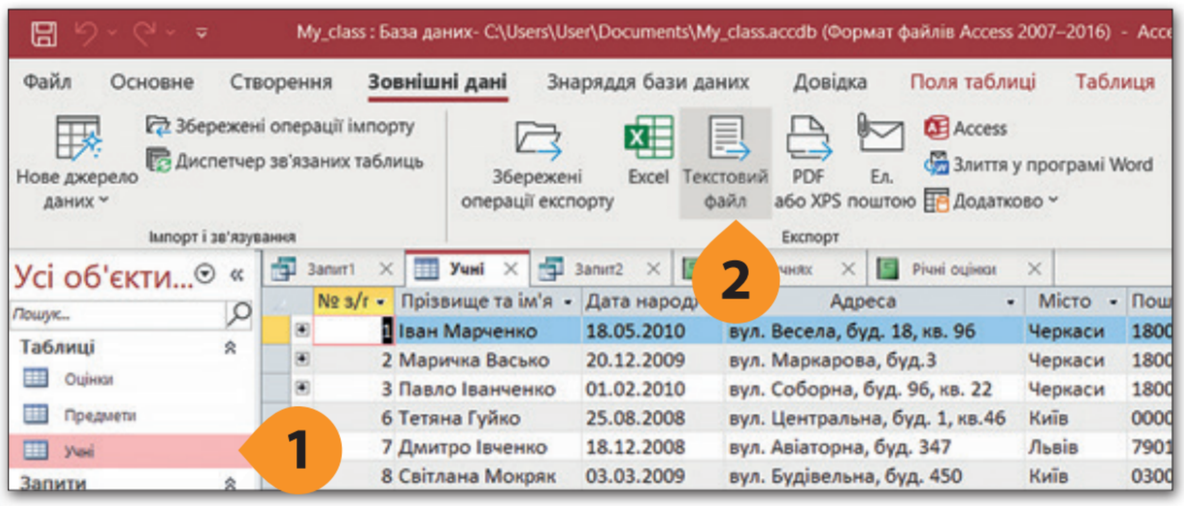


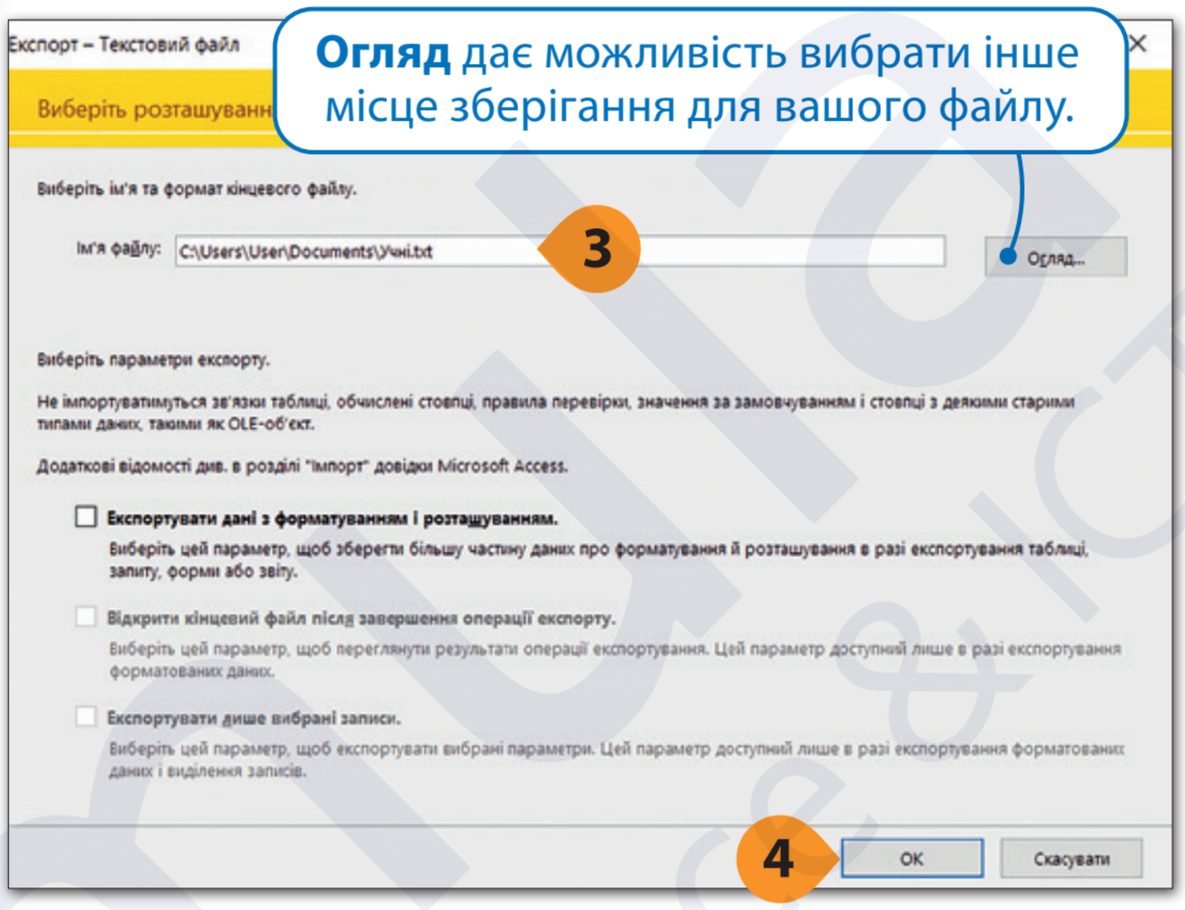
	A	B	C	D	E	F
1	№ з/п	Прізвище та ім'я	Дата народження	Адреса	Місто	Поштовий індекс
2	1	Іван Марченко	18.05.2010	вул. Весела, буд. 18, кв. 96	Черкаси	18002
3	2	Маричка Васько	20.12.2009	вул. Маркарова, буд.3	Черкаси	18004
4	3	Павло Іванченко	01.02.2010	вул. Соборна, буд. 96, кв. 22	Черкаси	18005
5	6	Тетяна Гуйко	25.08.2008	вул. Центральна, буд. 1, кв.46	Київ	00001
6	7	Дмитро Івченко	18.12.2008	вул. Авіаторна, буд. 347	Львів	79012
7	8	Світлана Мокряк	03.03.2009	вул. Будівельна, буд. 450	Київ	03005

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

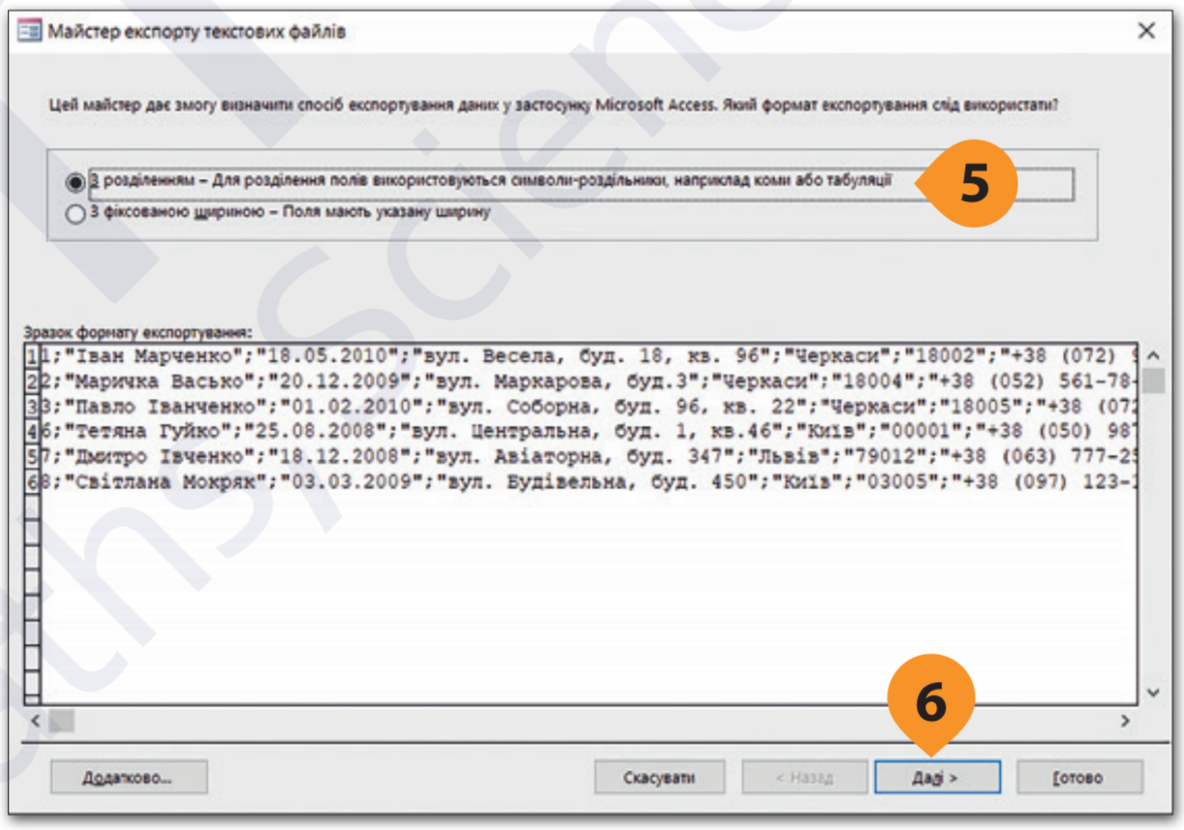
Експорт даних до текстового файлу за допомогою Comma Separated Values (CSV):

- в **Області переходів** оберіть об'єкт, з якого треба експортувати дані, наприклад таблицю **Учні** (1);
- на вкладці **Зовнішні дані** в групі **Експорт** натисніть **Текстовий файл** (2);
- у вікні **Експорт – Текстовий файл** уведіть ім'я текстового файлу, наприклад **Учні.txt** (3), клацніть **ОК** (4);
- у наступному вікні залиште відмітку **З розділенням** (5) і натисніть **Далі** (6);
- у наступному вікні відмітьте **Включити імена полів до першого рядка** (7), клацніть **Далі** (8);
- у наступному вікні клацніть **Готово** (9);
- у наступному вікні натисніть **Закрити** (10).

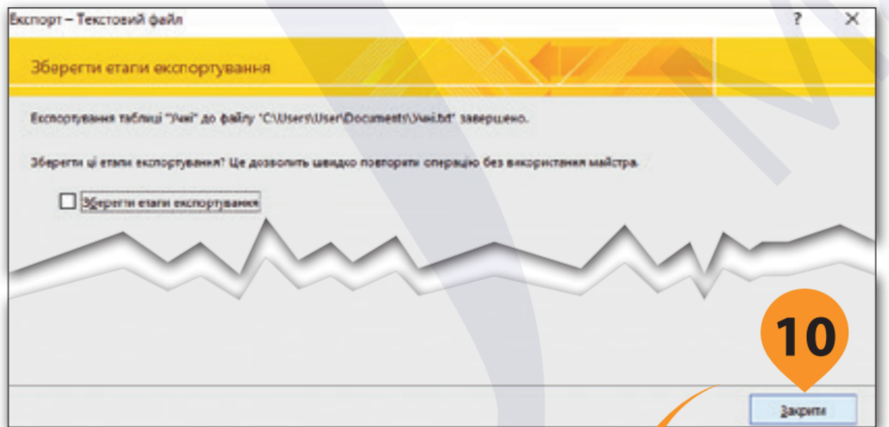


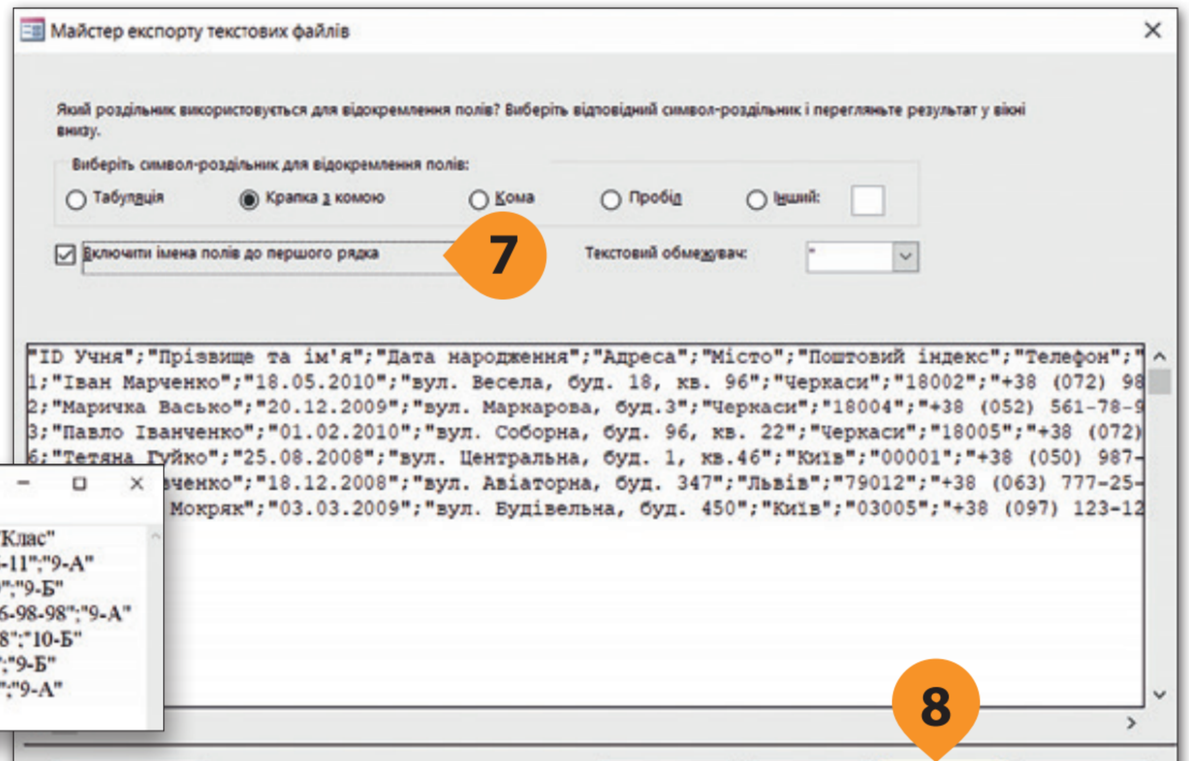


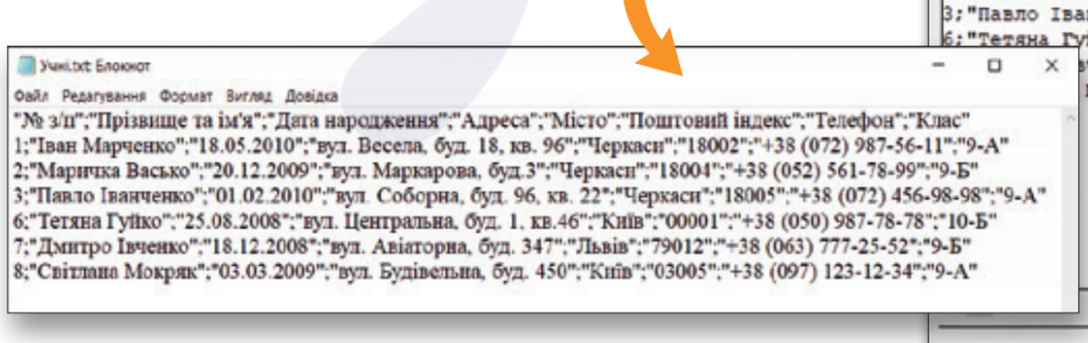
Огляд дає можливість вибрати інше місце зберігання для вашого файлу.







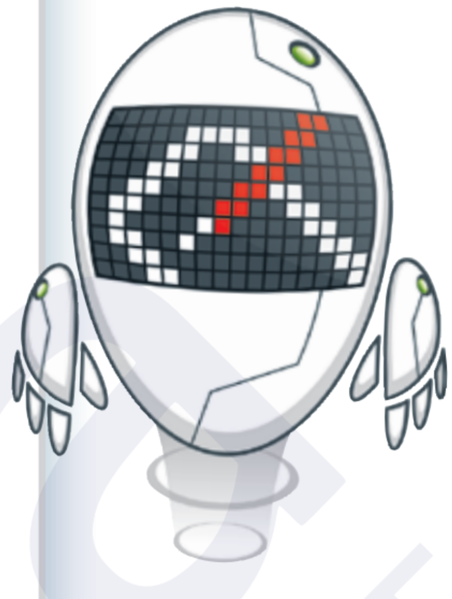


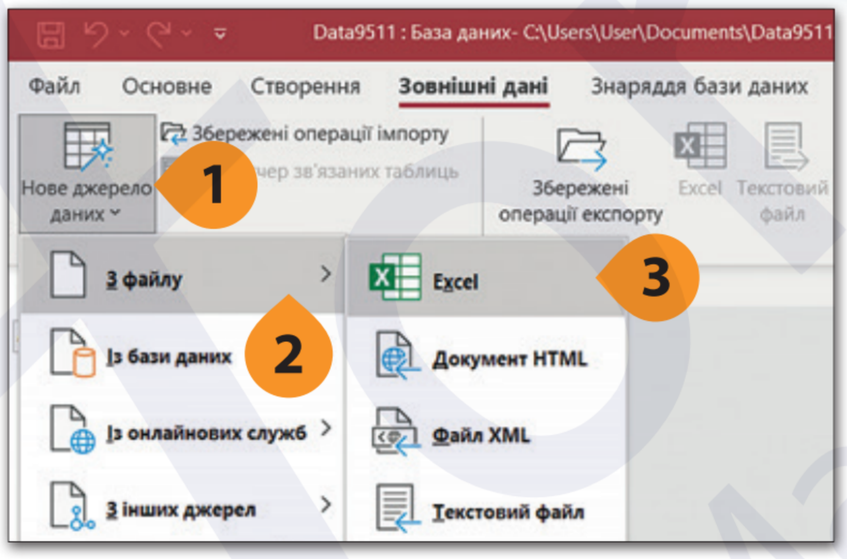


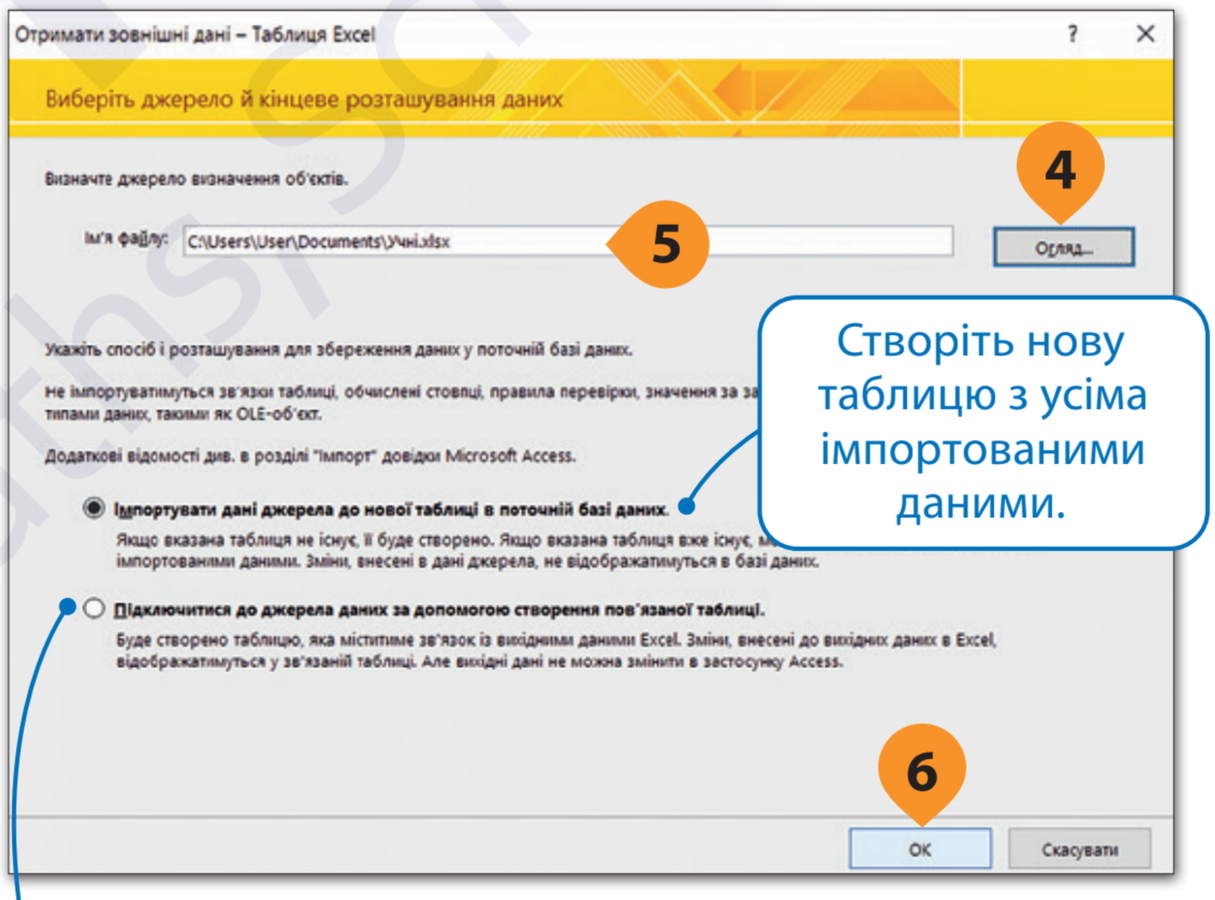
Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

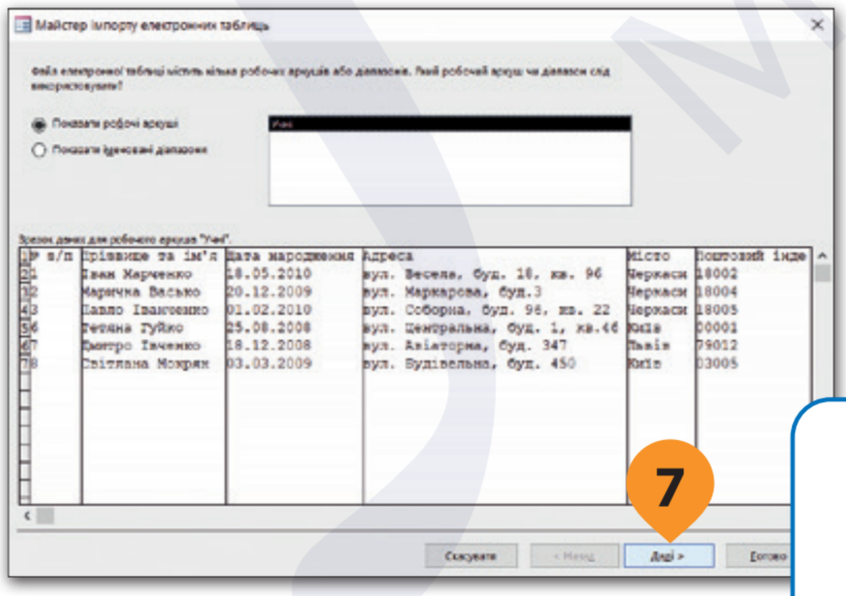
Щоб імпортувати дані з файлу Microsoft Excel:

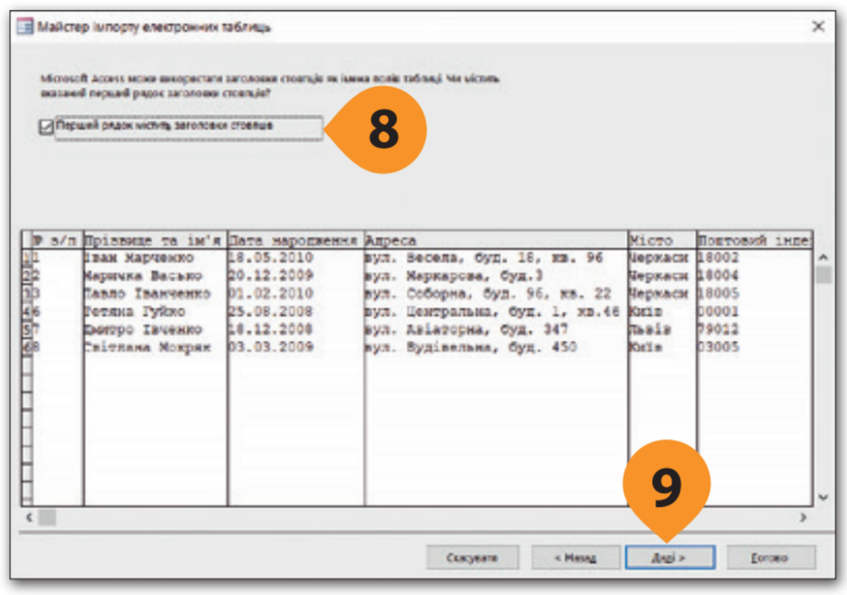
- відкрийте **Microsoft Access** і створіть нову порожню базу даних;
- на вкладці **Зовнішні дані** в групі **Імпорт і зв'язування** натисніть **Нове джерело даних** (1);
- у меню, що з'явиться, виберіть опцію **З файлу** (2) та клацніть **Excel** (3);
- у вікні **Отримати зовнішні дані – таблиця Excel** натисніть **Огляд** (4), щоб вибрати файл **Microsoft Excel,** який потрібно імпортувати, наприклад **Учні.xlsx** (5). Натисніть **ОК** (6);
- у наступному вікні клацніть **Далі** (7);
- у наступному вікні переконайтеся, що опція **Перший рядок містить заголовки стовпців** (8) відмічена, якщо у файлі **Microsoft Excel**, який імпортується, є рядок із заголовками, а потім натисніть **Далі** (9);
- у наступному вікні клацніть кожне з відображених полів і переконайтеся, що тип даних вибрано правильно (10). Якщо це не так, змініть його на коректне значення й натисніть **Далі** (11);
- у наступному вікні, якщо ваші дані з **Microsoft Excel** містять поле, яке є або може бути використане як ключове поле у вашій базі даних, натисніть **Вибрати власний первинний ключ** (12) й оберіть відповідне поле у спадному списку, наприклад **ID_учня / учениці** (13), й клацніть **Далі** (14);
- у наступному вікні, у полі **Імпортувати до таблиці**, введіть назву таблиці, яка буде заповнена вашими даними, наприклад **Учні** (15), натисніть **Готово** (16);
- у наступному вікні натисніть **Закрити** (17).











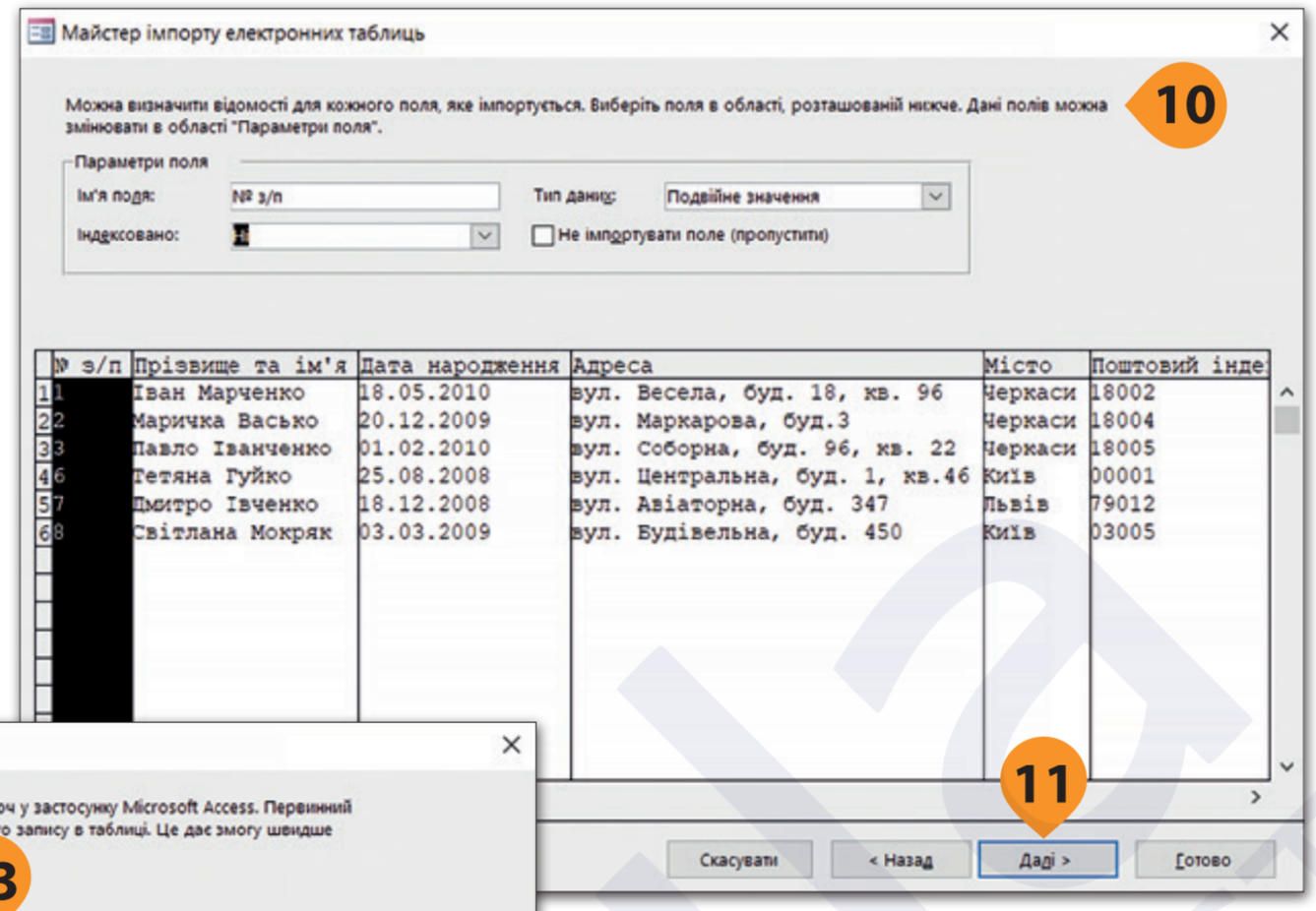
Створіть нову таблицю з усіма імпортованими даними.

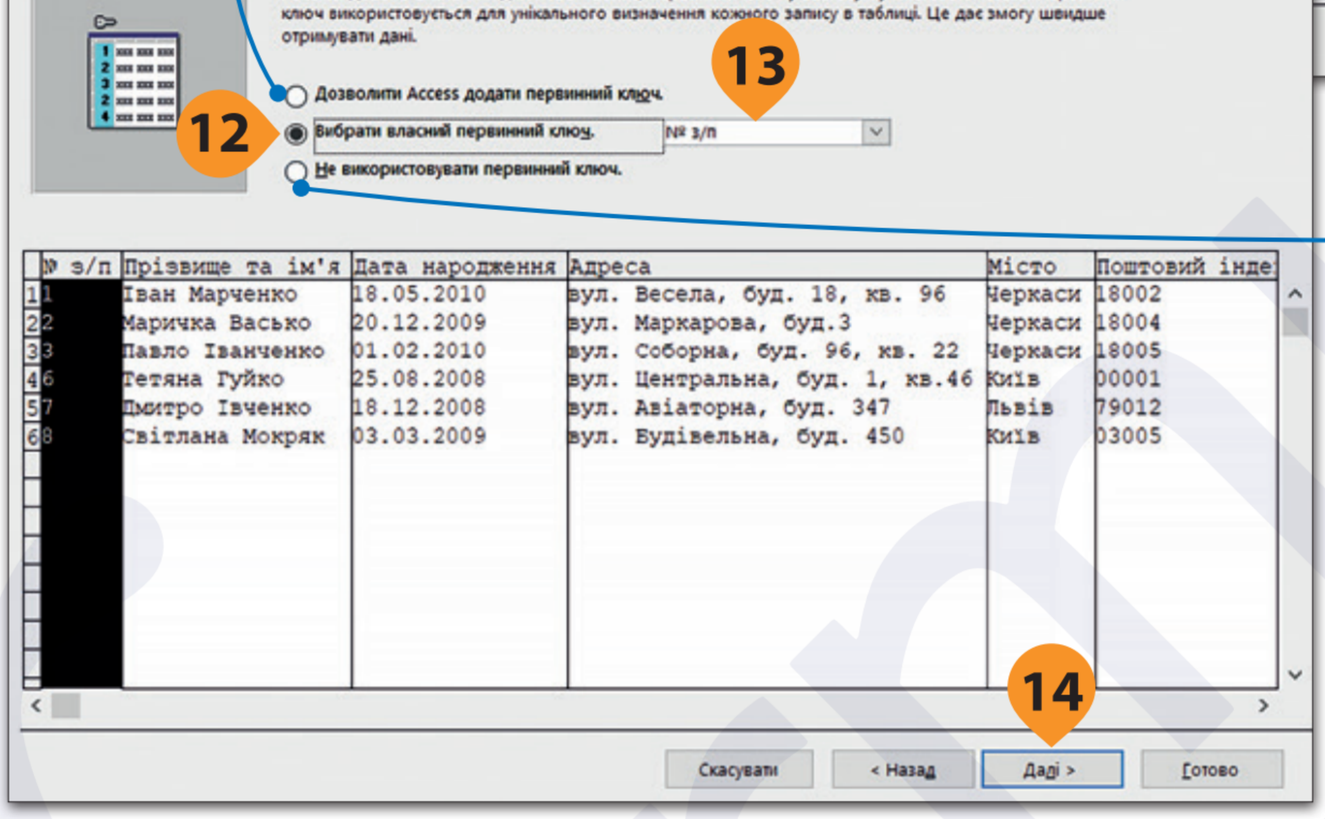
Створіть таблицю, підключивши її до Excel. Кожна зміна, яку ви робите в Excel, буде застосована до Access, але не навпаки.

До наявної бази даних можна додавати нові записи, не видаляючи старі – вони просто додаються в кінець таблиці.

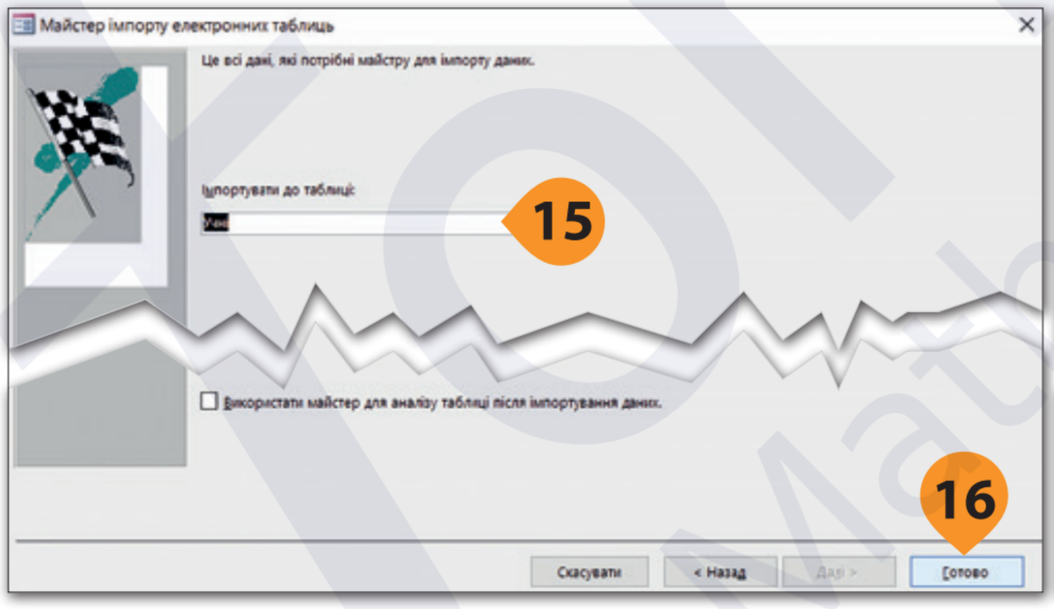
Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

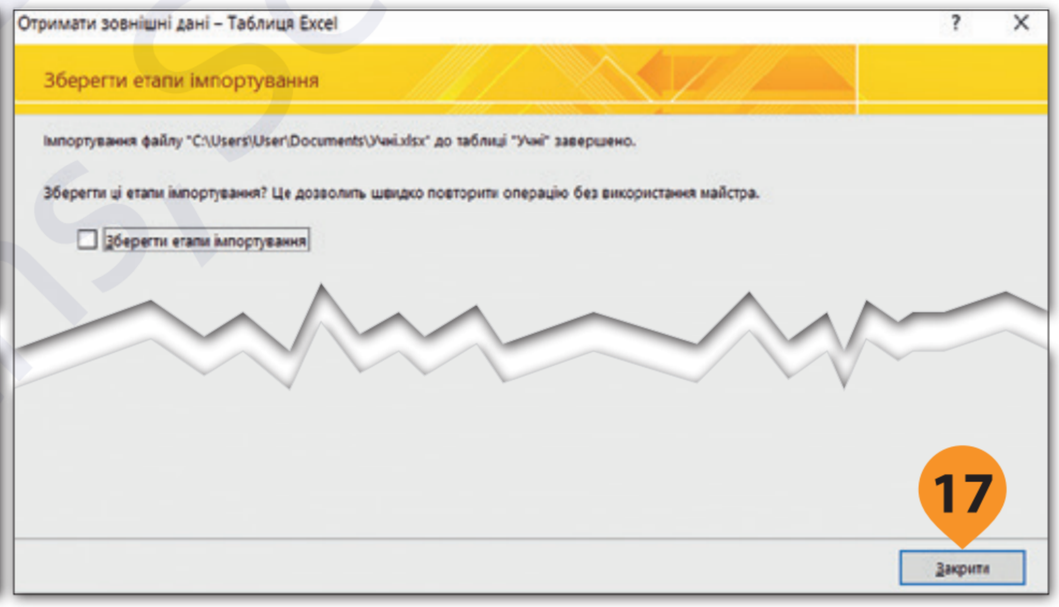
Оберіть це поле, і Microsoft Access вирішить, яке поле є **Ключовим**.

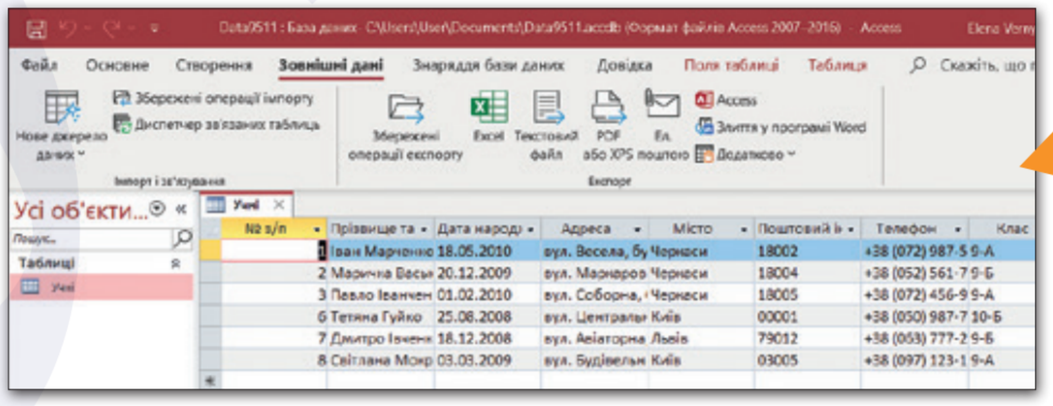


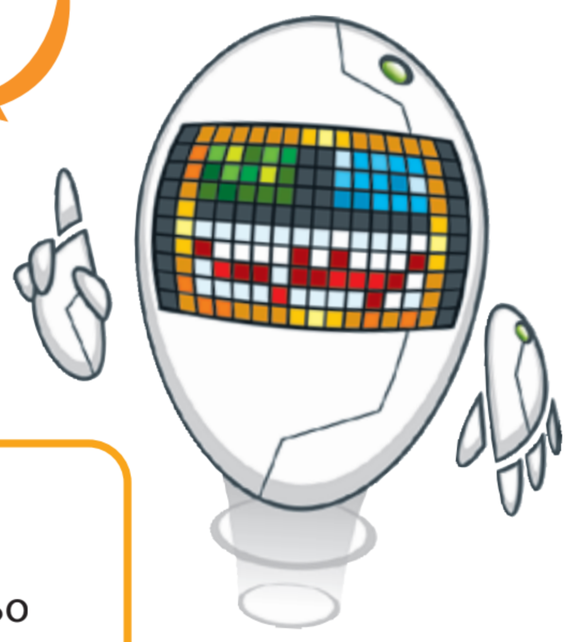


Виберіть це поле, і у вас не буде ключового поля.









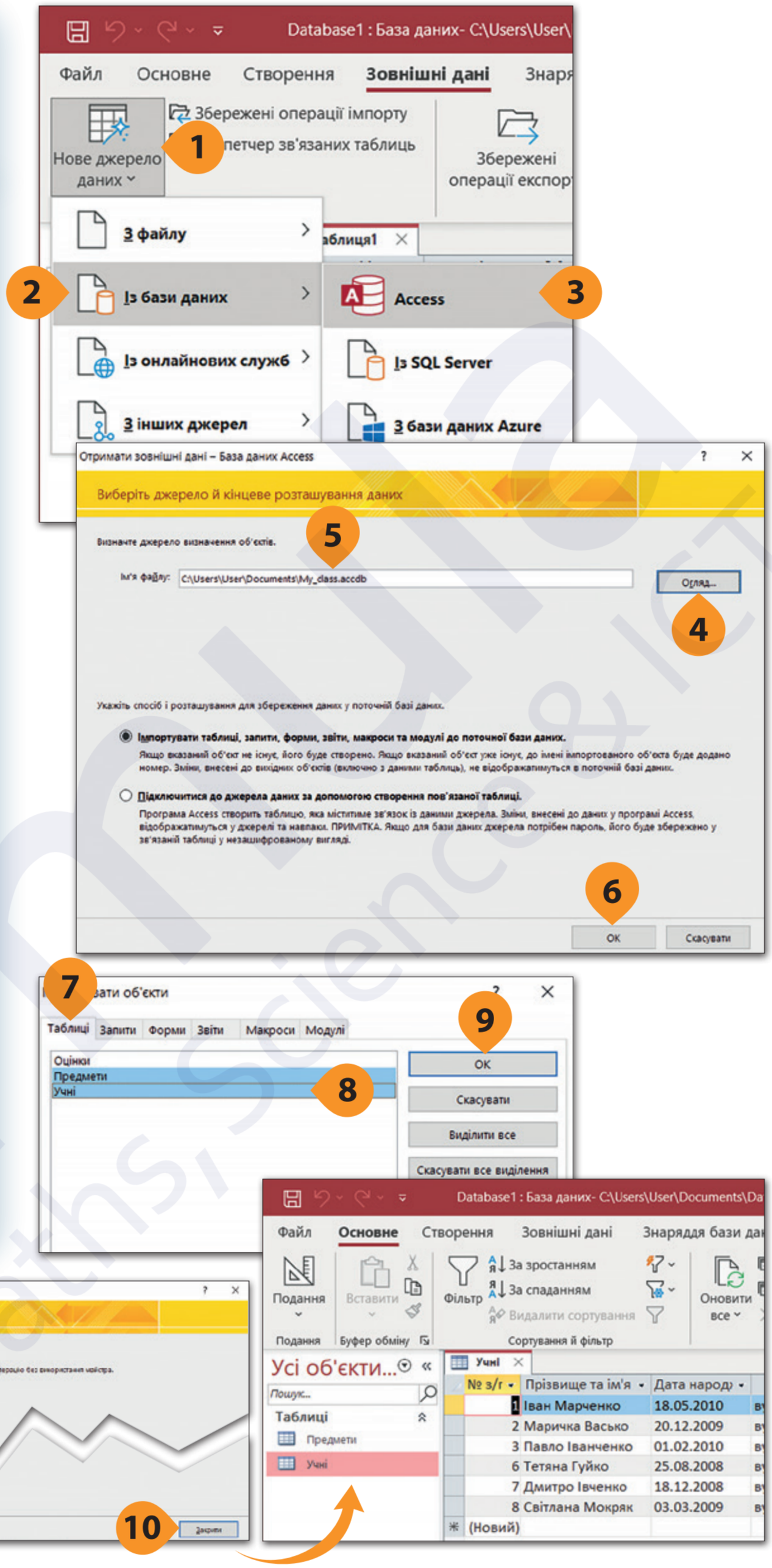
ДОВІДКА

Коли з'являється майстер, єдине, що треба робити, – це натискати **Далі**, доки не завершите всі кроки. Хоча процес повністю автоматизований, обов'язково перегляньте кожен екран, щоб переконатися, що попередньо вибрані опції саме ті, які вам потрібні.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Щоб імпортувати об'єкти бази даних: таблиці, запити, форми та звіти – з іншого файлу бази даних Microsoft Access:

- відкрийте **Microsoft Access** і створіть нову порожню базу даних;
- на вкладці **Зовнішні дані** у групі **Імпорт і зв'язування** клацніть **Нове джерело даних** (1);
- у меню, що з'явиться, оберіть опцію **Із бази даних** (2) і клацніть **Access** (3);
- у вікні **Отримати зовнішні дані – База даних Access** натисніть **Огляд** (4), щоб вибрати файл бази даних **Microsoft Access**, з якої потрібно імпортувати об'єкти, наприклад **My_class.accdb** (5);
- натисніть **ОК** (6);
- у вікні **Імпортувати об'єкти** виберіть об'єкти, які потрібно імпортувати, наприклад, у **Таблиці** (7) **Предмети** та **Учні** (8), натисніть **ОК** (9);
- натисніть **Закрити** (10), і ваші об'єкти з'являться на панелі **Область переходів**.



Можна імпортувати конкретний елемент з бази даних або базу даних цілком.

Практичне завдання

Завантажте файл-заготовку зі сторінки інтернет-підтримки. Експортуйте створену раніше спортивну таблицю до Microsoft Excel, відкрийте її, додайте ще кілька записів. Збережіть цей файл та імпортуйте до порожньої бази даних Microsoft Access.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Аналоги та альтернативи Microsoft Access

Google AppSheet

Google AppSheet – це основний low-code/no-code інструмент від Google, який тепер вважається найкращим аналогом Access у їхній екосистемі. AppSheet дає змогу створювати потужні мобільні та вебдодатки для збору та управління даними, використовуючи як джерело даних звичайні Google Таблиці, а також інші бази даних та хмарні сервіси. Він фокусується на автоматизації робочих процесів.

LibreOffice Base

LibreOffice Base – безкоштовний десктопний компонент з відкритим вихідним кодом, що є найближчим замінником Access, оскільки поєднує СКБД з візуальними інструментами для створення форм, запитів та звітів для локальних або невеликих баз даних.

Airtable

Airtable – це хмарна платформа, яка поєднує простоту електронної таблиці з можливостями реляційної бази даних. Ідеально підходить для спільної роботи, дає змогу легко керувати даними та створювати користувацькі інтерфейси без програмування.

Microsoft Power Apps

Microsoft Power Apps – low-code/no-code платформа від Microsoft, яка використовується для модернізації старих баз Access. Завдяки їй можна створювати мобільні та вебзастосунки, інтегровані з Office 365, SharePoint та іншими джерелами даних.

PostgreSQL

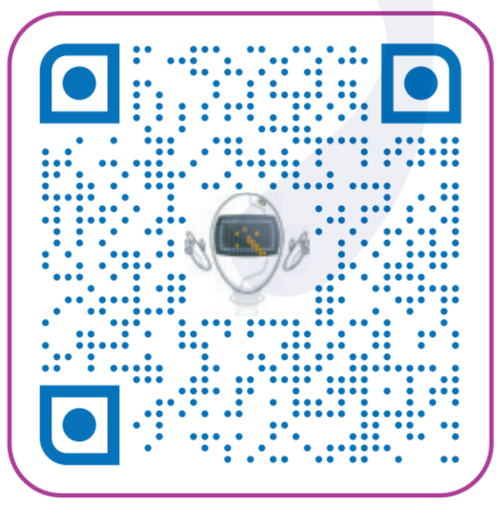
PostgreSQL: Потужна, відкрита, об'єктно-реляційна система керування базами даних (СКБД), яка є масштабованою та функціональною. Використовується для великих і критично важливих систем, але потребує додаткових інструментів для створення інтерфейсу.

Zoho Creator

Zoho Creator – це хмарне середовище для створення бізнес-застосунків, яке дозволяє користувачам розробляти прості та складні програми для роботи з базами даних (форми, звіти, робочі процеси) з мінімальним кодуванням. Creator добре підходить для автоматизації бізнес-процесів і пропонує широкі можливості інтеграції з іншими продуктами Zoho та сторонніми сервісами.

No-code (або zero-code) *– це технологія створення сайтів, чат-ботів, застосунків та ігор без необхідності писати код.*

https://formula.education/online-ict-9-klas/





Інші платформи



Проєкт



Підсумки



Глосарій

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

4. Програмні проєкти

Будь-який великий проєкт починається з ідеї. Але що перетворює ідею на надійний, конкурентоспроможний програмний продукт, здатний працювати роками?

Саме в цьому розділі ми пройдемо шлях від написання окремих скриптів до освоєння технології створення програмних продуктів.

Готові працювати за стандартами, які використовують професійні команди розробників по всьому світу?

Розпочнімо трансформацію вашого коду!



У цьому розділі ви навчитеся:

- розрізняти та застосовувати основні колекції Python: списки, кортежі та словники;
- професійно обробляти текстові дані;
- розрізняти та застосовувати інструменти стандартних бібліотек тазовнішніх бібліотек;
- користуватися технологією створення програмних продуктів;
- структурувати проєкти та підтримувати їхню якість.

Програмне забезпечення:

- Python IDLE;
- Sublime Text;
- PyCharm.

У цьому розділі ви дізнаєтеся:

- як ефективно використовувати основні структури даних Python;
- про повний життєвий цикл та технологію створення програмних продуктів – від планування до розгортання;
- як важливо документувати програмний проєкт;
- про сутність та ключові рівні тестування для забезпечення надійності продукту.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Тема 1

Списки та кортежі

У цій темі ви дізнаєтесь, що таке **списки** та **кортежі** в мові програмування **Python**. Навчитеся створювати ці структури даних, додавати й видаляти елементи, змінювати їх та здійснювати пошук потрібного значення. Окремо розглянете методи, які полегшують роботу зі списками, а також зрозумієте, у чому полягає відмінність між списками й кортежами.

Зверніть увагу, що елементи списку можуть бути будь-якого типу: текстові рядки, числа, логічні значення чи навіть інші списки.
У кожного елемента є свій номер, або індекс, починаючи з нуля.
Це означає, що перший елемент має індекс 0, другий – 1 і так далі.



Списки

Уявіть, що ви хочете зберегти в програмі кілька значень. Наприклад, перелік улюблених страв або результати контрольної роботи. Звичайна змінна дає змогу зберегти лише одне значення, тож цього буде замало. Саме тому у Python існує спеціальний тип даних – **список**.

Приклад списків:

```
fruits = ["яблуко", "банан", "вишня"]
number = [1, 2, 3, 4, 5]
mix = [1, "текст", True, 3.14]
```

Приклад індексації списку:

```
MyFavFood = ["вареники","піца", "суші", "гречка", "борщ",]
```

Індекс	0	1	2	3	4
Елемент	"вареники"	"піца"	"суші"	"гречка"	"борщ"

Якщо ви звернетеся до елемента з індексом 0, отримаєте слово «вареники», адже це перший елемент. А якщо використаєте індекс 4, отримаєте «борщ», бо це останній елемент у списку.

Програма:

```
MyFavFood = ["вареники","піца", "суші", "гречка", "борщ",]
print(MyFavFood [0])
print(MyFavFood [4])
```

Результат:

```
вареники
борщ
```

ДОВІДКА

Розробники Python хотіли створити тип даних, який міг би зберігати елементи **будь-якого** типу. Список у Python може містити цілі числа, рядки, інші списки тощо, що робить його надзвичайно гнучким інструментом.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Часто виникає потреба взяти не один елемент, а відразу кілька. У такому випадку зручно користуватися зрізами. Наприклад, ви можете отримати елементи з першого по третій, використавши запис страви [1 : 4]. У результаті буде створено новий список: ["піца", "суші", "гречка"]. Якщо вказати тільки початок або кінець зрізу, Python автоматично візьме решту елементів.

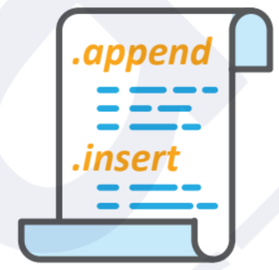
Програма:

```
MyFavFood = ["вареники","піца", "суші", "гречка", "борщ",]
print(MyFavFood [1:4])   # Виведе: ['піца', 'суші', 'гречка']
print(MyFavFood [:3])    # Виведе: ['вареники', 'піца', 'суші']
print(MyFavFood [2:])    # Виведе: ['суші', 'гречка', 'борщ']
```

Результат:

```
['піца', 'суші', 'гречка']
['вареники', 'піца', 'суші']
['суші', 'гречка', 'борщ']
```

Списки цінні тим, що їх можна змінювати. Ви можете додати новий елемент у кінець, використавши метод **append**, або вставити його на будь-яку позицію за допомогою **insert**.



Програма:

```
MyFavFood = ["вареники","піца", "суші", "гречка", "борщ",]
MyFavFood.append("морозиво")
print(MyFavFood)
```

Результат:

```
['вареники', 'піца', 'суші', 'гречка', 'борщ', 'морозиво']
```

```
MyFavFood = ['вареники', 'піца', 'суші', 'гречка', 'борщ',"морозиво"]
MyFavFood.insert(2, "салат")
print(MyFavFood)
```

```
['вареники', 'піца', 'салат', 'суші', 'гречка', 'борщ', 'морозиво']
```

Якщо ж якийсь елемент більше не потрібен, його можна видалити. Це робиться кількома способами: методом **remove** – за значенням, методом **pop** – за індексом, або ж за допомогою оператора **del**.



Програма:

```
MyFavFood = ['вареники', 'піца',"салат", 'суші', 'гречка', 'борщ',"морозиво"]
MyFavFood.remove("піца")  # Видаляє "піца"
print(MyFavFood)
```

Результат:

```
['вареники', 'салат', 'суші', 'гречка', 'борщ', 'морозиво']
```

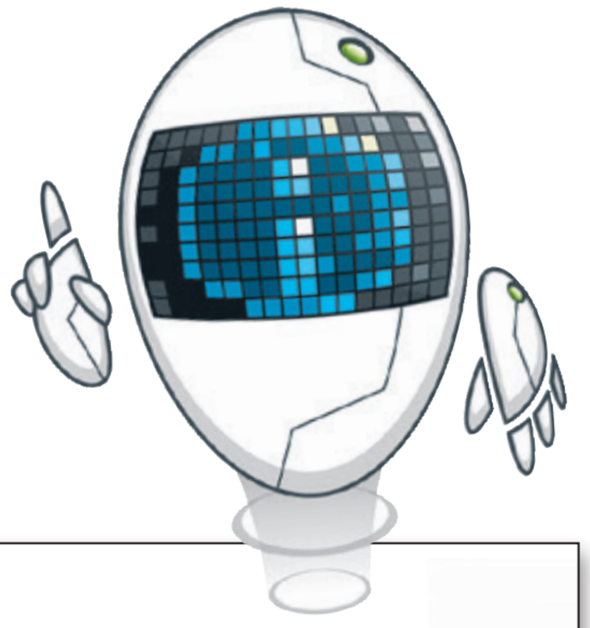
```
MyFavFood = ['вареники',"салат", 'суші', 'гречка', 'борщ',"морозиво"]
LastFood = MyFavFood.pop()  # Видаляє останній елемент
print(MyFavFood)
```

```
['вареники', 'салат', 'суші', 'гречка', 'борщ']
```

```
MyFavFood = ['вареники',"салат", 'суші', 'гречка', 'борщ']
del MyFavFood [0]  # Видаляє елемент з індексом 0
print(MyFavFood)
```

```
['салат', 'суші', 'гречка', 'борщ']
```

*Коли потрібно знайти певний елемент, стане у пригоді метод **index**, який повертає його номер у списку. А щоб перевірити, чи є взагалі таке значення у списку, можна використати ключове слово **in**.*



Програма:

```
MyFavFood = ["салат", 'суші', 'гречка', 'борщ']
Index = MyFavFood.index("суші")  # Поверне: 1
print(Index)
```

Результат:

```
1
```

```
MyFavFood = ["салат", 'суші', 'гречка', 'борщ']
print("гречка" in MyFavFood)          # Поверне: True
```

```
True
```


Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Методи та функції списків

Функція	Дія	Приклад

append( )	Додає елемент у кінець списку	**Програма:**
		`my_list=[8,'word','a',89,[1,2,3]]`
		`my_list.append('word2')`
		`print(my_list)`
		Результат: `>>> [8, 'word', 'a', 89, [1, 2, 3], 'word2']`

insert( )	Вставляє елемент в обрану позицію	**Програма:**
		`my_list=[8,'word','a',89,[1,2,3]]`
		`my_list.insert(2,'word2')`
		`print(my_list)`
		Результат: `>>> [8, 'word', 'word2', 'a', 89, [1, 2, 3]]`

del	Видаляє елемент списку за вказаним номером	**Програма:**
		`my_list=[8,'word','a',89,[1,2,3]]`
		`del my_list[1]`
		`print(my_list)`
		Результат: `>>> [8, 'a', 89, [1, 2, 3]]`

reverse( )	Здійснює реверс (зміну порядку елементів списку на зворотний)	**Програма:**
		`my_list=[8,'word','a',89,[1,2,3]]`
		`my_list.reverse()`
		`print(my_list)`
		Результат: `>>> [[1, 2, 3], 89, 'a', 'word', 8]`

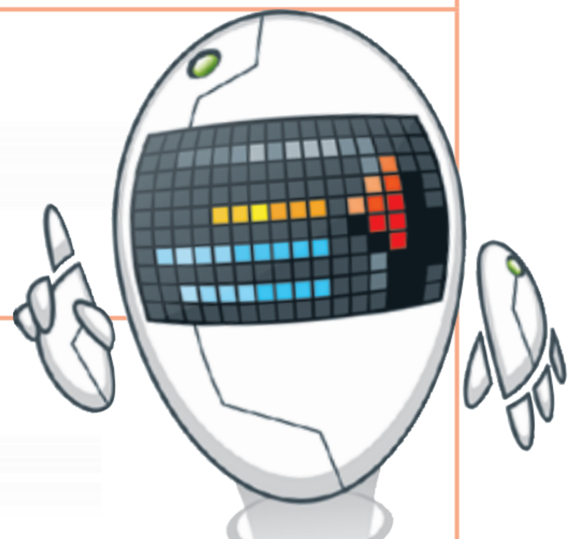
len( )	Повертає кількість елементів списку	**Програма:**
		`my_list=[8,'word','a',89,[1,2,3]]`
		`print(len(my_list))`
		Результат: `>>> 5`

max( )	Повертає максимальний елемент списку	**Програма:**
		`my_list=[8,10,89,99]`
		`print(max(my_list))`
		Результат: `>>> 99`

min( )	Повертає мінімальний елемент списку	**Програма:**
		`my_list=[8,10,89,99]`
		`print(min(my_list))`
		Результат: `>>> 8`

sum( )	Повертає суму елементів списку	**Програма:**
		`my_list=[8,10,89,99]`
		`print(sum(my_list))`
		Результат: `>>> 206`

count(n)	Повертає кількість елементів зі значенням *n*	**Програма:**
		`my_list=[8,10,89,99,8]`
		`print(my_list.count(8))`
		Результат: `>>> 2`



Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Інші методи та функції списків

Метод	Опис
extend()	Додає елементи іншого списку в кінець поточного списку
pop(індекс)	Видаляє елемент за індексом і повертає його значення
remove( )	Видаляє перше входження вказаного значення зі списку
sort( )	Сортує елементи списку
index( )	Повертає індекс першого входження елемента
copy()	*Створює та повертає копію списку*
list( )	Перетворює інший тип даних у список

Кортеж *– це фундаментальна структура даних у Python, яка являє собою впорядковану, незмінну колекцію елементів, що можуть бути різних типів. Назва походить із математики, де «n-кортеж» означає впорядковану послідовність n елементів.*



Кортежі

Окрім списків, у Python є ще одна схожа структура – **кортеж**. Вона дуже схожа на список, але має одну важливу відмінність: кортежі незмінні. Це означає, що після створення змінити їхній вміст не можна. Кортеж створюється за допомогою круглих дужок.

Програма:

```
coordinates = (10, 20)
color = ("червоний", "зелений", "синій")
oneElement = (42)

print(coordinates)
print(color)
print(oneElement)
```

Результат:

```
(10, 20)
('червоний', 'зелений', 'синій')
42
```

Незважаючи на свою «жорсткість», кортежі мають перевагу – вони працюють швидше за списки. Їх зручно використовувати там, де дані не повинні змінюватися.

Щоб краще зрозуміти відмінність, пригадайте, що списки ми записуємо у квадратних дужках, а кортежі – у круглих. Списки можна змінювати, тоді як кортежі залишаються сталими. Через це робота зі списками трохи повільніша, але вони зручніші у випадках, коли дані змінюються. Кортежі ж підходять для зберігання постійної інформації.

Порівняльна таблиця списків та кортежів

Властивість	Список (list)	Кортеж (tuple)
Розміщуються в	[ ]	( )
Змінюваність	Так	Ні
Швидкість обробки	Повільніше	Швидше
Використання	Для динамічних даних	Для постійних даних

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Ви вже знаєте, що списки зберігають елементи в певному порядку, а кортежі – це такі самі впорядковані колекції, але незмінні. Та в житті часто бувають ситуації, коли потрібно працювати не лише з наборами значень, а ще й з інформацією у вигляді:



«ключ → значення»

Наприклад, якщо ви хочете зберегти номери телефонів друзів, то незручно тримати два списки: один зі словами «Андрій, Марія, Оксана», а другий – з номерами. У Python для цього існує спеціальна структура – **словник**.

Приклад словника:

```
PhoneBook = { "Андрій": "011111111", "Марія": "0222222222", "Оксана": "0333333333" }
```

У цьому словнику ключами є імена, а значеннями – номери телефонів.

Щоб отримати значення за ключем, достатньо звернутися до словника так само, як ви це робили зі списком за індексом, тільки замість числа використовуйте ключ:

Програма:

```
PhoneBook = { "Андрій": "011111111", "Марія": "0222222222", "Оксана": "0333333333" }
print(PhoneBook ["Марія"])    # Виведе: 0222222222
```

Результат:

```
0222222222
```

Можна також додавати нові пари:

Програма:

```
PhoneBook = { "Андрій": "011111111", "Марія": "0222222222", "Оксана": "0333333333" }
PhoneBook ["Ігор"] = "04444444444"

print(PhoneBook)
```

Результат:

```
{'Андрій': '011111111', 'Марія': '0222222222',
'Оксана': '0333333333', 'Ігор': '04444444444'}
```

Або змінювати значення:

Програма:

```
PhoneBook = {'Андрій': '011111111', 'Марія': '0222222222', 'Оксана': '0333333333',
             'Ігор': '04444444444'}
PhoneBook ["Оксана"] = "055555555"
print(PhoneBook)
```

Результат:

```
{'Андрій': '011111111', 'Марія': '0222222222',
'Оксана': '055555555', 'Ігор': '04444444444'}
```

Якщо потрібно видалити запис:

Програма:

```
PhoneBook = {'Андрій': '011111111', 'Марія': '0222222222', 'Оксана': '0555555555',
             'Ігор': '04444444444'}
del PhoneBook ["Андрій"]
print(PhoneBook)
```

Результат:

```
{'Марія': '0222222222', 'Оксана': '0555555555',
'Ігор': '04444444444'}
```

ПІДКАЗКА

Словники надають семантичний доступ до даних (за іменем або ідентифікатором), а списки та кортежі – позиційний (за порядком).

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Вам відомо, що словники дають змогу зберігати пари «ключ → значення» та працювати з ними. Але інколи нам потрібно не просто звертатися до окремого елемента, а швидко отримати всі ключі, всі значення чи навіть переглянути словник повністю. Для цього в Python існують спеціальні методи, які значно полегшують роботу. Вони допомагають швидше знаходити потрібну інформацію, зручно виводити дані на екран і навіть уникати помилок, якщо ключа у словнику немає.

Списки, кортежі та словники використовуються в Python надзвичайно часто – вони є одними з найфундаментальніших і найбільш вживаних структур даних. Фактично, вони становлять основу майже будь-якої програми, від простих скриптів до складних наукових обчислень.

Перелік усіх методів та функцій списків

Функція	Опис
all()	Повертає True, якщо всі ключі словника дорівнюють True (або якщо словник порожній)
any()	Повертає True, якщо хоч один із ключів словника дорівнює True. Якщо словник порожній, повертається False
len()	Повертає довжину (кількість елементів) словника
sorted()	Повертає новий відсортований список ключів у словнику
clear()	Видаляє всі елементи зі словника
keys()	Повертає новий об'єкт ключів словника
values()	Повертає новий об'єкт значень словника

Отже, списки слугують для упорядкованих колекцій, де важлива гнучкість; кортежі – для незмінних даних, де важлива стабільність; а словники – для моделювання зв'язків «ключ → значення», де важлива швидкість доступу за ідентифікатором.

Вибір правильної структури даних є значущим чинником для створення ефективного та функціонального коду.



Знання ключових методів та функцій списків і кортежів є критично важливим, оскільки це не просто зазубрювання синтаксису; це основа ефективного програмування, що забезпечує написання швидкого, надійного і зрозумілого коду.

Практичне завдання

Завантажте файл-заготовку зі сторінки інтернет-підтримки. Виконайте запропоноване завдання.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Тема 2
Алгоритми впорядкування масивів / списків

Впорядкування даних (або сортування)

Масив даних (або просто масив) – це структура даних, яка зберігає колекцію елементів одного типу в суміжних (послідовних) комірках пам'яті. Завдяки цій властивості доступ до будь-якого елемента масиву відбувається за його індексом (порядковим номером), і це дуже швидка операція.

Сортування допомагає нам «навести лад» у масиві даних. Уявіть журнал із оцінками, де всі оцінки записані хаотично. Якщо ми впорядкуємо їх за зростанням, то легко зможемо знайти найнижчу і найвищу. Якщо ж за спаданням – одразу побачимо найкращі результати.

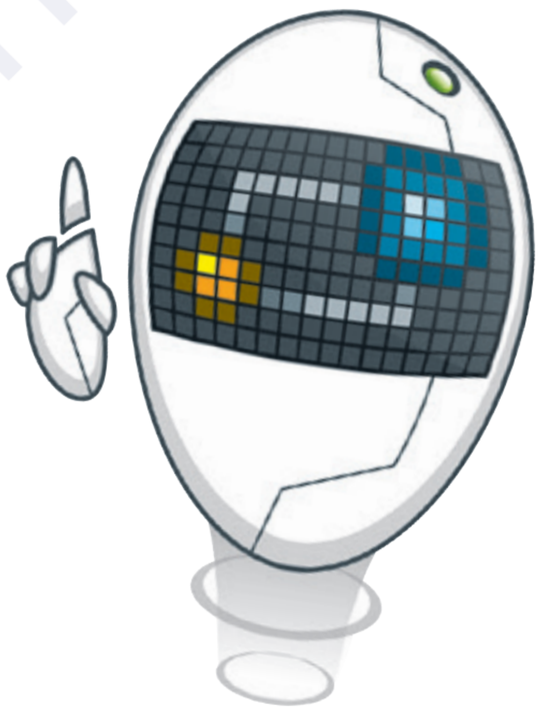
Впорядкування даних (або сортування) *– це процес розташування елементів множини даних у певному порядку відповідно до заданої ознаки або критерію, як-от за алфавітом, за числовим значенням (від меншого до більшого або навпаки), за датою чи будь-яким іншим критерієм.*

NumPy

У **Python** немає окремого вбудованого типу «масив», як у деяких інших мовах програмування. Для збереження колекцій даних ми користуємося списками. Проте, коли йдеться про обробку великих наборів числових даних (наприклад, у науці, статистиці чи машинному навчанні), використовують масиви бібліотеки **NumPy**.

Чому використовують бібліотеку **NumPy** замість списків? Масиви **NumPy** швидші та компактніші, ніж списки Python. Масив споживає менше пам'яті (що дуже важливо для оптимізації швидкості обчислень або використовуваної потужності) і зручний у використанні. **NumPy** потребує набагато менше пам'яті для зберігання даних і надає механізм вказівки типів даних. Це сприяє кращій оптимізації коду.

Масив **NumPy** призначений саме для чисел. Він працює набагато швидше за список, особливо з великими наборами даних, і має багато вбудованих функцій для математичних операцій та сортування.



Застосування NumPy

- Машинне навчання
- Аналіз даних
- Статистика
- Обробка зображень

ДОВІДКА

NumPy (скорочення від **Numerical Python**) насправді був створений у 2005 році Тревісом Оліфантом. Він об'єднав найкращі функції двох попередніх бібліотек – **Numeric** (розроблена Джимом Гаґуніним у 1995 році) та **Numarray**. Метою було створити єдину, потужну бібліотеку для числових обчислень, яка стала б стандартом у науковій спільноті Python. Завдяки своїй ефективності, швидкості та простоті використання NumPy став **фундаментальною бібліотекою** для наукових обчислень, обробки даних та машинного навчання в Python.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Python має вбудовану функцію **sorted()** та метод списку **.sort()**, які дають змогу впорядковувати дані за зростанням або спаданням.

Програма:

```
marks = [12, 7, 9, 10, 8]
print(sorted(marks))          # [7, 8, 9, 10, 12]
print(sorted(marks, reverse=True))  # [12, 10, 9, 8, 7]
```

Результат:

```
[7, 8, 9, 10, 12]
[12, 10, 9, 8, 7]
```

reverse=True *– це аргумент у функціях сортування, який вказує на впорядкування у зворотному або спадному порядку*

Це простий приклад, але коли ми працюємо з більшими наборами чисел – краще використовувати масиви **NumPy**, бо вони значно швидші.

Перш ніж використовувати готові інструменти Python чи бібліотеки, варто зрозуміти, як працюють класичні алгоритми сортування. Вони допомагають побачити загальну ідею: як із неупорядкованого списку отримати впорядкований. Розглянемо кілька прикладів.

Недолік методу – він досить повільний на великих наборах даних. У найгіршому разі час виконання зростає пропорційно квадрату кількості елементів.

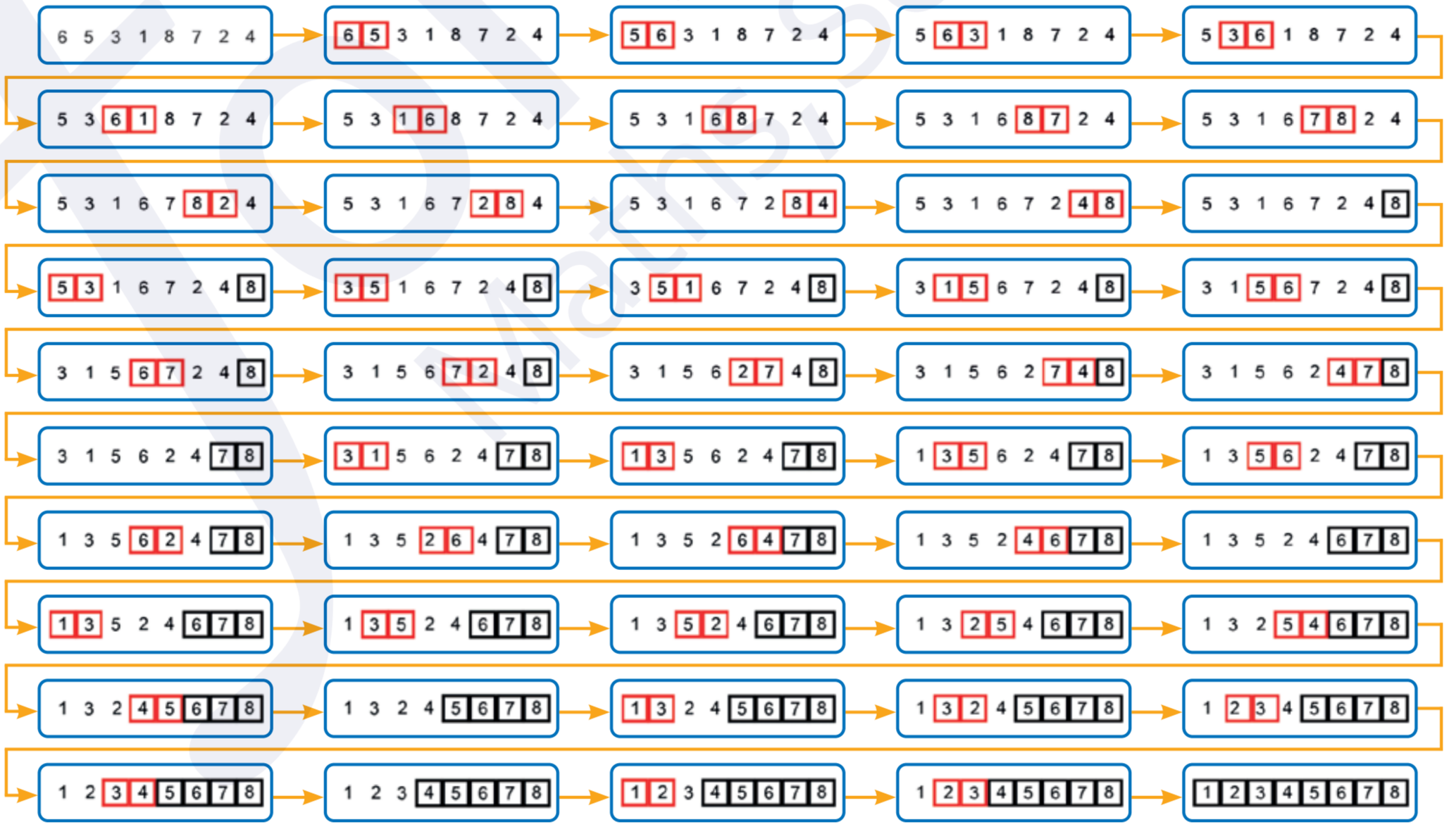


Сортування бульбашкою (Bubble Sort)

Це один із найперших алгоритмів, який вивчають під час знайомства з інформатикою. Його перевага – у наочності: він дуже добре демонструє принцип поступового впорядкування.

Ідея проста: ми багаторазово переглядаємо список, порівнюючи сусідні елементи. Якщо вони розташовані в неправильному порядку – міняємо їх місцями. Крок за кроком більші елементи «спливають» угору списку, ніби бульбашки у воді. Процес повторюється доти, доки весь список не стане впорядкованим.

Сортування бульбашкою (Bubble Sort):



Сортування даних виникло за потреби людини та машин впорядковувати інформацію для її пошуку, аналізу та зручного представлення.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

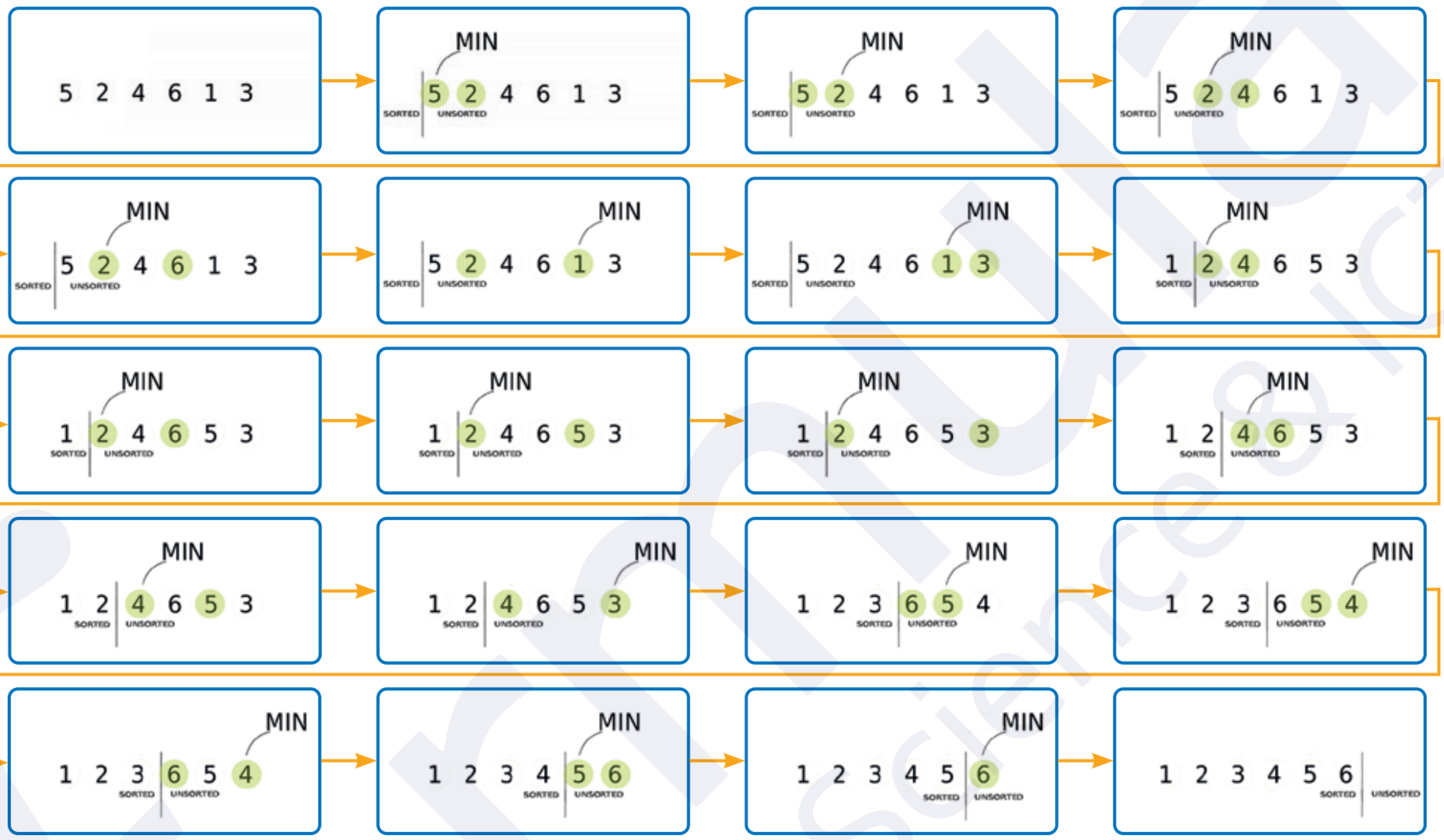
Сортування вибором (Selection Sort)



Алгоритм вибору теж належить до простих, але працює трохи швидше за бульбашкове.

Принцип такий: ми ділимо масив на дві частини – впорядковану та невпорядковану. На кожному кроці шукаємо найменший елемент серед невпорядкованих і переносимо його наприкінці впорядкованої частини. Отже, крок за кроком список поступово набуває впорядкованого вигляду.

Сортування вибором (Selection Sort):



Сортування злиттям (Merge Sort)

Більш ефективний підхід – це «розділяй і володарюй».

Алгоритм злиття складається з двох етапів:

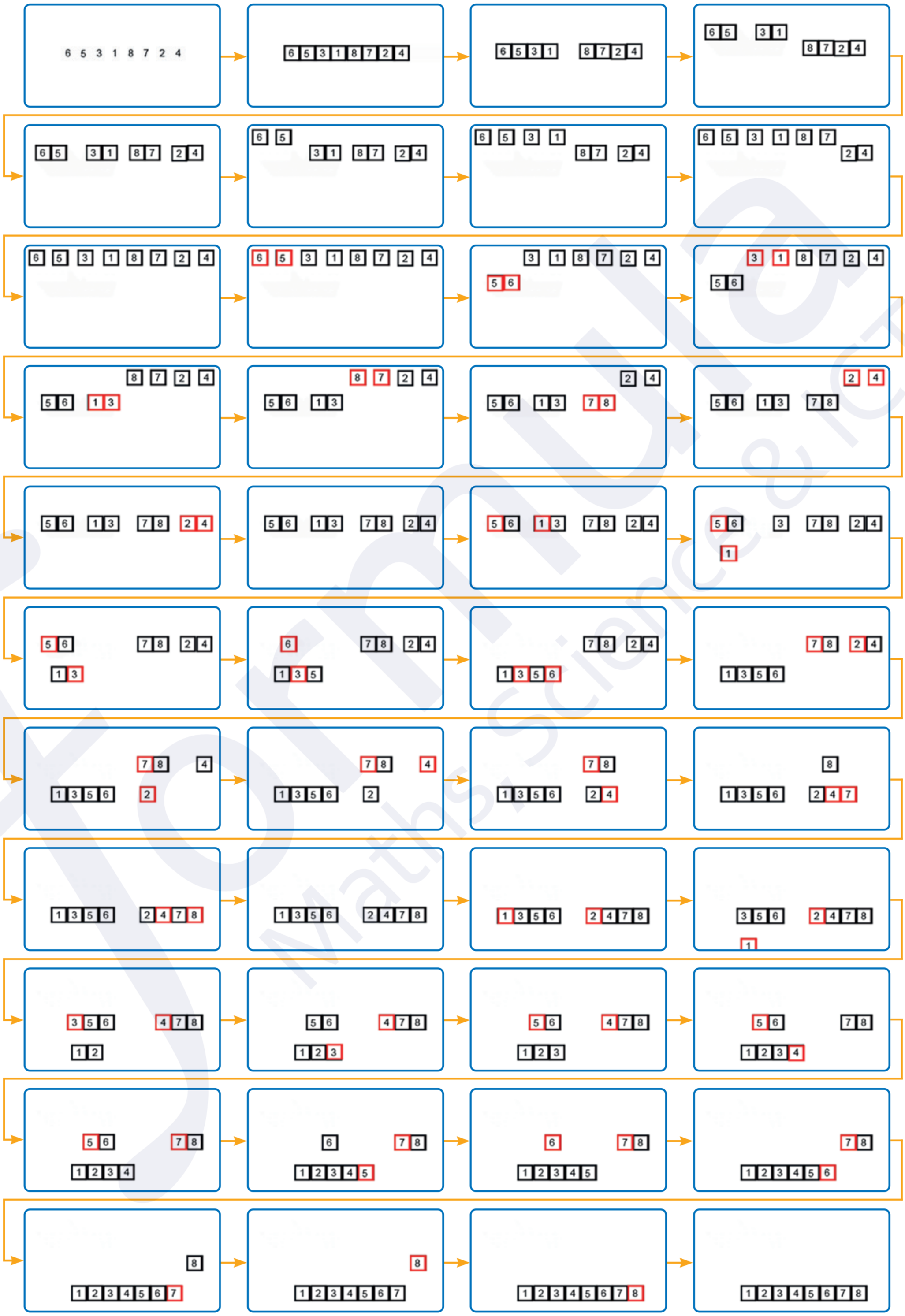
1. Масив ділиться на дрібніші частини, аж доки ми не отримаємо окремих елементів
2. Потім ці частини поступово «зливаються» парами, утворюючи впорядковані підмасиви, доки не збереться один великий відсортований список

ІСТОРІЯ

Сортування злиттям було винайдено американським математиком і вченим Джоном фон Нейманом у 1945 році. Він був першовідкривачем у галузі комп'ютерних наук і розробив цей алгоритм, працюючи над архітектурою ранніх комп'ютерів. Алгоритм базується на ідеї рекурсивного розбиття проблеми на менші, простіші частини. Фон Нейман зрозумів, що набагато легше об'єднати (злити) два вже відсортовані списки, ніж сортувати один неорганізований список.

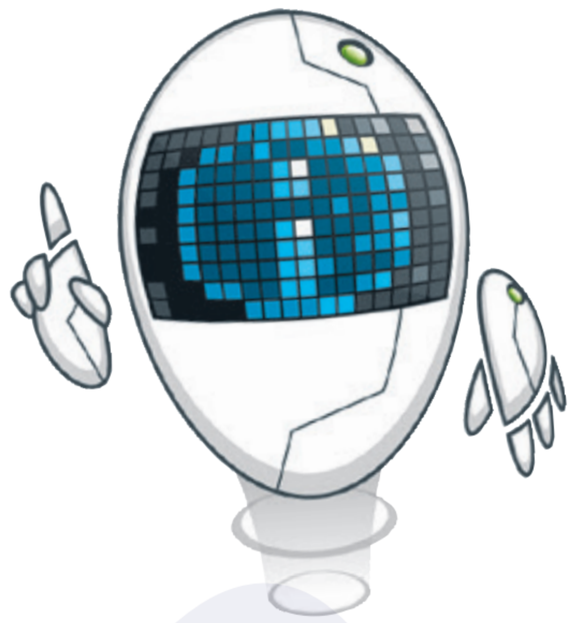
Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Сортування злиттям (Merge Sort):



Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

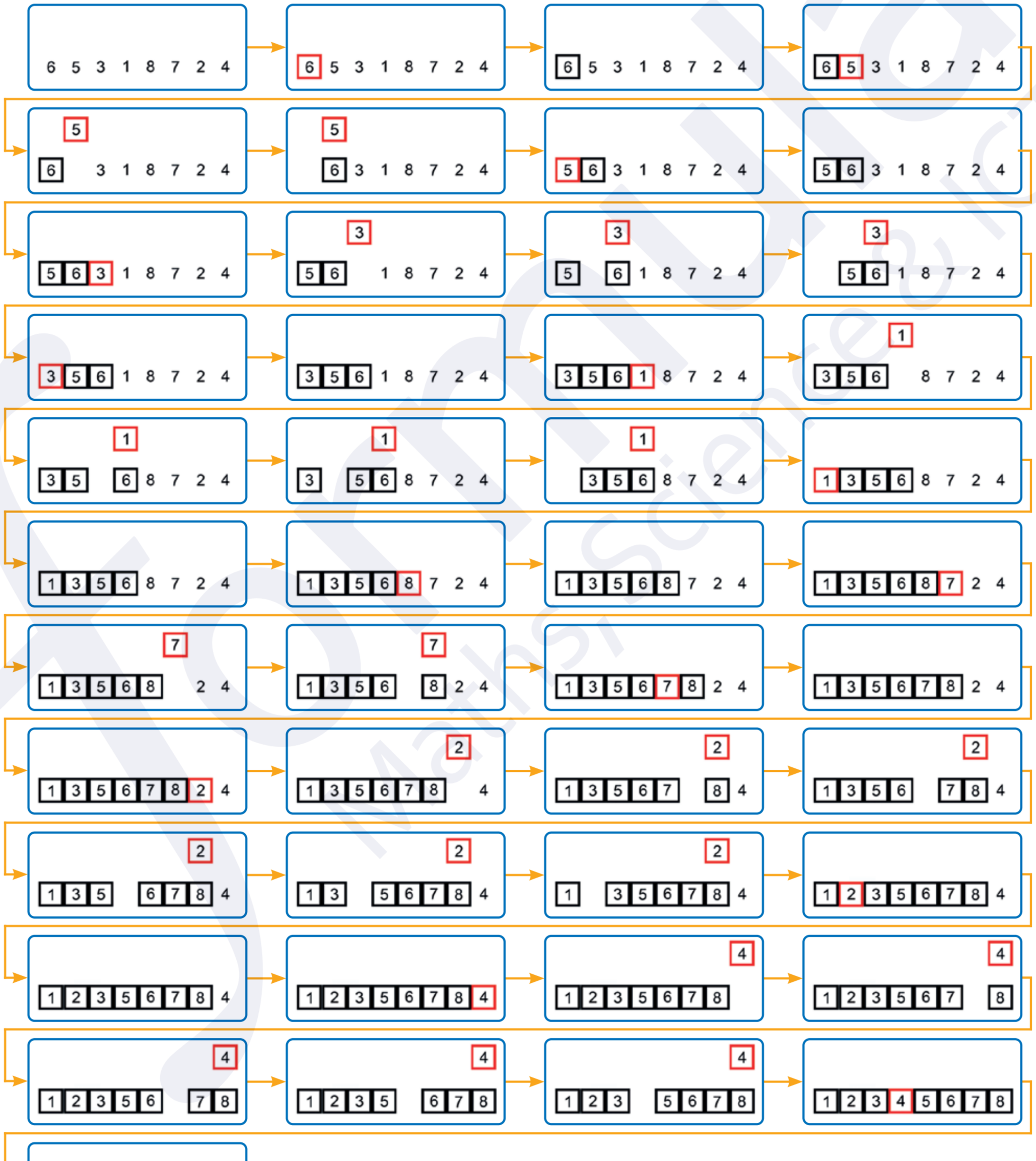
Сортування вставками (Insertion Sort)



Цей метод нагадує процес розкладання карт у руці. Ми беремо черговий елемент і вставляємо його на правильне місце серед тих, що вже впорядковані.

Так продовжується, доки всі елементи не займуть своїх позицій. Для невеликих масивів цей алгоритм досить зручний і швидкий, хоча на великих обсягах даних він поступається складнішим методам.

Сортування вставками (Insertion Sort):



Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Швидке сортування (Quick Sort)

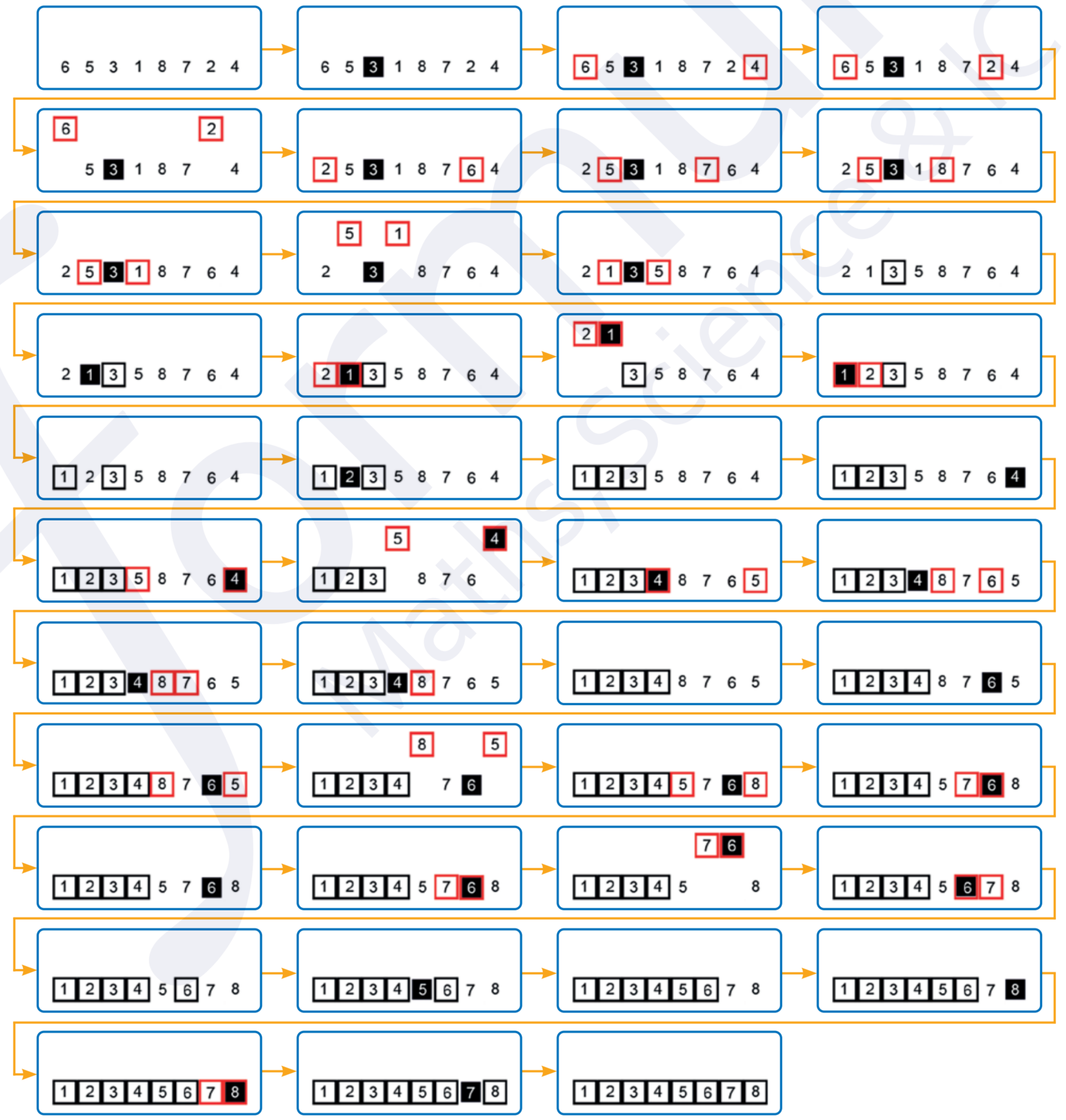
Ще один відомий алгоритм, який часто використовують на практиці. Він дещо складніший за попередні, але зазвичай працює швидше.

Загалом алгоритм працює швидко, хоча в деяких випадках його час виконання може зростати.

Ідея швидкого сортування:

1. Обираємо один елемент як опорний
2. Розміщуємо всі менші за нього значення ліворуч, а більші – праворуч (цей процес називається розбиттям)
3. Повторюємо попередні кроки для кожної з отриманих частин, доки всі підмасиви не будуть впорядковані

Швидке сортування (Quick Sort):



Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Ви вже знаєте, як можна сортувати списки у Python за допомогою вбудованих функцій. Проте коли йдеться про великі набори числових даних, зручніше і швидше працювати з масивами бібліотеки **NumPy**.



Основна функція для впорядкування – це ***np.sort()***. Вона повертає новий відсортований масив, не змінюючи початкового. Це дуже важливо: якщо ми хочемо зберегти і початкові дані, і впорядковані, то не доведеться робити копію вручну.

Програма:

```
import numpy as np
# Створення масиву з випадкових чисел
arr = np.random.randint(1, 100, size=10)
print("Початковий масив:", arr)
# Сортування за зростанням
print("За зростанням:", np.sort(arr))
# Сортування за спаданням
print("За спаданням:", np.sort(arr)[::-1])
```

Результат:

```
Початковий масив: [94 78  5 44 14 49 74 11 81 10]
За зростанням: [ 5 10 11 14 44 49 74 78 81 94]
За спаданням: [94 81 78 74 49 44 14 11 10  5]
```

Якщо після запуску отримали помилку:

Помилка: ModuleNotFoundError: No module named 'numpy':

```
Traceback (most recent call last):
  File "D:/Python/Тема  2/сортування.py", line 1, in <module>
    import numpy as np
ModuleNotFoundError: No module named 'numpy'
```



Це означає Python не може знайти бібліотеку NumPy, яка імпортується в першому рядку коду. Тому її потрібно додатково встановити.

У цьому прикладі ми отримуємо одразу два результати: відсортований список від меншого до більшого та від більшого до меншого.

Крім того, у **NumPy** є ще один метод – ***ndarray.sort()***, який виконує сортування «на місці», тобто змінює сам масив. Це може бути зручним, коли ми не плануємо більше використовувати початковий порядок даних.

Один з можливих варіантів додатково встановити бібліотеку NumPy:

- відкрийте **Командний рядок (Command Prompt, Terminal або PowerShell)**;
- введіть команду *py -m pip install numpy*.

Програма:

```
import numpy as np
# Створення масиву з випадкових чисел
arr = np.random.randint(1, 100, size=10)
print("Початковий масив:", arr)
arr.sort()
print("Масив після сортування:", arr)
```

Результат:

```
Початковий масив: [20 19 72 89  9 48 30 50 82  7]
Масив після сортування: [ 7  9 19 20 30 48 50 72 82 89]
```

ПІДКАЗКА

Основна вбудована функція Python, ***sorted()***, а також метод ***list.sort()***, використовують не Quick Sort і не Merge Sort, а гібридний алгоритм під назвою **TimSort**. Він був розроблений Тімом Пітерсом спеціально для Python. Це стабільний алгоритм, який є комбінацією **Merge Sort** та **Insertion Sort**. **TimSort** є не просто швидким, а й **інтелектуальним**. Він аналізує, наскільки добре дані відсортовані, і використовує це для значного прискорення процесу в реальних застосунках.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

NumPy дозволяє працювати не тільки з одномірними, а й з багатовимірними масивами. Упорядкування у такому разі можна робити різними способами – залежно від того, що саме ми хочемо отримати.

Цей приклад коду демонструє потужні можливості бібліотеки NumPy для роботи з багатовимірними масивами, що є ключовим у наукових обчисленнях та аналізі даних. Функція ***np.sort()*** та параметр **axis** забезпечують гнучкість у роботі зі складними матричними структурами, даючи розробнику повний контроль над даними.

У функції *np.sort()* існує параметр *axis*, який визначає напрямок сортування:

- ***axis=0*** – *сортування відбувається за стовпцями. Кожен стовпець упорядковується окремо від верхнього до нижнього значення*
- ***axis=1*** – *сортування відбувається за рядками. Кожен рядок упорядковується незалежно від інших*

Програма:

```
import numpy as np
# Створення двовимірного масиву
arr = np.random.randint(10, 100, size=(4, 4))
print("Початковий масив:\n", arr)
# Сортування за стовпцями
sorted_columns = np.sort(arr, axis=0)
print("Відсортований за стовпцями:\n", sorted_columns)
# Сортування за рядками
sorted_rows = np.sort(arr, axis=1)
print("Відсортований за рядками:\n", sorted_rows)
```

Результат:

```
Початковий масив:
 [[44 76 96 37]
 [18 19 90 95]
 [80 78 21 37]
 [59 19 64 84]]
Відсортований за стовпцями:
 [[18 19 21 37]
 [44 19 64 37]
 [59 76 90 84]
 [80 78 96 95]]
Відсортований за рядками:
 [[37 44 76 96]
 [18 19 90 95]
 [21 37 78 80]
 [19 59 64 84]]
```

Приклад сортування масивів

Заданий масив	Сортування за рядками *(axis=1)*	Сортування за стовпцями *(axis=0)*
[[45 72 11 90]	[[11 45 72 90]	[[12 24 11 19]
[33 81 56 19]	[19 33 56 81]	[33 72 56 40]
[78 24 65 40]	[24 40 65 78]	[45 81 65 51]
[12 99 84 51]]	[12 51 84 99]]	[78 99 84 90]]

Отже, ми бачимо, що один і той самий масив може бути впорядкований по-різному залежно від того, що для нас важливіше: порядок у рядках чи порядок у стовпцях. Розуміння цих принципів – ключ до ефективної роботи з даними.

Практичне завдання

Завантажте файл-заготовку зі сторінки інтернет-підтримки. Виконайте запропоноване завдання.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Тема 3

Опрацювання текстових даних

У попередній темі ми розглядали алгоритми впорядкування числових даних і вчилися працювати з масивами. Але в реальному житті люди частіше мають справу не лише з числами, а й з текстом: повідомленнями у месенджерах, статтями в мережі, документами чи навіть кодом програм. Текст – також дані, і Python має безліч можливостей для їхнього аналізу й обробки.

Вам відомо, що для опрацювання текстових даних у Python існує тип даних **str** (від англ. *string* – рядок). Рядок можна уявити як список символів, де кожен символ має свій індекс. До кожного символу рядка можна звернутися за допомогою його індексу – порядкового номера, який вказує позицію символу в рядку. Нумерація починається з нуля, тобто перший символ має індекс 0, другий – 1 і так далі. Зворотна нумерація починається з -1 (останній символ), далі -2, -3 тощо.

Ви вже знаєте, що рядки в Python – це послідовності символів, до яких можна звертатися за індексом, виділяти фрагменти за допомогою зрізів та виконувати прості операції, як-от об'єднання чи порівняння.



Нумерація даних

Пряма	0	1	2	3	4	5
String	P	y	t	h	o	n
Зворотна	-6	-5	-4	-3	-2	-1

Вбудовані методи для роботи з текстом

Уявіть собі ситуацію: ви отримали довгий текст і хочете швидко підрахувати кількість символів або замінити певне слово на інше. Звісно, можна було б писати довгі алгоритми, проте Python значно полегшує завдання – він має вбудовані методи для роботи з текстом.

Перелік найпоширеніших методів

Метод	Опис
len(text)	Повертає довжину рядка
text.upper()	Перетворює всі літери на великі
text.lower()	Перетворює всі літери на маленькі
text.capitalize()	Робить першу літеру великою, решту – маленькими
text.replace(a, b)	Замінює підрядок *a* на *b*
text.split()	Розбиває рядок на список слів (стандартно – за пробілами)
text.strip()	Видаляє пробіли на початку та в кінці рядка
text.isalpha()	Перевіряє, чи складається рядок лише з літер
text.find(sub)	Повертає індекс першого входження підрядка sub (або -1, якщо немає)

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Ці методи дають змогу аналізувати, змінювати й форматувати рядки без додаткових складних конструкцій. Саме вони роблять обробку текстових даних швидкою та зручною. Нижче наведено найпоширеніші з них, які варто знати кожному, хто починає програмувати мовою Python.

Функція *len()*

Функція ***len()*** – це вбудована функція Python, яка працює не тільки з рядками, але й з іншими колекціями (списки, кортежі, словники). Для рядків вона рахує кожен символ, включаючи пробіли, коми, крапки тощо.

*Наприклад, у рядку «Привіт, світ!» 13 символів: 6 літер, кома, пробіл, 4 літери та знак оклику. Важливо пам'ятати, що **len()** не є методом рядка, а функцією, тому викликається без крапки.*



Програма:

```
text = "Привіт, світ!"
print(len(text))   # Виведе 13
```

Результат:

```
13
```

Метод *upper()*

Метод ***upper*** корисний для стандартизації введення користувача або порівняння рядків без урахування регістру. Наприклад, якщо потрібно перевірити, чи ввів користувач «Так» або «так», порівняйте ***text.upper()*** з «ТАК». Метод ігнорує цифри, символи або літери, які вже у верхньому регістрі. Він перетворює на верхній регістр лише ті символи, які мають відповідний нижній регістр (наприклад, літери латиниці або кирилиці).

Програма:

```
s = "Python"
print(s.upper())   # Виведе "PYTHON"
```

Результат:

```
PYTHON
```

Метод *lower()*

Метод ***lower()*** працює аналогічно до ***upper()***, але перетворює всі літери на нижній регістр. Він часто використовується на рівні з ***upper()*** для нормалізації даних перед порівнянням, забезпечуючи нечутливість до регістру. При цьому ***lower()*** не змінює цифр, символів чи літер, які вже перебувають у нижньому регістрі.



Програма:

```
s = "ПрИвІт"
print(s.lower())   # Виведе "привіт"
```

Результат:

```
привіт
```

Метод *capitalize()*

Метод ***capitalize()*** корисний для форматування імен, назв або речень, оскільки він перетворює лише першу літеру всього рядка на верхній регістр, а всі інші літери (включно з тими, що можуть бути у верхньому регістрі) – на нижній.



Програма:

```
s = "пРиВіТ СвІт"
print(s.capitalize())   # Виведе "Привіт світ"
```

Результат:

```
Привіт світ
```


Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Це ідеально підходить для забезпечення стандартного написання, коли ви, наприклад, отримуєте введене від користувача ім'я. Тоді ***capitalize()*** перетворить «оЛЕКСАНДР» на «Олександр». Як і всі методи рядків, ***capitalize()*** не змінює оригінального рядка, а повертає новий відформатований рядок.

Важливо: *якщо перший символ не є літерою (наприклад, цифра або пробіл), то решта рядка все одно буде переведена у нижній регістр. Наприклад, "**1привіт".capitalize()** поверне **«1привіт»**, а не **«1Привіт»**.*

Метод *replace()*

Метод ***replace()*** шукає всі входження вказаного підрядка A у вихідному рядку і замінює їх на підрядок B. Якщо підрядок A не знайдено, метод повертає копію оригінального рядка без змін. Він чутливий до регістру: наприклад, виклик "Світ".replace("світ", "друг") не виконає заміни, оскільки регістр літер не збігається. Для заміни лише першого входження можна використовувати необов'язковий третій аргумент: text.replace(A, B, 1). Як і всі методи рядків, ***replace()*** не змінює оригінальний рядок, а повертає новий рядок із внесеними змінами.



Програма:

```
s = "Я люблю Python. Python дуже цікавий."
print(s.replace("Python", "Пайтон"))
# Виведе: "Я люблю Пайтон. Пайтон дуже цікавий."
```

Результат:

```
Я люблю Пайтон. Пайтон дуже цікавий.
```

Метод *split()*

Метод ***split()*** за стандартним налаштуванням розбиває рядок за будь-якою кількістю пробілів (включно з табуляцією та переведенням рядка) і повертає список отриманих підрядків. Це корисно для парсингу неформатованого тексту. Якщо ж вказати розділювач (наприклад, кому), розбиття відбуватиметься тільки за цим символом, що ідеально підходить для парсингу структурованих даних, зокрема рядків у форматі CSV.



Парсинг (Parsing) *– це процес аналізу рядка символів (наприклад, тексту, коду, даних) для визначення його граматичної структури та перетворення його у структурований, зрозумілий формат.*

Програма:

```
s = "Привіт як справи"
print(s.split())  # Виведе ['Привіт', 'як', 'справи']
```

Результат:

```
['Привіт', 'як', 'справи']
```

Метод *strip()*

Метод ***strip()*** використовується для очищення рядків від небажаних символів. За стандартним налаштуванням він видаляє всі пробіли (включно з пробілами, табуляцією та переведенням рядка) як на початку, так і в кінці рядка, при цьому не впливаючи на пробіли всередині тексту. Це є незамінним для нормалізації введення користувача або даних, отриманих із файлів.



Існують також спеціалізовані методи:

- ***lstrip()*** видаляє символи тільки зліва (на початку рядка)
- ***rstrip()*** видаляє символи тільки справа (у кінці рядка)

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Для точнішого контролю можна вказати, які саме символи (наприклад, коми, дефіси або певні літери) потрібно видалити, передавши їх як аргумент методу.

Програма:

```
txt = "аааааааабанани....."
x = txt.strip(".а")
print(x)
```

Результат:

```
банани
```

Метод *isalpha()*

Метод ***isalpha()*** перевіряє, чи складається рядок виключно з алфавітних символів (літер у будь-якому регістрі). Він повертає True лише у тому разі, якщо всі символи є літерами. Метод поверне ***False*** для порожнього рядка або рядка, що містить цифри, пробіли, знаки пунктуації чи будь-які інші символи, відмінні від літер. Метод ***isalpha()*** не чутливий до мови, тобто коректно працює з літерами кирилиці, латиниці та інших алфавітів, роблячи його корисним для валідації введення, наприклад, для перевірки, чи введено тільки ім'я без чисел.

Програма:

```
print("Python".isalpha())    # True
print("Python3".isalpha())   # False
print("Привіт".isalpha())    # True
```

Результат:

```
True
False
True
```

Метод *find()*

Метод ***find()*** шукає перше входження вказаного підрядка в основному рядку й повертає його початковий індекс. Цей метод чутливий до регістру. Його ключова перевага полягає в обробці помилок: якщо підрядок не знайдено, ***find()*** повертає –1, а не викликає помилку, на відміну від методу ***index()***. ***find()*** корисний для пошуку позиції підрядка перед його виділенням або заміною. Для обмеження області пошуку можна додатково вказати початковий та кінцевий індекси як необов'язкові аргументи: ***text.find(sub, start, end)***.

Програма:

```
s = "Мені подобається Python"
a = ""
print(s.find("Python"))    # 17
print(s.find("Java"))      # -1
print(s.find("Мені",4,10))# -1
print(a.find("п"))         # -1
```

Результат:

```
17
-1
-1
-1
```



У цій темі ми побачили, що рядки в Python – це не просто набір символів, а потужний інструмент для роботи з текстом.

Не менш важливо пам’ятати, що текст у програмуванні – це такі ж дані, як і числа, і він потребує уважної та системної роботи. Опанувавши методи для роботи з рядками, ви зробили ще один крок до того, щоб писати зручніші й «розумніші» програми.

Практичне завдання

Завантажте файл-заготовку зі сторінки інтернет-підтримки. Виконайте запропоноване завдання.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Тема 4

Бібліотеки мови програмування

Python – це сучасна та зручна мова програмування, яка відома своєю простотою та зрозумілістю. Однак її справжня цінність криється в бібліотеках. Бібліотеки Python – це готові набори інструментів, які допомагають швидко й ефективно вирішувати різноманітні завдання.



Бібліотеки – *це спеціальні набори функцій і класів, які можна підключати до своєї програми, щоб розширити її можливості.*

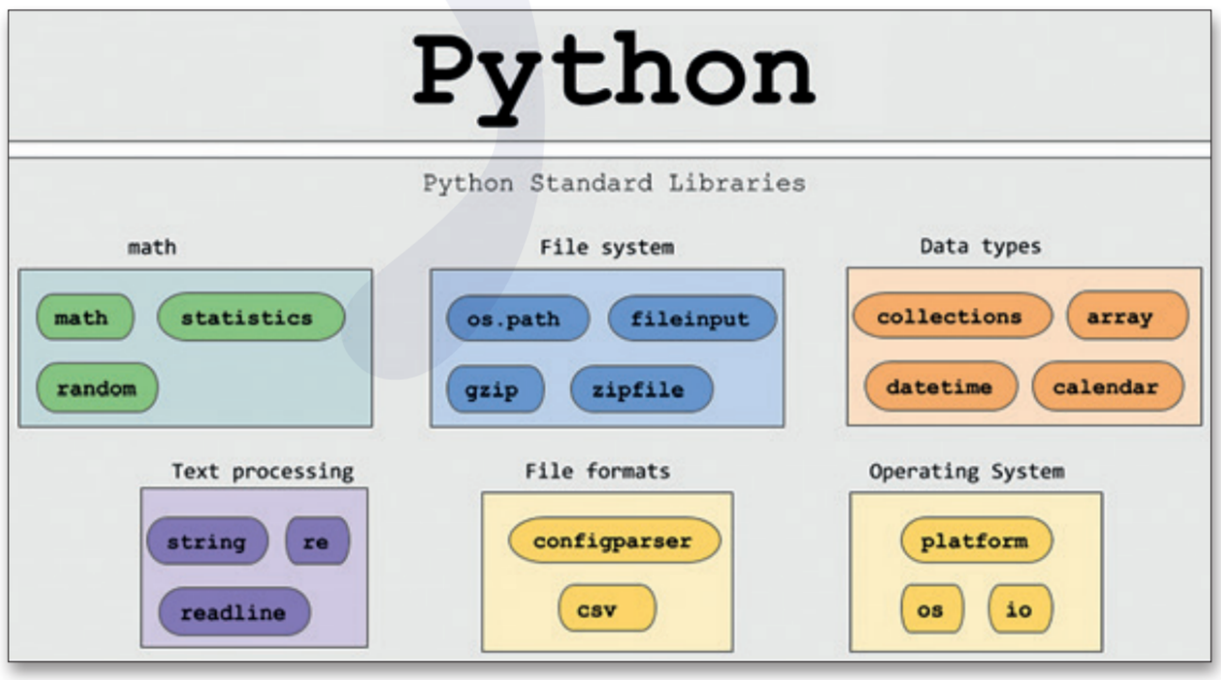


Завдяки бібліотекам можна легко створювати графіки, розробляти ігри, аналізувати дані або навіть автоматизувати роботу з вебсторінками. Вони економлять час і дають змогу зосередитися на творчій частині програмування, не витрачаючи сили на написання коду з нуля. У цьому розділі ви дізнаєтеся, як використовувати бібліотеки, щоб зробити свої програми потужнішими та цікавішими!

Бібліотеки Python

Стандартні бібліотеки	Зовнішні бібліотеки
Набір модулів, класів і функцій, які постачаються разом із самим дистрибутивом Python. Вони встановлюються автоматично, коли ви інсталюєте інтерпретатор Python, і не вимагають додаткового завантаження чи встановлення	Додаткові програмні модулі та пакети, які не входять до стандартної поставки Python. Їх потрібно встановлювати вручну за допомогою менеджера пакетів

Стандартні бібліотеки Python

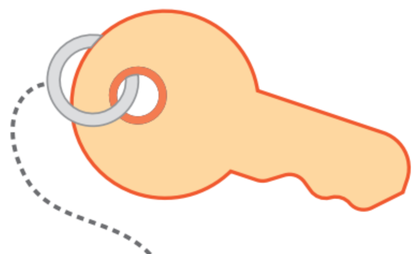


Стандартні бібліотеки Python – *це вбудований комплект інструментів та функцій, який є невід'ємною частиною мови і постачається разом із її інсталяційним пакетом.*

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Уявіть, що вам треба згенерувати випадкове число. Можна написати складний алгоритм самому, а можна просто під'єднати бібліотеку ***random*** й отримати результат в один рядок.

Стандартні бібліотеки можна уявити як комплект фірмових «батарейок», які ви отримуєте, купуючи пристрій (інтерпретатор Python). Вам не потрібно їх шукати, завантажувати чи встановлювати окремо.



Ключові особливості стандартних бібліотек

Доступність	Надійність	Широкий функціонал
Вони доступні одразу ж після встановлення Python. Щоб почати користуватися цими потужними інструментами, достатньо однієї команди **import** (наприклад, **import math**), і вони одразу готові до роботи.	Оскільки вони розробляються та підтримуються основними розробниками Python, вони є стабільними, надійними та сумісними з усіма версіями мови.	Бібліотеки охоплюють величезний спектр завдань, що дає змогу програмістам вирішувати більшість типових проблем без сторонніх інструментів.

Приклади популярних модулів стандартних бібліотек Python

Модуль	Основне призначення	Приклад використання
math	Виконання математичних функцій та використання констант	Обчислення sqrt(9), sin(x), використання π (число Пі)
random	Генерація випадкових чисел та випадковий вибір	Генерація випадкового числа в діапазоні (randint), перемішування списку (shuffle).
os	Взаємодія з операційною системою (файлова система, каталоги)	Створення папок (mkdir), перевірка існування файлу (path.exists)
sys	Доступ до параметрів і функцій інтерпретатора Python	Отримання аргументів командного рядка, вихід із програми (exit)
datetime	Робота з датою та часом	Отримання поточної дати, обчислення різниці між двома датами
json	Робота з даними у форматі JSON (кодування / декодування)	Збереження словників Python у файл або читання даних з веб-API
csv	Читання та запис файлів у форматі CSV (табличні дані)	Зручна обробка табличних даних із текстових файлів
re	Робота з регулярними виразами	Складний пошук, валідація та заміна шаблонів у тексті

Головна причина виникнення та постійного розширення стандартної бібліотеки Python полягає у філософії «Batteries included». Гвідо ван Россум та ранні розробники вважали, що користувачі не повинні витрачати час на пошук та встановлення базових утиліт для типових завдань. Ідея полягала в тому, що після встановлення мови користувач отримує комплексний набір інструментів, готовий до використання.



Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Щоб використовувати бібліотеки в Python, потрібно імпортувати їх за допомогою ключового слова **import**. Існує кілька способів підключення.



Імпорт усієї бібліотеки

import random # Підключаємо бібліотеку random

Після цього можна використовувати її функції через крапку: ***random.randint()***.

Програма:

```
import random  # Підключаємо бібліотеку random

a=random.randint(1,20) #випадкове числа від 1 до 20
print(a)
```

Результат:

```
8
```

Імпорт окремих функцій

from math import sqrt # Підключаємо тільки функцію sqrt з бібліотеки math

Тепер можна викликати ***sqrt()*** без вказівки назви бібліотеки.



Програма:

```
from math import sqrt  # Підключаємо тільки функцію sqrt з бібліотеки math

a=sqrt(25)       #повертає корінь квадратний
print(a)
```

Результат:

```
5.0
```

Використання псевдонімів (скорочень)

Це зручно, якщо назва бібліотеки довга.



Програма:

```
import numpy as np  # Тепер numpy можна викликати як np

a = np.array([1, 2, 3])
b = np.array([4, 5, 6])
print(a + b)        # [5 7 9]
print(a * b)        # [ 4 10 18
```

Результат:

```
[5 7 9]
[ 4 10 18]
```

Модуль *random*



Однією з найбільш важливих і корисних функцій у програмуванні є можливість запровадити елемент випадковості. Це потрібно не лише для розважальних цілей (наприклад, для створення ігор чи симуляцій), але й для серйозних завдань, як-от генерація ключів шифрування, випадкова вибірка даних у статистиці чи тестування алгоритмів. У Python робота з випадковістю реалізована через **стандартний модуль *random***. Цей модуль надає безліч функцій для генерування випадкових чисел у різних діапазонах, випадкового вибору елементів зі списку чи перемішування колекцій. Щоб скористатися цими інструментами, його необхідно імпортувати: ***import random***. Однак, важливо розуміти, як саме Python створює цю випадковість.





У Python випадковість реалізується за допомогою генератора псевдовипадкових чисел. Це означає, що числа «мають вигляд» випадкових, але насправді генеруються спеціальним алгоритмом. Для відтворюваності можна задати початкове значення генератора випадкових чисел за допомогою ***random.seed()***. Це корисно, коли потрібно, щоб програма завжди давала однаковий результат (наприклад, у тестуванні).

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Основні функції модуля *random*

Назва функції	Опис функції
random.randint(a, b)	ціле число в діапазоні [a, b]
random.randrange(start, stop, step)	число з діапазону з кроком (як range)
random.uniform(a, b)	випадкове дійсне число між a та b
random.random()	випадкове число від 0 до 1
random.choice(seq)	випадковий елемент зі списку
random.shuffle(list)	перемішує список
random.sample(seq, k)	випадковий піднабір із k елементів

*У технічному та науковому контексті «**random**» має точне значення і зазвичай стосується процесів, які неможливо передбачити або які імітують таку непередбачуваність.*



Приклад запису:

Програма:

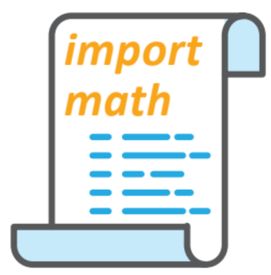
```
import random
random.seed(10)  # однаковий результат при кожному запуску
print("randint:", random.randint(1, 10))
print("randrange:", random.randrange(0, 100, 5))
print("uniform:", random.uniform(1.5, 9.5))
print("random:", random.random())
print("choice:", random.choice(["red", "green", "blue"]))
lst = [1, 2, 3, 4, 5]
random.shuffle(lst)
print("shuffle:", lst)
print("sample:", random.sample(lst, 3))
```

Результат:

```
randint: 10
randrange: 5
uniform: 4.931112437400917
random: 0.5780913011344704
choice: red
shuffle: [3, 1, 2, 5, 4]
sample: [3, 5, 1]
```

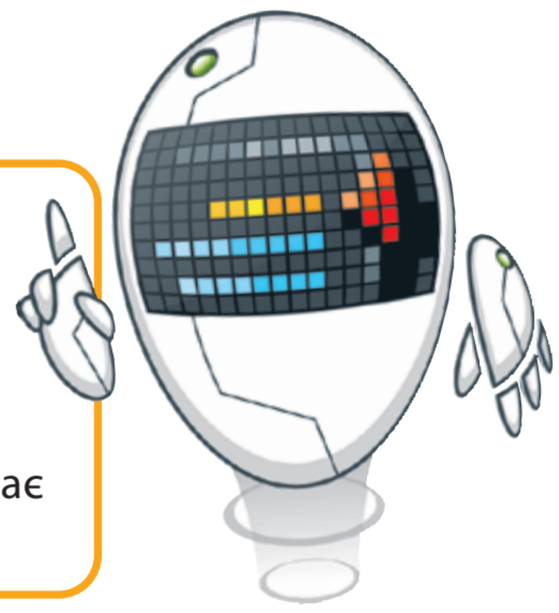
Модуль *math*

Ми вже ознайомилися з модулем ***random***, який допомагає працювати з випадковістю, генеруючи числа, перемішуючи списки чи вибираючи елементи. Проте в реальних задачах програмування лише випадкових чисел недостатньо. Дуже часто виникає потреба виконувати різноманітні **складні математичні обчислення** – від знаходження **квадратного кореня** та **логарифмів** до роботи з **тригонометричними функціями** *(sin, cos, tg)*. Для цього Python має ще одну потужну **стандартну бібліотеку** – **math**. Цей модуль містить численні математичні функції та константи, і є незамінним інструментом для наукових і технічних обчислень. Для початку роботи з ним необхідно лише імпортувати його: ***import math***.



ДОВІДКА

Будь-яка повноцінна мова програмування, особливо та, що претендує на використання в наукових обчисленнях, інженерії чи фінансах, повинна мати швидкий і точний доступ до базових математичних функцій. Вбудовувати ці функції безпосередньо в ядро мови було б неефективно та обтяжливо. Модуль math дав змогу відокремити складну математичну логіку від базових операцій Python, що зробило мову чистішою та допомагає користувачам викликати обчислення, коли вони потрібні.



Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Основні функції та константи модуля math

Тип функції	Функція / Константи	Призначення	Приклад
Константи	*math.pi*	Число π (Пі) ≈3.14159	*math.pi*
	math.e	Число e (число Ейлера) ≈2.71828	*math.e*
Корені та степені	*math.sqrt(x)*	Обчислює квадратний корінь числа *x*	*math.sqrt(16)* поверне 4.0
	math.pow(x, y)	Піднесення *x* до степеня *y* (x^y)	*math.pow(3, 2)* поверне 9.0
Логарифми	*math.log(x, base)*	Обчислює логарифм *x* за вказаною основою base	*math.log(100, 10)* поверне 2.0
	math.log10(x)	Обчислює десятковий логарифм (основа 10)	
Тригонометрія	*math.sin(x)*	Обчислює синус *x* (де *x* у радіанах)	*math.sin(math.pi / 6)*
	math.cos(x)	Обчислює косинус *x* (де *x* у радіанах)	
	math.radians(deg)	Перетворює градуси на радіани	*math.radians(90)*
Округлення	*math.ceil(x)*	Округлення до найближчого більшого цілого числа (вгору)	*math.ceil(7.1)* поверне 8
	math.floor(x)	Округлення до найближчого меншого цілого числа (вниз)	*math.floor(7.9)* поверне 7
Інші	*math.factorial(n)*	Обчислює факторіал числа *n*	*math.factorial(5)* поверне 120
	math.fabs(x)	Повертає абсолютне значення числа *x* (у форматі float)	*math.fabs(-10)* поверне 10.0
	math.gcd(x, y)	Визначає найбільший спільний дільник двох цілих чисел *x* та *y*	*math.gsd(6, 8)* поверне 2

Приклад їх використання:

Програма:

```
import math
print("pi:", math.pi)
print("sqrt(81):", math.sqrt(81))
print("pow(2, 10):", math.pow(2, 10))
print("factorial(5):", math.factorial(5))
print("gcd(18, 24):", math.gcd(18, 24))
print("sin(π/6):", math.sin(math.pi / 6))
print("log2(8):", math.log(8, 2))
print("ceil(3.14):", math.ceil(3.14))
print("floor(3.99):", math.floor(3.99))
```

Результат:

```
pi: 3.141592653589793
sqrt(81): 9.0
pow(2, 10): 1024.0
factorial(5): 120
gcd(18, 24): 6
sin(π/6): 0.49999999999999994
log2(8): 3.0
ceil(3.14): 4
floor(3.99): 3
```


Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Модуль *datetime*

Модуль ***math*** дає змогу розширити можливості звичайної арифметики і працювати з більш складними обчисленнями: тригонометрією, логарифмами, степенями. Проте життя програміста не обмежується лише числами. У багатьох завданнях потрібно оперувати ще й часом: дізнатися поточну дату, порахувати, скільки днів залишилося до події, або визначити різницю між двома моментами. Саме для цього в Python існує бібліотека ***datetime***, яка надає зручні інструменти для роботи з датами та часом.



Основні об'єкти та функції модуля datetime

Категорія	Об'єкт / Функція	Призначення	Ключові особливості
Основні об'єкти	*datetime.date*	Об'єкт, що містить лише календарну дату	Зберігає рік, місяць і день. Ідеально підходить, коли точний час не має значення
	datetime.time	Об'єкт, що містить лише час дня	Зберігає години, хвилини, секунди та мікросекунди. Використовується, коли потрібен час без прив'язки до конкретної дати
	datetime.datetime	Об'єкт, що поєднує дату і час (найбільш поширений)	Зберігає повну інформацію: рік, місяць, день, годину, хвилину, секунду. Використовується для позначення конкретного моменту часу
	datetime.timedelta	Об'єкт, що являє собою проміжок (інтервал) часу або різницю між двома моментами	Використовується для додавання або віднімання часу (наприклад, + 5 днів) і створюється функцією-конструктором
Функції створення	*datetime.now()*	Повертає поточну дату та час у вигляді об'єкта datetime	Отримує момент часу з урахуванням місцевої або вказаної часової зони
	datetime.today()	Повертає поточну дату та час, аналогічно datetime.now() без аргументів	Часто використовується як простіший синтаксис для отримання місцевого часу
	timedelta(days, hours, ...)	Функція-конструктор для створення проміжку часу (timedelta)	Дає змогу задати інтервал у днях, годинах, хвилинах, секундах. Приклад: timedelta(days=7)
Форматування	*strptime(string, format)*	Парсинг перетворює текстовий рядок у коректний об'єкт datetime	Вимагає точного формату (наприклад, '%Y-%m-%d'), щоб знати, як інтерпретувати рядок
	strftime(format)	Форматування перетворює об'єкт datetime у відформатований рядок	Дає змогу вивести дату у бажаному вигляді (наприклад, «15 Січня 2024» замість стандартного формату)

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Приклад їх використання:

Програма:

```
from datetime import datetime, timedelta
now = datetime.now()
print("Зараз:", now)
# Форматування
print(now.strftime("%d.%m.%Y %H:%M:%S"))
# Робота з timedelta
tomorrow = now + timedelta(days=1)
print("Завтра:", tomorrow.strftime("%d.%m.%Y"))
# Парсинг дати з рядка
date_str = "20.08.2025 12:45"
parsed = datetime.strptime(date_str, "%d.%m.%Y %H:%M")
print("Перетворене:", parsed)
```

Результат:

```
Зараз: 2025-09-21 17:39:48.922220
21.09.2025 17:39:48
Завтра: 22.09.2025
Перетворене: 2025-08-20 12:45:00
```

Зовнішні бібліотеки Python

Python має багато вбудованих модулів, наприклад math, random чи datetime. Вони допомагають розв'язувати найпоширеніші завдання. Але іноді цього може бути замало, особливо коли потрібно працювати з інтернетом, зображеннями чи створювати складні математичні обчислення. У такому разі програмісти використовують зовнішні бібліотеки, які створені іншими розробниками.



Ключові особливості та призначення зовнішніх бібліотек

Розширення функціоналу	Надійність	Екосистема
Зовнішні бібліотеки створюються спільнотою розробників для вирішення специфічних і складних завдань, які не вирішуються стандартними можливостями Python.	На відміну від стандартних бібліотек, зовнішні пакети потрібно встановлювати вручну, зазвичай за допомогою менеджера пакетів ***pip*** (наприклад, ***pip install numpy***).	Саме зовнішні бібліотеки зробили Python лідером у багатьох галузях (наука, аналіз даних, веброзробка).

Приклади популярних зовнішніх бібліотек Python

Бібліотека	Основне призначення	Приклад використання
NumPy	Числові обчислення та багатовимірні масиви	Наукові обчислення, основи машинного навчання
Pandas	Аналіз та маніпуляція даними (робота з таблицями / DataFrame)	Обробка CSV-файлів, статистика
Matplotlib	Створення графіків та візуалізація даних	Побудова діаграм та гістограм
Requests	Надсилання HTTP-запитів	Взаємодія з веб-API, завантаження даних із сайтів
Django / Flask	Веброзробка	Створення вебсайтів та сервісів

Зовнішні бібліотеки можна знайти у спеціальному онлайн-сховищі Python Package Index (PyPI), де зібрано тисячі корисних модулів.



Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Щоб встановити зовнішні бібліотеки, використовується програма ***pip***. Вона є менеджером пакетів Python і дає змогу встановлювати, оновлювати чи видаляти зовнішні модулі. Для цього потрібно відкрити командний рядок.



Щоб відкрити командний рядок у Windows, *натисніть на клавіатурі одночасно клавіші* **Win** + **R** *і введіть команду* ***cmd***.
На комп'ютерах з macOS чи Linux *зазвичай використовується застосунок* ***Terminal***.



Після цього достатньо ввести команду ***pip install*** і назву потрібного модуля. Наприклад, ***pip install requests*** встановлює модуль для мережевих запитів, а ***pip install numpy*** – бібліотеку для математичних обчислень. Якщо все виконано правильно, то у програмі можна одразу імпортувати цей модуль, як-от:



Програма:

```
import requests
print(requests.__version__)
```

Результат:

```
2.32.5
```

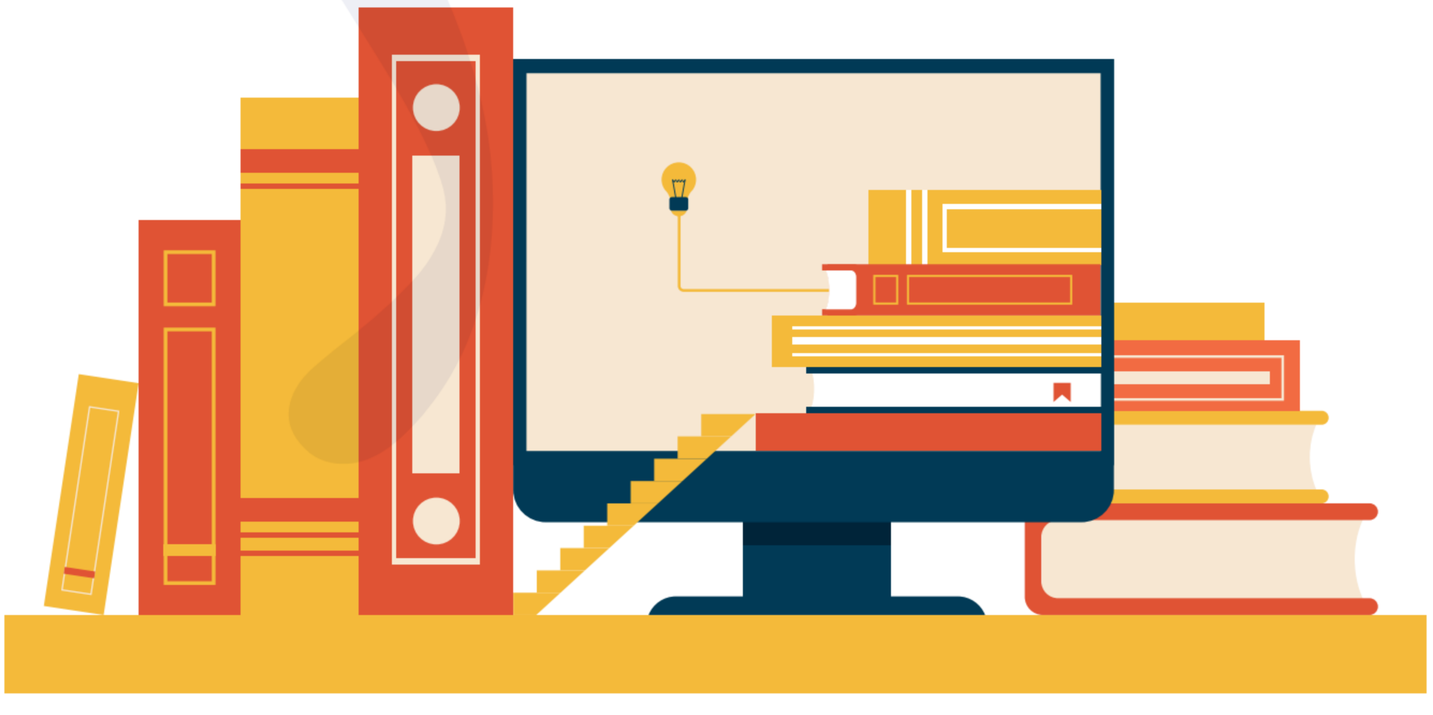
У цьому прикладі ми не тільки підключили модуль, а й перевірили його версію.

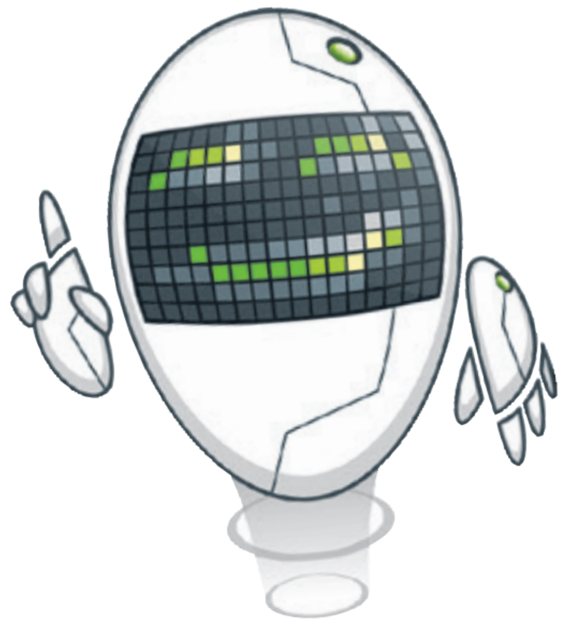
Хто створює зовнішні бібліотеки?

Зовнішні бібліотеки (або сторонні пакети) створюються і підтримуються спільнотою розробників, а не безпосередньо Python Software Foundation.

Розробники зовнішніх бібліотек

Приватні розробники та ентузіасти	Академічні кола	Комерційні компанії
Створюють інструменти для вирішення власних проблем, які потім відкривають для загального доступу.	Часто розробляють високопродуктивні бібліотеки (як NumPy та SciPy) для математичних та наукових обчислень, де важлива швидкість і точність.	Створюють фреймворки та інструменти (наприклад, Django для веброзробки або інструменти Google / Facebook для машинного навчання), щоб прискорити власні проєкти та сформувати екосистему навколо своїх технологій.





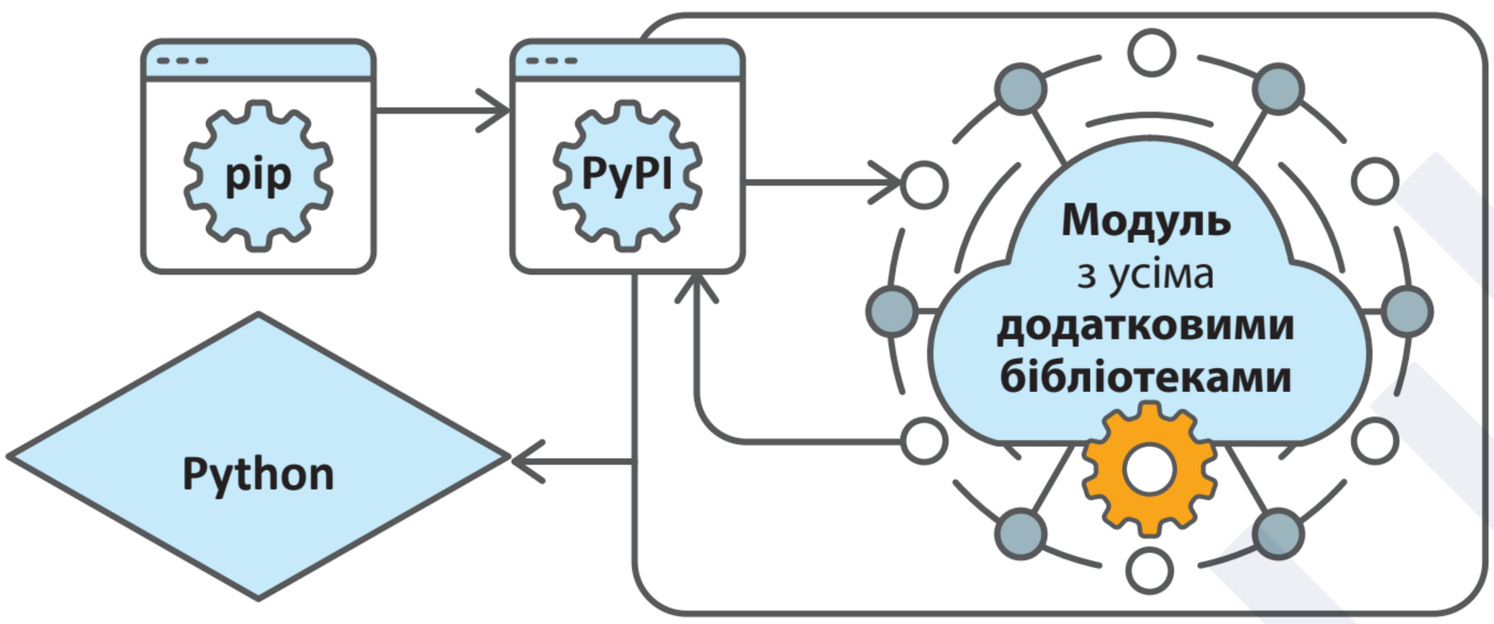
Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Принцип роботи pip



Принцип роботи **pip** дуже простий: спочатку програма звертається до сховища **PyPI** та шукає потрібний **модуль**. Потім вона завантажує його разом з усіма додатковими **бібліотеками**, від яких він залежить. І нарешті встановлює ці **пакети** в середовище Python на вашому комп'ютері. Після цього модуль можна використовувати у будь-якому вашому проєкті.

Принцип роботи pip





Текстовий файл *requirements.txt*

Коли у програмному продукті використовується багато різних бібліотек, зручно створювати спеціальний текстовий файл із назвою ***requirements.txt***. У ньому записують усі модулі, які потрібні для роботи програми, а також їхні версії. Це дає змогу швидко встановити всі залежності на новому комп'ютері за допомогою лише однієї команди: ***pip install -r requirements.txt***. Крім того, завдяки цьому інші розробники легко зможуть відтворити ваше середовище і працювати над проєктом разом із вами.



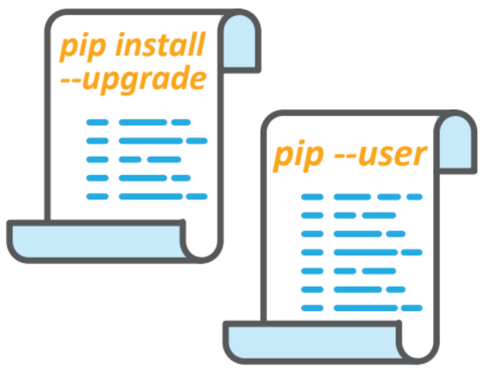
Керування оновленнями модулів

Іноді виникає потреба оновити вже встановлений модуль до новішої версії. Для цього використовується команда ***pip install --upgrade*** із вказаною назвою бібліотеки. Якщо ж певний модуль більше не потрібен, його можна видалити командою ***pip uninstall***. Щоб перевірити, які саме модулі вже встановлені у вашій системі, зручно скористатися командою ***pip list***.



Помилки у програмі *pip*

Як і будь-яка програма, ***pip*** може видавати помилки. Наприклад, якщо команда pip не розпізнається, це означає, що вона не встановлена або не додана до системних шляхів. У такому разі треба перевірити налаштування Python. Якщо система повідомляє про відсутність прав, можна додати параметр ***--user*** або запустити командний рядок від імені адміністратора. А коли під час імпорту з'являється повідомлення ***ModuleNotFoundError***, слід упевнитися, що модуль встановлено саме в те середовище Python, з яким ви працюєте.



Практичне завдання

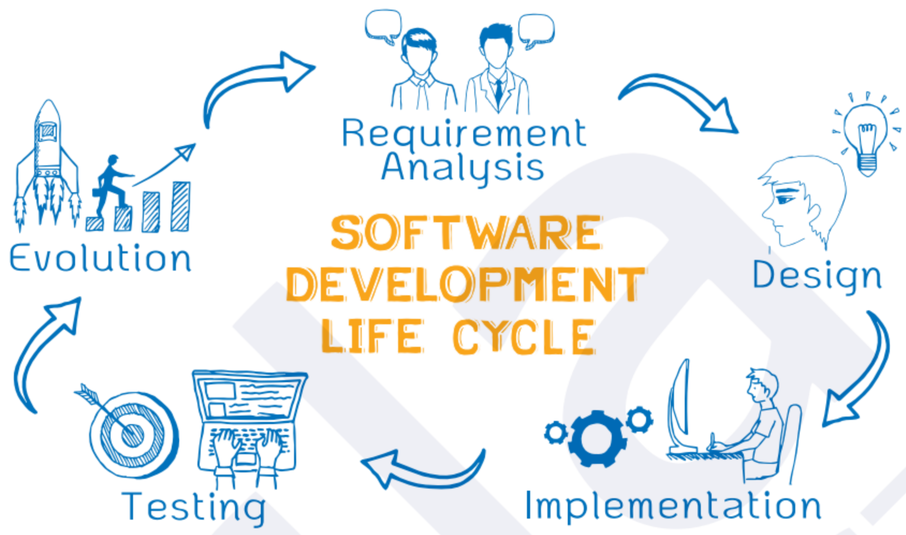
Завантажте файл-заготовку зі сторінки інтернет-підтримки. Виконайте запропоноване завдання.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Тема 5
Технологія створення програмних продуктів

У сучасному світі програмування використовується не лише для створення невеликих програм, а й для розробки складних програмних продуктів. Такі продукти допомагають людям у повсякденному житті, забезпечують роботу бізнесу, освіти, науки й навіть розваг.

Наприклад, можна створити короткий скрипт на Python, який обчислює середнє арифметичне чисел або виводить повідомлення користувачеві.



Програма (Computer program) *– це набір інструкцій, написаних мовою програмування, які комп'ютер може виконати для досягнення певної мети або вирішення конкретної задачі.*

Програмний продукт (Software product) *– це комплексне, цілісне рішення, створене для задоволення потреб користувача або бізнесу.*

Програмний продукт складається з багатьох взаємопов'язаних файлів, модулів і бібліотек, об'єднаних у чітку структуру. На відміну від простого скрипту, програмний продукт має повний життєвий цикл: він не лише створюється, а й підтримується, вдосконалюється (через оновлення та виправлення помилок) і розвивається протягом багатьох років. Його розробкою, як правило, займається ціла команда спеціалістів. До яскравих прикладів програмних продуктів належать операційні системи, корпоративні системи керування, складні вебсайти, а також мобільні додатки та комп'ютерні ігри.

Кожен розробник повинен швидко орієнтуватися у проєкті та мати можливість доповнювати його, не ламаючи наявних функцій.



Для забезпечення цього потрібні такі ключові елементи:

Чітка архітектура та модульність

Код має бути розділений на невеликі, незалежні модулі з чітко визначеними інтерфейсами (контрактами). Це мінімізує залежності та дає змогу кільком розробникам працювати над різними частинами одночасно, не створюючи конфліктів.

Стандарти кодування та документація

Усі члени команди мають дотримуватися єдиного стилю кодування (наприклад, PEP 8 для Python) та регулярно створювати вичерпну документацію. Це гарантує, що новий розробник може швидко зрозуміти, як працює будь-який компонент.

Автоматизоване тестування

Жодна зміна не повинна потрапляти у фінальний продукт без попереднього проходження набору автоматичних тестів (юніт-тести, інтеграційні тести). Тестування слугує «страховкою», яка гарантує, що доповнення або виправлення в одній частині коду не призвели до регресії (помилок) в іншій.

Системи контролю версій

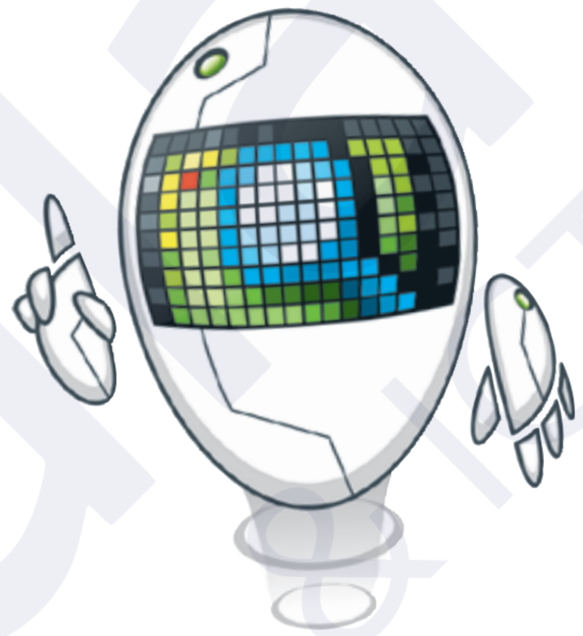
Використання систем, як-от Git, є обов'язковим. Вони дають змогу команді співпрацювати безпечно, відстежувати всі зміни, повертатися до попередніх робочих версій та ефективно «зливати» код від різних авторів.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Чому важливо структурувати код?

Уявімо, що ви написали програму з тисячі рядків. Якщо зберігати весь код в одному файлі, то знайти помилку буде надзвичайно складно, додавати нові функції – незручно, а працювати в команді – майже неможливо. Якщо ж розподілити код на окремі модулі, створити папки для тестів і документації, то програма стає зрозумілою і легкою для підтримки.

Модуль (Module) у Python – *це файл, який містить визначення Python (функції, класи, змінні) та виконуваний код.*



Модулі

У попередніх темах ви вже дізналися, що таке бібліотеки Python. Вони дають змогу використовувати готові функції для обчислень, роботи з файлами чи створення графіки. Але для того, щоб скористатися бібліотекою або власним модулем, потрібно виконати імпорт.

*У Python будь-який файл з розширенням **.py** є модулем. Використання модулів уможливлює розділення великої програми на менші частини. Наприклад, у модулі **greetings.py** може зберігатися функція для вітання користувача, а в **main.py** ця функція викликається. Так головний файл стає зрозумілим і не перевантаженим деталями.*

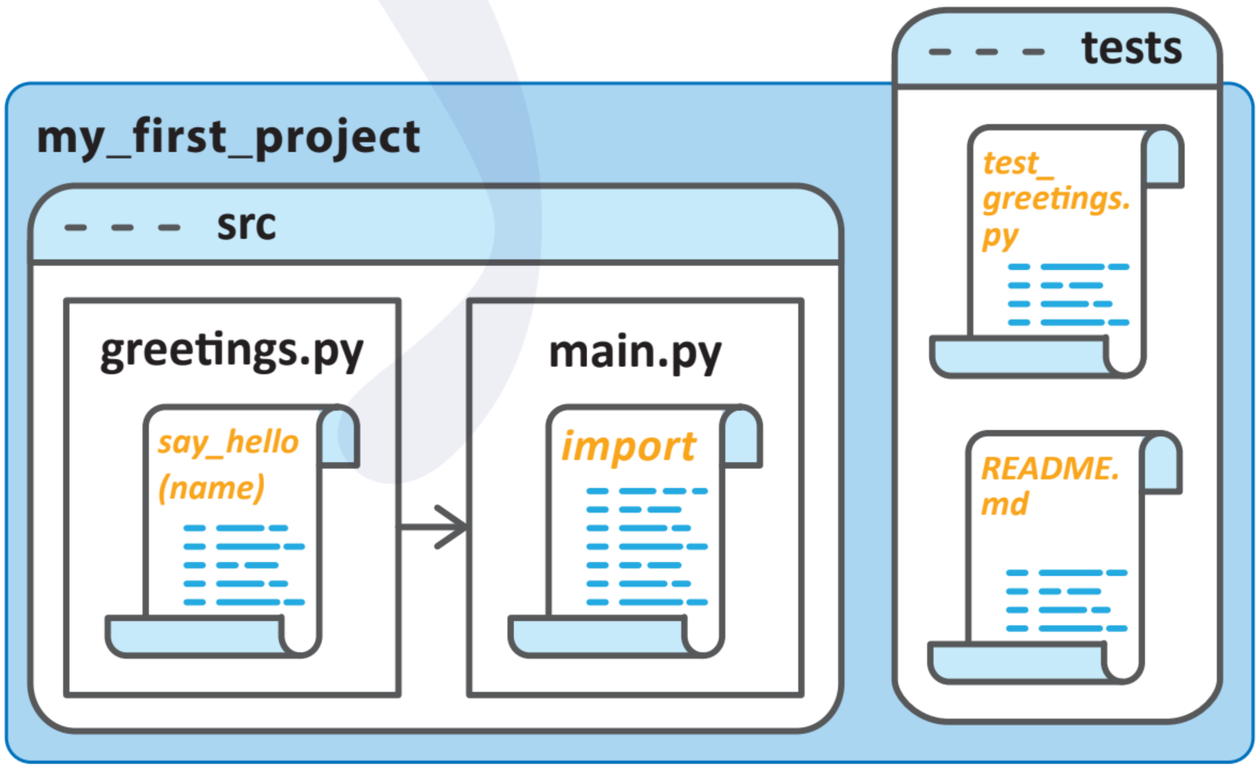
Способи імпорту модуля

import	Допомагає підключати модулі. Можна імпортувати весь модуль або окрему функцію з нього
import math	Підключає всю бібліотеку математичних функцій
from math import sqrt	Дає змогу використати лише одну функцію для обчислення квадратного кореня



Так само можна створювати й власні модулі, які легко повторно використовувати в різних проєктах. Це робить код більш універсальним, а роботу над проєктами – ефективною.

Уявімо, що ми створюємо невеликий проєкт під назвою ***my_first_project***. У папці ***src*** розмістимо два файли: ***main.py*** та ***greetings.py***. У модулі ***greetings.py*** буде функція ***say_hello(name)***, яка виводить вітання. А в головному файлі ***main.py*** ми імпортуємо цю функцію та викликаємо її. Для перевірки роботи програми створимо у папці ***tests*** файл ***test_greetings.py***. У файлі ***README.md*** поясним, як запускати програму.



Такий простий приклад показує, що навіть невеликий проєкт можна і варто структурувати. Це допоможе вам у майбутньому створювати більш складні продукти, використовуючи бібліотеки, які ви вже знаєте, та додаючи власні модулі.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Модулі *– це логічно завершені частини програми, представлені у вигляді файлів із кодом.*

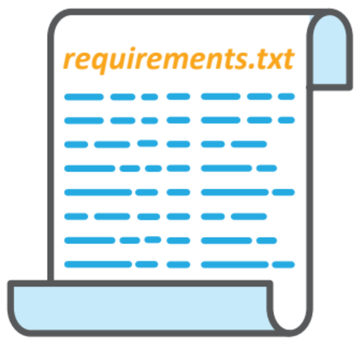
Модулі повинні:

- Виконувати одну конкретну задачу (наприклад, робота з файлами, обчислення, інтерфейс користувача)
- Бути незалежними (модуль повинен працювати сам по собі, без залежності від інших модулів, якщо це можливо)
- Мати зрозумілі назви (наприклад, ***file_utils.py*** для роботи з файлами, ***math_operations.py*** для обчислень)

Приклад правильної та неправильної організації модулів

Неправильно	Правильно
Один файл ***all_functions.py*** з 50 функціями для всього: математики, роботи з файлами, візуалізації	Розбиття на модулі: ***math_operations.py***, ***file_utils.py***, ***visualization.py***
Назва файлу ***module1.py*** – незрозуміло, для чого він	Назва файлу ***data_processing.py*** – одразу зрозуміло, мету створеного модулю.

Часто програмний продукт складається з багатьох зовнішніх бібліотек. Щоб зручно зберігати інформацію про них, у Python використовують спеціальний файл ***requirements.txt***. У ньому записують усі залежності проєкту, тобто бібліотеки, без яких він не зможе працювати. Завдяки цьому файл має кілька важливих переваг: новий учасник або учасниця команди може швидко встановити всі потрібні модулі, на різних комп'ютерах можна уникнути конфліктів версій, а також легко поділитися проєктом з іншими розробниками.



Приклад такого файлу:

```
requests==2.31.0
flask==2.3.2
pillow==10.0.0
```

Тут зазначено не лише назви бібліотек, а й їхні версії. Це допомагає зробити роботу більш передбачуваною: усі члени команди матимуть однакове середовище. ***requirements.txt*** значно полегшує процес оновлення середовища розробки, даючи змогу швидко відновити всі необхідні пакети на новій машині чи віртуальному середовищі. Його використання сприяє кращій організації роботи, підвищуючи ефективність та якість кінцевого продукту.

Щоб створити файл *requirements.txt*:

- використайте команду ***pip freeze > requirements.txt***.

Щоб встановити бібліотеки з файлу *requirements.txt*:

- виконайте команду ***pip install -r requirements.txt***.

ПІДКАЗКА

Важливо знати: уникайте імпорту з використанням * (наприклад, ***from math_operations import ****), тому що це може призвести до конфліктів імен функцій. Якщо ви імпортуєте функцію з псевдонімом, використовуйте зрозумілі скорочення (наприклад, ***np*** для ***numpy***, ***pd*** для ***pandas***).

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Віртуальне середовище

Ще одне важливе поняття – віртуальне середовище. Воно дає змогу створити для кожного проєкту власний простір із потрібними бібліотеками. Це дуже зручно, коли ви працюєте над кількома програмами одночасно і кожна з них потребує різних версій модулів. Активувати віртуальне середовище можна по-різному залежно від операційної системи.

Віртуальне середовище (Virtual environment) *у Python – це ізольоване робоче середовище, яке дає змогу створювати незалежні інсталяції та керувати пакетами для кожного окремого проєкту.*

Щоб активувати віртуальне середовище на *Windows*:

- використайте команду ***venv\Scripts\activate***.

Щоб активувати віртуальне середовище на *macOS* чи *Linux*:

- використайте команду ***source venv/bin/activate***.



Пакет (Package) у Python *– це спосіб організації пов'язаних модулів у єдиний каталог (папку) та ієрархію просторів імен.*

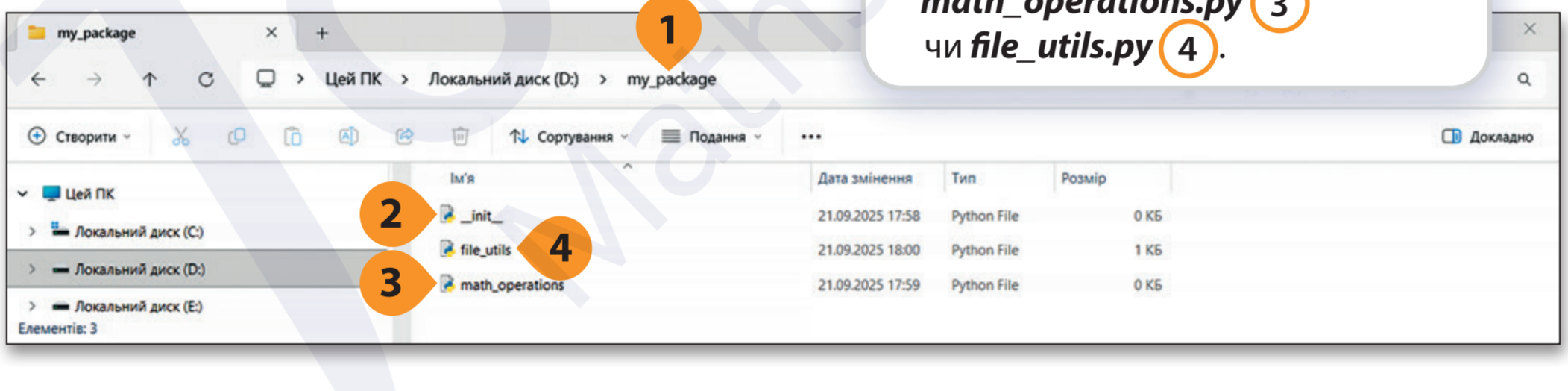
Створення власних пакетів

З часом ви можете створювати настільки великі проєкти, що виникає потреба організовувати власні пакети. Пакет у Python – це набір модулів, об'єднаних у спільну структуру. Наприклад, у пакеті **my_package** (1) можна зібрати всі модулі для роботи з графікою чи математичними операціями. Це дозволяє підтримувати порядок у проєкті й робить імпорт модулів більш зручним.

Щоб створити пакет:

- потрібно створити папку з його назвою і додати в неї спеціальний файл ***__init__.py*** (2);
- цей файл може бути порожнім, але він сигналізує Python, що папка є пакетом;
- далі в цю папку можна додати модулі, наприклад ***math_operations.py*** (3) чи ***file_utils.py*** (4).

Тоді структура буде такою:



Тепер у головному файлі можна імпортувати потрібну функцію безпосередньо з пакета:

```
from my_package.math_operations import add # імпортує функцію add (сума двох чисел)

print(add(2,5))
```

```
7
```

Отже, використання модулів і пакетів допомагає структурувати програмний продукт, а робота з ***requirements.txt*** і віртуальними середовищами робить проєкт більш надійним і зручним для співпраці.

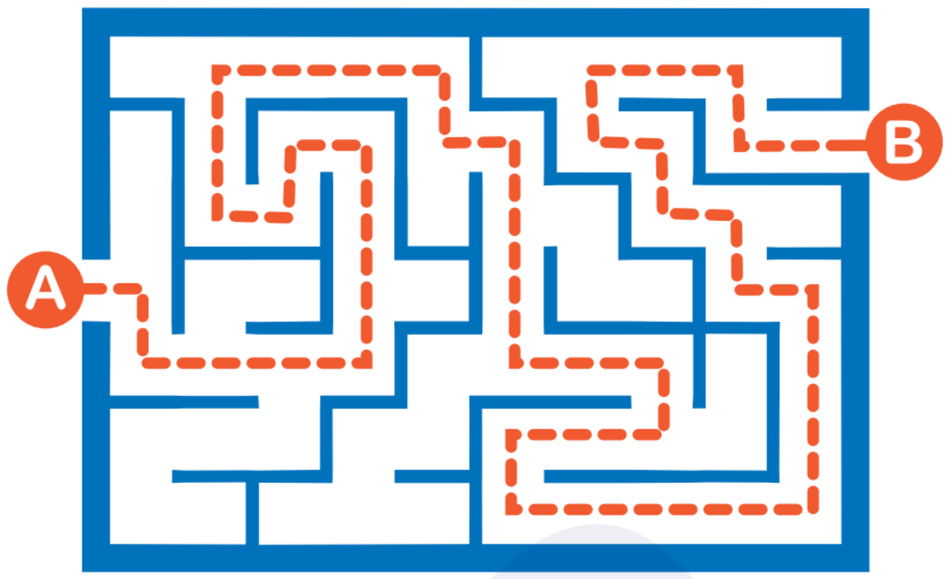
ПІЗКАЗКА

Головний файл і **my_package** мають бути в одній папці.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Планування

Уявіть, що ви вирушаєте в подорож. Щоб дістатися до пункту призначення, потрібно знати маршрут, підготувати карту або завантажити навігатор. Якщо їх не буде, то легко збитися зі шляху. Точно так само і з програмуванням: перш ніж написати перший рядок коду, варто скласти план роботи.



Планування в програмуванні *– це критично важливий початковий етап розробки програмного продукту, під час якого визначаються мета, вимоги, архітектура та послідовність дій для створення, тестування та впровадження програми.*

Планування в програмуванні – *це як складання карти подорожі. Воно допомагає зрозуміти, куди ви йдете, які кроки потрібно зробити і які труднощі можуть виникнути.*



Планування перетворює абстрактну ідею на **конкретний, структурований план дій**.

Завдяки цьому команда розробників:

1. **Розуміє, ЩО** має бути створено (вимоги)
2. **Розуміє, ЯК** це буде працювати (архітектура та дизайн)
3. **Знає, КОЛИ** очікувати результати (графік та терміни)

Основні етапи планування

1. Визначення мети

- Поставте собі запитання: «Що саме має робити моя програма?»
- **Приклад:** «Моя програма повинна вітати користувача його іменем».



2. Розбиття на маленькі завдання

- Великі проєкти здаються складними. Але якщо поділити їх на невеликі кроки, виконувати стає значно легше.
- **Приклад:**
  - створити функцію ***say_hello***;
  - написати тест для цієї функції;
  - додати головний файл ***main.py***.



3. Визначення пріоритетів

- Важливо розуміти, які завдання треба зробити першими, а які можуть зачекати.
- **Наприклад**, неможливо написати тест до функції, якщо сама функція ще не створена.



4. Оцінка часу

- Спробуйте передбачити, скільки часу знадобиться для виконання кожного завдання.
- Це допомагає організувати роботу та уникнути стресу перед дедлайном.



Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Приклад плану для простого проєкту

1. Створити папки ***src, tests, docs.***
2. Написати функцію ***say_hello*** у файлі ***greetings.py***.
3. Створити файл ***main.py*** та перевірити роботу функції.
4. Написати тест для функції.
5. Задокументувати проєкт.

Запам'ятайте: *навіть найпростіший план допомагає уникнути хаосу в роботі.*



Матриця пріоритетності

Не всі завдання мають однакову важливість. Деякі – ключові для проєкту, інші – допоміжні. Якщо витратити багато часу на дрібниці, можна не встигнути зробити головне.

Щоб правильно розподілити сили, програмісти користуються методами пріоритезації. Один з найпопулярніших інструментів – матриця пріоритетності.

Ця матриця допомагає «розкласти по поличках» усі справи та зосередитися на головному.

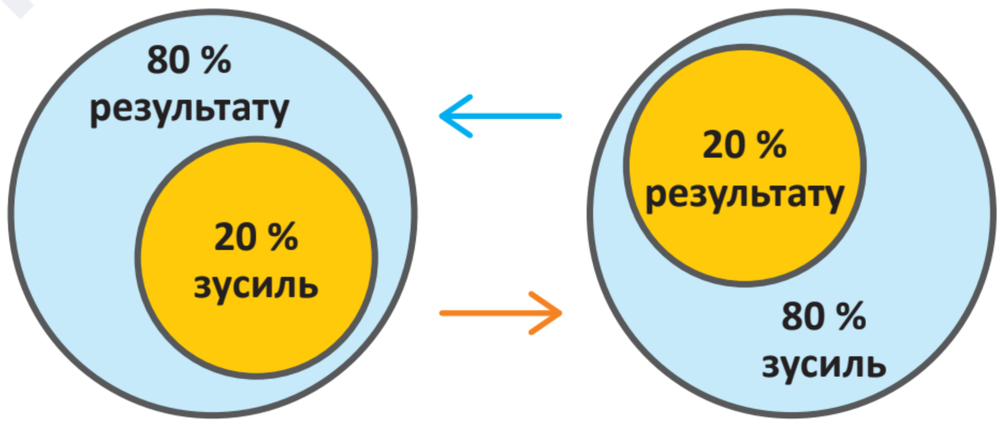
Матриця пріоритетності

Важливість	Терміновість	Дії	Приклад
Висока	Терміново	Робити відразу!	Написати головну функцію програми
Висока	Не терміново	Запланувати на найближчий час	Написання документації
Низька	Терміново	Делегувати або відкласти	Дрібні правки в коді
Низька	Не терміново	Можна не робити	Додаткові прикраси інтерфейсу

Принцип Парето (правило 80/20)

Ще один важливий інструмент – **правило Парето**. Воно стверджує:

- 20 % зусиль приносять 80 % результату.
- І навпаки: 80 % зусиль можуть дати лише 20 % результату.



Уявімо, що команда працює над шкільним застосунком для розкладу уроків. Завжди починайте з найважливішого. Дрібні завдання можна виконати пізніше або навіть виключити з плану.

- **Головне завдання** – щоб учні й учениці бачили розклад на завтра.
- **Додаткове завдання** – зробити зміну кольорів у застосунку.

Якщо команда витратить тиждень на дизайн, але забуде реалізувати показ розкладу – програма буде гарною на вигляд, але зовсім не корисною.

Наприклад:

- Написання основної функції ***say_hello*** – це лише невелика частина роботи (20 %), але без неї програма взагалі не працюватиме. Це і є ті самі «80 % результату».
- А от додавання нових кольорів в інтерфейсі чи декоративних елементів може зайняти багато часу (80 % зусиль), але практичної користі буде менше.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Розробка програмного продукту

Життєвий цикл програмного продукту

Хоча конкретні методології можуть відрізнятися, зазвичай процес розробки має основні етапи.

Основні етапи розробки програмного продукту

1. Планування та аналіз вимог

Визначення цілей, збір та документування функціональних і нефункціональних вимог продукту, а також оцінка ресурсів і термінів.

2. Проєктування та дизайн

Створення архітектури системи (структури модулів, баз даних, інтерфейсів) та розробка користувацького інтерфейсу (UI/UX).

3. Реалізація

Власне написання коду згідно з проєктними рішеннями, з використанням обраних мов програмування та технологій.

4. Тестування

Систематична перевірка коду для виявлення та виправлення помилок (багів). Сюди входять юніт-тести, інтеграційні та системні тести.

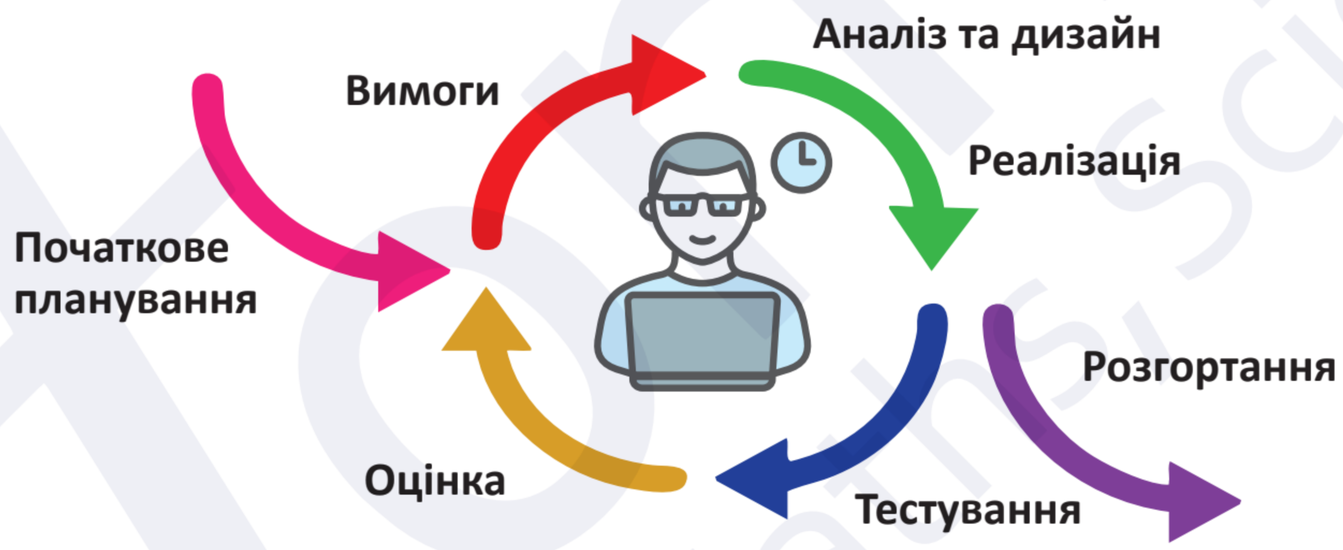
5. Впровадження

Розгортання продукту для кінцевих користувачів, запуск його у виробниче середовище.

6. Супровід та підтримка

Постійне оновлення, виправлення виявлених після релізу помилок і додавання нового функціоналу для підтримки актуальності продукту.

Отже, розробка програмного продукту – це безперервний, циклічний процес, який вимагає командної роботи, дисципліни та постійного вдосконалення.



Коли код написаний, обов'язково відбувається **тестування**. Це перевірка, чи правильно працюють усі частини програми, чи немає помилок. Наступний етап – **документування**: пояснення, як користуватися програмою, які функції вона має та як її можна розвивати далі. І, нарешті, **підтримка**: удосконалення програми, виправлення недоліків, додавання нових можливостей. Усе це разом утворює **життєвий цикл програмного продукту**.

Розробка програмного продукту (Software development) *– це систематичний процес створення, тестування та супроводу програмного забезпечення.*



ІСТОРІЯ

У 1896 році Парето вивчав розподіл багатства в Італії. Він виявив, що 80 % усієї землі та багатства країни належить лише 20 % населення. Він провів подібні дослідження в інших країнах і виявив, що ця нерівномірна закономірність (де велика частина ресурсів контролюється невеликою часткою людей) є сталою. Тому принцип виник як економічне спостереження Вільфредо Парето, але був систематизований і популяризований Джозефом Джураном як універсальний інструмент для підвищення ефективності та управління якістю.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Життєвий цикл програмного продукту

- **Тестування**
- **Документування**
- **Підтримка**

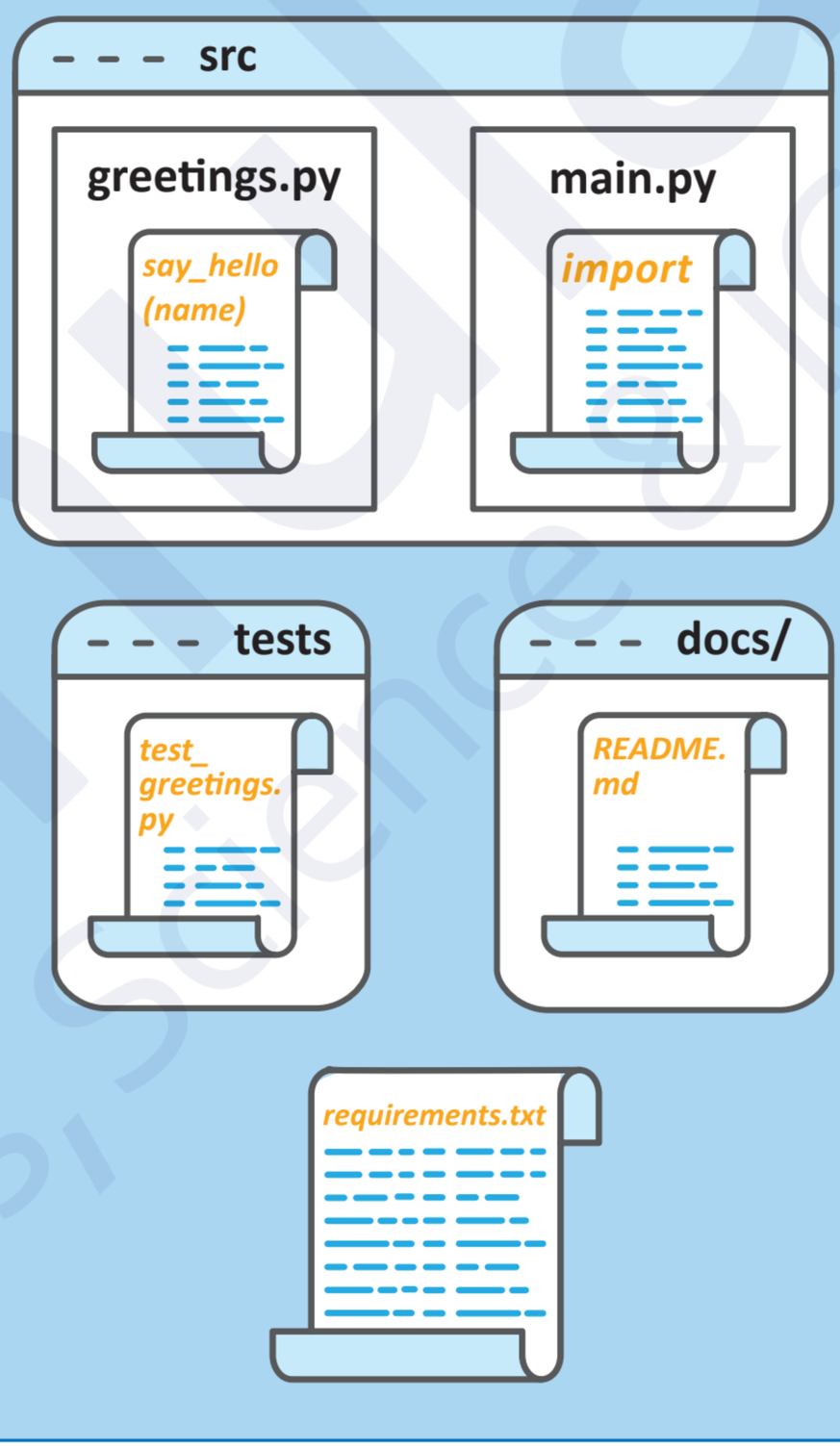
Структура програми

Щоб програмний продукт був зручним у роботі, його зазвичай організовують у вигляді папок і файлів. Навіть невеликі програми рекомендується розділяти на кілька частин.

Типова структура програми може мати такий вигляд:

- папка з вихідним кодом (наприклад, ***src/***), де зберігається головний файл програми ***main.py*** та інші модулі, як-от ***greetings.py***;
- папка з тестами (***tests/***), у якій містяться файли для перевірки роботи коду;
- папка з документацією (***docs/***), де розташований файл ***README.md*** з описом проєкту;
- файл ***requirements.txt***, у якому записано список бібліотек, необхідних для роботи програми.

Типова структура програми



Завдяки такій структурі код легко зрозуміти не лише автору, а й будь-кому, хто долучиться до проєкту в майбутньому.

Отже, технологія створення програмних продуктів поєднує точність алгоритмів, гнучкість структур даних та дисципліну організації коду, забезпечуючи створення масштабованих і підтримуваних рішень. Успіх проєкту визначається не лише робочим кодом, а й його організованістю, надійністю та здатністю до масштабування.

Існує правило, що ілюструє критичну важливість технології створення програмних продуктів та раннього тестування: що раніше ви знайдете помилку, то дешевше й легше її виправити. Це стратегія мінімізації фінансових та репутаційних ризиків проєкту.

Практичне завдання

Завантажте файл-заготовку зі сторінки інтернет-підтримки. Виконайте запропоноване завдання.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Тема 6
Документація програмного проєкту

Створення програмного продукту можна порівняти з будівництвом будинку. Спочатку потрібно скласти план, підготувати креслення, звести фундамент – і лише потім можна облаштовувати кімнати. Так само і в програмуванні: написання коду є лише частиною великої справи. Програма проходить кілька етапів – від визначення потреб і написання документації до тестування та підтримки.



У житті траплялися ситуації, коли доводилося встановлювати програму чи гру, але було невідомо, як нею користуватися без підказки. Це є яскравим прикладом того, коли продукт був створений без достатньо зрозумілої інструкції.

Документація програмного проєкту *– це набір текстових та графічних матеріалів, які описують мету, структуру, функціональність та правила використання програмного забезпечення.*

У мові Python існують правила, які допомагають організувати програмний код, створювати пакети та працювати з бібліотеками. Але цього недостатньо. Важливо навчитися створювати документацію: інструкції, технічні завдання, схеми, чеклісти. Саме вони забезпечують зрозумілість і зручність програмного продукту.

Документація *– це мапа, яка пояснює користувачеві, куди рухатися і що робити. Вона відповідає на прості, але важливі питання: що це за програма, як її встановити, які можливості вона має і що робити, якщо виникнуть труднощі.*



Цільові групи документації

Технічна (внутрішня) документація	Користувацька (зовнішня) документація
Технічний паспорт програмного продукту, призначений насамперед для розробників, архітекторів та інженерів, які працюють над його створенням та супроводом. Її **ключова функція** – забезпечити знання, необхідні для модифікації, масштабування та виправлення помилок у коді. Найважливішою формою внутрішньої документації є коментарі в коді та **Docstrings** (спеціальні рядки документації, що описують функції та класи). Вони пояснюють складну логіку, аргументи функцій та очікувані результати. Цей вид документації є критично важливим для підтримки програмного продукту протягом його багаторічного життєвого циклу	**Набір матеріалів, створений для кінцевих користувачів програмного продукту**, а не для розробників. Її **головна мета** – пояснити, як правильно й ефективно використовувати програму для вирішення своїх завдань. Вона слугує мостом між складністю коду та простотою досвіду користувача. Цей вид документації має бути написаний зрозумілою, нетехнічною мовою і сфокусований на функціях, а не на реалізації. Вона зазвичай включає **Посібники користувача** з детальним описом інтерфейсу, покроковими інструкціями щодо встановлення та налаштування, а також оглядом основних функціональних можливостей

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Docstrings *(від англ.* ***documentation strings*** *– рядки документації) у Python – це спеціальні рядкові літерали, які розміщуються одразу після визначення модуля, класу, функції або методу. Docstrings є ключовою формою внутрішньої (технічної) документації в Python, оскільки забезпечує самодокументованість коду, що є життєво важливим для підтримки та масштабування будь-якого програмного продукту.*

Уявімо мобільний застосунок **«Електронний щоденник»**, який допомагає здобувачам і здобувачкам освіти зберігати домашні завдання та контролювати їх виконання. Якщо людина вперше відкриє його без інструкції, вона може не здогадатися, де додати завдання чи як увімкнути нагадування. Саме тому інструкція має бути написана простою мовою, з прикладами та підказками.



Документація потрібна не лише користувачам, а й тим, хто створює програму. Здебільшого програмний продукт розробляє команда, де кожен відповідає за свою частину роботи: код, дизайн, перевірку, підтримку. Щоб уникнути плутанини, створюють **опис структури проєкту** – своєрідну «карту для розробників».

Припустімо, ви створюєте сайт для шкільної бібліотеки. Якщо дизайнер знає, що на головній сторінці має бути пошукова стрічка і блок «Новинки», він створить макет саме з цими елементами. А програміст, маючи цю схему, швидко зрозуміє, які функції реалізувати.

Якщо кожен учасник команди буде працювати без спільного плану, проєкт може зіткнутися із серйозними проблемами.

Жодна програма не може вважатися готовою, доки її не перевірили. Тестування дає змогу знайти помилки та виправити їх до того, як користувач почне працювати з продуктом. Для цього складають **чеклісти** – таблиці або списки із завданнями для перевірки.

Уявімо, що ви тестуєте онлайн-гру «Шкільний квест». Чекліст допомагає не пропустити жодного важливого етапу. Це своєрідний план перевірки, який робить тестування системним і зрозумілим.

Опис мобільного застосунку «Електронний щоденник»:

«Електронний щоденник» допомагає учням і ученицям зручно контролювати домашні завдання та переглядати оцінки. Для початку роботи:

- завантажте застосунок із **Google Play** або **App Store**;
- зареєструйтеся за допомогою шкільної пошти й увімкніть сповіщення, щоб отримувати нагадування.

Опис структури проєкту сайту для шкільної бібліотеки

Щоб усі розуміли, який вигляд матиме сайт, потрібно зафіксувати в документації:

- які сторінки будуть на сайті;
- які елементи матиме кожна з них;
- які кольори та зображення потрібно використовувати.

Чекліст тестування онлайн-гри «Шкільний квест»

Ви перевіряєте:

- чи можна зареєструватися;
- чи правильно відкриваються рівні;
- чи зберігається прогрес;
- чи коректно працюють підказки і звук.

Також важливо впевнитися, що дані захищені від сторонніх змін.

ПІЗКАЗКА

Зверніть увагу! Чітка інструкція завжди зрозуміла навіть тим, хто вперше стикається з програмою.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Документування програмного проєкту в Python

Python-проєкт – це не просто один файл із кодом, а ціла організована структура. У ній є кілька обов'язкових складників. Усі разом вони утворюють чітку й зрозумілу структуру.

Обов'язкові складники програмного проєкту в Python

Модулі та пакети
де кожен виконує свою функцію (наприклад, математичні обчислення або робота з файлами)

Файл *requirements.txt*
який містить перелік усіх потрібних бібліотек

Віртуальне середовище
що зберігає ці бібліотеки в окремій «папці» і не дозволяє їм конфліктувати з іншими проєктами

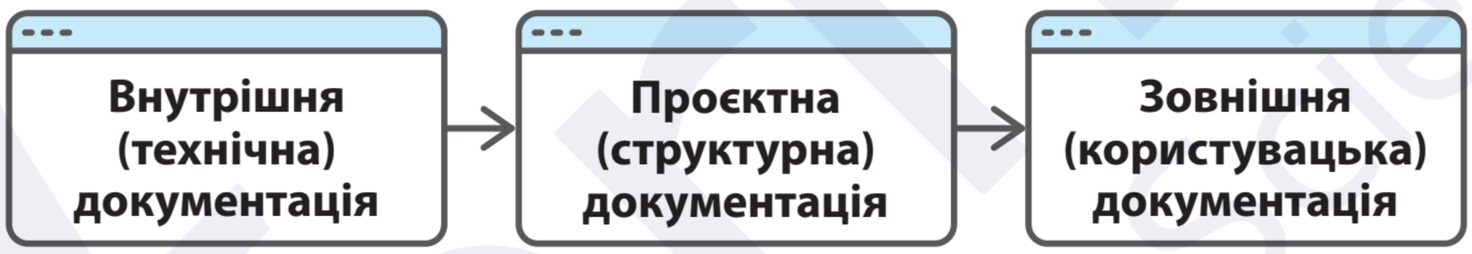
Тести
які перевіряють правильність роботи програми

Документація
яка пояснює роботу проєкту й окремих функцій

Під час роботи над шкільним проєктом, ви домовляєтеся з друзями, хто що робить: один готує презентацію, другий шукає матеріали, третій оформлює. Так само і з Python-проєктом: кожна частина має свою роль, але лише разом вони створюють готовий продукт.

***Документація** – це не додатковий «баласт», а невід'ємна частина будь-якого програмного продукту. Вона допомагає користувачам зрозуміти програму, команді – узгоджувати роботу, а тестувальникам – оцінювати якість.*

Схема документування програмного проєкту в Python



Внутрішня (технічна) документація: на рівні коду

Ця документація призначена для розробників і забезпечує зрозумілість, підтримуваність та масштабованість коду.

Внутрішня (технічна) документація: на рівні коду

Компонент	Місце розташування	Призначення
Docstrings ("""...""")	Одразу після визначення модуля, класу, функції чи методу	Детально описує призначення, параметри, тип поверненого значення та винятки. Доступна через **help()**
Коментарі (#)	Усередині складних блоків функцій та алгоритмів	Пояснює складну логіку, чому було прийнято певне рішення, або позначає місця для покращення
Анотації типів (Type hinting)	У сигнатурах функцій (наприклад, ***def func(arg: str) -> bool:***)	Забезпечує статичну перевірку типів і значно покращує читабельність та передбачуваність коду
requirements.txt	Кореневий каталог проєкту	Фіксує всі зовнішні залежності та їхні версії для відтворення середовища (***pip install -r***)

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Проєктна (структурна) документація: на рівні проєкту

Ці документи описують організацію, архітектуру та загальні правила роботи з проєктом.

Проєктна (структурна) документація: на рівні проєкту

Документ	Основний зміст	Цільова аудиторія
README.md	Основний вступний файл. Містить назву, коротке призначення, інструкції зі швидкої інсталяції та запуску, приклади використання	Всі користувачі та розробники
ARCHITECTURE.md	Детальний опис архітектури (модулі, пакети, бази даних, API)	Розробники, архітектори
CONTRIBUTING.md	Правила для співавторів (як вносити зміни, як оформлювати запити на злиття, стандарти кодування)	Розробники, що долучаються
Журнал змін ***(CHANGELOG/HISTORY)***	Хронологічний список усіх змін, виправлень і доданих функцій для кожної версії	Користувачі та розробники

Зовнішня (користувацька) документація: на рівні споживача

Зовнішня (користувацька) документація: на рівні споживача

Тип матеріалу	Призначення	Приклад
Посібник користувача	Покрокові інструкції щодо встановлення, налаштування та використання основного функціоналу	Детальний опис меню, кнопок, робочих процесів
API-референс	Автоматично згенерований опис всіх класів та функцій, доступних для програмного використання	Список усіх методів та їхніх параметрів, отриманий з Docstrings
FAQ та усунення несправностей	Відповіді на типові запитання та посібники для вирішення поширених проблем	«Що робити, якщо програма не запускається?»

Ви довідалися, що у Python-проєктах існує чітка структура, яка робить роботу зрозумілою і впорядкованою. Правильна організація коду та документації – ознака професійного підходу. Вона націлена на створення продуктів, якими приємно й безпечно користуватися.

Документація може бути навіть важливішою за сам код, оскільки вона забезпечує довговічність, підтримуваність та спільне розуміння проєкту.

Практичне завдання

Завантажте файл-заготовку зі сторінки інтернет-підтримки. Виконайте запропоноване завдання.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Тема 7
Тестування та налагодження програмних проєктів

Створення програмного продукту – це не лише написання коду. Навіть найкраща програма може містити приховані помилки, які не завжди помітні під час розробки. Тому важливо перевірити, чи працюють усі функції, чи зручний інтерфейс і чи не виникає технічних збоїв. **Тестування** – це процес пошуку та виявлення помилок, а **налагодження** – їх виправлення. Разом ці етапи допомагають зробити програму надійною, безпечною та зручною для користувачів.

Але робота над проєктом не закінчується після випуску. Коли програма потрапляє до користувачів, саме їхні відгуки стають цінним джерелом інформації. Вони сприяють виявленню не лише технічних проблем, а й недоліків у дизайні, відсутніх функцій або незручностей у роботі. Тому відгуки користувачів – це продовження тестування, яке відбувається вже в реальних умовах.



Тестування програмного проєкту

Процес тестування зазвичай циклічний

Планування:
визначення цілей, стратегії та створення тестових сценаріїв

Виконання:
запуск тестів і фіксація результатів

Аналіз:
виявлення помилок (багів) та їх виправлення

Регресія:
повторне тестування після виправлення помилок, щоб переконатися, що виправлення не спричинило нових проблем в інших частинах програми

Тестування програмного проєкту – *це систематичний процес перевірки програмного забезпечення з метою виявлення помилок (багів), оцінки його якості та підтвердження того, що воно відповідає всім визначеним вимогам.*



Отже, ***тестування*** *– це невід'ємне страхування якості, яка робить програмний продукт надійним і придатним до довготривалої підтримки.*

Розробники використовують різні методи, щоб переконатися, що продукт працює так, як очікується. Одним із найефективніших інструментів є чекліст – послідовний перелік кроків, які потрібно виконати, щоб перевірити всі функції програми.

Чекліст тестування програмного проєкту (Testing checklist) – *це структурований перелік пунктів, умов або кроків, які необхідно послідовно перевірити, щоб систематично оцінити якість та відповідність вимогам програмного забезпечення.*

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Ключові види тестування програмного проєкту

Вид тестування	Об’єкт перевірки	Основна мета	Хто виконує
Юніт-тестування (Unit testing)	Найменша ізольована частина коду (окрема функція, метод класу)	Підтвердити, що найменші блоки коду працюють коректно та незалежно від іншої системи	Розробники
Інтеграційне тестування (Integration testing)	Взаємодія та зв'язок між різними модулями, компонентами чи підсистемами	Перевірити, чи правильно компоненти обмінюються даними та працюють як єдине ціле	Розробники та тестувальники
Системне тестування (System testing)	Уся система в цілому в максимально наближеному до реального середовищі	Перевірити відповідність початковим функціональним і нефункціональним вимогам (безпека, продуктивність)	Тестувальники
Приймальне тестування (Acceptance testing)	Продукт, готовий до релізу, з огляду на кінцевого користувача / замовника	Підтвердити, що продукт задовольняє бізнес-вимоги і готовий до випуску	Кінцеві користувачі та замовник

Призначення чекліста

Чекліст виконує роль «дорожньої карти» для тестувальника, забезпечуючи:

1. Повноту перевірки

Гарантує, що жоден критичний аспект функціоналу, інтерфейсу чи безпеки не буде пропущено.

2. Уніфікацію

Сприяє тому, що тестування проводиться однаково різними людьми та в різних циклах (наприклад, під час регресії).

3. Ефективність

Дає змогу швидко оцінити статус проєкту, просто відзначаючи, які пункти вже виконані (checked), а які ще потребують уваги.

Приклад: Уявіть, що ви створюєте сайт для шкільної їдальні. Учні й учениці повинні мати можливість зареєструватися, обрати страви з меню, оформити замовлення та отримати підтвердження на електронну пошту.

Чекліст для тестування може включати такі дії:

- Перевірити, чи можна успішно зареєструватися за допомогою шкільної пошти.
- Спробувати обрати кілька страв та оформити замовлення.
- Переконатися, що система надсилає підтвердження на вказану електронну адресу.
- Перевірити, чи не можна оформити замовлення без вибору страви.

Якщо під час тестування виявляються помилки (наприклад, система дозволяє оформити порожнє замовлення або не надсилає лист-підтвердження), їх фіксують і виправляють до запуску сайту.



Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Чекліст можна оформити у вигляді таблиці, де вказуються дії користувача, очікуваний результат і статус перевірки.

Такий підхід допомагає не пропустити жодного важливого моменту й забезпечити якість програми.

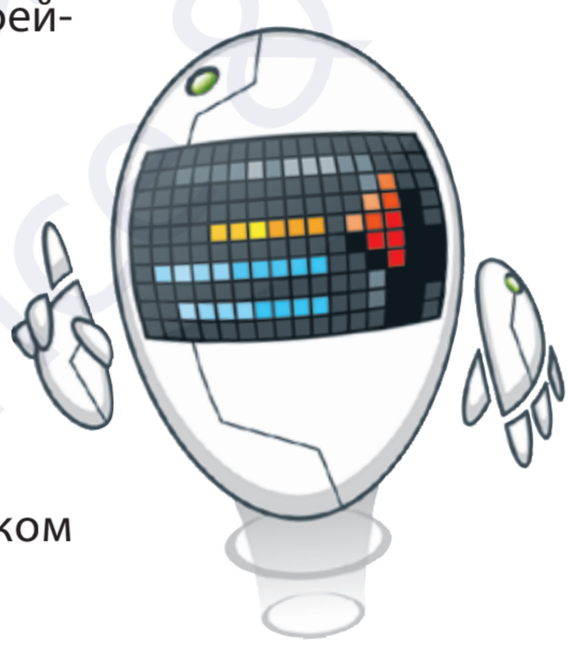
Чекліст

Дія	Очікуваний результат	Статус
Реєстрація за шкільною поштою	Користувач успішно зареєстрований	✓
Вибір страв з меню	Страви додаються до кошика	✓
Оформлення замовлення без вибору	Система виводить помилку	✗
Надсилання підтвердження	Лист приходить на електронну пошту	✓

Відгуки користувачів програмного проєкту

Після випуску програми користувачі починають залишати відгуки. Вони можуть стосуватися різних аспектів: технічних помилок, зручності інтерфейсу або побажань щодо нових функцій.

Відгуки користувачів програмного проєкту (User feedback) *– це інформація, думки, пропозиції або звіти про проблеми, які кінцеві споживачі надають розробникам щодо свого досвіду використання продукту.*



Для прикладу розглянемо користування мобільним застосунком «Шкільний розклад».

Мобільний застосунок «Шкільний розклад»

Користувач	Відгук	Статус
Відгук 1	*«Програма вилітає, коли намагаюся відкрити розклад на наступний тиждень»*	Це свідчить про критичну технічну помилку, яку потрібно виправити якомога швидше
Відгук 2	*«Кольори надто яскраві, болять очі»*	Це не заважає роботі програми, але робить її менш зручною
Відгук 3	*«Додайте, будь ласка, можливість переглядати оцінки в особистому кабінеті»*	Це побажання щодо нової функції, яке можна реалізувати в майбутніх оновленнях

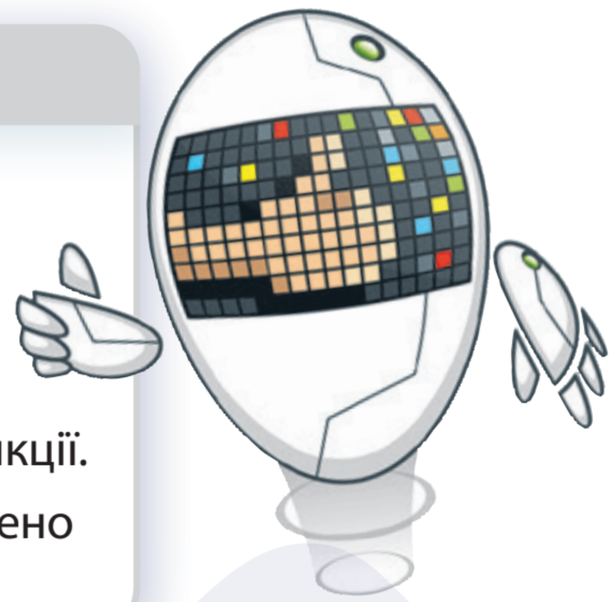
Відгуки можна поділити на три групи:

- **Критичні технічні проблеми** (програма не працює або «вилітає») – вимагають негайного виправлення
- **Проблеми зручності або дизайну** (незручний інтерфейс, надто яскраві кольори) – важливі, але не блокують роботу програми
- **Побажання щодо нових функцій (додати оцінки, змінити шрифт)** – можна врахувати в майбутніх версіях

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Як працювати з відгуками?

- Збирайте та фіксуйте всі коментарі в одному місці (наприклад, у таблиці або спеціальному файлі).
- Аналізуйте, які проблеми повторюються найчастіше.
- Визначайте пріоритети: спочатку виправляйте критичні помилки, потім – проблеми зручності, а вже пізніше – реалізовуйте нові функції.
- Відповідайте користувачам, повідомляйте, коли помилку виправлено або коли планується оновлення.



Приклад роботи з відгуками

Відгук	Тип проблеми	Пріоритет	Дії команди
Програма «вилітає» при відкритті розкладу	Критична технічна помилка	Високий	Виправити в найближчому оновленні
Кольори надто яскраві	Дизайн	Середній	Змінити палітру в наступній версії
Хотілося б бачити оцінки в особистому кабінеті	Нова функція	Низький	Додати до плану розвитку програми

Налагодження програмного проєкту

Після виявлення помилок команда розробників береться до їх виправлення. Важливо не лише усунути проблему, а й документувати зміни, щоб інші члени команди знали, що було зроблено.

Налагодження програмного проєкту (Debugging) *– це систематичний процес виявлення, ізоляції та виправлення помилок (багів) у програмному коді.*

Головна мета налагодження *– змусити програму працювати відповідно до її вимог та логіки, усуваючи будь-яку несподівану або неправильну поведінку. Це є невіддільною частиною циклу розробки та тестування, що забезпечує коректну та стабільну роботу програми.*

Процес налагодження зазвичай включає такі етапи:

1. Виявлення

Визначення того, що програма працює некоректно (часто відбувається під час тестування або звітів про відгуки користувачів).

2. Локалізація (ізоляція)

З'ясування точного місця в коді, де виникає помилка. Це найскладніша частина процесу.

3. Аналіз

Розуміння причини помилки (чому виникає неправильне значення, чому код «падає» тощо).

4. Виправлення

Внесення змін у код для усунення помилки.

5. Верифікація (регресія)

Повторний запуск коду та відповідних тестів, щоб переконатися, що помилка виправлена і що виправлення не спричинило нових помилок (регресій) в інших частинах програми.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Приклад: У грі «Шкільний квест» виявлено, що на третьому рівні гра «зависає».

Тестувальник складає звіт:

- **Опис проблеми:** Гра перестає реагувати на клавіші при переході на третій рівень.
- **Кроки для відтворення:** Запустити гру, пройти перші два рівні, спробувати перейти на третій.
- **Очікуваний результат:** Гра повинна продовжувати роботу.
- **Фактичний результат:** Гра «зависає».
- **Пріоритет:** Високий.

Документація по тестуванню та налагодженню повинна містити:

- Опис помилки та кроки для її відтворення.
- Дату виявлення та виправлення.
- Ім'я розробника, який усунув проблему.
- Версію програми, у якій помилку виправлено.

Це допомагає відстежувати прогрес та уникати повторення помилок у майбутньому.

Тестувальник програмного проєкту

Окрім звичайних користувачів, існують тестувальники – фахівці, які спеціально займаються пошуком помилок. Вони перевіряють програму більш детально, складають звіти та пропонують, що саме потрібно виправити.



Тестувальник програмного проєкту (Software tester) *– це фахівець, який систематично перевіряє програмне забезпечення з метою знайти помилки (баги), оцінити його якість та підтвердити, що воно відповідає початковим технічним і користувацьким вимогам.*

Тестувальники визначають рівень пріоритету кожної помилки

Високий пріоритет	→	Середній пріоритет	→	Низький пріоритет
Проблеми, які роблять програму непридатною для використання (наприклад, збій під час запуску)		Проблеми, які ускладнюють роботу, але не блокують її (наприклад, надто гучна музика)		Дрібні недоліки, пов'язані з естетикою або додатковими функціями (наприклад, некоректний переклад кнопок)

Приклад: У грі «Шкільний квест» тестувальник виявляє три проблеми.

Команда спочатку виправляє критичну помилку, потім регулює гучність, а підказки додає в одному з наступних оновлень.

Пріоритетність проблем у грі «Шкільний квест»:

1. Гра «вилітає» під час переходу на третій рівень (високий пріоритет).
2. Музика надто гучна (середній пріоритет).
3. Відсутні підказки для новачків (низький пріоритет).

ДОВІДКА

Професія **Тестувальник програмного забезпечення** як окрема і визнана роль у команді розробки почала формуватися та виокремлюватися у 1970-х роках, коли пошук недоліків став окремою, критично важливою спеціалізацією. До цього часу тестування не було окремою спеціалізацією, а вважалося частиною роботи розробника.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Отже, тестування та налагодження є невід'ємною частиною технології створення програмного продукту, забезпечуючи його якість і надійність.

Тестування *– це систематична перевірка, яка охоплює кілька рівнів: від юніт-тестування (перевірка окремих функцій) до системного та приймального тестування (перевірка всього продукту з погляду кінцевого користувача).*

За допомогою чеклістів та тестувальників виявляються помилки, а відгуки користувачів слугують життєво важливим джерелом для покращення продукту в наступних ітераціях.

Паралельно цьому процесу відбувається налагодження, метою якого є виявлення, ізоляція та виправлення знайдених помилок.

Успішне завершення циклу тестування та налагодження гарантує, що програмний продукт не лише відповідає технічним вимогам, але й є стабільним та зручним, що критично важливо для його довготривалого життєвого циклу.

Професія тестувальника є гарантом того, що програмний продукт буде функціональним, надійним і безпечним для кінцевого споживача.

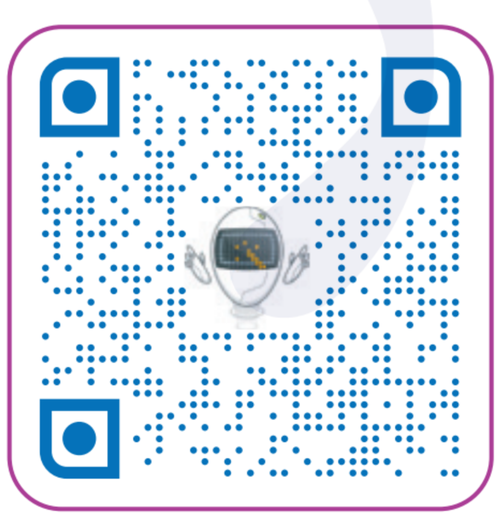


Програмні проєкти є надзвичайно важливими в сучасному світі, оскільки вони є рушійною силою інновацій та ефективності майже в усіх сферах життя. Вони дають змогу створювати, підтримувати та покращувати програмне забезпечення, яке, своєю чергою, лежить в основі функціонування бізнесу, науки, освіти, державного управління та нашого повсякденного життя. Тому програмні проєкти – це не просто розробка коду, а стратегічний процес, що забезпечує розвиток і стійкість сучасного суспільства та економіки.

Практичне завдання

Завантажте файл-заготовку зі сторінки інтернет-підтримки. Виконайте запропоноване завдання.

https://formula.education/online-ict-9-klas/





Інші платформи



Проєкт



Підсумки



Глосарій

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

5. 3D-моделювання



Будь-який проєкт починається з ідеї. Щоб зрозуміти, який вигляд вона матиме, її зображають на папері або у графічному редакторі, а потім створюють її тривимірну модель.

3D-моделювання – це процес створення тривимірних об'єктів та сцен за допомогою спеціальних комп'ютерних програм. Ця технологія використовується майже всюди: від кіно та комп'ютерних ігор до архітектури, медицини й навіть космічних досліджень.

За допомогою 3D-моделей можна створити віртуальний макет будівлі, розробити дизайн майбутнього гаджета, оживити вигаданого героя чи відтворити історичні артефакти. Це відкриває можливості для творчості, експериментів та інновацій.

У цьому розділі ви навчитеся:

- створювати прості 3D-об'єкти у спеціальних програмах;
- поєднувати базові фігури у складніші моделі;
- застосовувати текстури та кольори до своїх об'єктів;
- налаштовувати освітлення і камеру для отримання кращого вигляду сцени;
- експортувати готові моделі для використання у проєктах (іграх, відео чи презентаціях).

У цьому розділі ви дізнаєтеся:

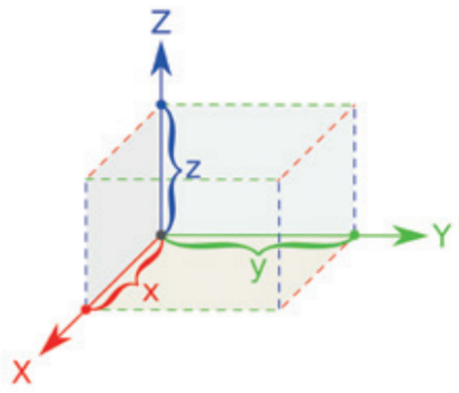
- що таке 3D-моделювання, для чого його використовують;
- які етапи проходить модель – від ідеї до готового об'єкта;
- як створювати прості 3D-фігури, як перетворювати їх у складні об'єкти;
- як застосовуються текстури, матеріали та освітлення, щоб зробити моделі реалістичними.

Програмне забезпечення:

- Tinkercad;
- Blender;
- FreeCAD;
- Daz 3D;
- Autodesk Maya;
- Autodesk 3ds Max.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Тема 1
Моделі



Для того, щоб пояснити складну ідею або побачити об'єкт, який поки що існує лише в задумі. Як це зробити? Створіть **модель** – спеціальну спрощену копію реальності чи концепції. Модель дає змогу краще уявити, дослідити та навіть протестувати те, що поки важко охопити безпосередньо.

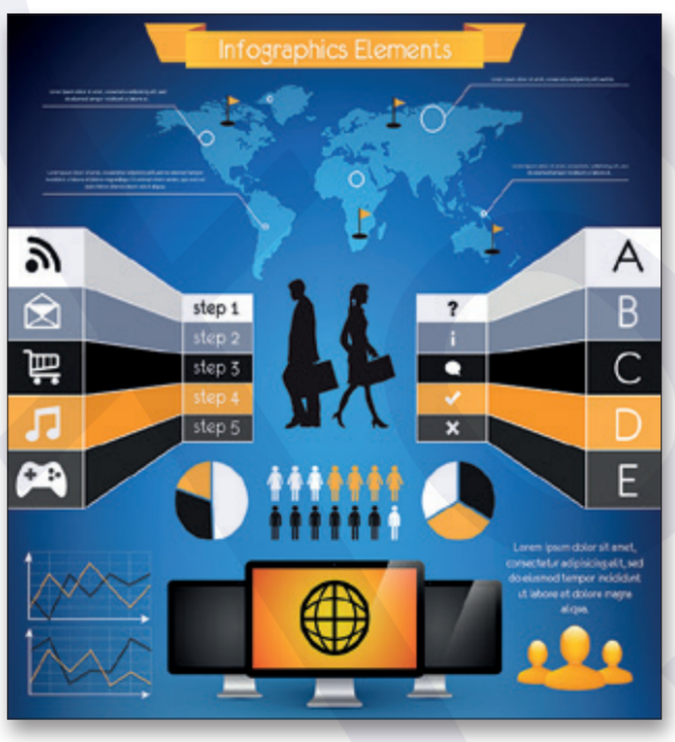
У світі тривимірного моделювання моделі допомагають нам відтворювати реальність у віртуальному просторі. У такий спосіб ми здатні створювати об'єкти, що на вигляд як справжні, але існують тільки на екрані. Моделі створені для того, щоб можна було краще зрозуміти, пояснити й відтворити складні речі у простій та зручній формі. Завдяки цьому можна будувати міста, придумувати героїв і навіть проєктувати цілі світи для ігор чи фільмів.

Види моделей

Загалом моделі бувають різними.

Існують **фізичні моделі** – це матеріальні об'єкти, які в зменшеному або збільшеному вигляді відтворюють структуру або поведінку іншого об'єкта чи системи. Уявіть собі модель літака в аеродинамічній трубі, яка допомагає інженерам вивчати його опір повітрю, або архітектурний макет будівлі, що дає змогу замовнику ще до початку будівництва побачити, який вигляд матиме майбутня споруда. Ці моделі можна помацати, роздивитися з усіх боків, і вони безпосередньо взаємодіють з фізичним світом, хоч і в обмеженому масштабі.





Але коли говорять про тривимірне моделювання, на передній план виходять **інформаційні моделі**. Це вже не матеріальні об'єкти, а радше їхні віртуальні двійники, що існують у цифровому просторі. Вони являють собою набір даних, закодованих у певній формі, які описують структуру, властивості та поведінку об'єкта чи системи. Тривимірна модель автомобіля, створена в системі автоматизованого проєктування, або віртуальний персонаж у відеогрі – все це приклади інформаційних моделей.

Артефакт *– небажана, аномальна або неправильно сформована особливість у цифровій моделі або у фінальному об'єкті.*



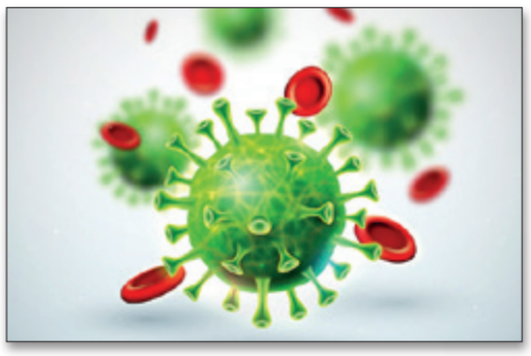
Тривимірне моделювання є надзвичайно потужним інструментом, оскільки воно дає змогу створювати інформаційні моделі з неймовірною деталізацією та реалістичністю. Ці віртуальні двійники можуть бути не лише візуальним представленням об'єктів, а й містити інформацію про їхні властивості, поведінку та навіть історію створення. Вони стають своєрідними цифровими артефактами, які можна досліджувати, аналізувати та використовувати для найрізноманітніших цілей – від розробки нових технологій до збереження культурної спадщини.

Візуалізація та наукове моделювання

Уявіть поєднання візуалізації та наукового моделювання! Особливо в контексті тривимірного світу. Це ніби поєднання двох потужних сил: здатності людини бачити та здатності людини мислити абстрактно й аналізувати. Тривимірне моделювання стає тут тим чарівним інструментом, який перетворює сухі цифри та складні рівняння на зрозумілі та наочні образи.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.





Уявіть собі наукове дослідження, де вчені намагаються розгадати таємниці будови молекули ДНК. Раніше їм доводилося оперувати складними хімічними формулами та абстрактними просторовими уявленнями. Але завдяки тривимірній візуалізації ця невидима структура постає перед їхніми очима у всій своїй красі та складності: подвійна спіраль, взаємодія атомів, динаміка процесів. Візуалізація не просто ілюструє наукові дані, вона стає інструментом для глибшого розуміння, допомагає побачити закономірності та взаємозв'язки, які могли б залишитися непоміченими в таблицях чи графіках.



В астрономії тривимірні моделі космічних тіл та явищ дають нам змогу досліджувати будову галактик, рух планет та наслідки зіткнень у космосі, роблячи абстрактні відстані та масштаби відчутними. У медицині візуалізація внутрішніх органів людини, процесів на клітинному рівні або моделювання поширення захворювань відкриває нові горизонти для діагностики, лікування та наукових досліджень.



В інженерії тривимірні моделі майбутніх конструкцій допомагають проводити складні симуляції, аналізувати міцність, аеродинаміку та інші критично важливі параметри ще до фізичного втілення проєкту.



Тривимірне моделювання сприяє візуалізації гіпотез, їхній перевірці шляхом симуляцій, виявленню нових закономірностей та висвітленню відкриттів у зрозумілій та захопливій формі. Це справжнє мистецтво перетворення абстрактного на конкретне, невидимого на видиме, складного на зрозуміле.

Моделювання не просто інструмент візуалізації, а й важливий метод наукового пізнання

- **Спростити складність** – виділити ключові елементи та взаємодії досліджуваного явища
- **Проводити віртуальні експерименти** – тестувати гіпотези та досліджувати поведінку систем у різних умовах без реальних ризиків і витрат
- **Покращити комунікацію** – ефективно передавати складні наукові концепції іншим дослідникам та громадськості
- **Візуалізувати невидиме** – зробити наочними абстрактні концепції та процеси, що відбуваються на мікроскопічному чи макроскопічному рівнях
- **Генерувати нові знання** – виявляти неочікувані закономірності та формулювати нові гіпотези на основі спостереження за моделлю

Практичне завдання

Завантажте файл-заготовку зі сторінки інтернет-підтримки. Виконайте запропоноване завдання.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Тема 2

Методи тривимірного моделювання

Уявіть, що перед вами стоїть завдання – створити щось, чого ще не існує. Це може бути персонаж для анімації, деталь для складної машини чи навіть модель майбутньої будівлі. Як це зробити? Саме тут стануть у пригоді методи тривимірного моделювання – набір інструментів і технік, які дають змогу створювати форму в цифровому просторі. Кожен метод має свої особливості, принципи та сферу застосування.



Основні типи моделювання

Пустотіле
знайшло широке застосування в дизайнерських програмах і допомагає створювати об'єкти, що мають внутрішні порожнини. Наприклад, під час розробки дизайну пляшки чи корпусу комп'ютера важливо враховувати не лише зовнішню форму, а й внутрішній простір

Твердотіле
є незамінним інструментом у системах автоматизованого проєктування (САПР), орієнтоване на створення цілісних, заповнених об'єктів, що мають об'єм і масу. Наприклад, 3D-моделі гайки та болта з точними параметрами різьби

Методи моделювання

Об'єктне
Передбачає створення об'єктів за допомогою геометричних форм, які можна масштабувати та змінювати.

Полігональне
Використовує полігони для створення об'єктів з деталями різного рівня складності.

Сплайнове
Базується на створенні об'єктів за допомогою плоских форм, ліній та кривих.

Поверхневе
Розширює сплайнове моделювання, дозволяючи натягувати поверхні на каркас об'єктів.

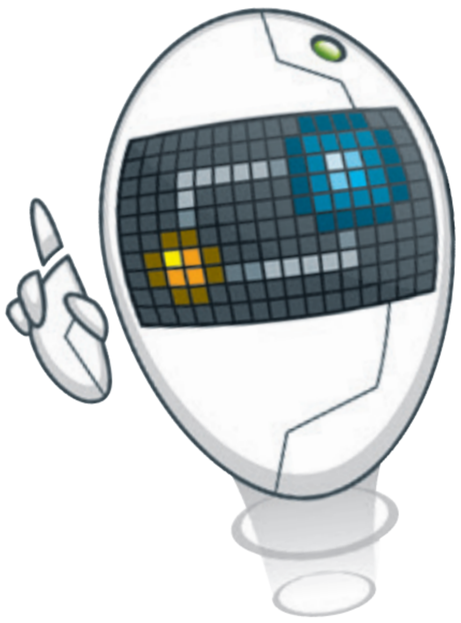
Художня ліпка, або **скульптинг**
Це метод художнього моделювання, завдяки якому можна вільно формувати об'єкти, немовби виробляючи їх з глини.

NURBS
Використовує нерегулярні математичні криві для створення об'єктів.

Промислове моделювання для САПР
Спрямоване на створення деталізованих моделей для потреб САПР та інших промислових застосувань.

ІСТОРІЯ

Історія 3D-моделювання розпочалась у 1963 році. Саме тоді Айвен Сазерленд, американський учений-інформатик, створив революційну програму для інтерактивної комп'ютерної графіки – Sketchpad. За допомогою неї користувачі могли малювати безпосередньо на дисплеї комп'ютера, використовуючи світлове перо. Така техніка дала змогу створювати складні малюнки, а також різноманітні графічні моделі. Це стало вражаючим досягненням, адже до цього часу будь-яке зображення на комп'ютері створювали вручну за допомогою кодів.

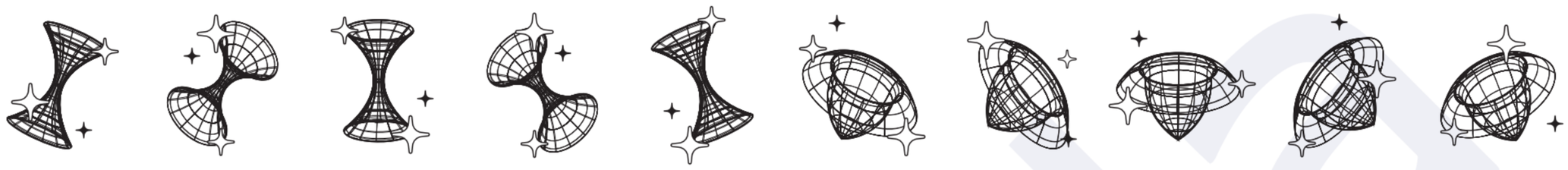


Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Об’єктне моделювання

Об’єктне тривимірне моделювання, яке часто називають параметричним, являє собою інтуїтивно зрозумілий підхід до створення тривимірних віртуальних об’єктів у програмах для роботи з 3D-графікою. У його основі лежить використання готових геометричних примітивів. Користувач додає фігури на сцену й трансформує їх, змінюючи розміри, положення в просторі, орієнтацію, кількість сегментів та інші параметри.

Примітив *– це базовий, найпростіший геометричний об'єкт, який використовується як основний будівельний блок у 3D-моделюванні.*





Головна перевага – **параметричність**: у будь-який момент можна повернутися до вихідних налаштувань і відредагувати модель. Це зручно для швидкого прототипування та ітеративного дизайну.

Примітиви можна поєднувати булевими операціями (додавання, віднімання, перетин), деформувати, скручувати, згинати, а також застосовувати модифікатори (наприклад, згладжування кутів чи збільшення щільності сітки).

Завдяки точності та гнучкості метод широко використовується в дизайні, архітектурі, інженерії та мистецтві як основа для складних моделей.

Полігональне моделювання

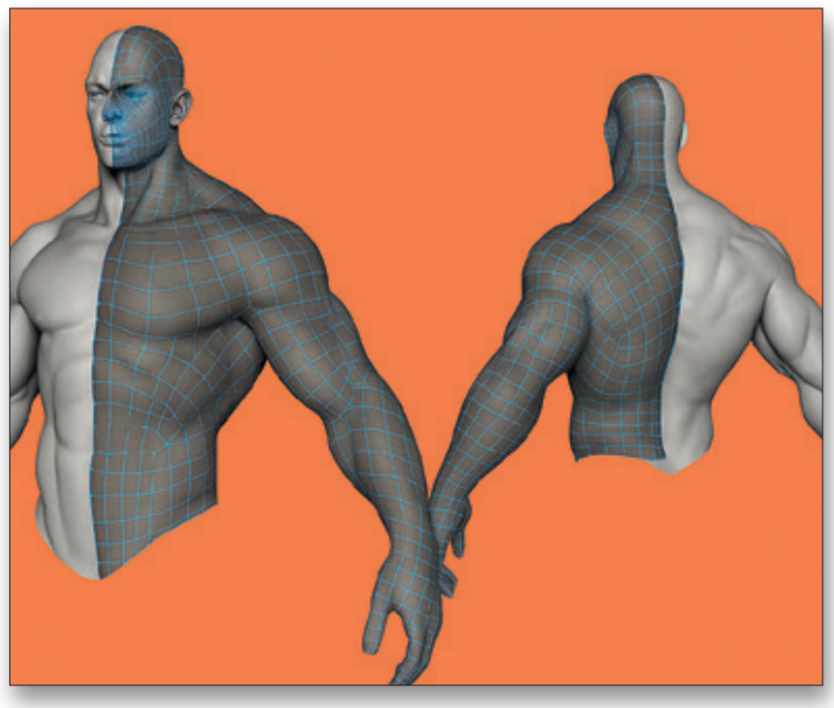
Полігональне тривимірне моделювання – це створення об’єктів із многокутників (полігонів), що складаються з вершин, ребер і граней. Полігони можуть бути трикутними, чотирикутними чи складнішими, завдяки чому можна відтворювати як прості, так і дуже деталізовані форми.

Метод вирізняється гнучкістю та простотою: змінюючи вершини, ребра чи грані, легко формувати різноманітні об’єкти. Для вигнутих поверхонь використовують більше полігонів, для плоских – менше.

Полігональні моделі зручні для анімації та ігор, оскільки добре масштабуються, легко деформуються. Уявіть собі персонажа, що рухається, – його полігональна сітка може згинатися та розтягуватися, імітуючи рухи м’язів та суглобів. Вони також оптимізуються за кількістю полігонів, що робить їх «легкими» для обчислень і придатними для ігор та інтерактивних середовищ.

Завдяки цьому полігональне моделювання є універсальним інструментом 3D-художників і дизайнерів, який поєднує деталізацію, ефективність та продуктивність.





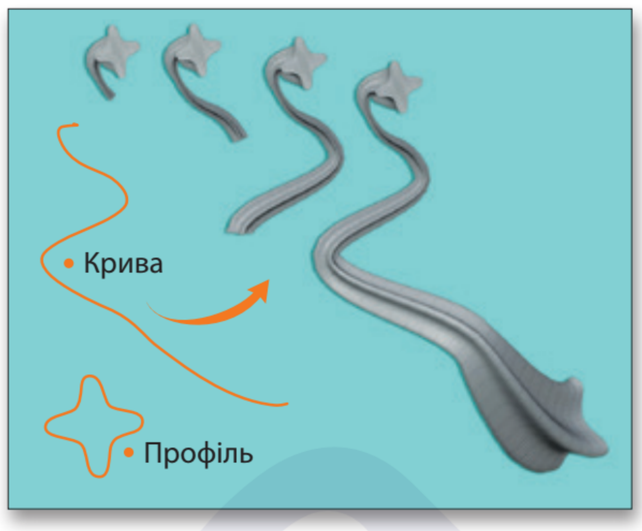
Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Сплайнове моделювання

Сплайнове моделювання – це універсальний метод, який часто застосовується в багатьох сферах, де потрібне створення точних і плавних 3D-об'єктів. Його головна перевага полягає в тому, що він дає змогу будувати поверхні, які зберігають свою якість і плавність незалежно від масштабу, оскільки базується на математичних формулах, а не на полігонах.

Слово «сплайн» походить від англійського слова *spline* – гнучка лінійка, яку креслярі використовували для проведення плавних кривих. Це підкреслює практичне походження методу.

Сплайни відкривають широкі можливості для побудови різноманітних тривимірних об'єктів завдяки кільком ключовим технікам.



Ключові техніки сплайнів

Обертання сплайнової кривої навколо осі. Обертаючи двовимірну форму на 360°, можна легко отримати об'ємну модель. Цей метод ідеально підходить для створення об'єктів, що мають осьову симетрію: вази, келихи, пляшки, тарілки

Екструзія (видавлювання) для надання об'єму контуру об'єкта. Якщо є контур деталі, його можна «витягнути» вздовж певної осі, щоб надати йому об'єму: карнизи, молдинги, труби, балки тощо

З'єднання кількох кривих для створення складних гладких поверхонь, як-от корпус автомобіля, крила літака. Їхні складні вигнуті форми часто моделюються шляхом «зшивання» разом кількох сплайнових кривих, утворюючи гладкі та аеродинамічні поверхні

Використання сплайнів як напрямних – для формування об'єктів або розташування елементів уздовж траєкторії. Наприклад, можна створити сплайнову криву, що імітує звивисту річку, а потім «протягнути» вздовж неї поперечний переріз човна, автоматично отримуючи модель човна, що повторює всі вигини річки. Це може бути корисним для створення перил, ланцюгів або рядів дерев

В **анімації** сплайни застосовують для побудови траєкторій руху, що забезпечує плавність і контроль швидкості

Застосування сплайнового моделювання

Автомобілебудування та авіація

Сплайни є основою для проєктування зовнішніх поверхонь транспортних засобів, де важливі аеродинаміка та естетика.

Промисловий дизайн

Метод використовується для моделювання споживчих товарів, дозволяючи швидко вносити зміни в дизайн зі збереженням точності та пропорцій.

Будівництво

Допомагає проєктувати складні криволінійні форми будівель, які неможливо створити стандартним полігональним підходом.

Інженерні розрахунки

Сплайни використовуються для аналізу деформації об'єктів під навантаженнями, що важливо для оцінки їхньої міцності.

Комп'ютерна графіка та анімація

Сплайни є важливим інструментом у 3D-анімації для створення плавних траєкторій руху, що забезпечує реалістичність.

Відеоігри

У розробці відеоігор сплайни використовуються для створення доріг, річок, трас та інших елементів ігрового світу, які мають реалістичну форму.

Протезування

Сплайни створюють точні 3D-моделі кісток на основі МРТ, КТ, що допомагає хірургам у плануванні операцій.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

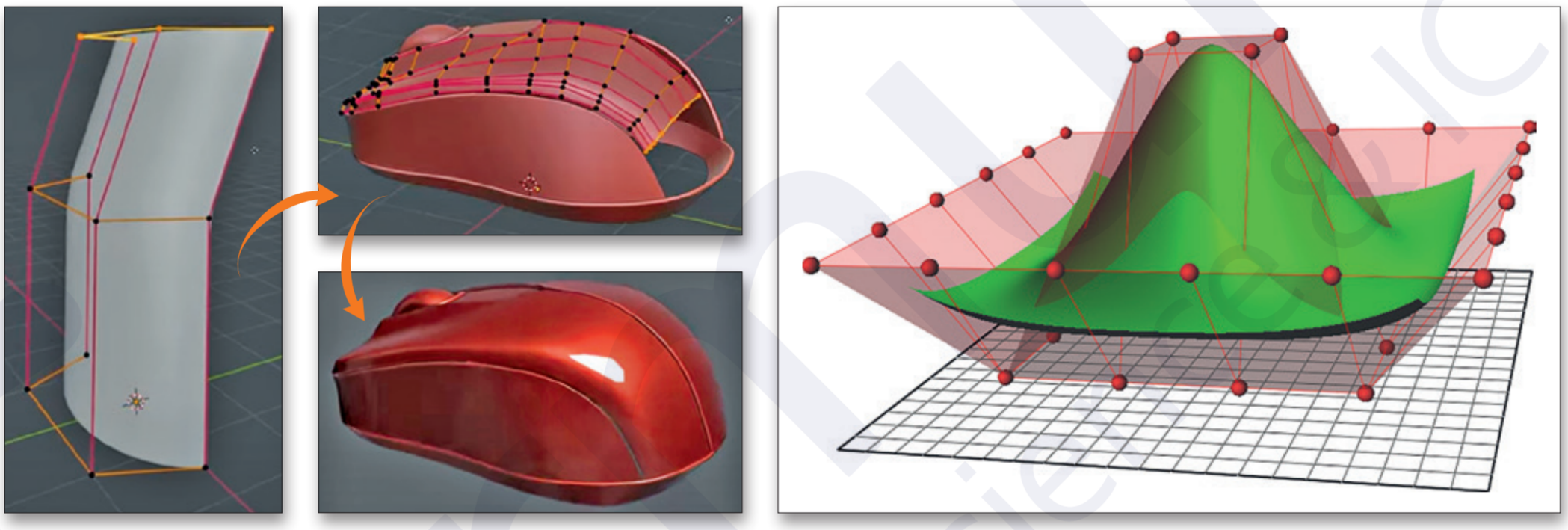
NURBS-моделювання

NURBS (Non-Uniform Rational B-Splines) – це метод створення 3D-об'єктів на основі математично точних кривих і поверхонь. На відміну від полігонального моделювання, де форма складається з многокутників, **NURBS** використовує криві та поверхні для створення форм. Замість складання фігури з окремих граней, ви «ліпите» її з гладких, неперервних ліній.

Форма визначається **контрольними точками**: крива огинає їх і змінюється залежно від їхнього положення та ваги (ступеня впливу). Це дає гнучкість і високу точність.

NURBS-моделювання є ключовим інструментом у сферах, де важлива математична точність і плавність форм. На відміну від полігонального моделювання, яке складається з безлічі плоских граней, NURBS-поверхні визначаються математично, що забезпечує їхню високу якість незалежно від масштабу.

*Якщо ви бачите обтічний кузов сучасного автомобіля, крило літака, елегантну форму побутової техніки, або навіть складні органічні поверхні в анімаційних фільмах, дуже висока ймовірність, що ці об'єкти були створені саме за допомогою **NURBS-моделювання**.*



Застосування NURBS-моделювання

Автомобілебудування

NURBS-моделювання є основою для проєктування зовнішніх поверхонь і деталей. Це дає змогу інженерам і дизайнерам створювати ідеально гладкі, аеродинамічно ефективні та візуально привабливі форми, які неможливо отримати за допомогою інших методів.

Дизайн

Метод **NURBS** застосовується для створення корпусів телефонів, меблів, побутової техніки, де важлива елегантність та ергономічність.

Медицина

У медицині **NURBS** використовують для точного моделювання органів на основі даних сканування (МРТ, КТ) для планування хірургічних операцій або створення індивідуальних протезів.

Кіноіндустрія

NURBS уможливлюють створення органічних форм, як-от тіла персонажів, що вимагають плавних і неперервних переходів між поверхнями.

Архітектура

NURBS-моделювання допомагає архітекторам проєктувати та візуалізувати складні, криволінійні фасади та елементи будівель, які не мають аналогів у стандартній геометрії.

Ювелірне мистецтво

Завдяки надзвичайній точності та гнучкості, **NURBS** слугує для створення дрібних і деталізованих ювелірних виробів зі складними вигинами, що потім можуть бути відтворені за допомогою 3D-друку.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

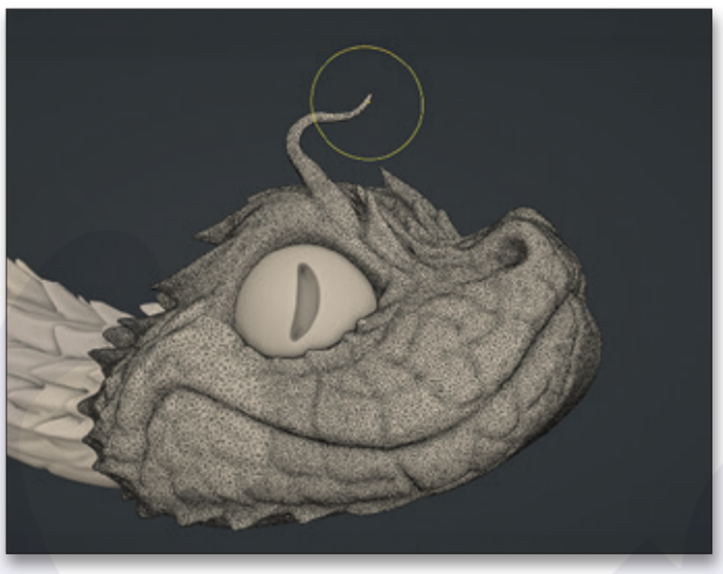
Скульптинг (художнє моделювання)

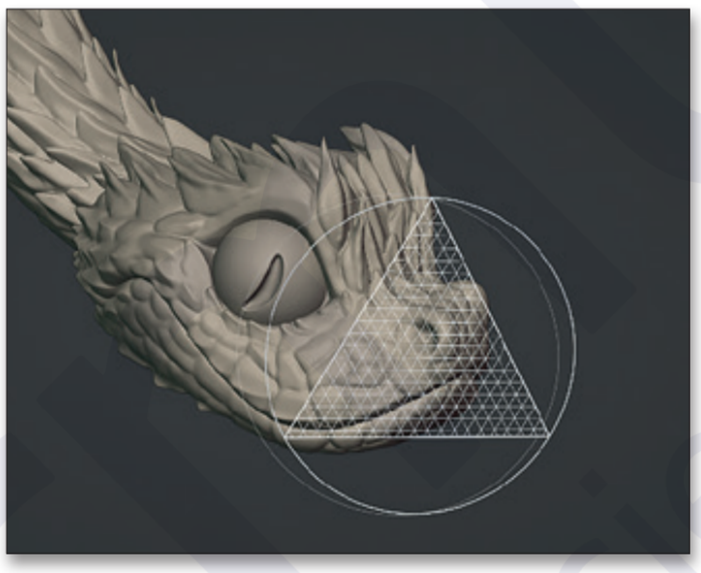
Цифровий скульптинг – це метод 3D-моделювання, який нагадує роботу зі справжньою глиною. Замість редагування полігонів художник «ліпить» модель за допомогою віртуальних пензлів: тягне, штовхає, згладжує чи загострює поверхню, створюючи складні та органічні форми.

Цей підхід зручний для **органічного дизайну** – фантастичних істот, портретів, складних драпірувань тканин, анатомічно точних моделей м'язів і дрібних деталей.

Однак моделі, створені методом скульптингу, часто мають надзвичайно високу щільність полігонів, що може бути проблематичним для анімації та використання в реальних проєктах, як-от відеоігри чи інтерактивні інсталяції, де важлива оптимізація та продуктивність. Для вирішення цієї проблеми застосовують процес **ретопології** – створення нової, більш простої полігональної сітки поверх високодеталізованої скульптури. Це дає змогу зберегти візуальну складність моделі, але зробити її придатною для подальшої анімації та рендерингу в реальному часі.

***Рендеринг** – це процес перетворення тривимірної моделі у готове зображення або відео з урахуванням світла, тіней, матеріалів і текстур. Простіше, це «фотографування» 3D-сцени комп'ютером. Саме завдяки рендерингу можна побачити реалістичні картинки чи анімацію замість «сіток» і каркасів моделей.*







Скульптинг активно використовують у кіно, іграх, 3D-друку, реставрації артефактів та створенні текстур. За допомогою спеціальних карт нормалей можна відобразити дрібні нерівності (пори, зморшки, тріщини) без перевантаження моделі полігонами, що робить її реалістичною й оптимізованою.

У галузі реставрації історичних артефактів скульптинг використовується для точного відновлення пошкоджених фрагментів, дозволяючи повернути до життя безцінні предмети минулого.

Скульптинг є незамінним інструментом для створення складних скульптурних композицій, які потім можуть бути відтворені за допомогою 3D-друку, відкриваючи нові горизонти для сучасного мистецтва.

Завдяки можливості працювати з високодеталізованими моделями та використовувати інтуїтивний інтерфейс, 3D-скульптинг сприяє створенню надзвичайно реалістичних і деталізованих об'єктів. З розвитком технологій, таких як VR та AR, можливості 3D-скульптингу лише розширюються, надаючи художникам ще більше інструментів для втілення своїх ідей у життя.

***Карта нормалей** – це спеціальна текстура, яка зберігає інформацію про дрібні нерівності поверхні (пори, зморшки, подряпини). Вона створює ілюзію складної форми за допомогою світла й тіней. Карта нормалей дає змогу зробити модель більш деталізованою та реалістичною, але при цьому «легкою» для комп'ютера.*

ІСТОРІЯ

Французький інженер **П'єр Безьє (Pierre Bézier)**, який працював у компанії Renault, незалежно розробив свої **«криві Безьє»** у 1960-х роках для комп'ютерного дизайну кузовів автомобілів. Це стало основою для подальшого розвитку **сплайнів** та **NURBS**. Хоча криві Безьє не є NURBS у чистому вигляді, вони заклали фундаментальні принципи.

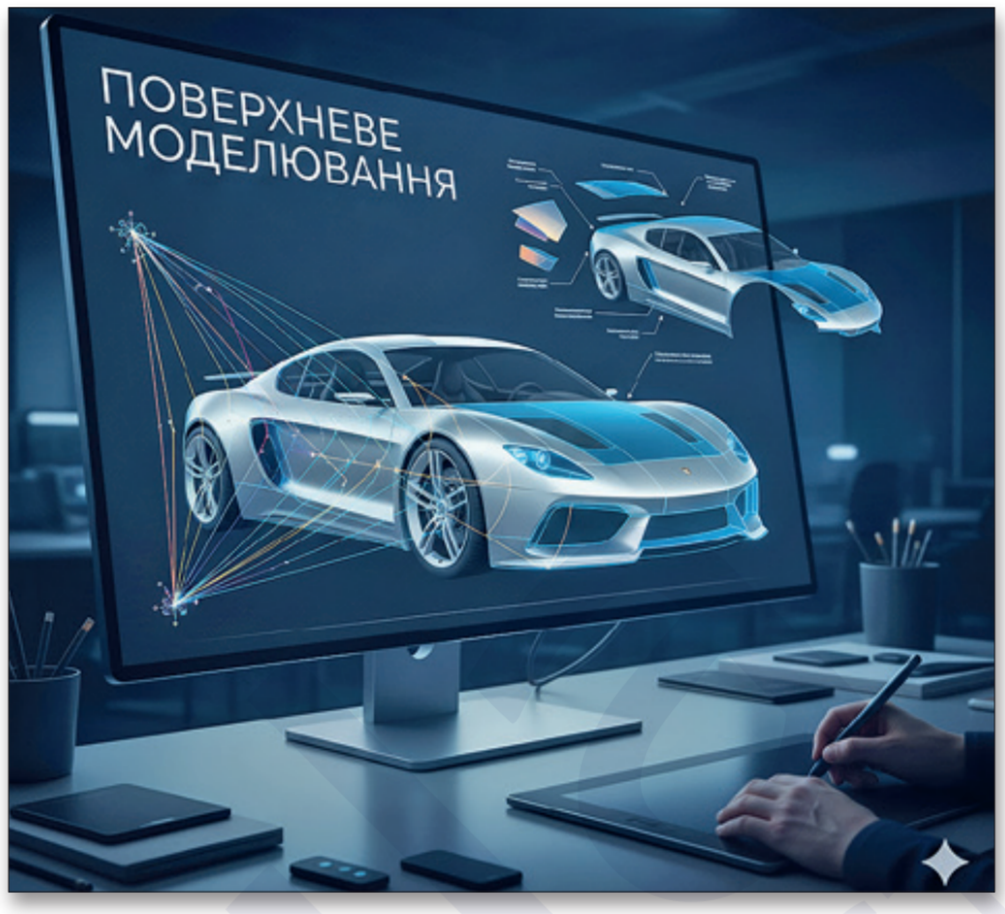
Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Поверхневе моделювання

Поверхневе моделювання – це метод створення 3D-моделей, який зосереджується виключно на зовнішній оболонці об'єкта, не враховуючи його внутрішньої структури. У цьому методі моделі будуються як порожні оболонки, визначені набором кривих і поверхонь, дозволяючи створювати складні, вільно сформовані та гладкі поверхні.

Поверхневе моделювання базується на математичних формулах, які описують криві та поверхні. Найчастіше використовуються сплайни та NURBS. Завдяки використанню математичних кривих, поверхневе моделювання забезпечує високу точність і гладкість форм. Це робить його ідеальним для промислового дизайну та інженерії, де важлива якість поверхні та її відповідність суворим стандартам.

Поверхневе моделювання, особливо з використанням NURBS, дає змогу інженерам і дизайнерам створювати ідеально гладкі, аеродинамічні форми для кузовів автомобілів, фюзеляжів літаків й інших транспортних засобів. Математична точність поверхневих моделей допомагає проводити симуляції повітряних потоків і краш-тести. Також цей метод корисний для архітекторів та інженерів, щоб проєктувати складні криволінійні будівлі, мости та інші споруди. Поверхневе моделювання дає змогу візуалізувати й детально пропрацювати футуристичні фасади та елементи.



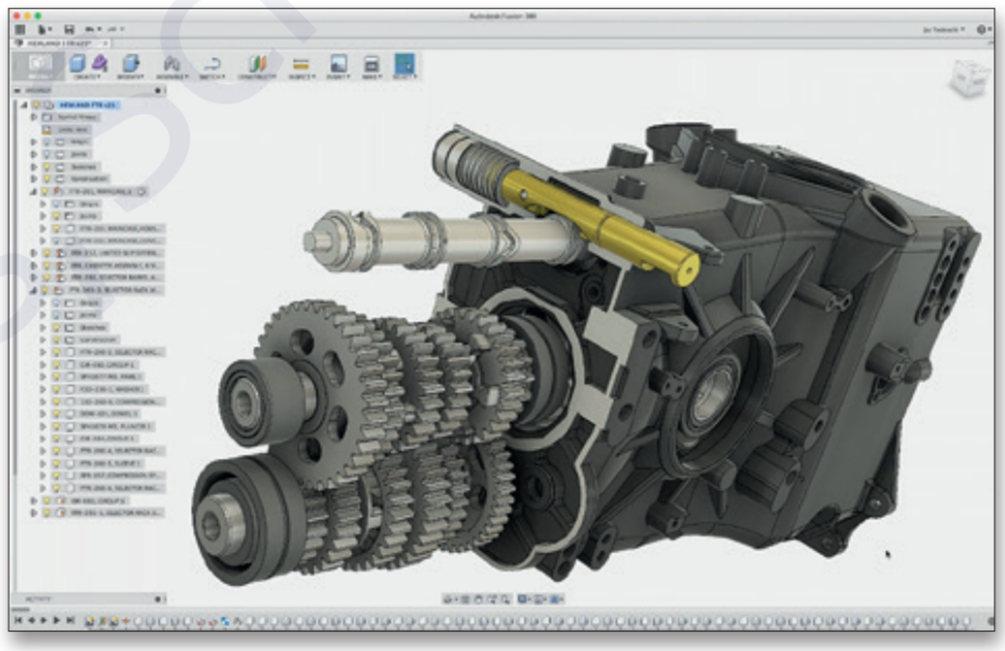
Поверхневе моделювання часто поєднують з твердотільним моделюванням, використовуючи перше для створення складних зовнішніх форм, а друге – для додавання внутрішніх інженерних елементів.

Твердотіле моделювання у САПР

На відміну від методів, що оперують лише поверхнями або оболонками, **твердотіле моделювання** в системах автоматизованого проєктування (САПР) має справу з повноцінними віртуальними твердими тілами, що володіють об'ємом та внутрішньою структурою. Це один із найбільш точних та математично обґрунтованих способів 3D-моделювання. Він використовується в інженерії, архітектурі, промисловому дизайні та машинобудуванні.

Головна перевага – висока точність. Моделі містять усі дані про форму, розміри та зв'язки між елементами, тому підходять не тільки для візуалізації, а й для виробництва.

Такі моделі можна застосовувати для симуляцій (міцність, теплопередача, гідродинаміка), а також для автоматизованого створення креслень і програм для верстатів із числовим програмним керуванням (ЧПК) у подальшому виготовленні деталей.



Практичне завдання

Завантажте файл-заготовку зі сторінки інтернет-підтримки. Виконайте запропоноване завдання.

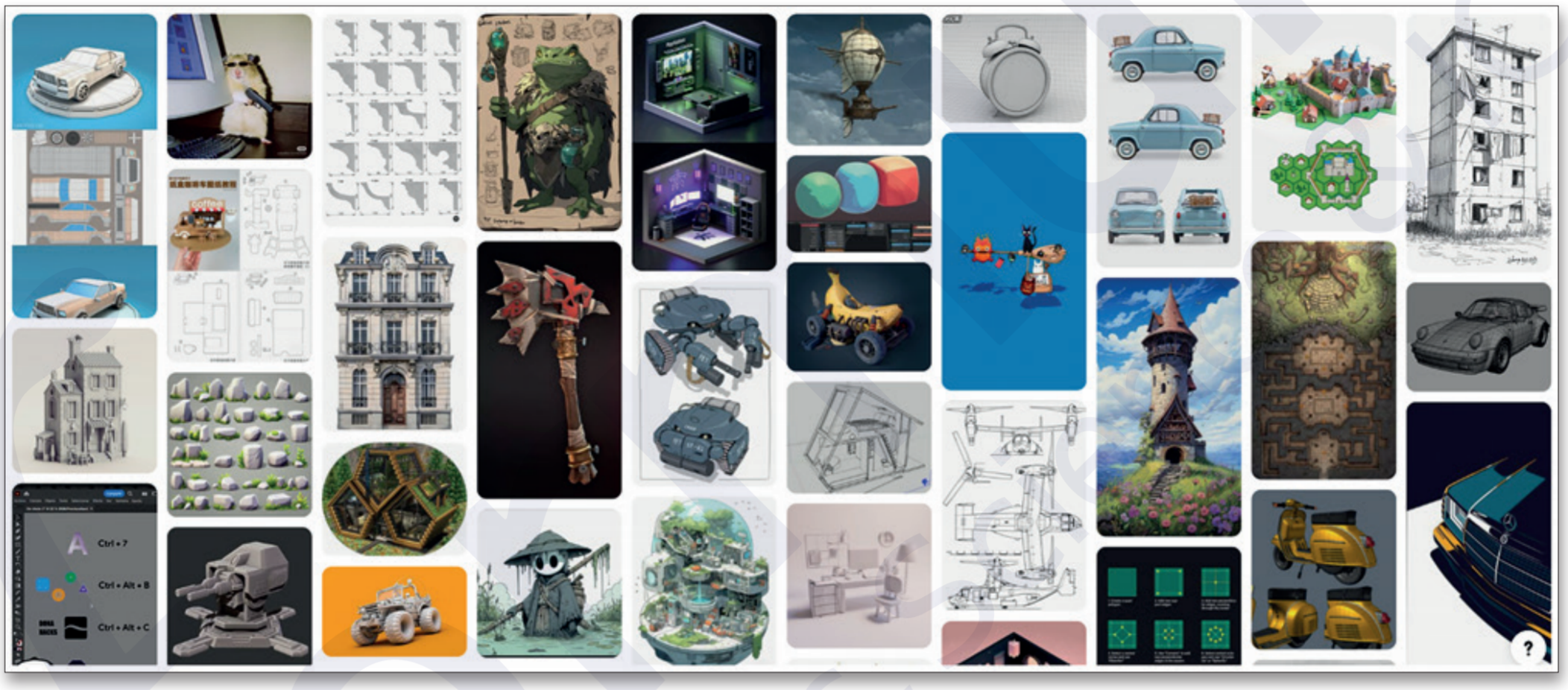
Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Тема 3
Етапи створення 3D-зображення

Етапи створення тривимірного зображення

Перший крок у створенні 3D-моделі – це пошук ідеї та натхнення. Так само як художник або дизайнер перед початком роботи, 3D-дизайнер спочатку шукає приклади – це можуть бути фотографії, малюнки, креслення чи вже готові моделі. Такі приклади називаються **референсами** – вони допомагають краще уявити, який вигляд буде мати майбутній об'єкт.

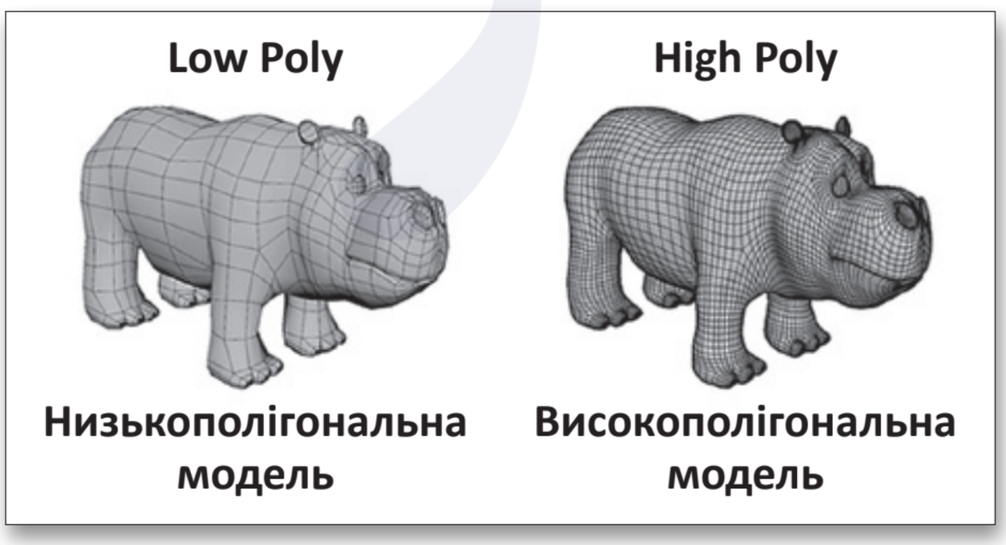
Щоб зібрати всі ідеї в одному місці, часто створюють **дошку настрою (mood board)** – це набір зображень, які показують стиль, форму, кольори та деталі, щоб краще уявити, що саме хочемо зробити. Це дуже корисно, бо допомагає не загубити задум і зробити модель більш реалістичною або цікаво стилізованою.



Наступний важливий етап – це моделювання форми об'єкта. Зазвичай він складається з двох частин. Спочатку створюється **низькополігональна** модель – це як каркас майбутнього об'єкта, зроблений із простих багатокутників (їх називають полігонами). Така модель має загальні обриси та форму, але без дрібних деталей.

Низькополігональна модель дуже зручна для роботи: її легко змінювати, додавати нові елементи або експериментувати з формою. Це дає змогу швидко перевірити, який вигляд матиме об'єкт, перш ніж переходити до складнішого етапу – додавання деталей.

Коли базова форма готова, настає час додати деталі – це називається **високополігональне моделювання**. На цьому етапі до простої моделі додається багато нових полігонів (многокутників), щоб зробити її більш реалістичною. Це схоже на те, як скульптор поступово надає своєму творінню остаточних рис. Можна створити складні вигини, нерівності, зморшки, текстури – усе, що робить об'єкт реалістичним і цікавим.

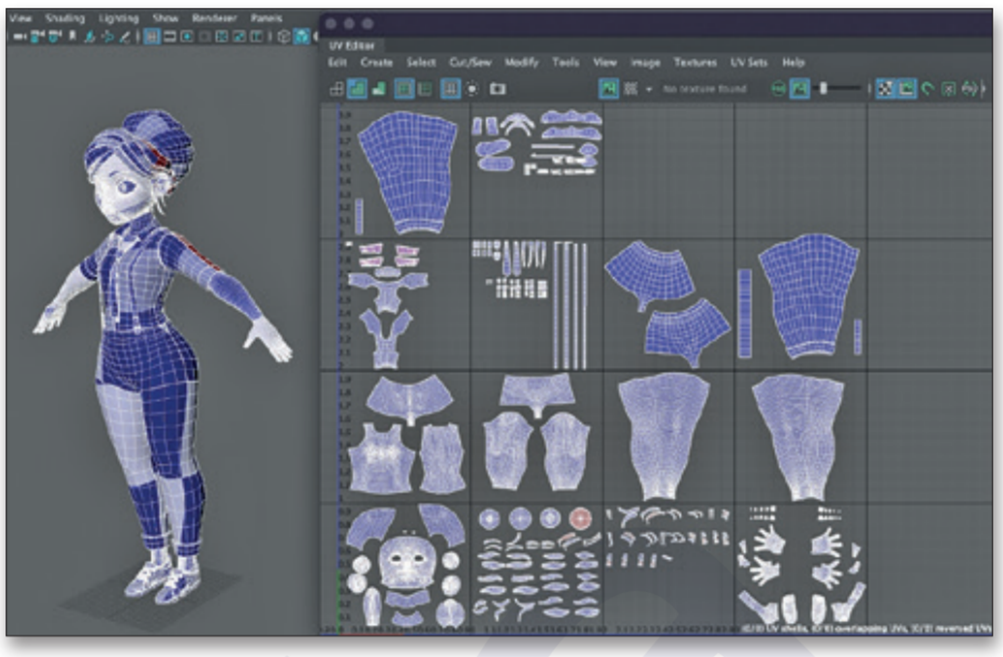


Для створення високополігональних моделей часто використовуються техніки:

- **Сабдівіжн** – згладжування сітки, щоб її вигляд був плавнішим
- **Цифровий скульптинг** – ліплення форми за допомогою віртуальних пензлів

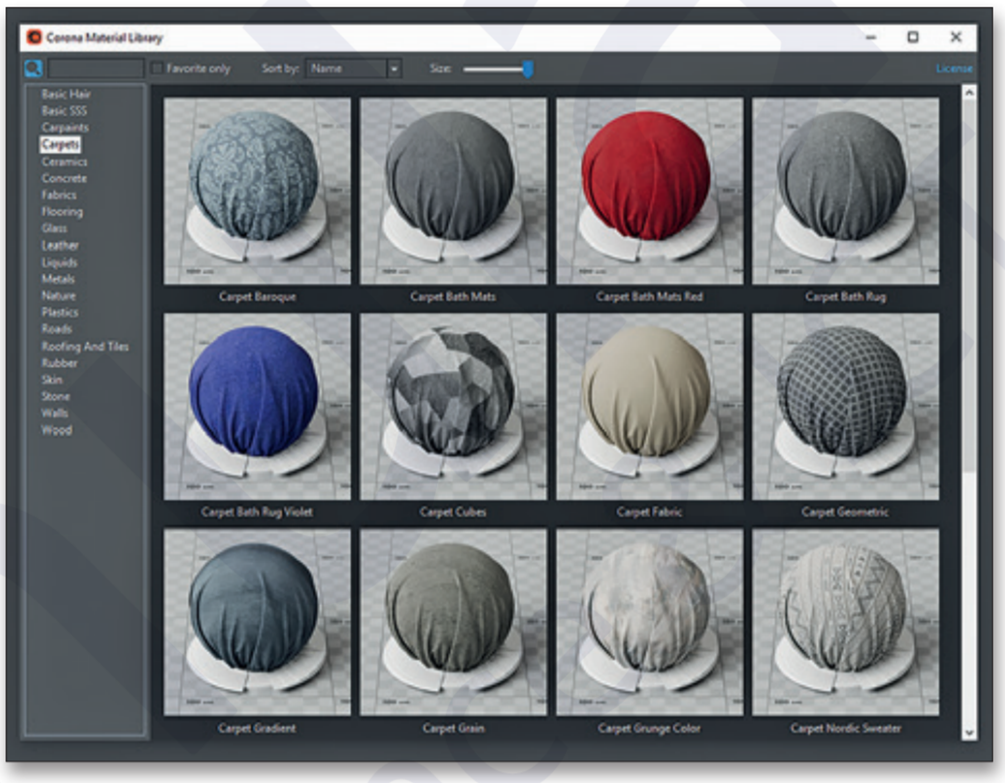
Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Коли форма об’єкта готова, настає черга «одягнути» його в **текстури**. Але перш ніж це зробити, варто створити **текстурну розгортку моделі (UV mapping)**. Уявіть, що 3D-модель розрізається та розкладається на площині, подібно до викрійки одягу. Це допомагає точно накласти двовимірне зображення (текстуру) на тривимірну поверхню. Правильно зроблена розгортка мінімізує спотворення текстур та забезпечує якісне відображення деталей.



Далі відбувається **розробка матеріалів**. Це процес визначення властивостей поверхні об’єкта, як-от кольору, блиску, шорсткості, прозорості, здатності відбивати світло та багато інших.

Сучасні програми уможливлюють створення складних і дуже реалістичних матеріалів. Для цього використовують різні текстури та карти.



Ці елементи разом створюють матеріал, який робить модель цікавою та реалістичною – ніби вона справжня.

Карти та текстури

Карта нормалей додає дрібні деталі (наприклад, зморшки чи тріщини) без збільшення кількості полігонів

Карта блиску показує, які частини об’єкта блищать більше, а які – менше

Кольорова текстура відповідає за основний вигляд поверхні

Роздільна здатність текстури – *це один з параметрів, який визначає якість і чіткість двовимірного зображення, що накладається на тривимірну модель.*

Шорсткість – *це характеристика поверхні, яка описує наявність нерівностей, виступів і западин, що відхиляються від ідеально гладкої форми.*

Щоб побачити створений об’єкт у всій красі, необхідно обрати та налаштувати візуалізатор (рендерер). **Візуалізатор** – це спеціальна програма або частина програми, яка перетворює тривимірну сцену в двовимірне зображення – тобто створює фінальну картинку, яку бачимо на екрані.



Існує багато різних візуалізаторів, і кожен має свої особливості. Одні створюють дуже реалістичне освітлення та тіні. Інші працюють швидше, але з меншою деталізацією. Деякі краще підходять для ігор, інші – для фільмів або архітектури.



Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Серед найвідоміших візуалізаторів – **V-Ray**, **Corona** та **Arnold**. Вибір залежить від того, наскільки реалістичне зображення потрібно, скільки часу є на рендеринг і які ресурси має комп'ютер.

Щоб модель мала красивий і реалістичний вигляд, потрібно правильно налаштувати освітлення. У 3D-сцені можна використовувати різні типи світла.

Типи світла

- **Точкове світло** – як лампочка, світить в усі боки
- **Спрямоване світло** – як ліхтарик, світить в одному напрямку
- **Площинне світло** – як світло з екрана, що світить з поверхні
- **Глобальне освітлення** – як світло з усього оточення (імітує розсіяне світло)

Правильне розташування джерел світла, їхня яскравість, колір і тіні можуть повністю змінити вигляд сцени та зробити її більш атмосферною.

Ще один важливий елемент – це камера. **Віртуальна камера** в 3D-програмі працює як справжня: вона має положення, кут огляду, фокусну відстань і глибину різкості. Завдяки цьому можна обрати найкращий ракурс, зробити акцент на важливих деталях і створити цікаву композицію.

Фінальний етап – створення композиції та запуск рендерингу. Розставте об'єкти, налаштуйте світло, оберіть ракурс камери – запустіть **рендеринг**. Це процес, коли програма обчислює всі налаштування і створює фінальне зображення. Результатом цього кропіткого та творчого процесу стає тривимірне зображення, яке оживає на екрані, готове вражати своєю деталізацією та реалістичністю. Саме воно покаже 3D-сцену у всій красі – з текстурами, світлом і тінями.

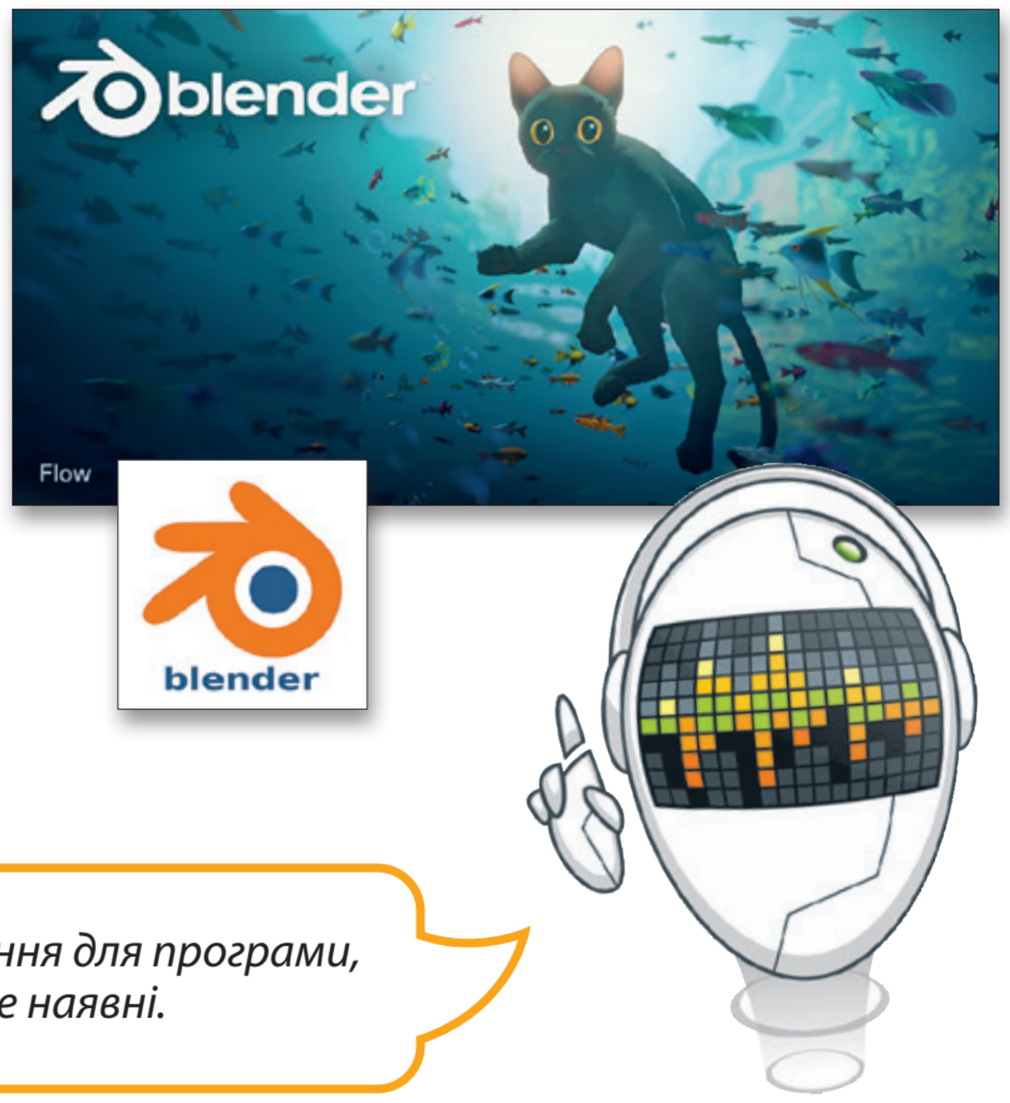
Програми для 3D-моделювання

У світі тривимірного моделювання існує багато програм, кожна з яких має свої особливості та можливості. Серед них можна виділити найпопулярніші інструменти, що допомагають створювати й оживляти віртуальні світи.

Blender

Blender – це справжній універсальний інструмент для творчості у 3D. Уявіть собі програму, в якій можна створювати об'ємні моделі, ліпити фігури, як із глини, створювати анімації, додавати спецефекти – і все це абсолютно безкоштовно.

Blender працює з різними операційними системами (Windows, macOS, Linux) і має величезну спільноту користувачів, які завжди готові допомогти. Його використовують і новачки, і професіонали, які створюють мультфільми, ігри та навіть фільми.



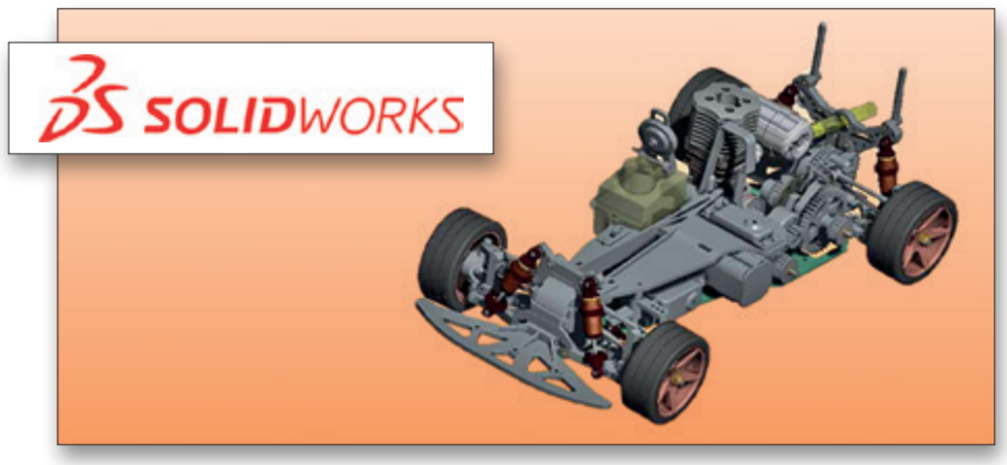
***Плагін** – це додатковий компонент або розширення для програми, який додає нові функції або покращує вже наявні.*

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

SolidWorks

SolidWorks – це програма, яку використовують інженери для створення точних 3D-моделей реальних об'єктів. Вона не просто малює красиві картинки, а уможливлює створення моделей з реальними розмірами, формами та властивостями – як у справжньому виробництві.

У SolidWorks можна спроєктувати деталі машин, механізми, побутову техніку та багато іншого. Програма допомагає конструкторам і дизайнерам перевірити, як працюватиме об'єкт ще до того, як його виготовлять у реальному житті.



Autodesk 3ds Max

Autodesk 3ds Max – це потужна програма для створення 3D-моделей, яка вже понад 20 років залишається однією з найпопулярніших серед професіоналів. Уявіть собі майстерню, де можна створювати круті спецефекти, будівлі, персонажів для ігор та навіть цілі віртуальні світи.

Однією з ключових особливостей 3ds Max є її універсальність. Вона має багато інструментів для моделювання: можна створювати точні форми з полігонів, плавні криві або гладкі поверхні. Програма постійно оновлюється і має велику кількість плагінів та доповнень, як платних, так і безкоштовних. Ці доповнення значно розширюють функціональність програми, додаючи нові інструменти, текстури та можливості для вирішення специфічних завдань.

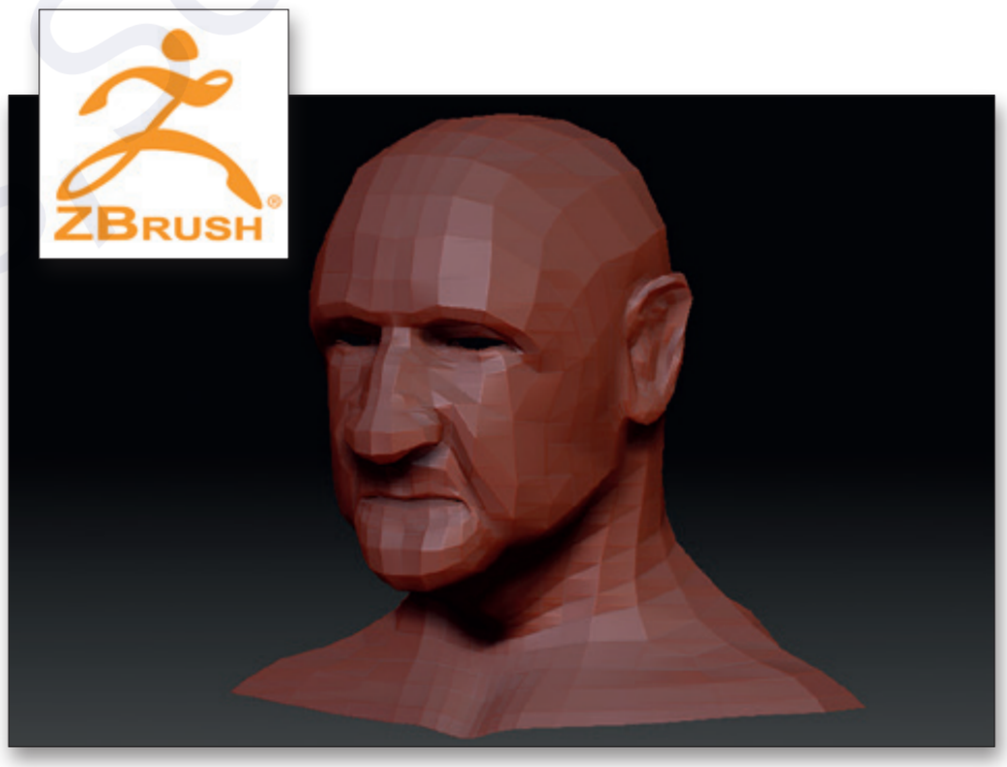


Цю програму використовують архітектори, дизайнери, аніматори та розробники ігор, бо вона дозволяє створювати дуже якісні та деталізовані проєкти.

ZBrush

ZBrush – це програма, яка дає змогу «ліпити» 3D-моделі, як із справжньої глини. Замість звичайних інструментів використовуються спеціальні пензлі, кожен з яких має свою «суперсилу»: один витягує форму, інший згладжує поверхню, а ще інший додає дрібні деталі, як-от зморшки чи текстуру шкіри.

Цю програму найчастіше використовують художники та дизайнери, які створюють фантастичних істот, героїв для ігор, реалістичні обличчя та скульптури. ZBrush допомагає зробити моделі дуже деталізованими, ніби вони справжні.



Autodesk Maya

Autodesk Maya – це потужна програма для створення 3D-світів, яку використовують у кіно, телебаченні та відеоіграх. Уявіть собі велику фабрику, де кожен «цех» відповідає за щось своє: моделювання, анімацію, симуляцію, рендеринг.

Maya дуже гнучка та має багато складних, але точних інструментів. Завдяки їм професіонали можуть створювати неймовірно реалістичні сцени, персонажів і вражаючі візуальні ефекти.

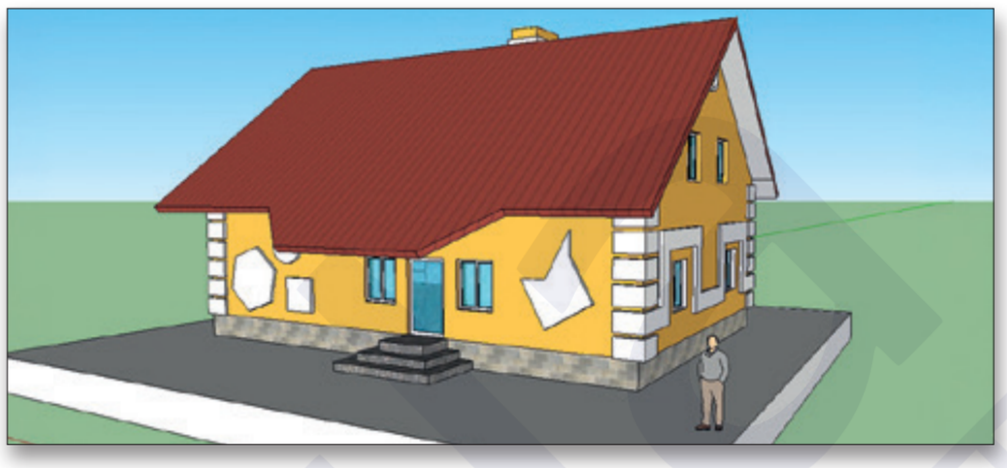


Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

SketchUp

SketchUp – проста та зручна програма для створення 3D-моделей будівель, кімнат і ландшафтів. Її дуже легко освоїти, тому вона чудово підходить для тих, хто тільки починає працювати з тривимірним моделюванням або хоче швидко візуалізувати свою ідею.

SketchUp особливо популярний серед архітекторів, дизайнерів інтер'єрів та ландшафтних дизайнерів, адже з ним легко та швидко створювати ескізи будівель, приміщень та ландшафтів, ніби ви малюєте їх олівцем у тривимірному просторі. З програмою можна працювати і на комп'ютері, і безпосередньо в браузері.



Наведена нижче таблиця допоможе швидко зорієнтуватися у програмах для створення тривимірних об'єктів. Це своєрідна шпаргалка, завдяки якій можна порівняти їхні основні можливості та обрати той інструмент, який найкраще відповідає вашим творчим задумкам та потребам – чи то створення персонажів, будівель, техніки або цілих віртуальних світів.

Програми для створення тривимірних об'єктів

Програма	Наявність безкоштовної версії	Підтримка моделювання NURBS	Підтримка скульптингу	Підтримка полігонального моделювання
Blender	+	+	+	+
SketchUp	+ (вебверсія, SketchUp Free)	–	–	+
Autodesk Maya	– (30-денна пробна версія)	+	+	+
Autodesk 3ds Max	– (30-денна пробна версія)	+	+	+
SolidWorks	– (є освітні ліцензії)	+	–	+
ZBrush	–	–	+	+ (не основний метод)

Практичне завдання

Завантажте файл-заготовку зі сторінки інтернет-підтримки. Виконайте запропоноване завдання.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Тема 4
Графічний редактор Blender

Blender – не просто програма, це творча лабораторія, де ідеї оживають у тривимірному просторі. Вона універсальна і дає змогу реалізувати різноманітні задуми – від створення персонажів та візуальних ефектів до дизайну інтер'єрів та віртуальних пейзажів.

Що можна робити за допомогою Blender:

3D-моделювання
Створювати складні деталізовані моделі, від простих об'єктів до персонажів і сцен

Скульптурування
«Ліпити» органічні моделі, використовуючи інструменти для скульптурування та текстурування

Анімація
Створювати складні анімації, включаючи анімацію персонажів та симуляції частинок

Рендеринг
Генерувати високоякісні статичні зображення та відео

Відеомонтаж та компонування
Об'єднувати відео та зображення, додавати спеціальні ефекти та керувати кольором

Створення ігор
Моделювання ігрових активів, які потім використовуються у зовнішніх рушіях.

***Blender** не є графічним редактором у традиційному розумінні, а є повноцінним пакетом для створення тривимірної комп'ютерної графіки.*



Ключові особливості Blender

Безкоштовний та з відкритим кодом	Універсальність	Активний розвиток	Розширення функціоналу
Використовувати програму можна безкоштовно, а її код відкритий для всіх.	Надає повний набір інструментів для всього процесу створення 3D-графіки.	Програма постійно оновлюється та вдосконалюється, що сприяє її популярності.	Можливості програми можуть бути розширені за допомогою застосунків, створених користувачами.

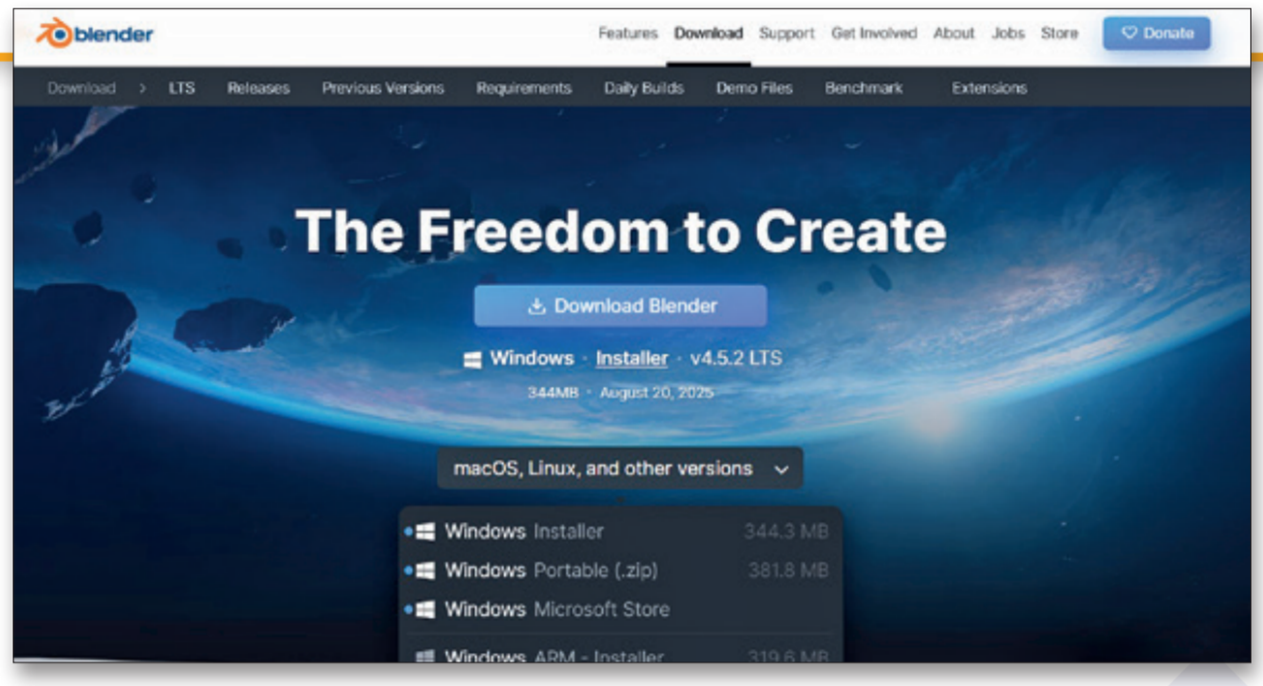
ДОВІДКА

Графічний редактор **Blender** був створений голландським арт-директором Тон Розендалем у 1995 році для використання в 3D-анімаційній студії «NeoGeo».

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Завантаження програми Blender

1. Перейдіть за посиланням blender.org/download/ та завантажте інсталятор **Blender**.
2. Запустіть програму встановлення на вашому пристрої.



Для застарілих систем та комп'ютерів можна завантажити попередні версії.



Налаштування інтерфейсу

Після запуску застосунку в першому діалоговому вікні буде можливість налаштувати стиль користування програмою. Це спрощує перехід, якщо раніше користувалися іншим тривимірним редактором.

Натискаємо **ОК**. Після бачимо **вікно привітання (Splash Screen)**.

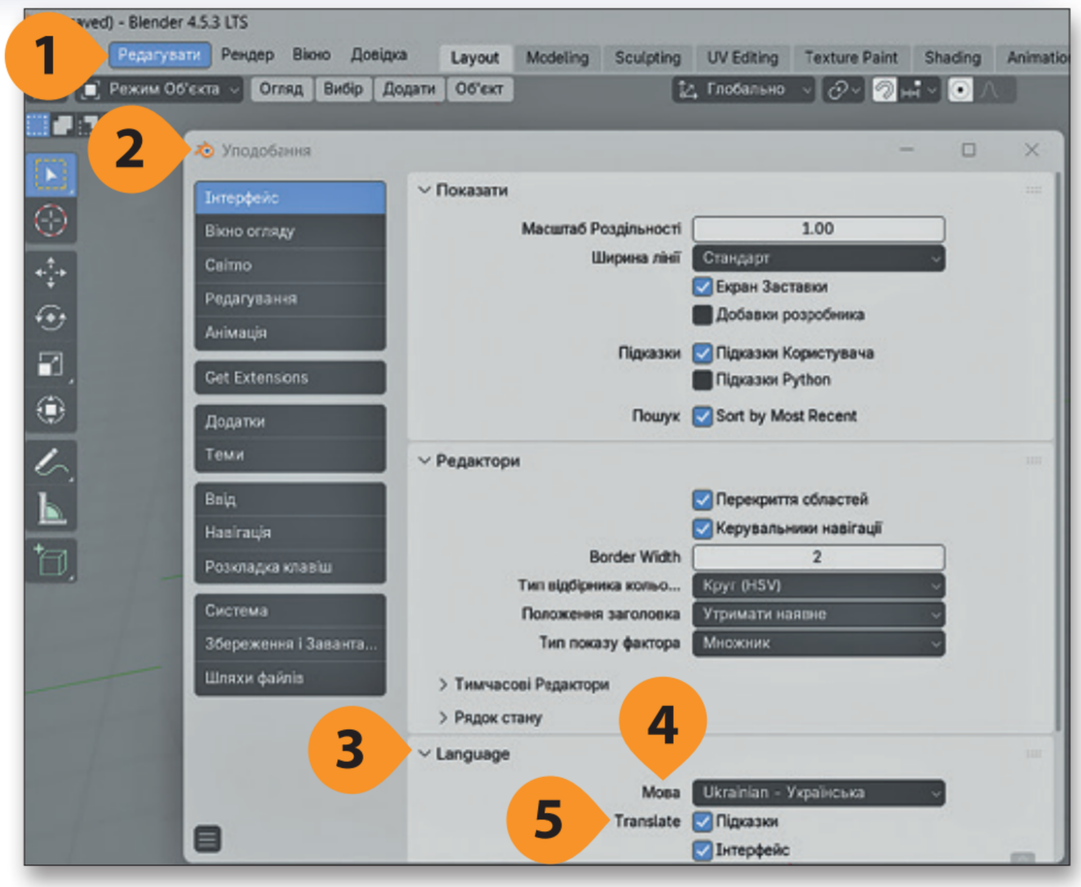
Вікно привітання (Splash Screen) складається з двох рядків:

- рядок створення **Нового документа** (1);
- рядок посилань на **Недавні матеріали** (якщо вже працювали в застосунку, то буде посилання на останні документи) (2).

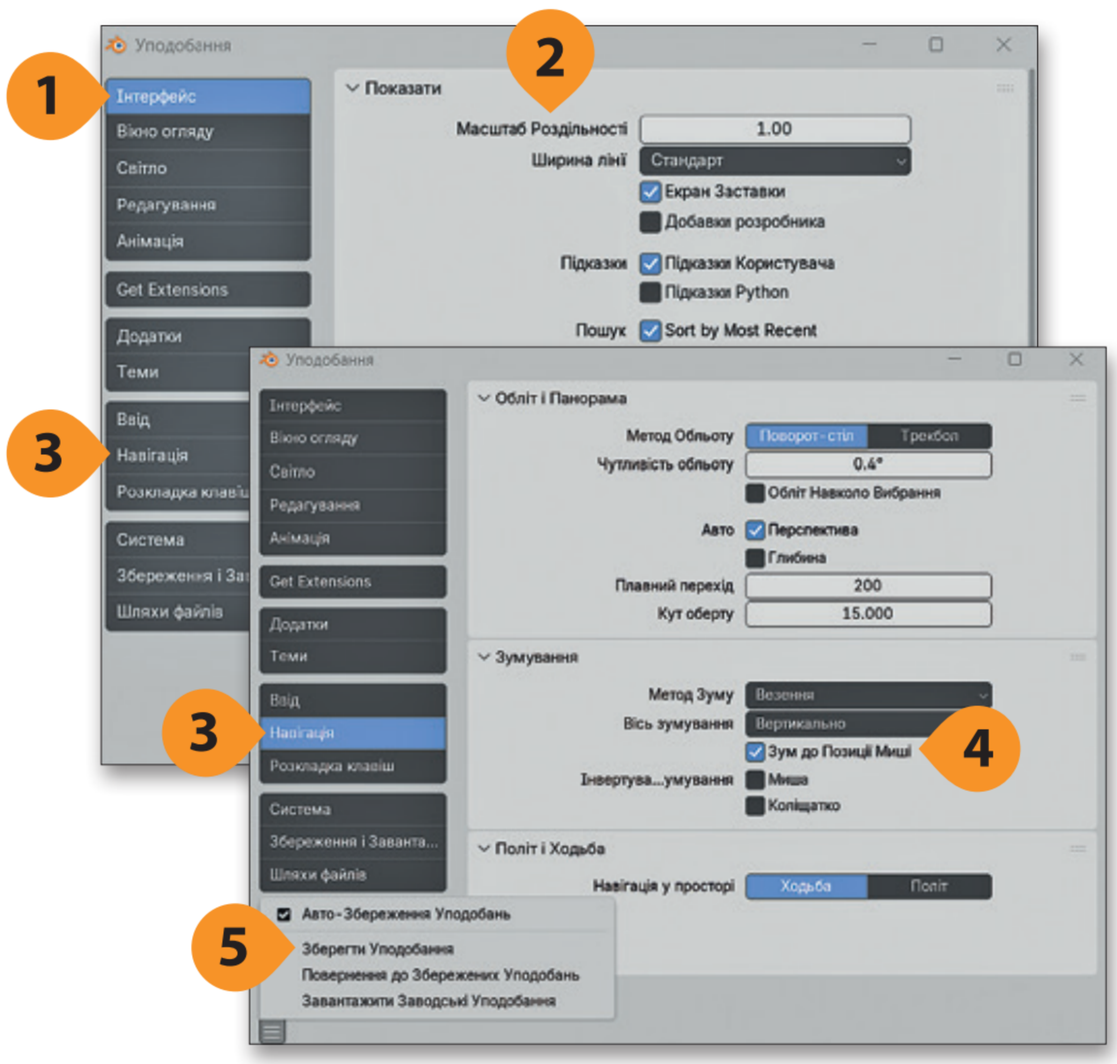
Щоб змінити мову інтерфейсу:

- натисніть головне меню **Редагувати** (1);
- оберіть пункт **Уподобання** (2);
- у вікні налаштувань, що відкриється, знайдіть меню **Мова (Language)** (3).
- У переліку **Мова** (4) оберіть **українську**;
- установіть усі 2 прапорці під **Переліком мов** у пункті **Translate** (Перекласти) (5): **Підказки** (переклад підказок); **Інтерфейс** (переклад інтерфейсу).





Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.



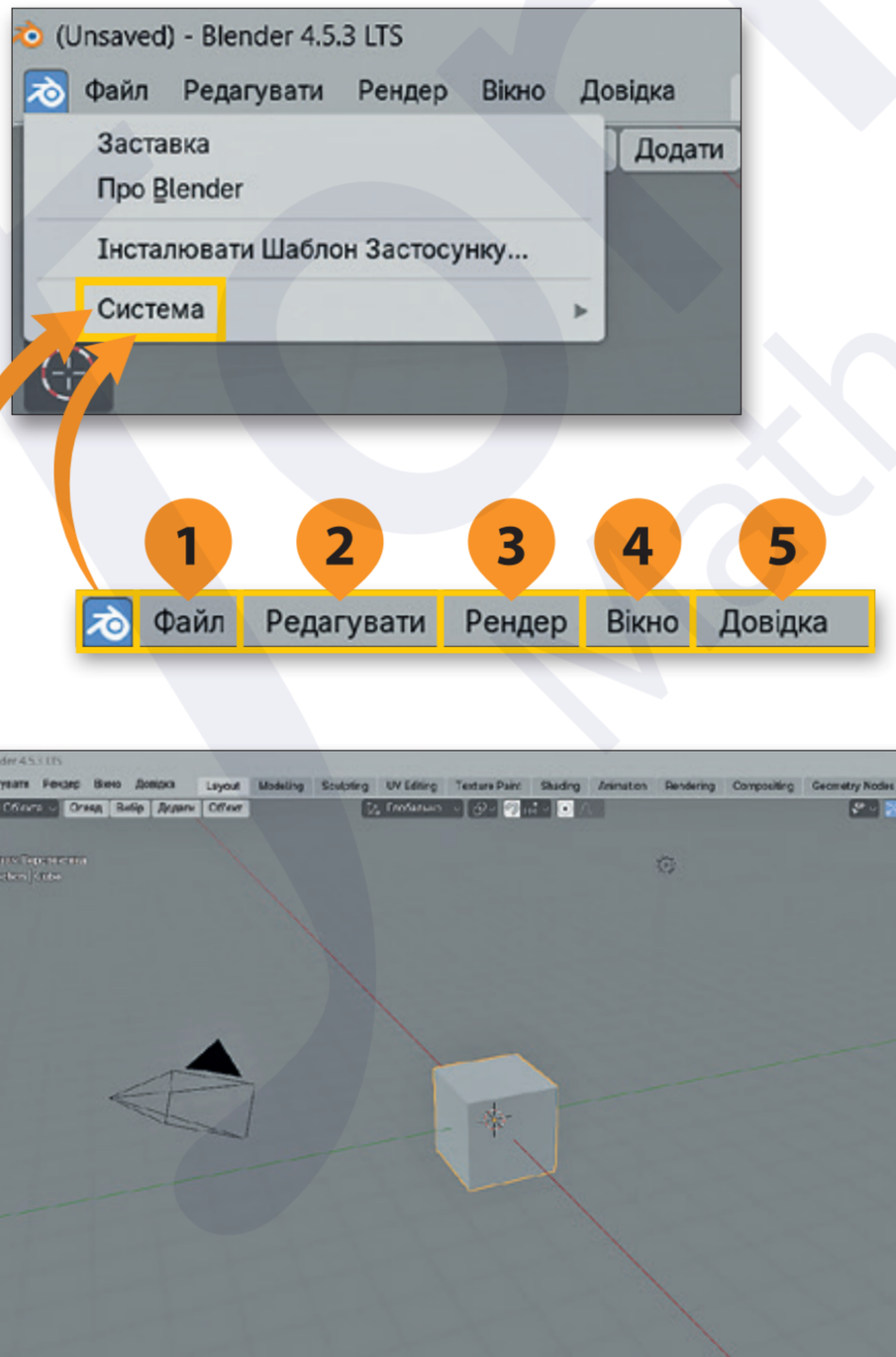
Щоб налаштувати інтерфейс:

- з метою зменшення напруги на очі зробіть більшими елементи **інтерфейсу** (1) завдяки пункту **Масштаб роздільності** (2);
- у пункті **Навігація** (3) встановіть прапорець **Зум** (4) до позиції миші для спрощення процесу навігації;
- після зміни налаштувань збережіть **Уподобання**, користуючись бутерброд-меню, розташованим у нижньому лівому куті палітри (5).

Огляд інтерфейсу

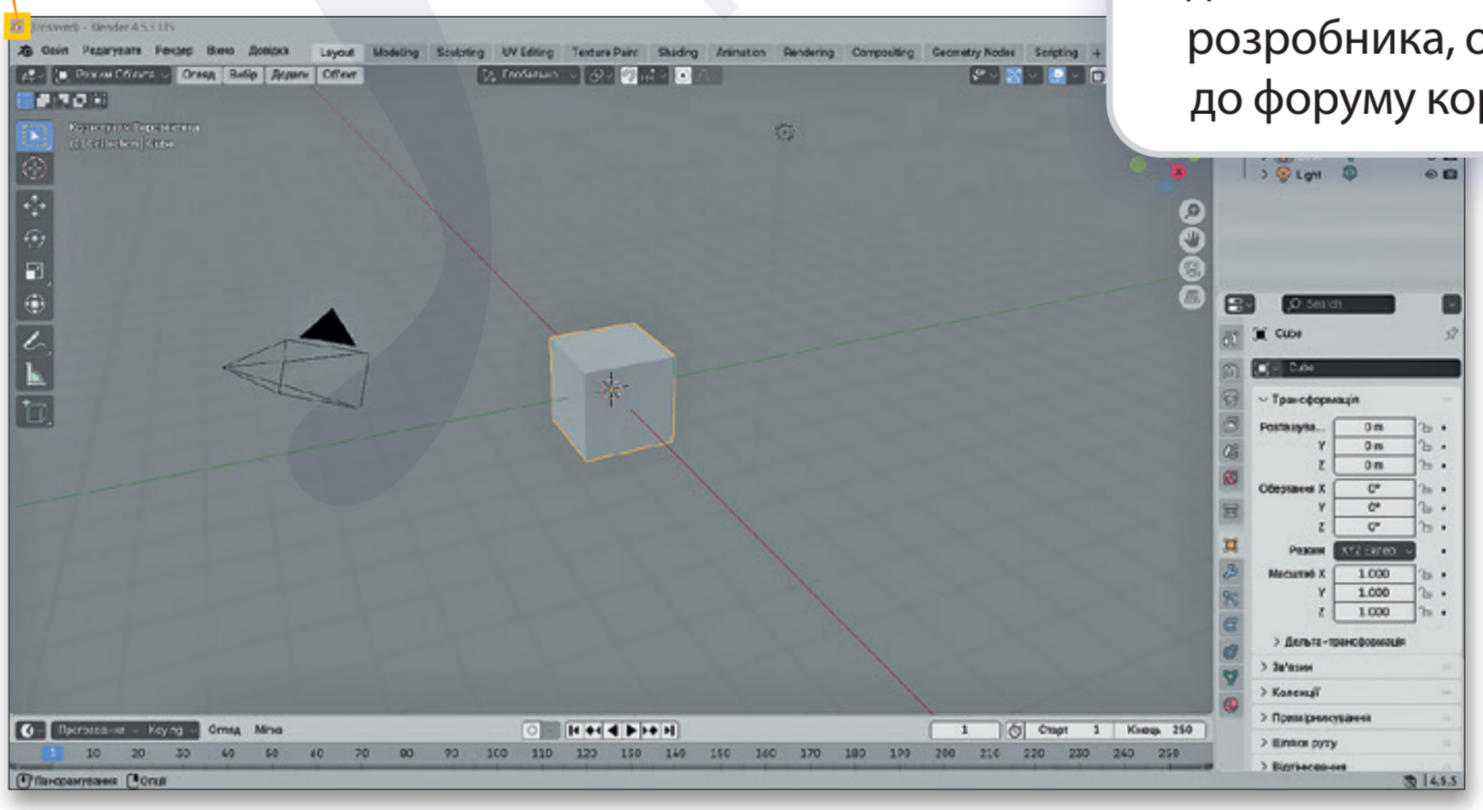
Головне меню

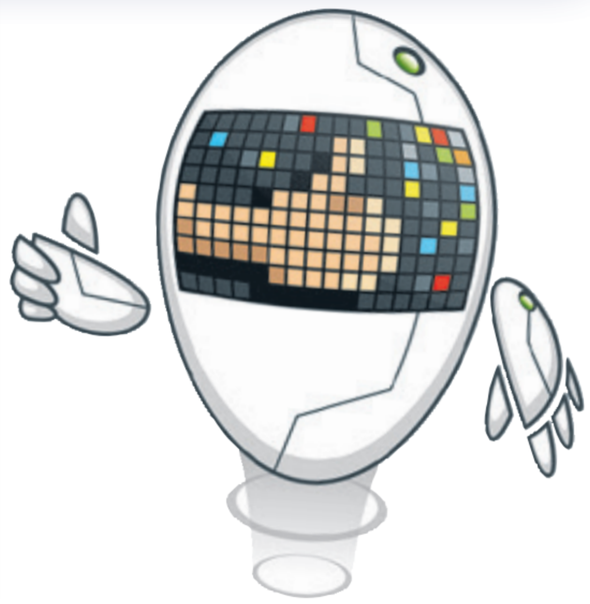
Згори вікна **Blender** під заголовком розташоване **Головне меню**, яке складається з 5 пунктів. Перед ними є значок застосунку, який відкриває меню виклику початкового екрана завантаження (вікна привітання).



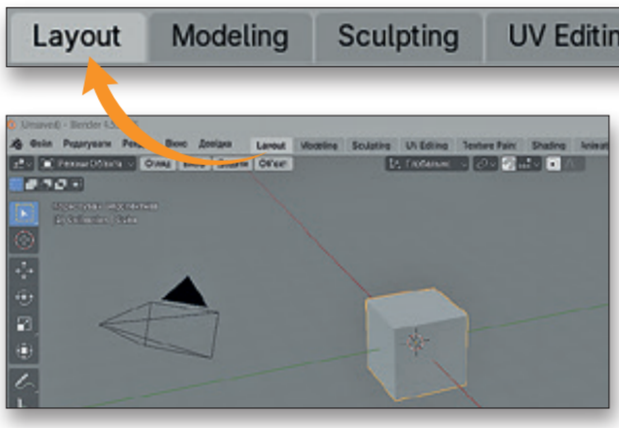
Головне меню містить:

- меню **Файл** (1), яке дає змогу зберігати документи, робити експорт та імпорт документів в інші формати;
- меню **Редагувати** (2), яке дає доступ до налаштувань програмного середовища, відміни дії, пошуку, перейменування та кілька команд редагування;
- меню **Рендер** (3), яке запускає процес візуалізації або з камер, або з вікна редагування;
- меню **Вікно** (4), яке дає змогу керувати вікнами, робити скриншоти, відображати рядок стану;
- меню **Довідка** (5), яке дає змогу перейти до довідкової системи, до системи навчання на сайті розробника, отримати доступ до форуму користувачів тощо.





Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.



Робочі середовища

Праворуч від **Головного меню** розташована смуга з перемикачами робочих середовищ. Загалом там 11 перемикачів та знак «+», завдяки якому можна власноруч створити нове середовище, налаштоване певним чином.

Призначення середовищ, якими будемо користуватися на практиці

Layout – загальне середовище, яке відкривається за стандартним налаштуванням, здебільшого робота в ньому і буде здійснюватися

Modeling – середовище для полігонального моделювання – базового методу моделювання, містить панель з інструментами моделювання

Sculpting – середовище для художнього моделювання, містить панелі з пензлями редагування та відповідними палітрами їх налаштування

UV Editing – середовище розробки текстурних розгорток, потрібне для коректного налаштування матеріалу на об'єкті

Texture Paint – малювання на растровій карті

Shading – редактор матеріалів

Видовий екран

Видовий екран є найбільш важливою частиною середовища, весь процес моделювання відбувається саме в цій палітрі. Згори палітри розташована смуга з базовим інструментарієм. Розгляньмо його.

Видовий екран містить:

- **Перемикач змісту палітри** (1);
- **Перемикач режимів роботи** (2);
- **Меню палітри** (налаштування, інструментарій виділення, бібліотеки об'єктів, команди дій з об'єктами) (3);
- **Перемикач центрів координат** (4);
- **Перемикач центру трансформацій** (5);
- **Налаштування прив'язок** (6);
- **Режим симетричного моделювання** (м'якого) (7);
- **Меню налаштування відображення** (8);
- **Перемикання у напівпрозорий режим** (9);
- **Режими візуалізації** у видовому вікні (10).

Ліва частина екрана містить **Панель інструментів**, сховати або відкрити її можна, якщо натиснути клавішу з латинською літерою T на клавіатурі.

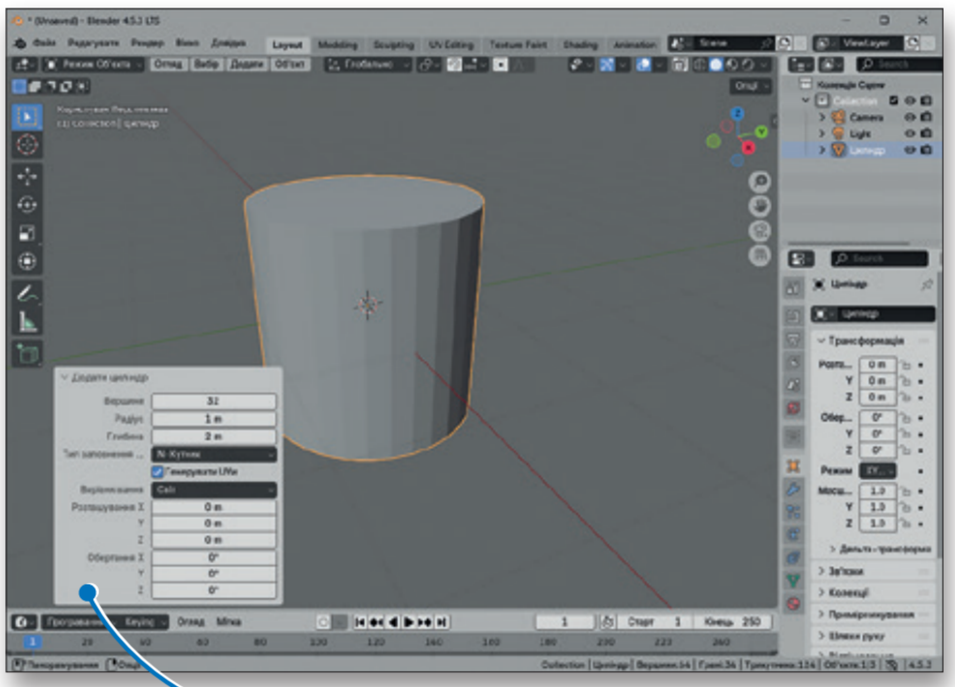


Права частина екрана містить **Палітру властивостей**, сховати або відкрити її можна, якщо натиснути клавішу з латинською літерою N на клавіатурі.

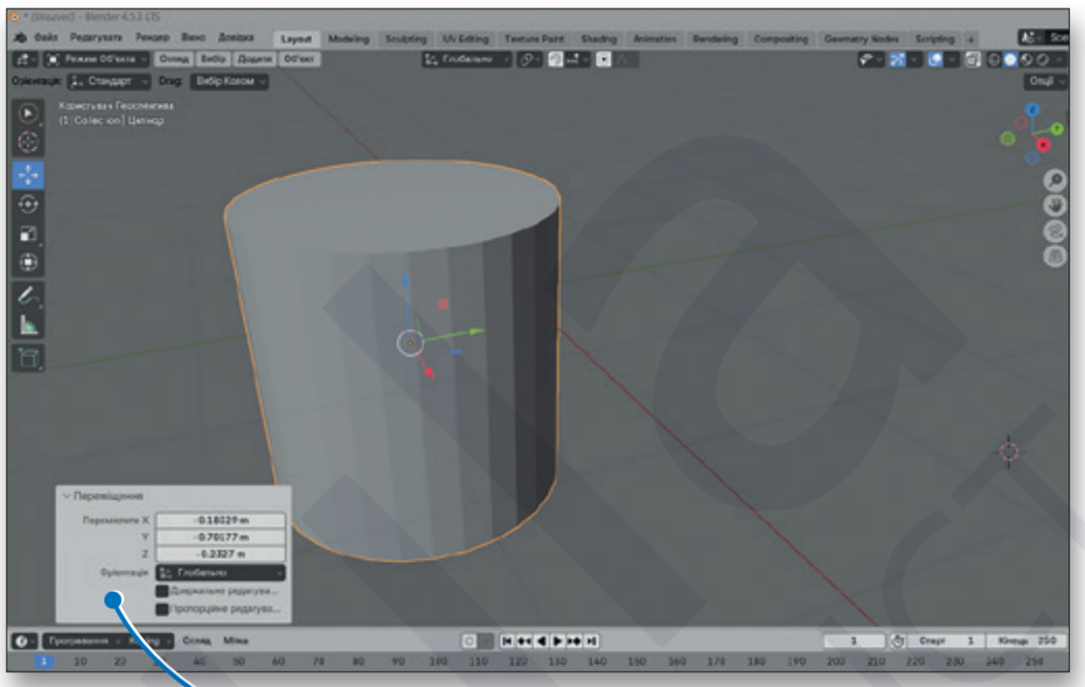
Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Палітра останньої дії

Палітра останньої дії з'являється унизу при виконанні більшості дій чи команд. Наприклад, під час додавання нового об'єкта на сцену в **Палітрі останньої дії** можна налаштовувати параметри об'єкта. Або при користуванні інструментами трансформації, як-от інструментом пересування, у **Палітрі** відображається інформація щодо можливості коригування позиції об'єкта на сцені.



Вигляд **Палітри** після додавання циліндра



Вигляд **Палітри** після пересування циліндра

Базові вікна

У нижній частині інтерфейсу типово розташовано вікно із **Шкалою часу** для налаштування анімацій та симуляцій. Вона містить 250 кадрів.



Шкала часу містить 250 кадрів



Праворуч від вікна **Видового екрана** розташований **Оглядач сцени**, або **Структуратор**. Він містить перелік усіх об'єктів у сцені.

Сцена – це тривимірний документ.

Структуратор дає змогу також керувати відображенням об'єктів на сцені та виводом їх на візуалізацію. Це здійснюється відповідно вмиканням та вимиканням **знака ока** (5) навпроти назви об'єкта та перед знаком фотоапарата.

Оглядач сцени, або Структуратор містить:

- перелік усіх об'єктів у **Сцені**, які розташовані у **Колекції** (порожні теки) для зручності (1);
- стандартно під час створення у **Сцені** завжди буде 3 об'єкти: **Камера (Camera)** (2), **Джерело освітлення (Light)** (3) та **Куб (Cube)** (4).

Якщо під час створення складної композиції розташувати об'єкти в різних колекціях, то можна вмикати та вимикати усі об'єкти, розташовані в колекції, або швидко їх виділяти.



Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.



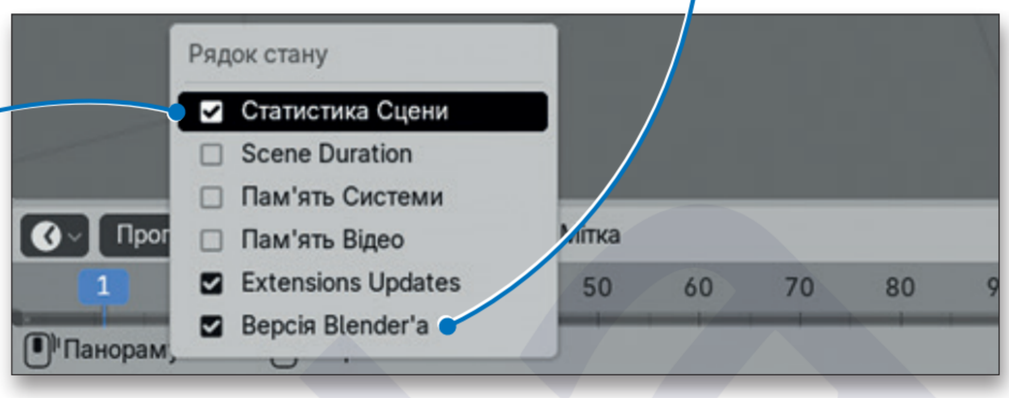
Рядок стану

Рядок стану містить підказки щодо активних у цей момент інструментів та методів їх використання.

Для налаштувань відображених даних на рядку стану, потрібно клацнути правою кнопкою миші та обрати потрібне.

Також можна увімкнути на **Рядку стану** відображення статистики.

У правому куті **Рядка стану** висвітлюється версія програмного продукту.



Вікно Властивостей

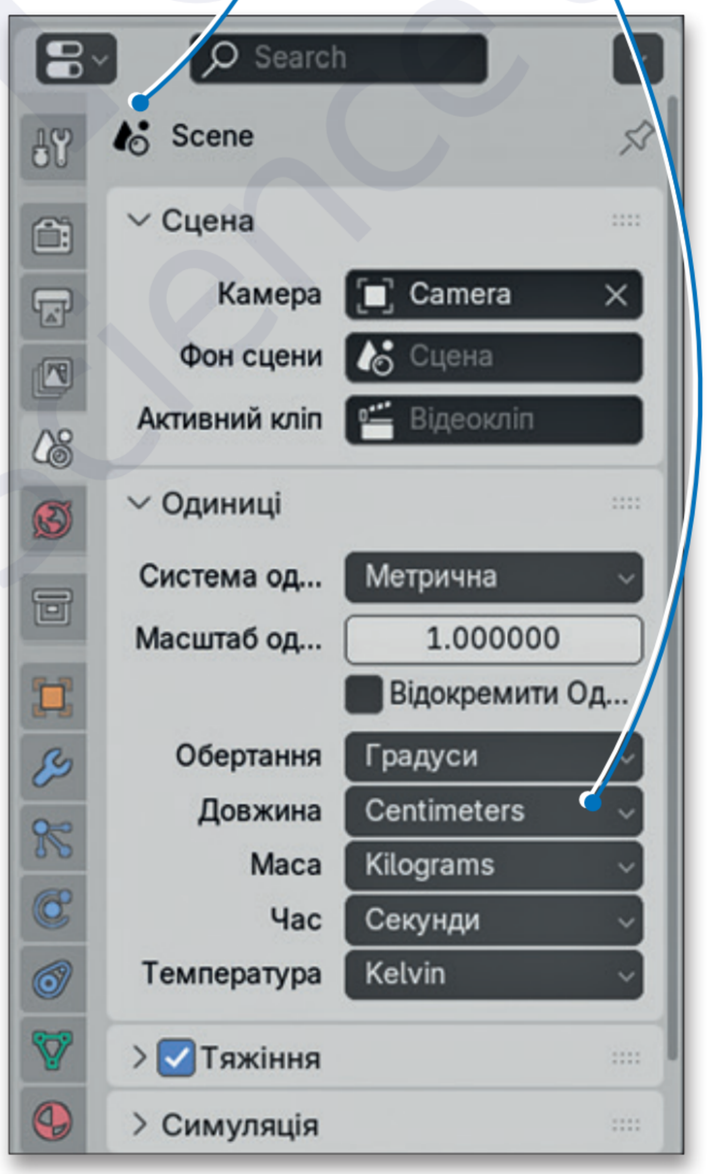
Під вікном структуратора розташовано **вікно властивостей**. У процесі роботи можна користуватися налаштуванням різних інтерфейсів, команд або палітрою властивостей.



Палітра вікна властивостей містить:

- **інструмент** (1);
- **візуалізацію** (2);
- **виведення** (3);
- **шари** (4);
- **сцену** (5);
- **світло** (6);
- **об’єкт** (7);
- **модифікатори** (8);
- **частки** (9);
- **фізику** (10);
- **обмеження** (11);
- **дані** (12);
- **матеріал** (13);
- **текстуру** (14).

Перш ніж почати роботу у **Blender**, потрібно налаштувати одиниці виміру програмного середовища. Для цього у палітрі необхідно обрати вкладку **Властивостей сцени**, іконка якої схожа на крапельку. Одиницями довжини варто встановити сантиметри.



ІСТОРІЯ

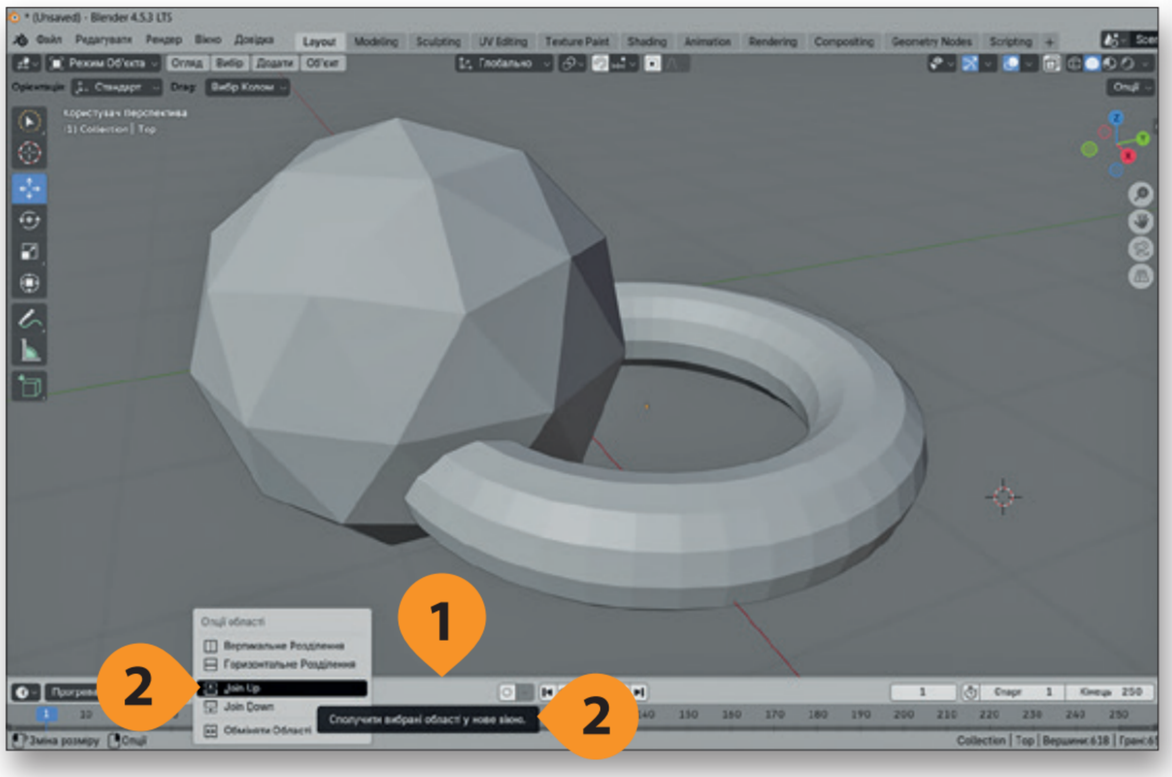
Оскільки **Blender** був створений голландською компанією **NeoGeo** для внутрішнього використання, то багато ранніх назв могли відображати їхній внутрішній стиль або переклад з нідерландської. Пізніше, коли **Blender** став відкритим проєктом, термінологія еволюціонувала, щоб відповідати загальноприйнятим стандартам індустрії, але деякі «оригінальні» назви могли залишитися. Отже, більшість назв у **Blender** є результатом поєднання усталеної термінології комп'ютерної графіки, математичних концепцій та власної еволюції програмного забезпечення, що прагне бути інтуїтивно зрозумілим для користувачів.



Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

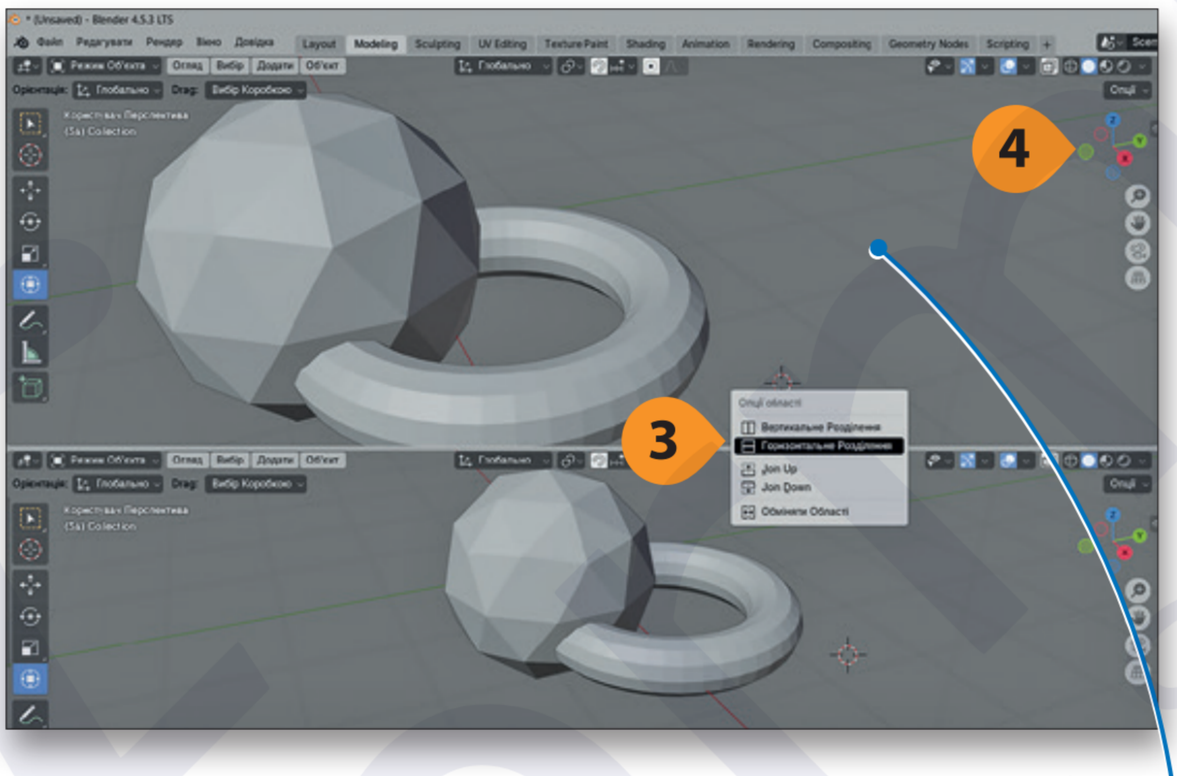
Налаштування вікон в інтерфейсі

У **Blender** є можливість власноруч створювати бажану кількість вікон та налаштовувати потрібні інструменти для комфортної роботи в середовищі.



Щоб сполучити чи розділити вікна у робочому середовищі:

- клацніть правою клавішею миші на межу між вікнами (1), це прибере палітру з **Рядком стану** і відкриє палітру **Опції області**;
- у відкритій палітрі **Опції області** оберіть пункт **Сполучити області** (2) та вкажіть на вікно, яке треба прибрати;
- у відкритій палітрі **Опції області** оберіть **Розділення горизонтальне** (3) та вкажіть на вікно, яке треба розділити.



Щоб отримати 4 вікна моделювання,
натисніть на клавіатурі одночасно клавіші **Shift** + **Ctrl** + **Alt** + **Q**

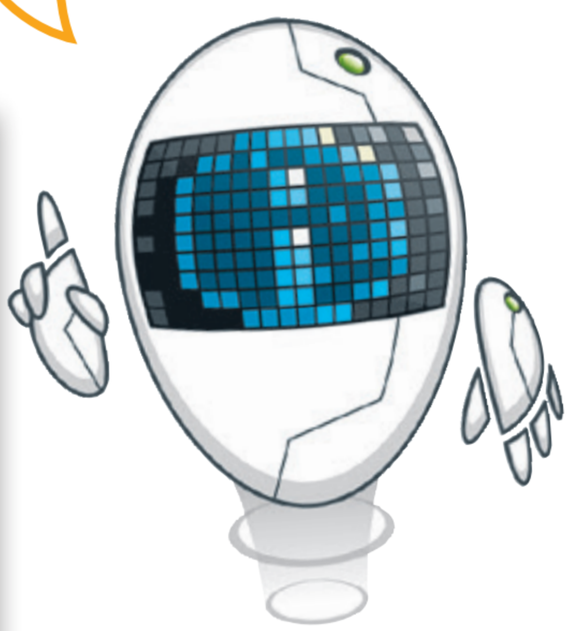
Щоб розширити будь-яке вікно на весь екран,
натисніть на клавіатурі одночасно клавіші **Ctrl** + **Alt** + **Пробіл**

Навігація у Видовому екрані

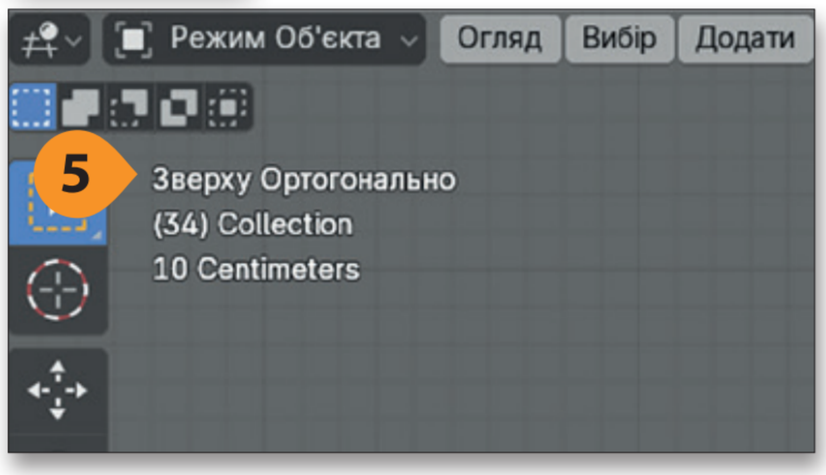
Навігацію в **Blender** можна робити за **Іменованими виглядами**.

Щоб змінити кут огляду, можна натискати на **Різнокольоровий трипод** у верхньому правому куті видового екрана (4). Наприклад, натиснувши на вісь (Z), можна отримати вигляд згори.





У верхньому лівому куті палітри відображається назва вигляду. Це може бути звичайний користувацький вигляд або іменований (5).



Також для навігації можна використовувати інструменти, що містяться під триподом з осями. Це інструменти ***Масштабування***, ***Панорамування***, ***Вигляду з камери*** *та* ***Перемикач до режиму ортогонального***.

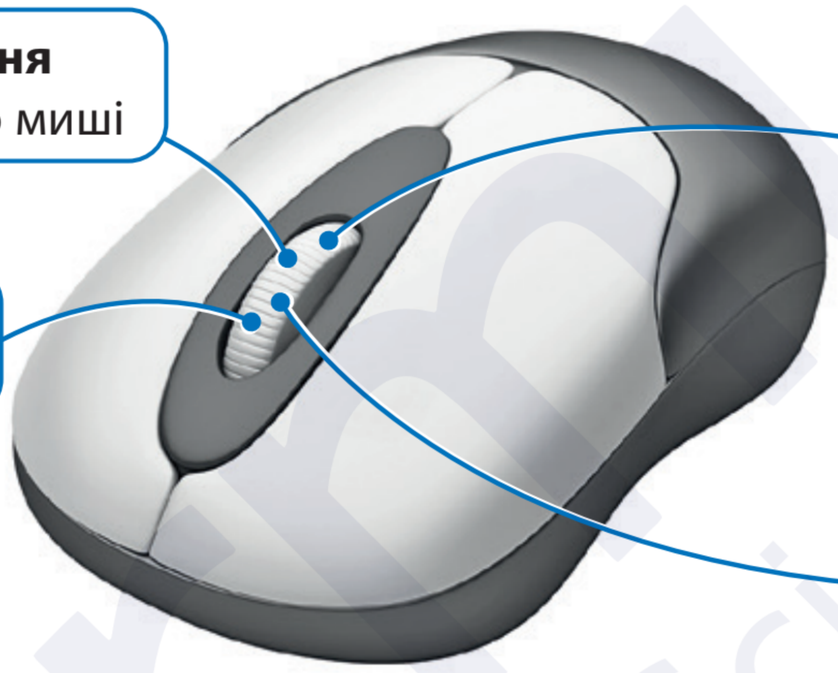
Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.



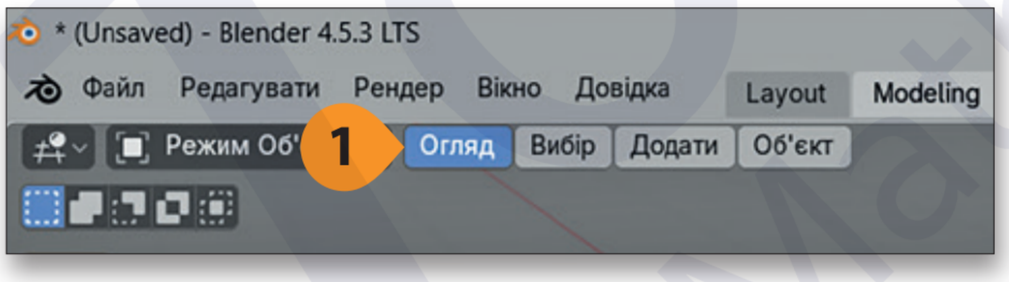
Інструменти для навігації:

- **Масштабування** – зміна ступеня наближення або віддалення від сцени (1);
- **Панорамування** – переміщення всього 3D-вигляду (вікна перегляду) в площині екрана (вгору, вниз, вліво або вправо) без зміни кута огляду (2);
- **Вигляд з камери** перемикає вигляд у 3D-вікні перегляду на точний вигляд, який бачить активна камера сцени (3);
- **Перемикач між перспективною та ортогональною проєкціями** – зміна типу проєкції, яка використовується для відображення 3D-сцени (4).

Бажано під час навігації користуватися мишею

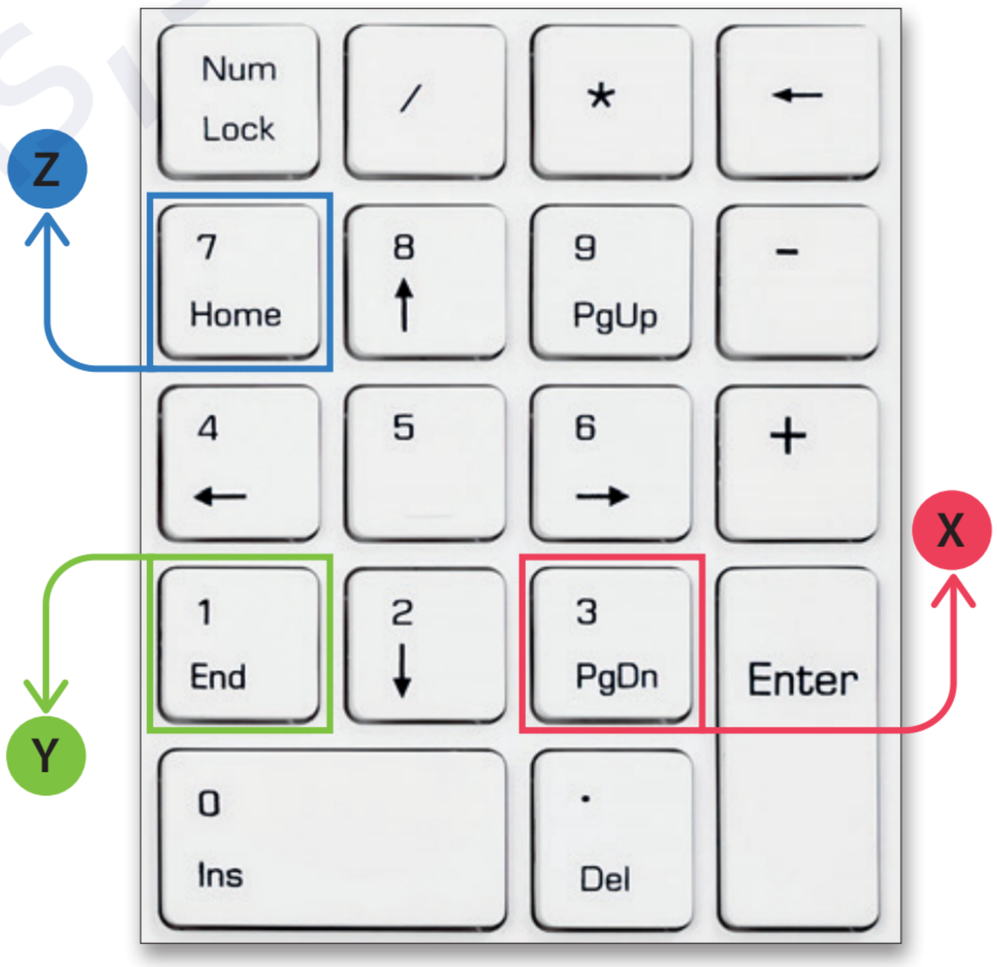


Також перемикатися між іменованими виглядами можна за допомогою меню **Огляд** (1).



Крім цього, наявна можливість користуватись клавішами на додатковій цифровій клавіатурі **(NumPad)**.

Подвійним натисканням на цифрові клавіші на додатковій цифровій клавіатурі можна перемикатися на вигляди з протилежних боків (наприклад, з праворуч на ліворуч).



ДОВІДКА

Якщо ноутбук не має вбудованої додаткової цифрової клавіатури **NumPad**, ви можете придбати зовнішню NumPad-клавіатуру. Це стандартне та дуже поширене рішення для 3D-художників та інших користувачів, яким потрібне швидке введення цифр або, як у випадку з Blender, швидка навігація.

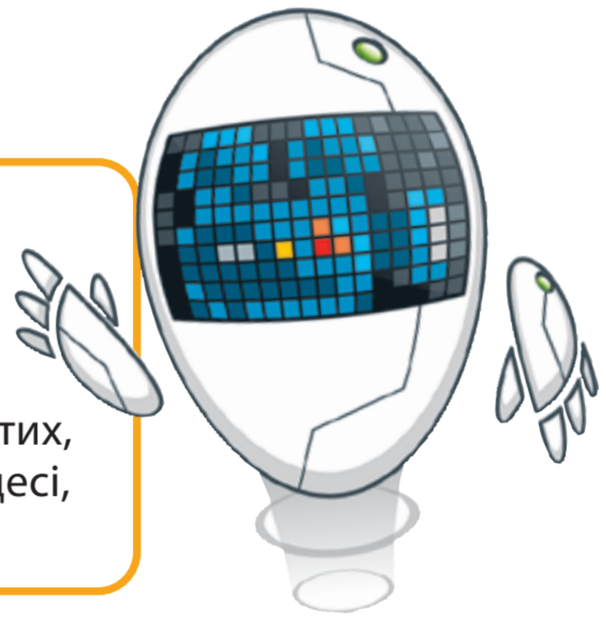
Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Навігація за допомогою клавіш на додатковій цифровій клавіатурі (NumPad)

Клавіша NumPad	Призначення	Опис
1	**Вигляд спереду**	Перемикає на ортогональний вигляд спереду
Ctrl + 1	**Вигляд ззаду**	Перемикає на ортогональний вигляд ззаду
3	**Вигляд справа**	Перемикає на ортогональний вигляд справа
Ctrl + 3	**Вигляд зліва**	Перемикає на ортогональний вигляд зліва
7	**Вигляд зверху**	Перемикає на ортогональний вигляд зверху
Ctrl + 7	**Вигляд знизу**	Перемикає на ортогональний вигляд знизу
9	**Обернути вигляд**	Обертає поточний вигляд на 180 градусів (змінює ракурс на протилежний)
5	**Перемикач проєкції**	Перемикає між перспективним та ортогональним оглядом
0	**Вигляд з камери**	Перемикає на вигляд активної камери
. *(крапка)*	**Зумувати до виділеного**	Центрує вигляд навколо виділеного об'єкта або елемента геометрії.
/ *(ділення)*	**Локальний вигляд**	Ізолює вибраний об'єкт, приховуючи все інше на сцені, що спрощує фокусування на деталях
+ -	**Зумування**	Збільшення та зменшення масштабу (приблизно як прокрутка коліщатка миші)

ДОВІДКА

Опанування гарячих клавіш є ознакою професійного рівня роботи в Blender. Це дозволяє виконувати складні операції з легкістю та швидкістю, що необхідні для реалізації ваших проєктів. Використовуючи клавіатуру, ви буквально «розмовляєте» з програмою. Почніть із вивчення найбільш вживаних клавіш для навігації (наприклад тих, що представлені в таблиці). Це дозволяє зосередитися на творчому процесі, мінімізуючи час на пошук функцій у меню.



Практичне завдання

Завантажте файл-заготовку зі сторінки інтернет-підтримки. Виконайте запропоноване завдання.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Тема 5
Об’єктне моделювання у Blender

Об’єктне моделювання є першим і найважливішим кроком у роботі з Blender, оскільки саме воно визначає форму та структуру майбутнього 3D-актива. Після моделювання об’єкт зазвичай проходить етапи текстурування, освітлення та рендерингу.



Об’єктне моделювання у Blender (та інших програмах 3D-графіки) *– це процес створення, маніпулювання та модифікації тривимірних об’єктів (моделей), які складаються з геометричних елементів (вершин, ребер і граней).*

Ключові режими та елементи у Blender

Ключові режими та елементи у Blender

Режим об’єкта (Object mode)	Режим редагування (Edit mode)
Це базовий режим роботи, який дає змогу маніпулювати об’єктами як єдиним цілим, а не їхньою внутрішньою геометрією. Також це основний режим для організації сцени, розміщення об’єктів та застосування до них загальних трансформацій. Завдяки цьому ви можете керувати загальними властивостями та положенням об’єктів у 3D-просторі.	Це ключовий режим для детального об’єктного моделювання, який допомагає вам змінювати внутрішню геометрію об’єкта. Режим редагування дає вам контроль над його основними будівельними блоками: вершинами, ребрами та гранями. Він використовується для того, щоб взяти базову форму (наприклад, куб або сферу) і «виліпити» з неї складну, унікальну модель, детально контролюючи кожен її аспект.

Інструменти трансформації

Blender пропонує три основних інструменти трансформації: **Переміщення (Move)**, **Масштабування (Scale)** та **Обертання (Rotate)**.

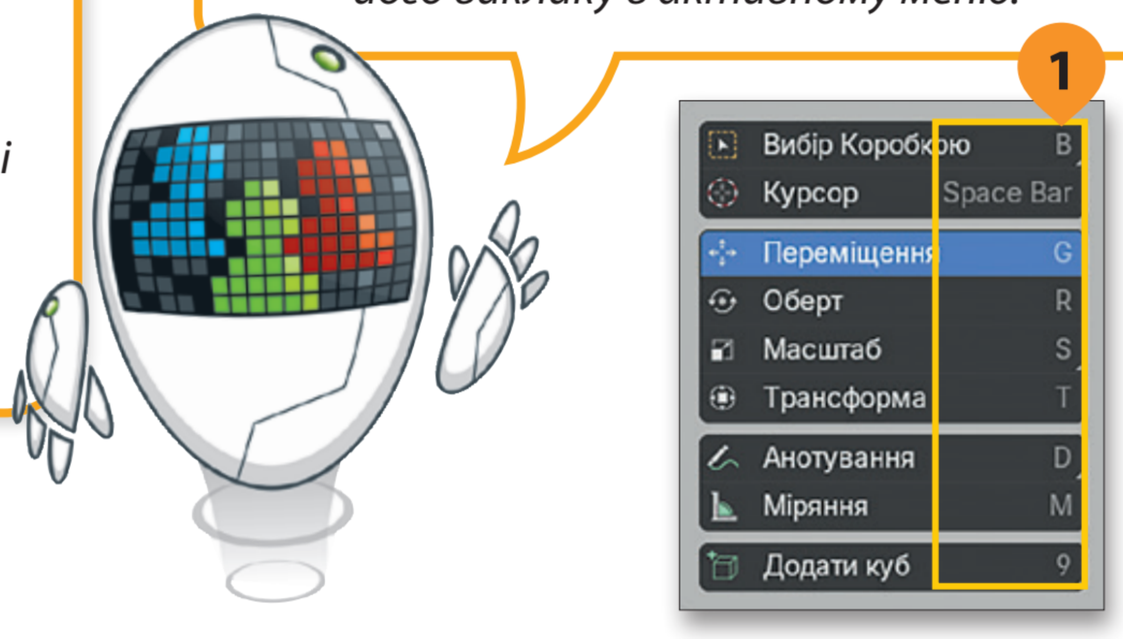
*Варто згадати про «**гарячі» клавіші**, які значно прискорюють робочий процес:*

- **G** **(Grab)** – для **переміщення**;
- **R** **(Rotate)** – для **обертання**;
- **S** **(Scale)** – для **масштабування**.

*Під час роботи в повноекранному режимі **активувати меню з інструментами** можна комбінацією клавіш*

Shift + **Пробіл** / **Space**

Меню з інструментами трансформації (1) *містить підпис інструмента, і «гарячу» клавішу його виклику в активному меню.*

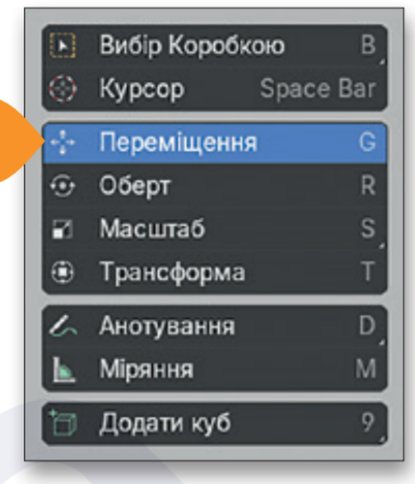


Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Основні інструменти трансформації у Blender

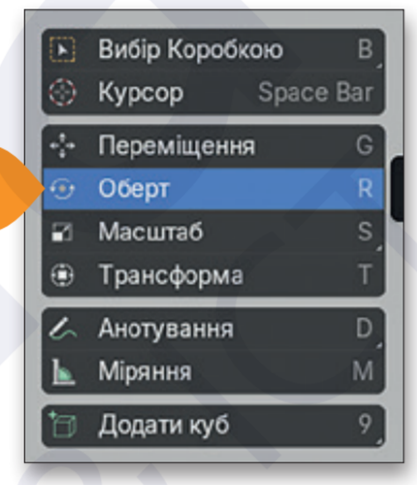
Переміщення (Move) 1

Цей інструмент дає змогу зміщувати об’єкт у просторі без зміни його розміру чи орієнтації.
Коли ви активуєте інструмент «Переміщення», на виділеному об’єкті з’являється маніпулятор – три кольорові стрілки (осі X, Y, Z).
Ви можете клацнути та перетягнути одну зі стрілок, щоб перемістити об’єкт лише вздовж цієї осі, або клацнути в центрі маніпулятора, щоб вільно переміщати об’єкт у площині екрана



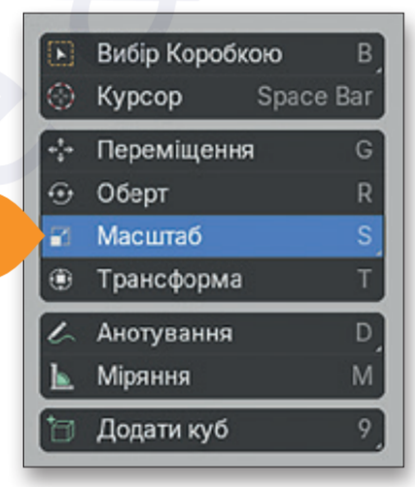
Обертання (Rotate) 2

Цей інструмент слугує для зміни орієнтації об’єкта в просторі навколо його центру (або іншої точки трансформації). Після активації інструмента **Обертання** навколо об’єкта з’являється маніпулятор у вигляді кілець (також пофарбованих у червоний, зелений та синій кольори для осей X, Y, Z відповідно). Клацаючи та перетягуючи одне з кілець, ви обертаєте об’єкт навколо відповідної осі. Якщо клацнути за межами кілець і перетягнути, об’єкт обертатиметься вільно відповідно до руху миші



Масштабування (Scale) 3

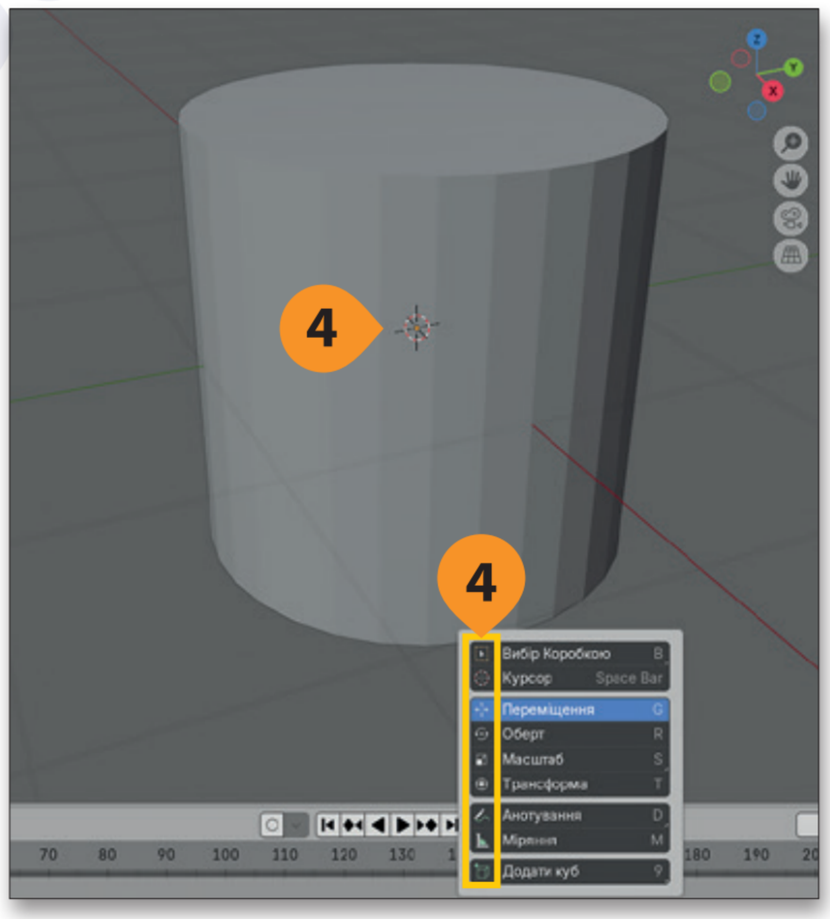
Цей інструмент здатен змінювати розмір об’єкта – збільшувати або зменшувати його. Після активації інструмента **Масштабування** на об’єкті з’являється маніпулятор з кольоровими квадратами на кінцях осей. Клацаючи та перетягуючи один з квадратів, ви змінюєте розмір об’єкта лише вздовж цієї осі. Клацання на центральному колі маніпулятора (або просто без вибору осі) масштабує об’єкт пропорційно по всіх трьох осях, зберігаючи його форму



Опорна точка у Blender

Перш ніж почати користуватися **інструментами трансформації**, оберіть об’єкт чи об’єкти, до яких потрібно застосувати інструмент. При активації інструмента трансформації в центрі обертання (опорній точці) з’явиться керуючий елемент (гізмо). **Гізмо** 4 мають для різних інструментів різний вигляд.

У Blender опорну точку ще називають **Origin**. Вона визначає центр обертання та масштабування об’єкта в **режимі об’єкта** і є його координатою, незалежно від того, де він розташовується у світовому просторі. Положення **Origin** не змінюється при маніпуляціях у **режимі редагування**.



Опорна точка *– це центральна точка* 4*, яка визначає, де об’єкт обертається та масштабується в* ***режимі об’єкта****. Вона є невід’ємною частиною ідентичності об’єкта, її положення є ключовим для правильного анімування та розміщення моделі.*



Неправильне розташування опорної точки *може повністю зіпсувати анімацію обертання та масштабування моделі.*

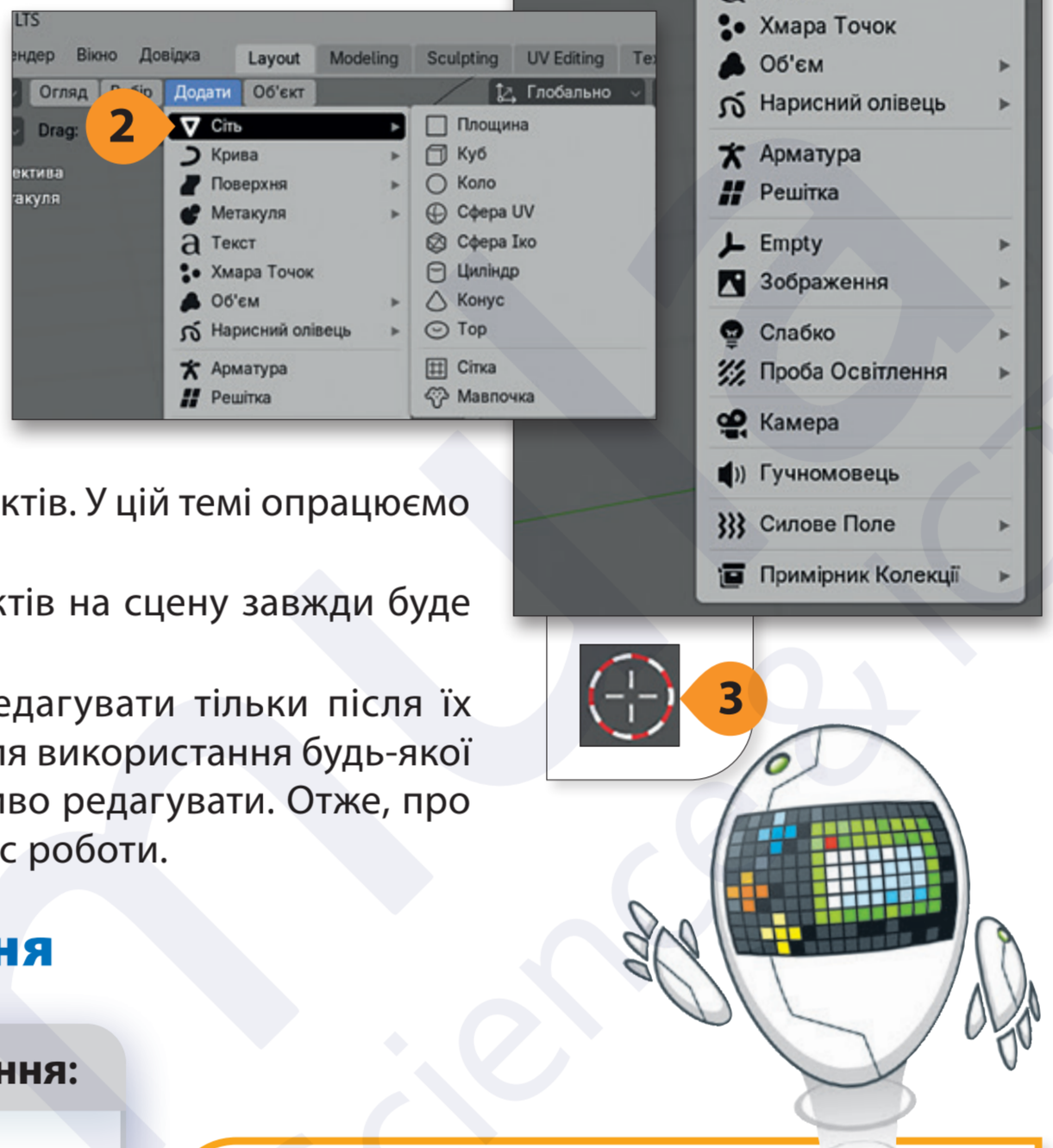
Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Додавання геометричних примітивів

Для початку моделювання на сцену потрібно додати об'єкти. Можна обрати їх з бібліотек Blender або імпортувати із зовнішніх документів.

Розгляньмо **бібліотеки Blender**.

*Отримати доступ до **бібліотек Blender** можна з меню видового вікна **Додати** (1) або користуючись комбінацією клавіш* **Shift** + **A**



У бібліотеках містяться різни типи об'єктів. У цій темі опрацюємо лише полігональні типу **Сіть** (2).

Варто знати, що точкою вставки об'єктів на сцену завжди буде тривимірний курсор (3).

Базові параметри об'єктів можна редагувати тільки після їх вставки до сцени у вікні останньої дії. Після використання будь-якої команди ці параметри вже буде неможливо редагувати. Отже, про цю незручність потрібно пам'ятати під час роботи.

Інструменти виділення

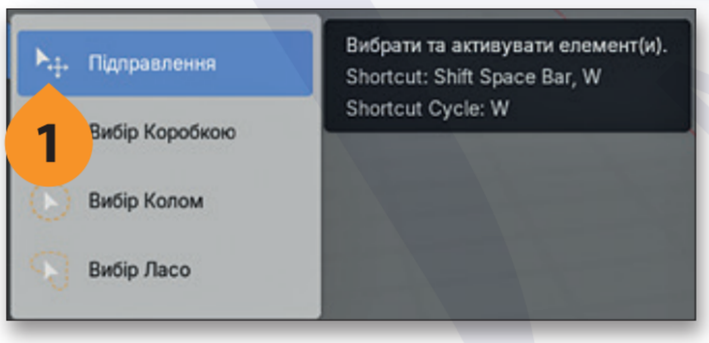
Щоб активувати Інструмент виділення:

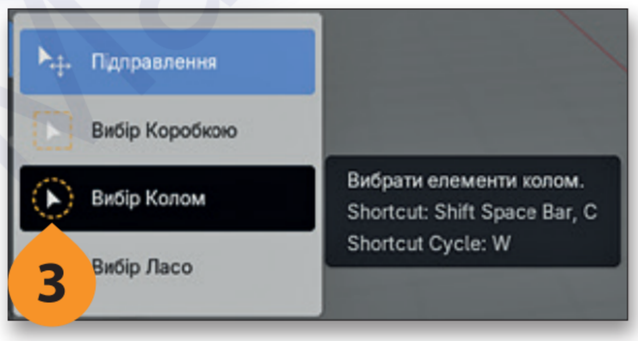
- натисніть на клавіатурі «гарячу» клавішу **W**;

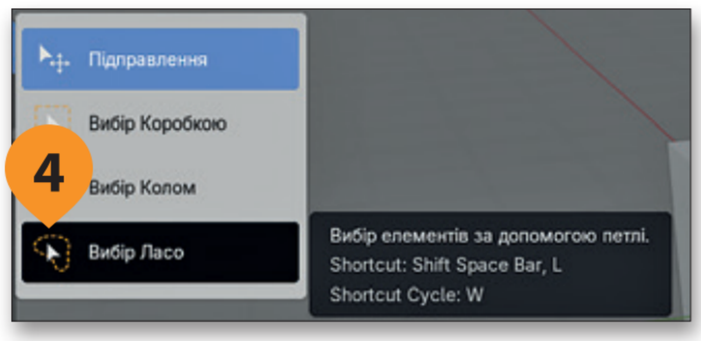
  або
- оберіть з **Панелі інструментів**;

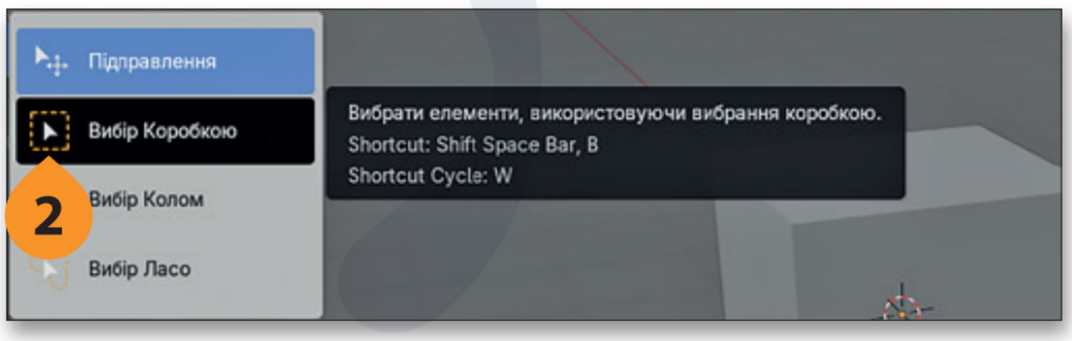
  або
- оберіть з меню **Вибір видового екрана**.

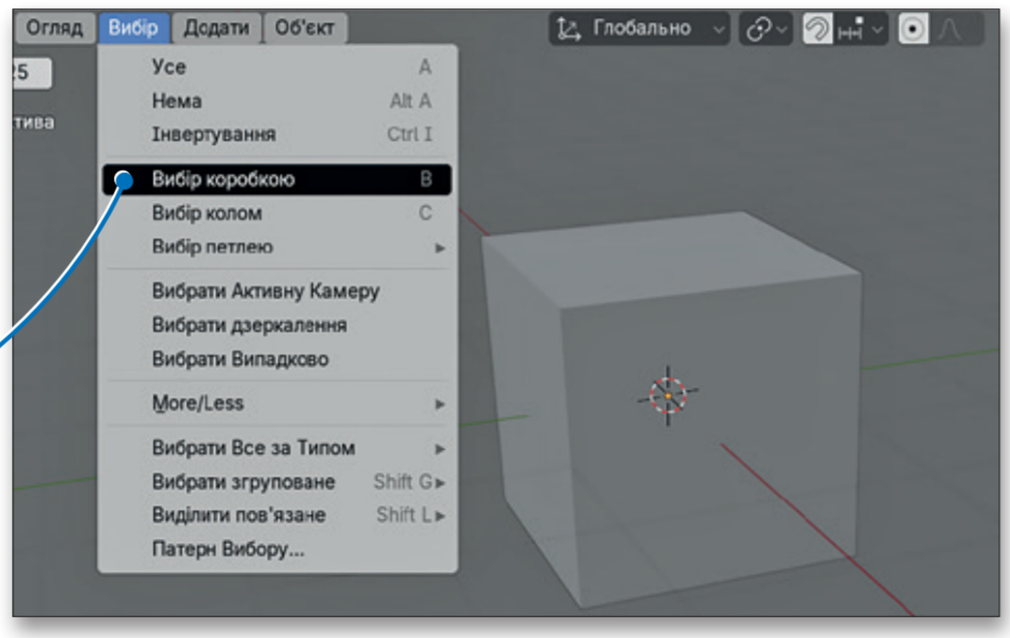
Натискаючи **W**, *можна перемикатися між режимами виділення послідовно:* ***об'єкт (Підправлення)*** (1), ***Вибір коробкою*** (2), ***Вибір колом*** (3), ***Вибір ласо*** (4).











Найчастіше використовується **Вибір коробкою**.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Дублювання об'єктів

Існує декілька способів для копіювання чи дублювання об'єктів:

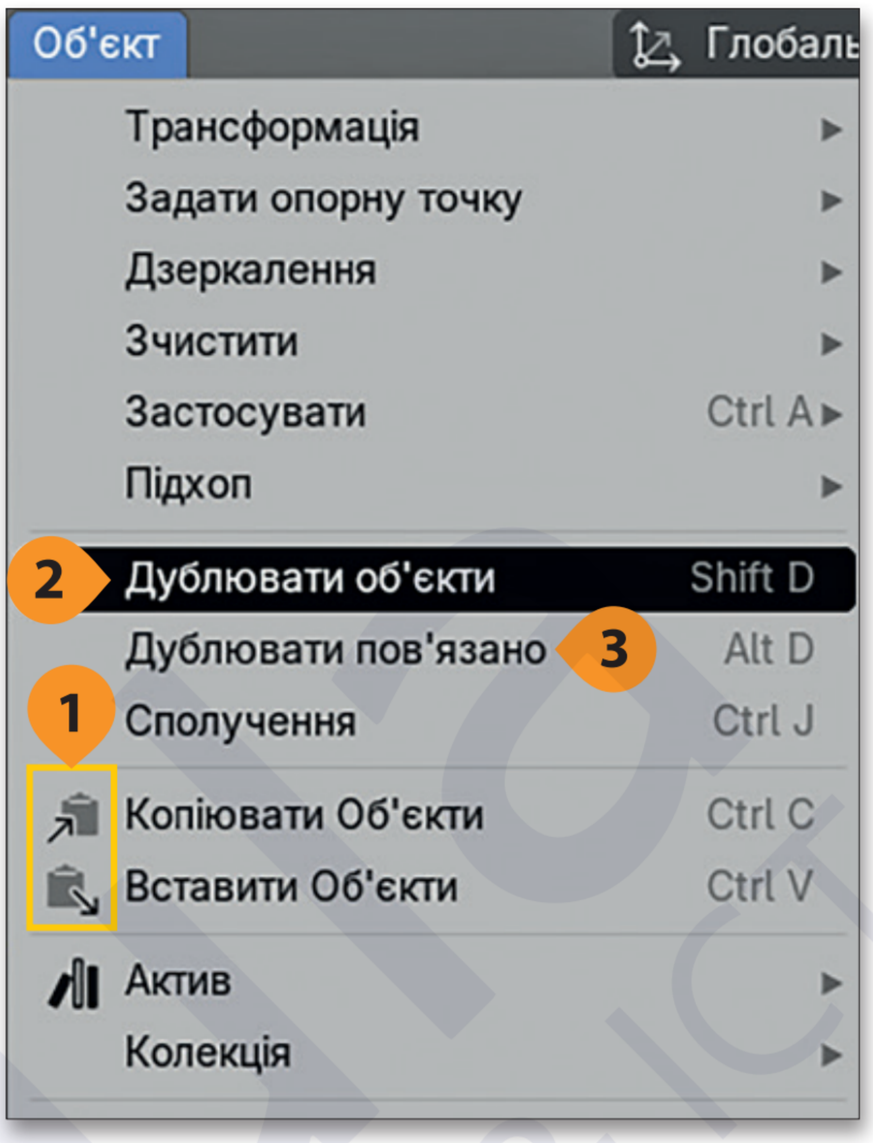
- по-перше, виділіть об'єкт, який треба дублювати;
- по-друге, застосуйте відповідну команду.

Найзручніший спосіб – це скористатися командами **Копіювання** і **Вставки** (1). Дублікат об'єкта буде накладено в тому ж місці, що і його оригінал.

Також є можливість дублювати однією командою **Дублювати об'єкти** (2). Після дублювання об'єкт буде рухатися по сцені за курсором. Тобто це одночасно і команда **Дублювання**, і команда **Пересування**.

Перелічені методи створюють незалежні копії.

Ви можете створити залежні копії об'єкта, користуючись командою **Дублювати пов'язано** (3). Залежні копії не навантажують систему та під час редагування одного з об'єктів, дії будуть застосовуватись на усі залежні об'єкти.



Щоб скопіювати чи дублювати об'єкти:

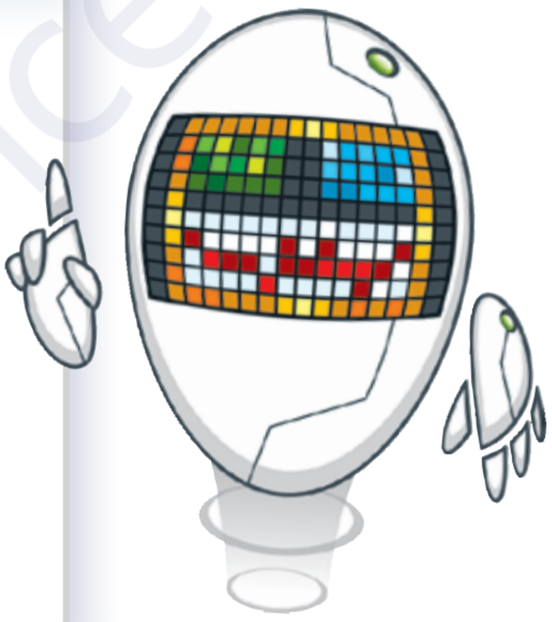
- найзручніший спосіб – скористатися командами **Копіювання** і **Вставки** з меню або використати відповідні комбінації клавіш: Ctrl + C – для **копіювання**, Ctrl + V – для **вставки**;

  або

- скористайтеся командою **Дублювати об'єкти** за допомогою комбінації клавіш Shift + D;

  або

- можна створити залежні копії об'єкта завдяки команді **Дублювати пов'язано** чи комбінацією клавіш Alt + D.



ДОВІДКА

Система інструментів трансформації вимоглива до користувача. Вона потребує інвестицій часу у вивчення «мови клавіатури» та її унікальної логіки, особливо щодо орієнтації та необхідності застосування трансформацій. Ігнорування цього кроку може призвести до неочікуваних спотворень геометрії при подальшому моделюванні, що є головною проблемою для новачків під час 3D-моделювання.

Практичне завдання

Завантажте файл-заготовку зі сторінки інтернет-підтримки. Виконайте запропоноване завдання.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Тема 6
Полігональне моделювання у Blender

Полігони

Будь-яка полігональна модель у Blender складається з трьох базових елементів.

Що більше полігонів використовується, то **детальнішою** буде модель.

Полігональне моделювання – *це метод створення 3D-моделей, який використовується з ранніх років розвитку комп'ютерної графіки. Його суть полягає у візуалізації об'ємних об'єктів за допомогою полігонів – багатокутних фігур.*

Базові елементи полігональної моделі

Компонент	Англійська назва	Опис
Вершина	**Vertex**	Одиночна точка в 3D-просторі. Це найменший компонент
Ребро	**Edge**	Лінія, що з'єднує дві вершини
Грань	**Face / Polygon**	Плоска поверхня, утворена трьома або більше ребрами, з'єднаними послідовно

Полігони *– це будівельні блоки 3D-моделей. Їх можна порівняти з мозаїкою, де з маленьких шматочків складається цілісна форма.*

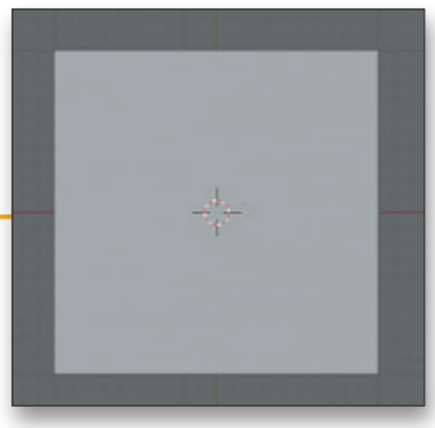


Існують різні типи полігонів:

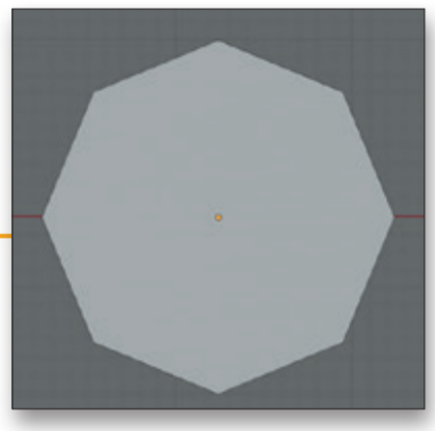
Трикутники
Найпростіший тип полігона, що складається з трьох вершин. Вони прості для обчислення, але погано поводяться при згладжуванні



Чотирикутники
Більш складний тип полігона, що складається з чотирьох вершин. Це золотий стандарт у моделюванні, оскільки вони забезпечують найкращу поведінку при деформації та використанні модифікатора



N-гони (многокутники)
Полігони з ***n*** вершинами (де ***n > 4***). Їх слід уникати на кривих або деформованих поверхнях, оскільки вони можуть спричиняти проблеми з освітленням та згладжуванням



Сітка – *це сукупність полігонів, що утворює каркас 3D-моделі. Вона подібна до сітки, що використовується для створення скульптур.*



Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Сітка

Сітка – це каркас, що складається з трьох основних компонентів, які з'єднані між собою: вершин, ребер та граней. Вона є фундаментальною структурою, яка надає об'єкту його форму – від простого куба до складного анімованого персонажа.

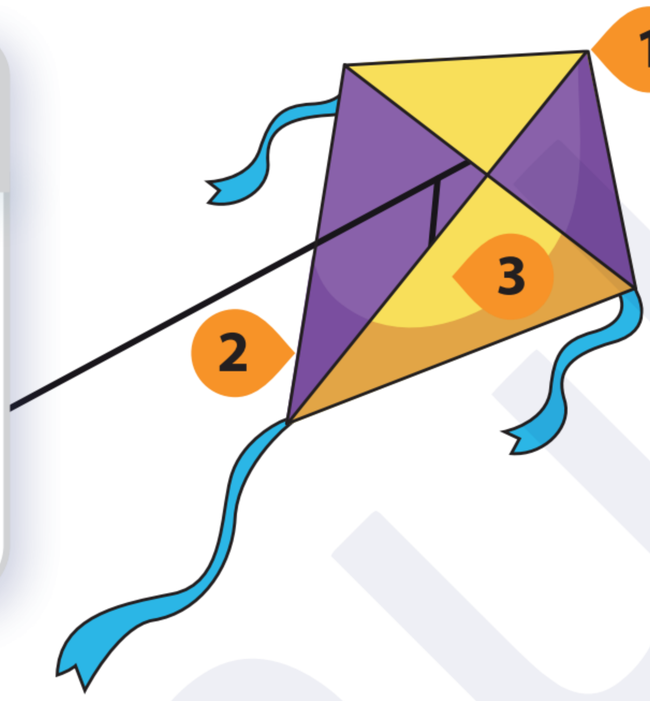
Маніпулювання компонентами сітки (вершинами, ребрами, гранями) у режимі редагування є основою полігонального моделювання.

Сітка в контексті 3D-моделювання – *це цифрова структура, яка визначає форму та геометрію тривимірного об'єкта.*



Уявіть, що ви будуєте повітряного змія:

- **вершини** – це кути каркаса (1);
- **ребра** – це палички, що з'єднують кути (2);
- **грані** – це папір або тканина, якою ви обклеюєте каркас (3).



Режим редагування (Edit mode)

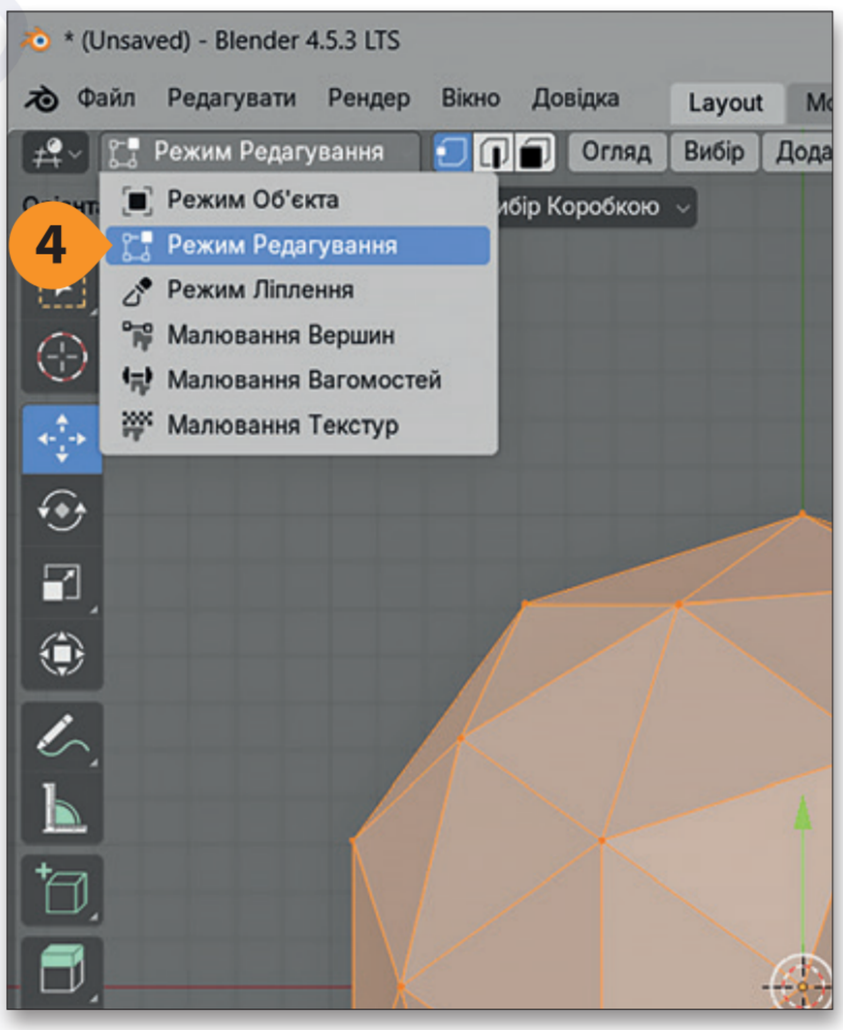
Для переходу в **режим полігонального моделювання** треба спочатку виділити об’єкт редагування, потім у видовому вікні обрати з переліку відповідно **режим редагування** (4).

Це ввімкне **режим полігонального моделювання**. Також замість меню можна користуватися клавішею **Табуляції** на клавіатурі для перемикання між режимами моделювання та об’єктним.

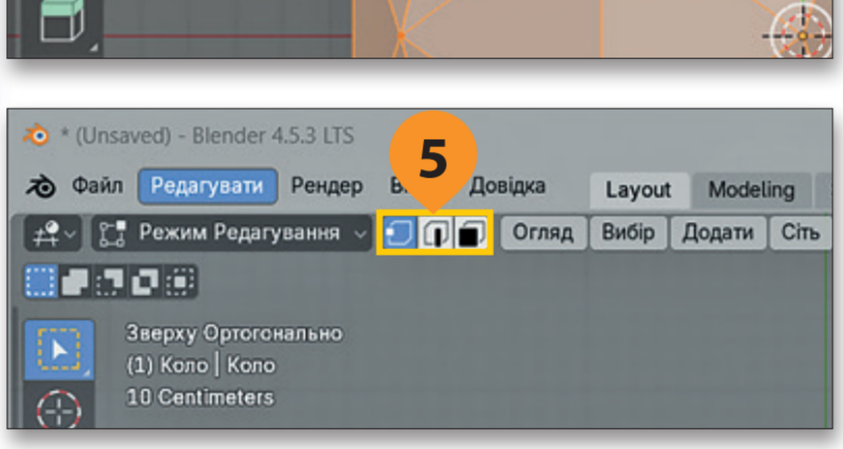
Режим редагування (Edit mode) *– це спеціальний робочий простір, який дає вам змогу маніпулювати індивідуальними компонентами 3D-сітки (вершинами, ребрами та гранями) для зміни форми об'єкта.*

Усі маніпуляції з полігонами виконуються в режимі редагування (Edit mode)

- **Створення примітиву:**
  натисніть Shift + A → **Mesh** та додайте базовий об'єкт (наприклад, куб).
- **Перехід у режим редагування**: виділіть об'єкт і натисніть клавішу Tab
  або виберіть **Edit Mode** у спадному меню у верхньому лівому куті **3D-вікна**.
- **Вибір компонентів**: у цьому режимі ви можете перемикатися між вибором **Вершин (1)**, **Ребер (2)** або **Граней (3)**, використовуючи цифрові клавіші 1, 2 та 3.



У **режимі редагування** зміниться зміст **Інструментальної панелі**, з’являться інструменти редагування у відповідному режимі (5).

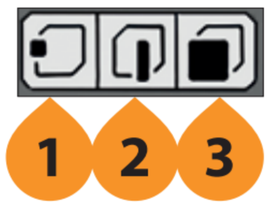


Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Типи елементів редагування

Існує 3 типи об'єктів, положення котрих можна міняти під час полігонального моделювання, це – вершини, ребра та грані. Поруч з перемикачем режимів розташовані **Іконки перемикання Типів елементів редагування**.

Іконки перемикання Типів елементів редагування

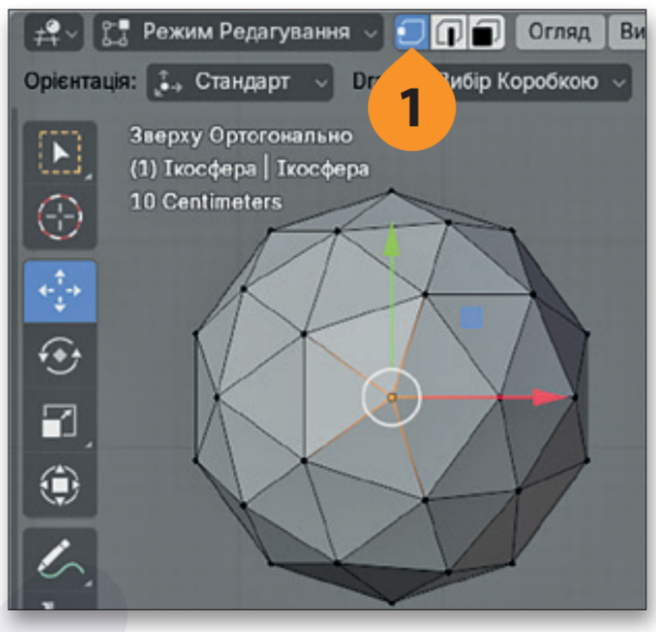


*Для перемикання **Типів елементів редагування** можна користуватися «гарячими» клавішами, відповідно це 1, 2 та 3.*

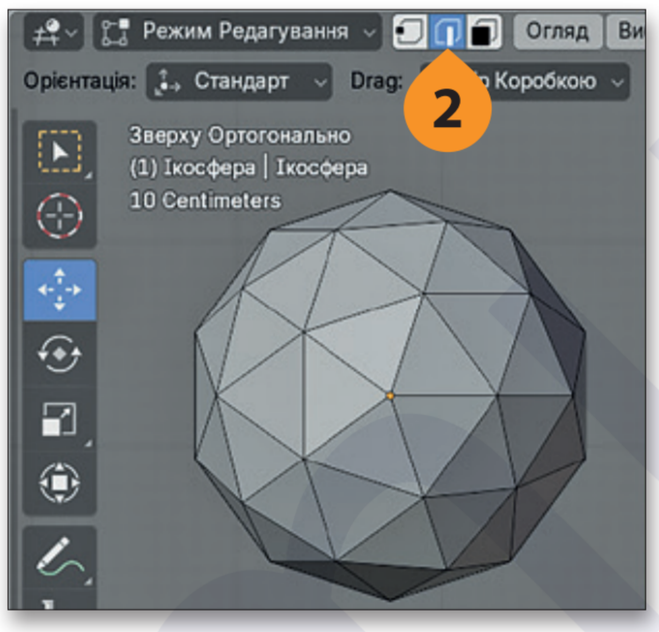
1 – **Вершини (Vertices)** – окремі точки, що утворюють сітку

2 – **Ребра (Edges)** – лінії, що з'єднують дві вершини

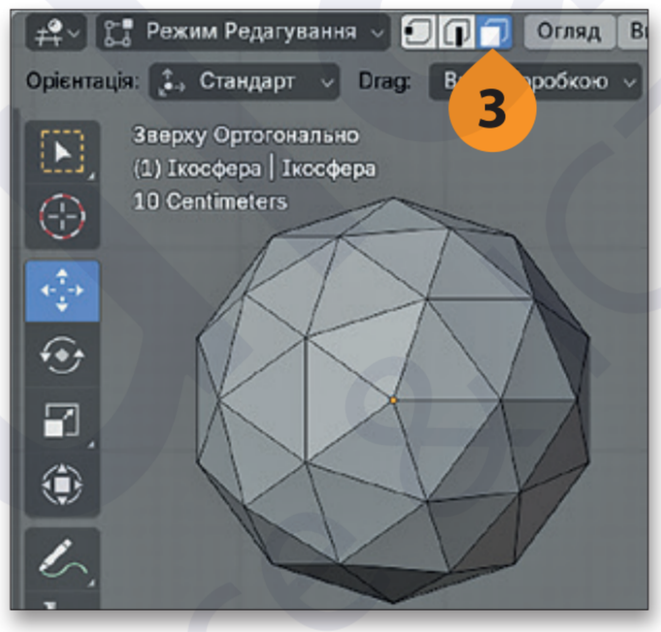
3 – **Грані (Faces)** – плоскі поверхні, утворені ребрами



Редагування вершин



Редагування ребер



Редагування граней

Інструменти трансформації під час редагування

У **режимі ведагування вершин** під час використання **Інструментів трансформації** можна редагувати форму об'єкта.

*Інструмент **Пересування** змінює позицію вибраної вершини чи групи вершин. У вікні останньої дії можна змінювати відповідні параметри.*



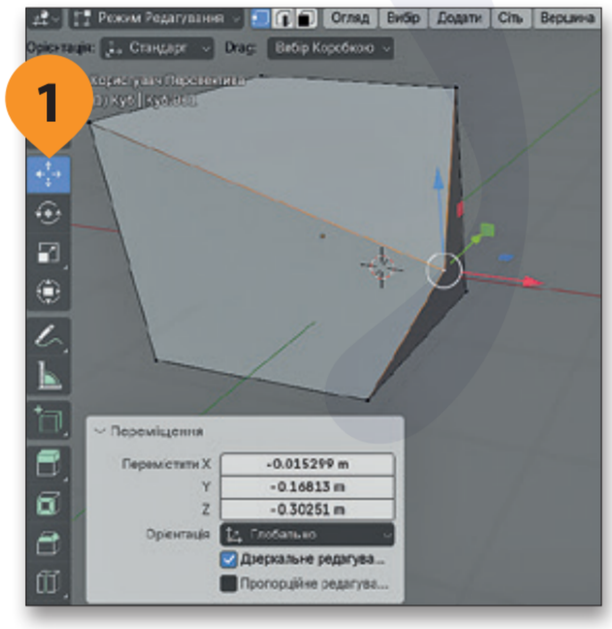
1 – інструмент **Пересування**

2 – інструмент **Обертання**

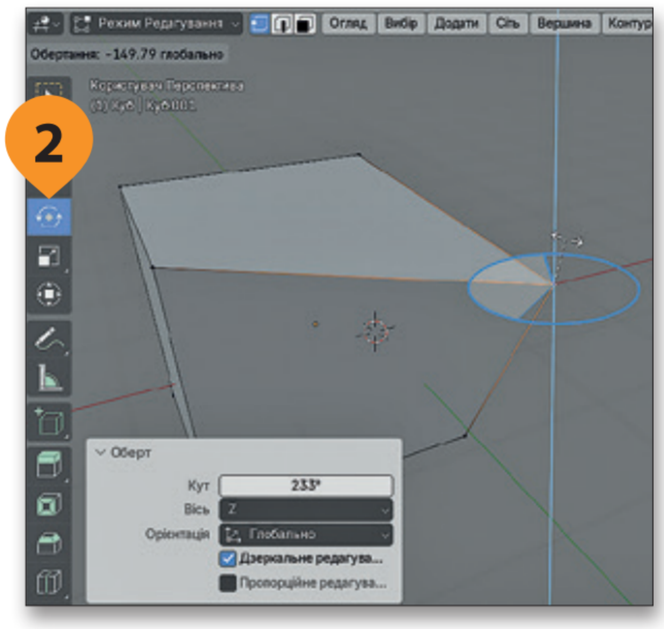
3 – інструмент **Масштабування**

4 – інструмент **Трансформації**

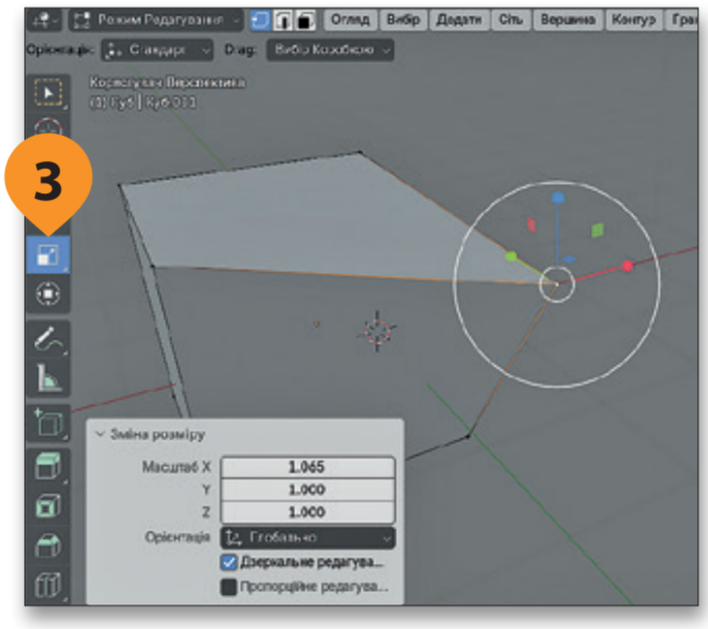
*Інструмент **Обертання** або **Масштабування** на одну вершину діяти ніяк не буде, але можна обрати декілька вершин, центром оберту при цьому буде геометричний центр, розрахований між обраними вершинами.*



Інструмент Пересування



Інструмент Обертання



Інструмент Масштабування

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

*Інструмент **трансформації** – це комбінований інструмент, який містить одночасно елементи для переміщення (стрілки), обертання (дуги) та масштабування (квадрати). Це дає вам змогу швидко перемикатися між трьома трансформаціями без зміни інструмента.*

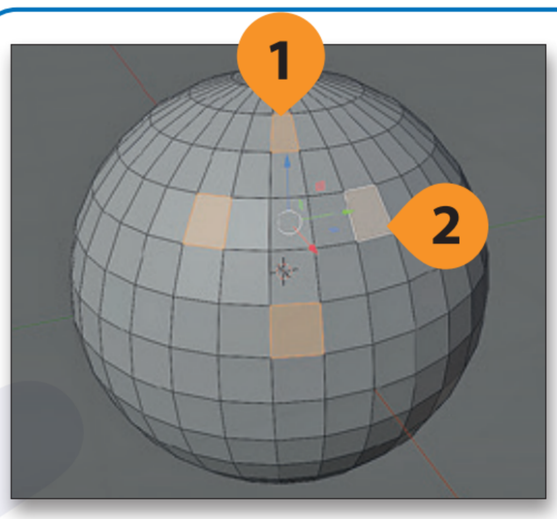


Виділення у режимі редагування

Важливою складовою під час моделювання є **вміння виділяти потрібні елементи** та робити це швидко й ефективно. Тому інструментарій виділення у більшості редакторів є гнучким та потужним. У цьому розділі ми розглянемо лише команди, без яких буде неможливо виконати вправи.

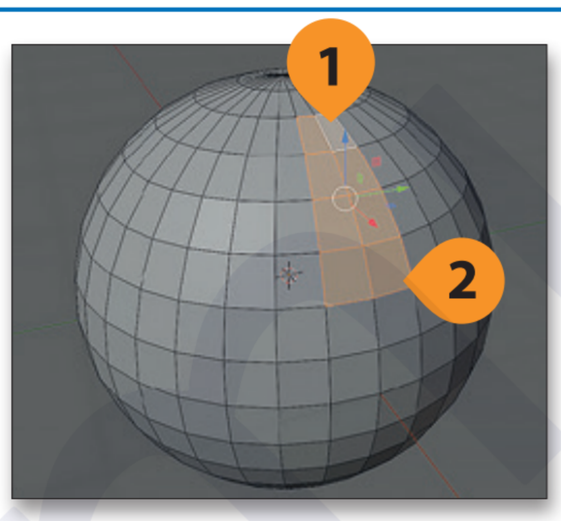
На зображеннях нижче надані результати послідовного клацання кнопкою миші на один (1) та другий (2) осередок із відповідно затиснутими клавішами клавіатури.

*Виділення замкненої геометрії називають **Виділення петлею**.*



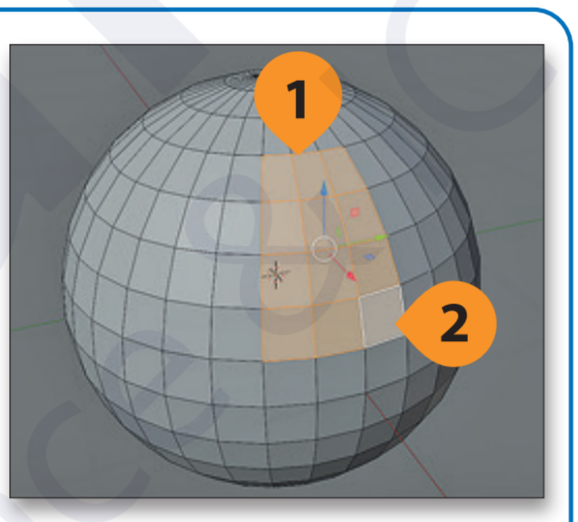
Затиснута клавіша

Shift



Затиснута клавіша

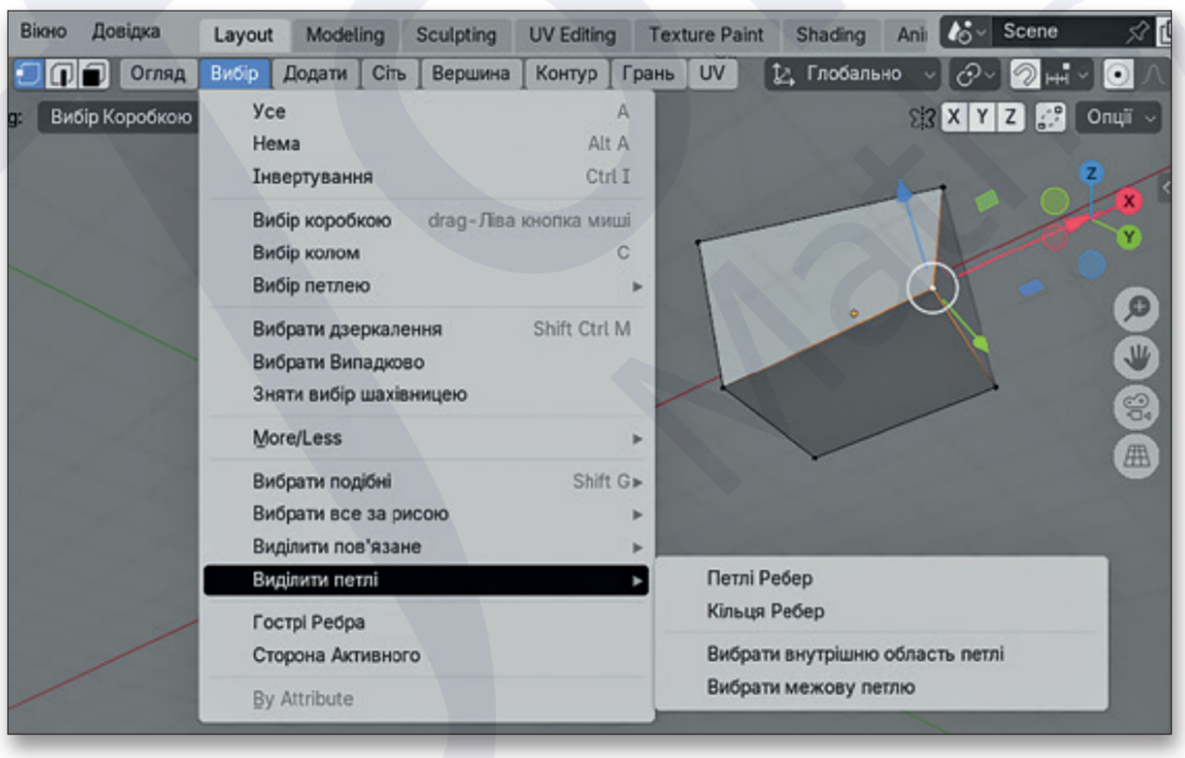
Ctrl



Затиснуті клавіші

Shift + Ctrl

Існує **Виділення петель** та **Виділення кілець** (послідовно розташованих елементів).



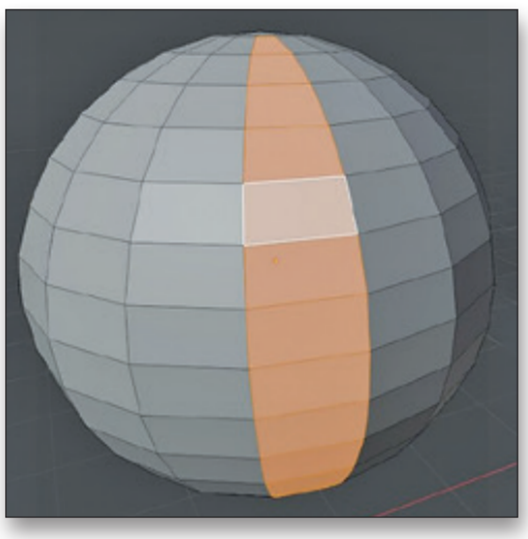
*Для **виділення елементів** треба затискати клавішу Shift.*

*Для того щоб **прибрати зайвий елемент з виділення**, слід ще раз клацнути по ньому кнопкою миші із затиснутою клавішею Shift.*

*Якщо потрібно **прибрати із зони виділення декілька елементів,** використайте клавішу Ctrl.*

*Для **виділення** частіше користуються затиснутою клавішею Alt, але можна обирати і команду в меню вікна моделювання **Вибір**.*

Затиснуті клавіші Shift + Alt дають змогу обрати декілька петель.



Затиснута клавіша

Alt

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

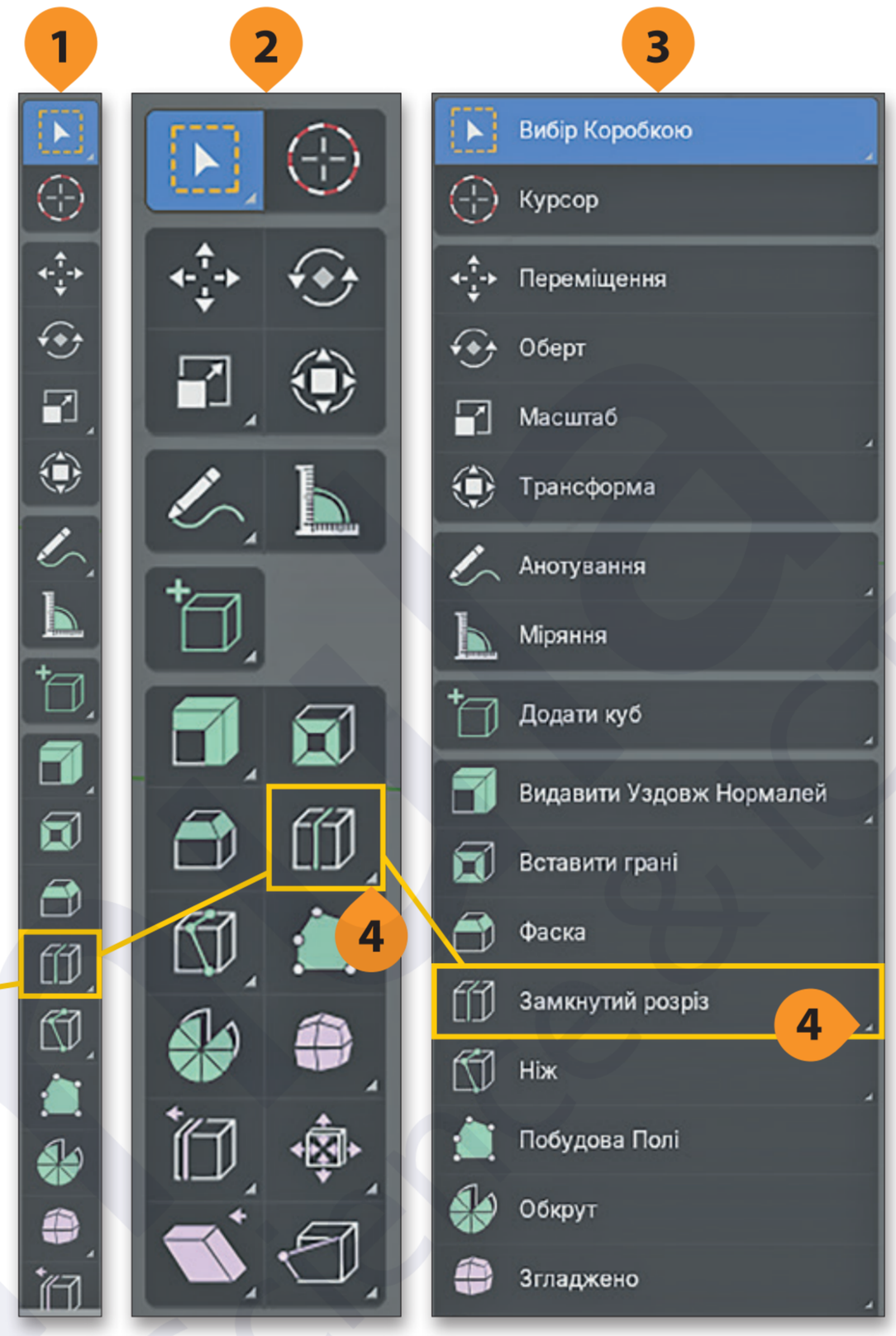
Панель інструментів режиму редагування

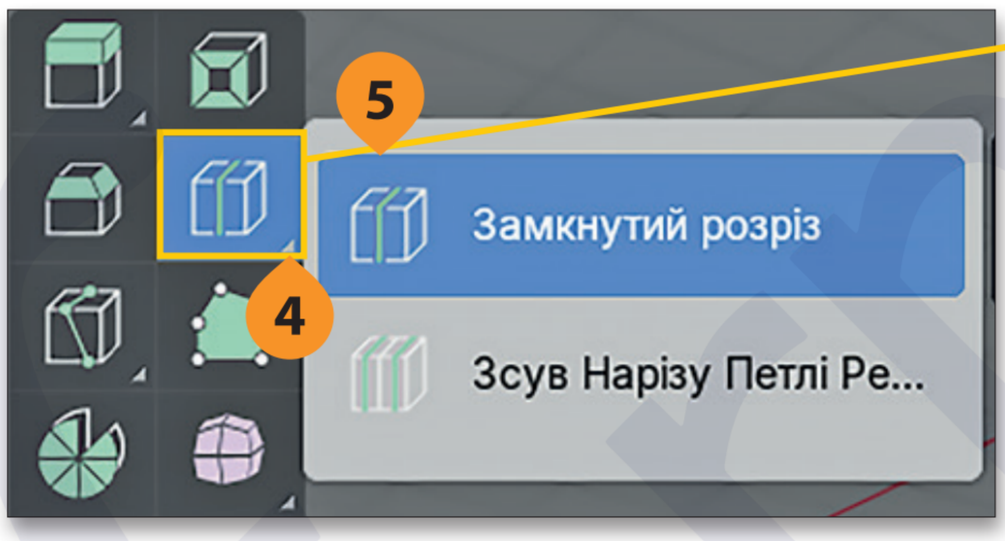
На **Інструментальну панель** режиму редагування виведені інструменти, що уможливлюють використання найбільш потужних команд редагування. Інші команди можна активувати з меню видового вікна.

Панель інструментів за стандартним налаштуванням розміщена у вертикальний рядок (1). Але можна змінити тип відображення, потягнувши за її ліву межу, й отримати два рядки інструментів (2).

Також за потреби можна панель розтягнути далі, щоб бачити не тільки іконки команд, а й підписи з назвами команд редагування (3).

Якщо команда має у лівому нижньому куті невеличкий трикутник (4), це означає, що при натисканні на іконку лівою кнопкою миші й затримці на 2 секунди кнопки затиснутою можна побачити сховані інші інструменти наданої групи команд (5).





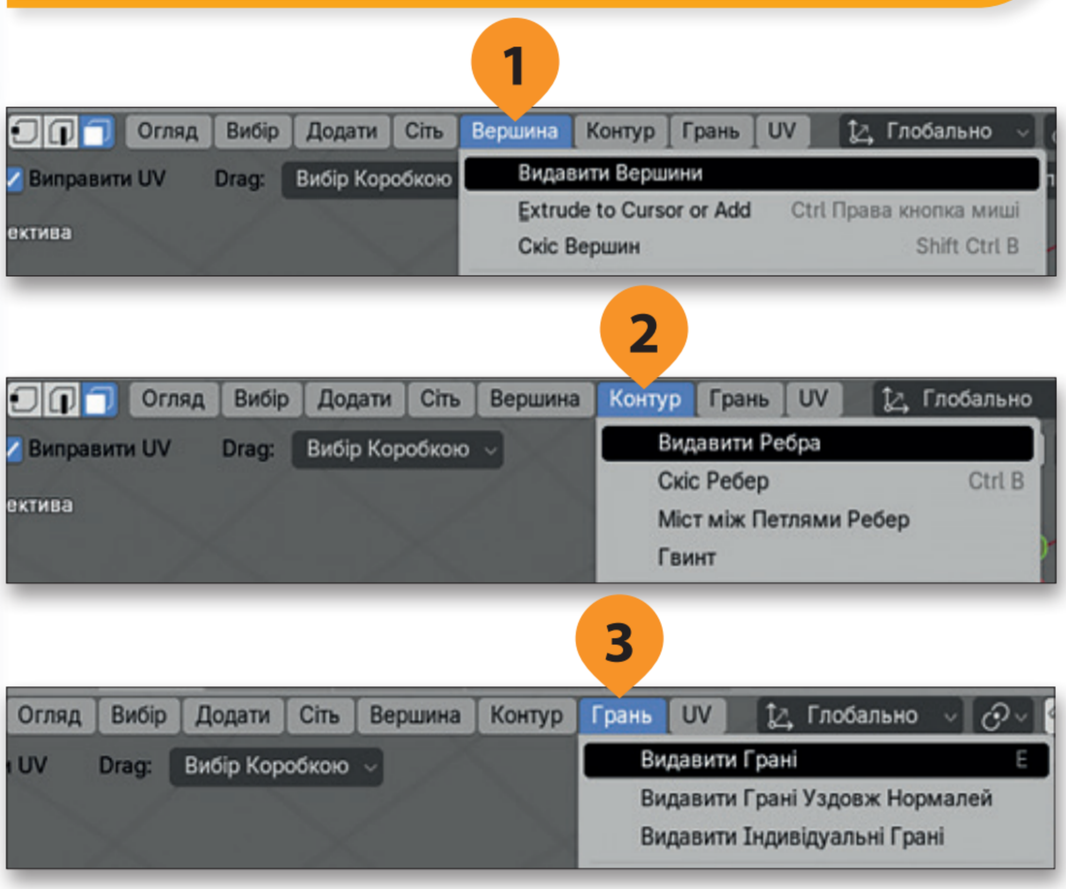
Команда Видавлювання

Однієї з базових команд під час моделювання є команда **Видавлювання елементів (Extrude)**. Вона по-різному діє у режимах редагування вершин, ребер та граней.

Команда Видавлювання (Extrude) *– це фундаментальний інструмент, який дає змогу створити нову геометрію, витягуючи (видавлюючи) грані, ребра або вершини. Вона бере виділений елемент і, по суті, копіює його, з'єднуючи копію з оригіналом новими гранями. Це додає об'єкту глибину або нові деталі.*

Щоб застосувати видавлювання елементів (Extrude):

- потрібно у відповідному режимі обрати елемент, активувати інструмент **Видавлювання**, керувати нарощуванням нової геометрії чи її прорізанням;
- можна також скористатися меню: **Вершина** (1), **Контур (ребро)** (2) та **Грань** (3);
- також можна активувати команду з меню правої кнопки миші або «гарячою» клавішею (E) на клавіатурі.

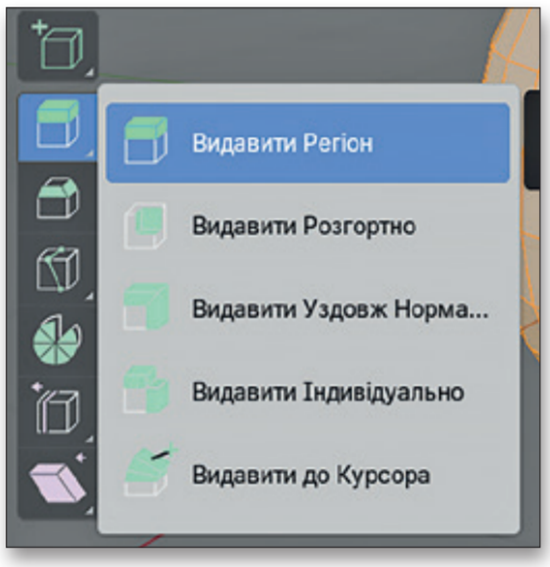


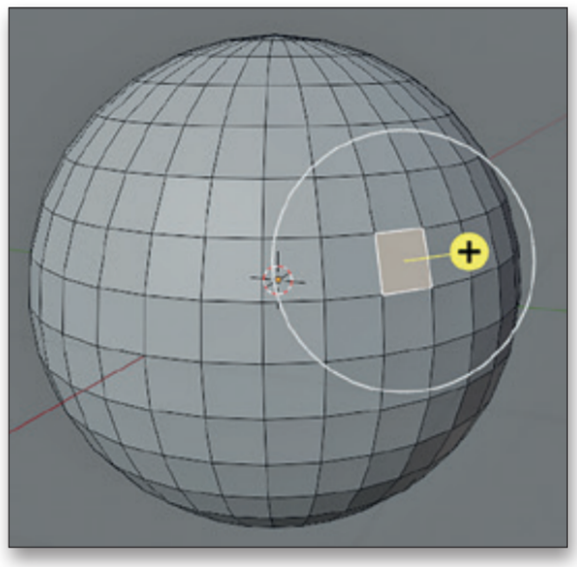
Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Різновиди команди Видавлювання

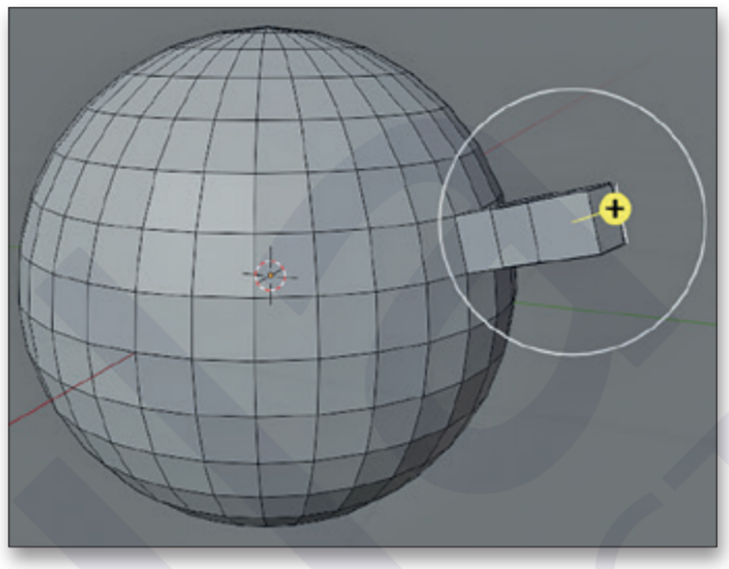
Існує декілька різновидів **команди Видавлювання**, як в **Інструментальній панелі**, так і в **Панелях команд**. Спочатку розгляньмо, як працює команда у **режимі редагування полігонів** на прикладі сфери.

Активація команди

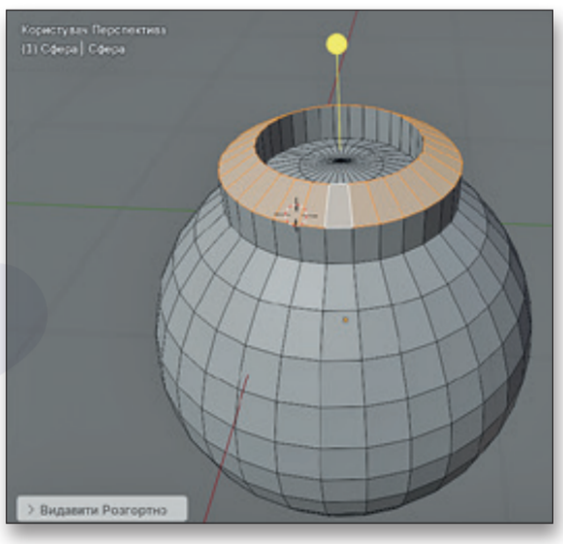


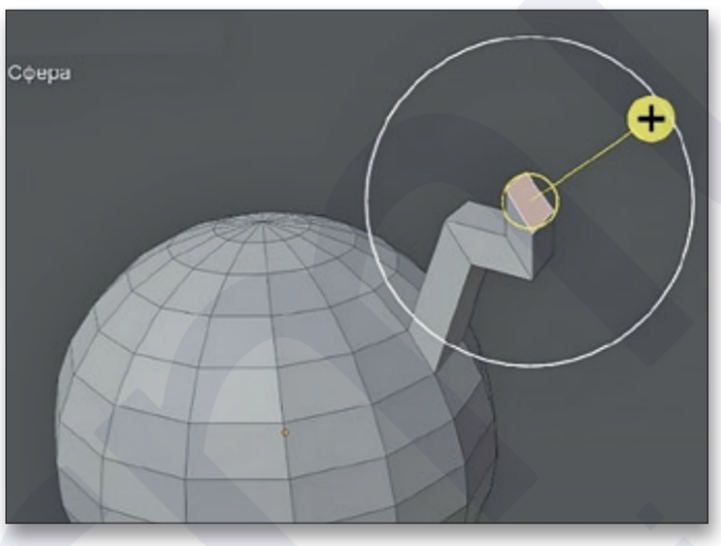


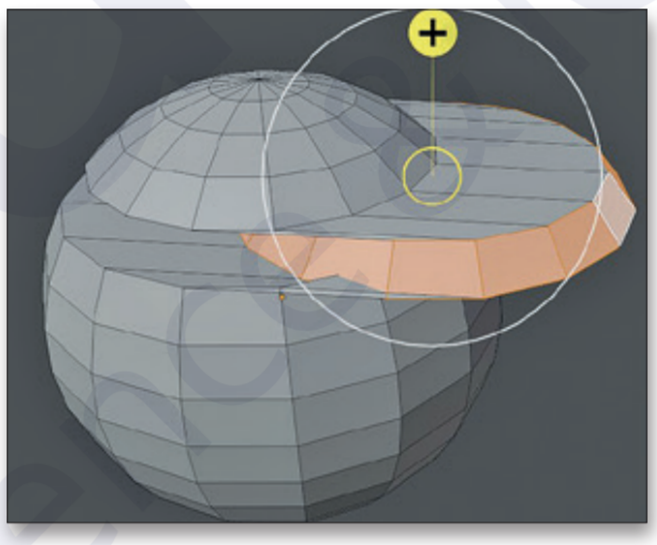
Користування елементом +



Користування елементом коло



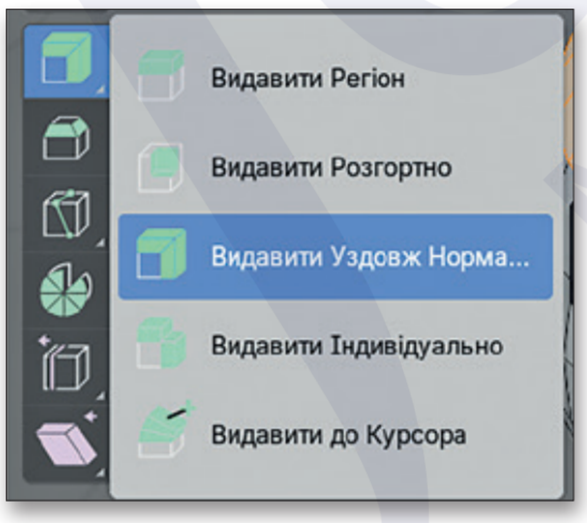


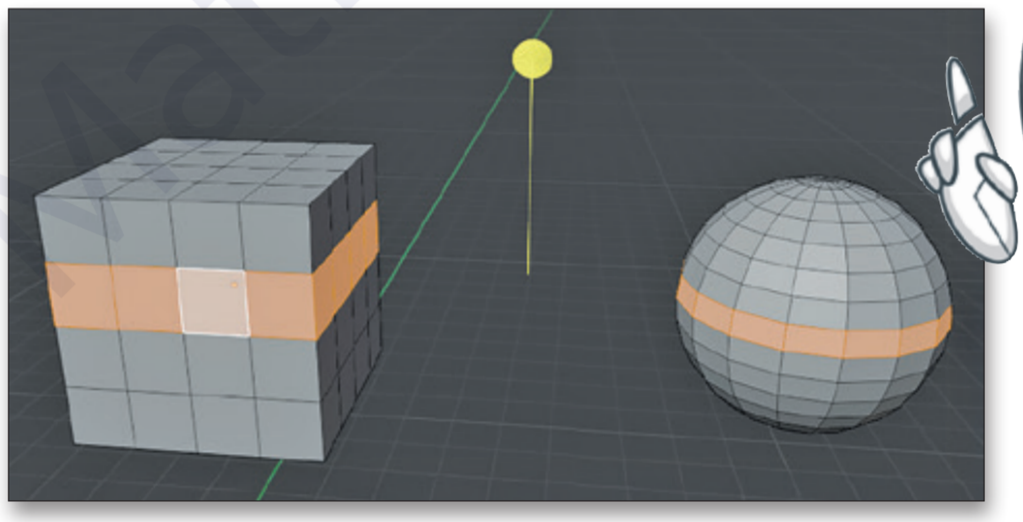


Команда Видавити уздовж нормалі

Команда **Видавити уздовж нормалі** видавлює елементи відносно нормалі до обраного елемента, або якщо обрано декілька – відносно нормалі до геометричного центру обраних елементів.

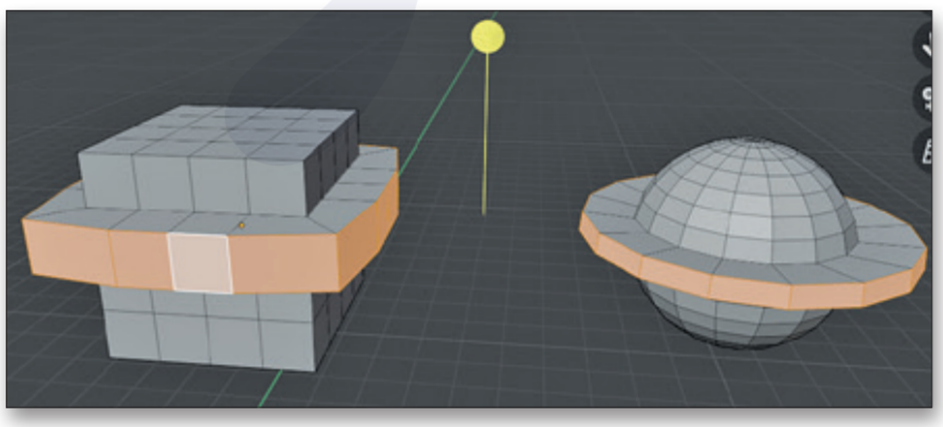
Використання команди для змінення форми



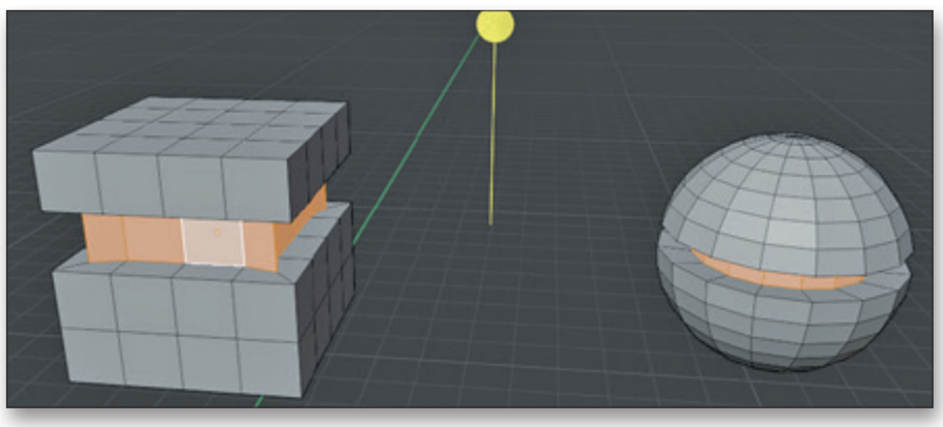




Додавання геометрії



Утворення отворів

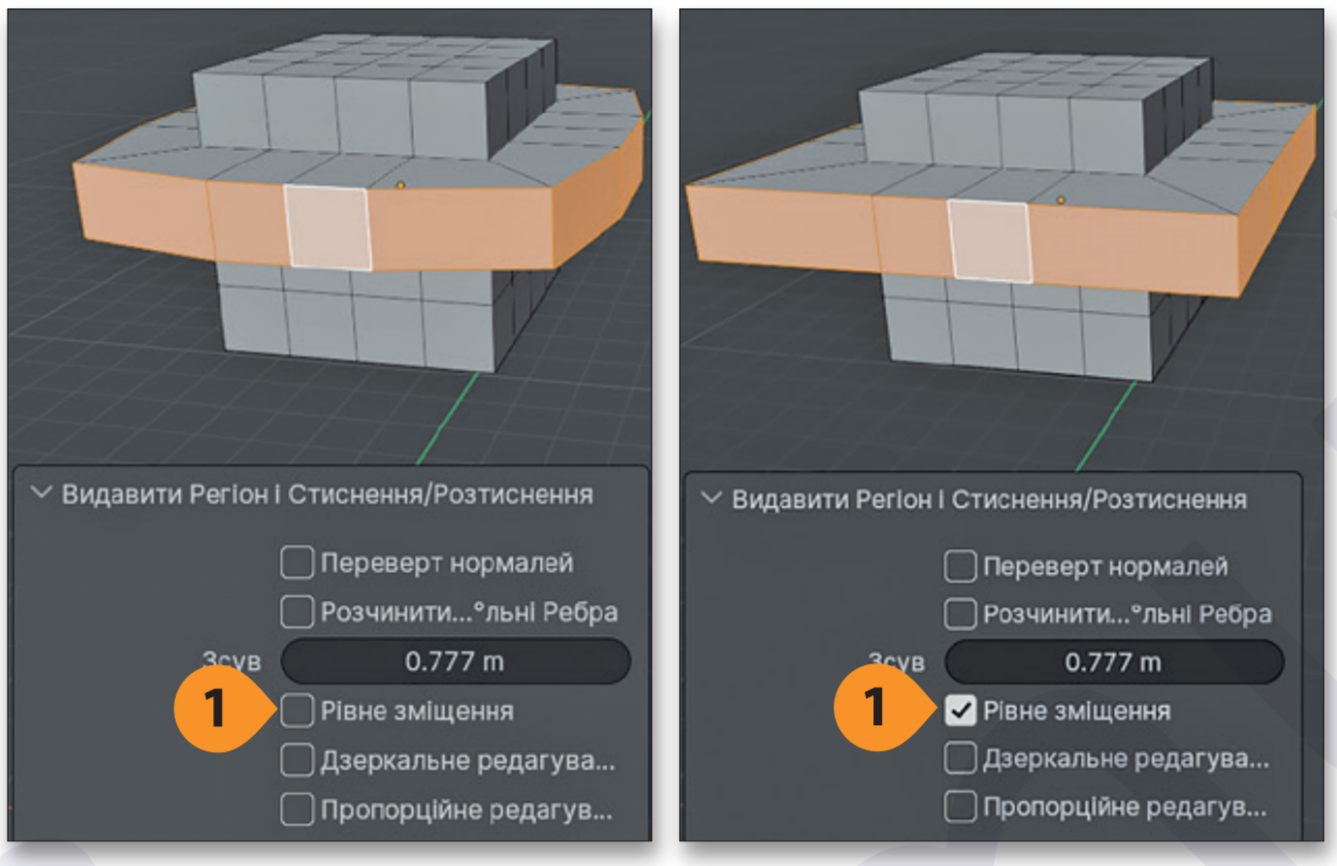


Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Рівне зміщення

Після **Видавлювання уздовж нормалі** може вийти викривлена геометрія на кутах елементів. Скоригувати отриману форму можна за допомогою палітри останньої дії. Потрібно в палітрі встановити прапорець на пункті **Рівне зміщення** (1).

Палітра останньої дії під час уточнення форми



Перемикання між загальним та індивідуальним кресленням *можна робити подвійним натисканням «гарячої» клавіші виклику команди*

Вставки граней (I)

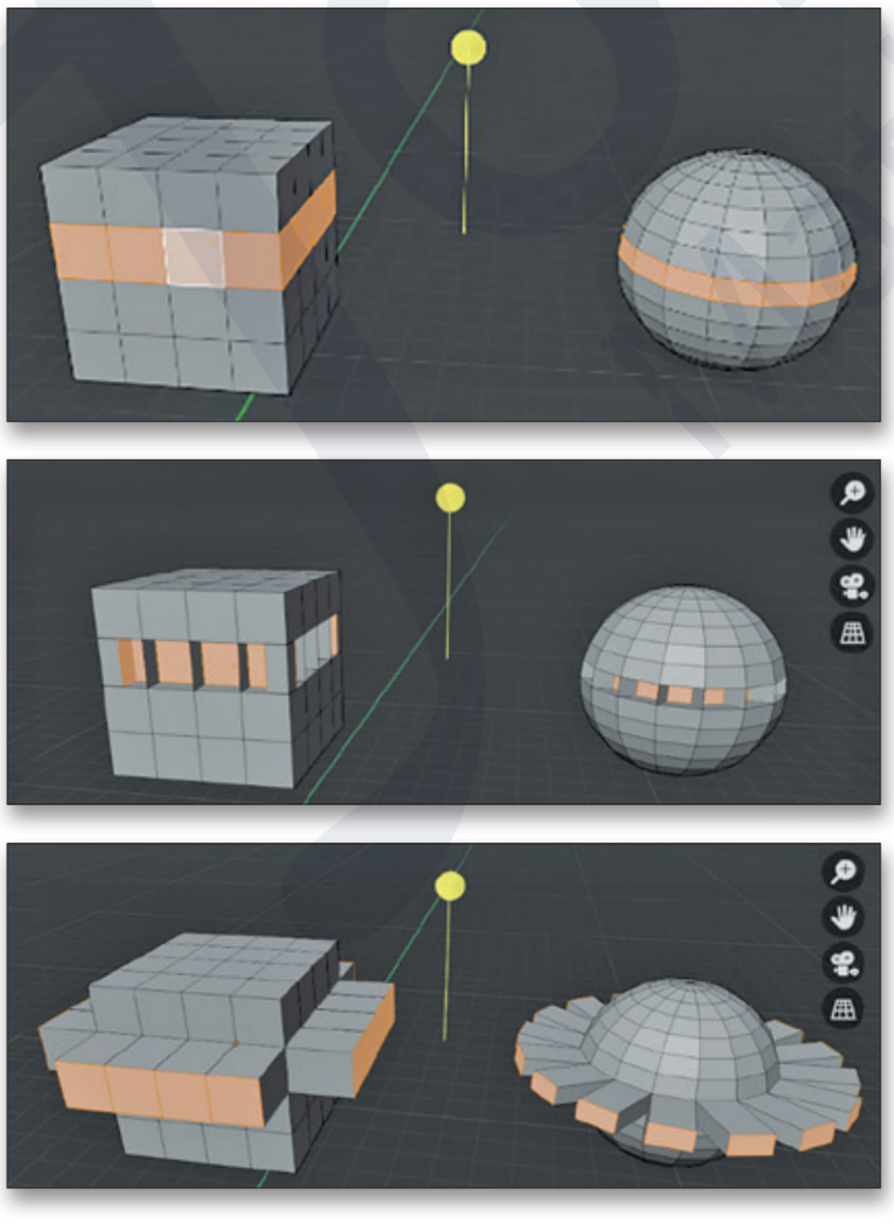
або за допомогою ***Меню останньої дії*** *після виконання команди.*



Команда Видавити індивідуально

Команда **Видавити індивідуально** дає змогу редагувати одночасно окремо кожний вибраний елемент на підставі його власної нормалі.

Команда Видавити індивідуально



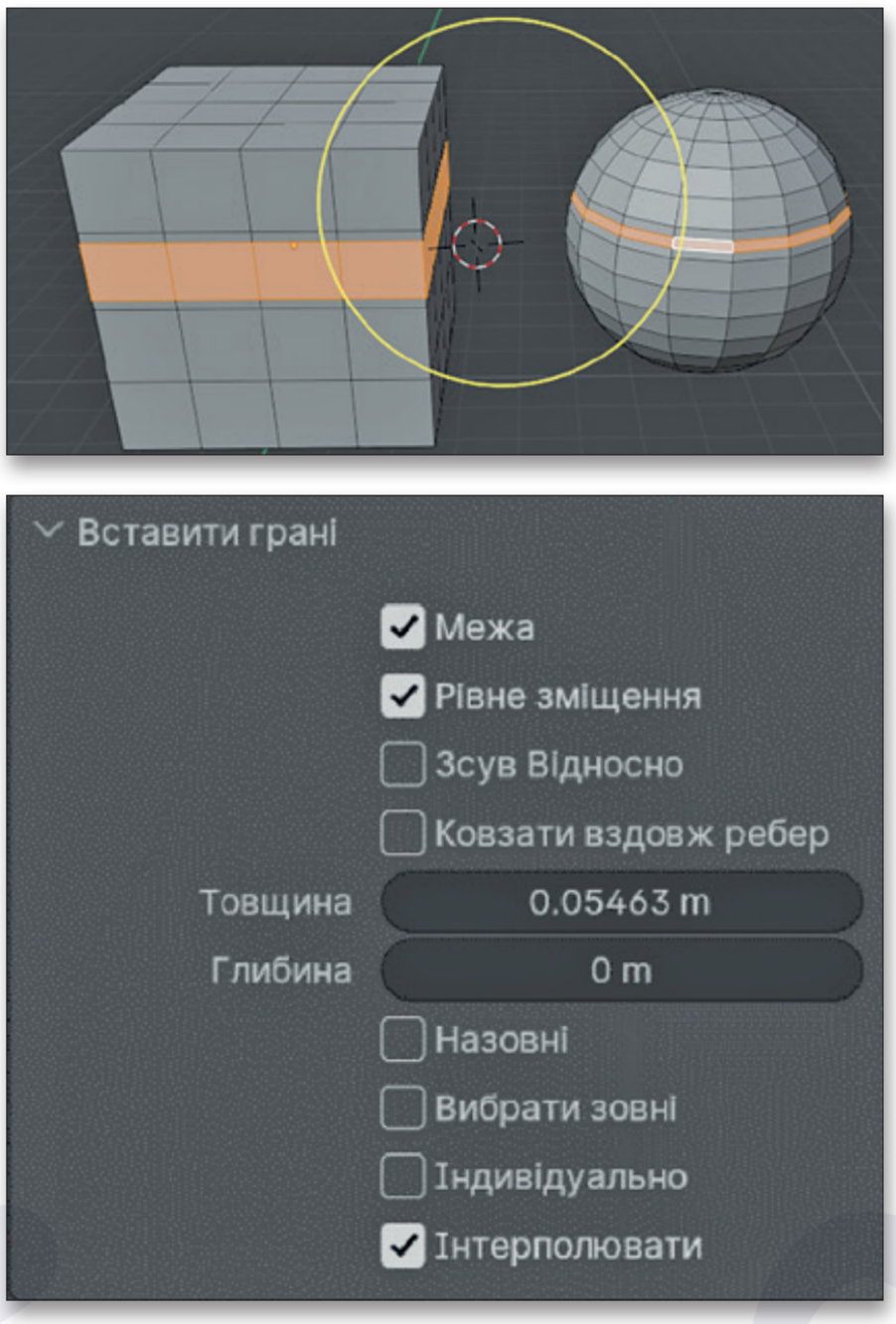
Команда Вставки (обведення)

Команда **Вставка граней** дає змогу прокреслити геометрію усередині обраних осередків. Можна робити загальне креслення або індивідуальне.

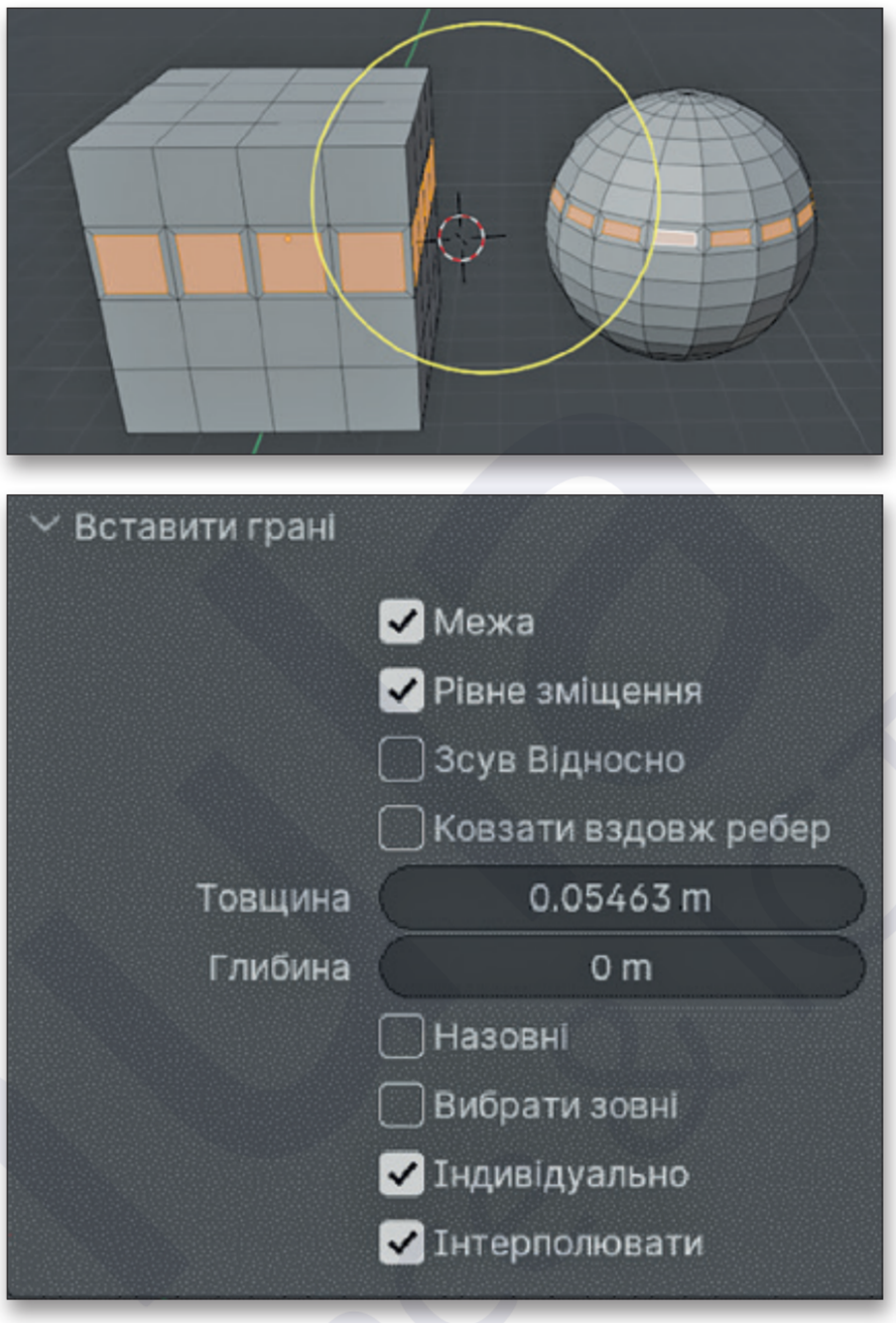
Команда Вставка граней

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

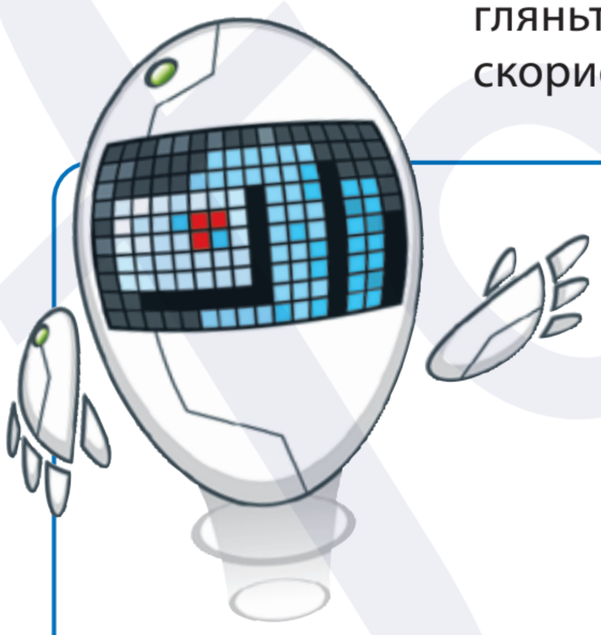
Спільна геометрія обведення



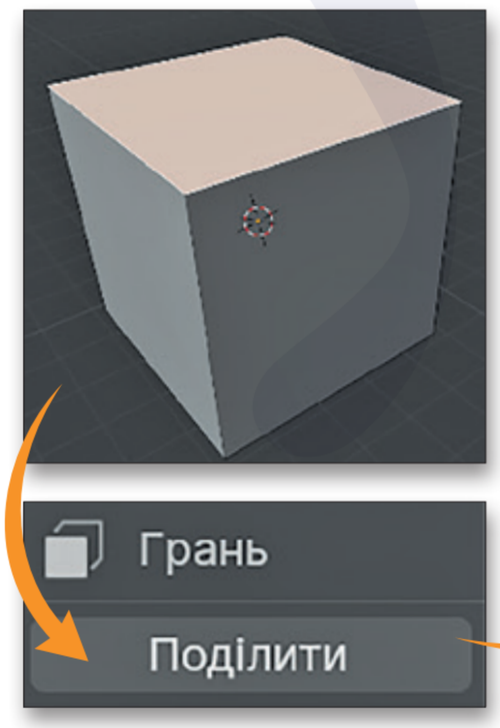
Відокремлена геометрія обведення

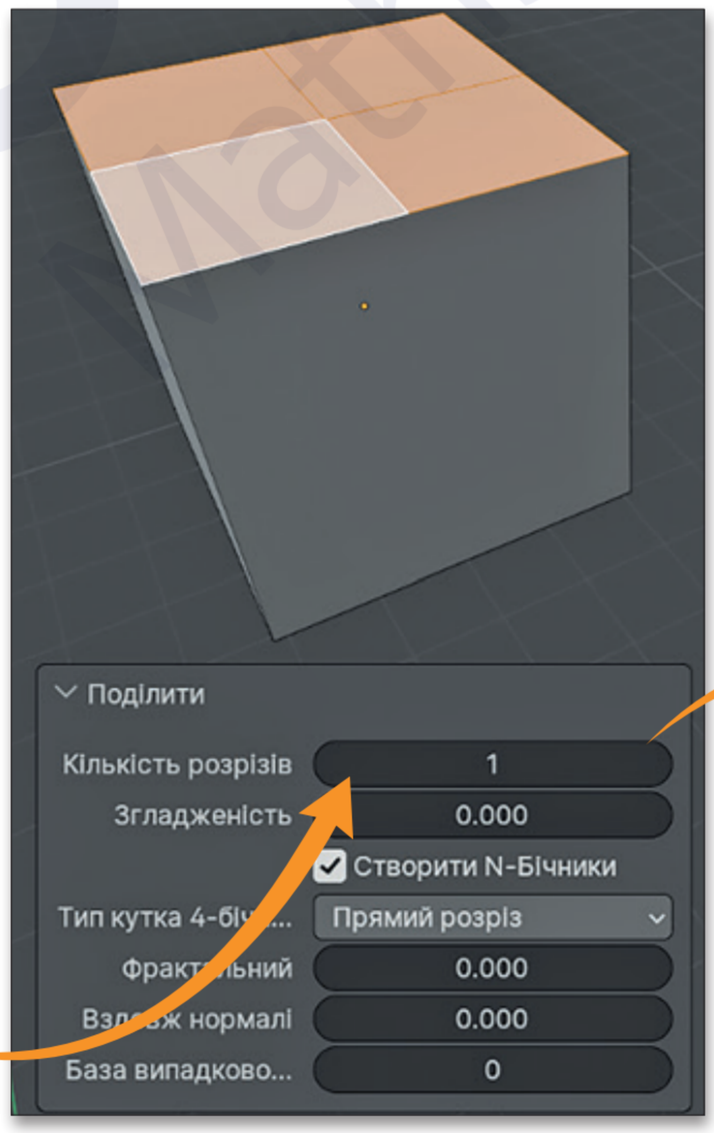


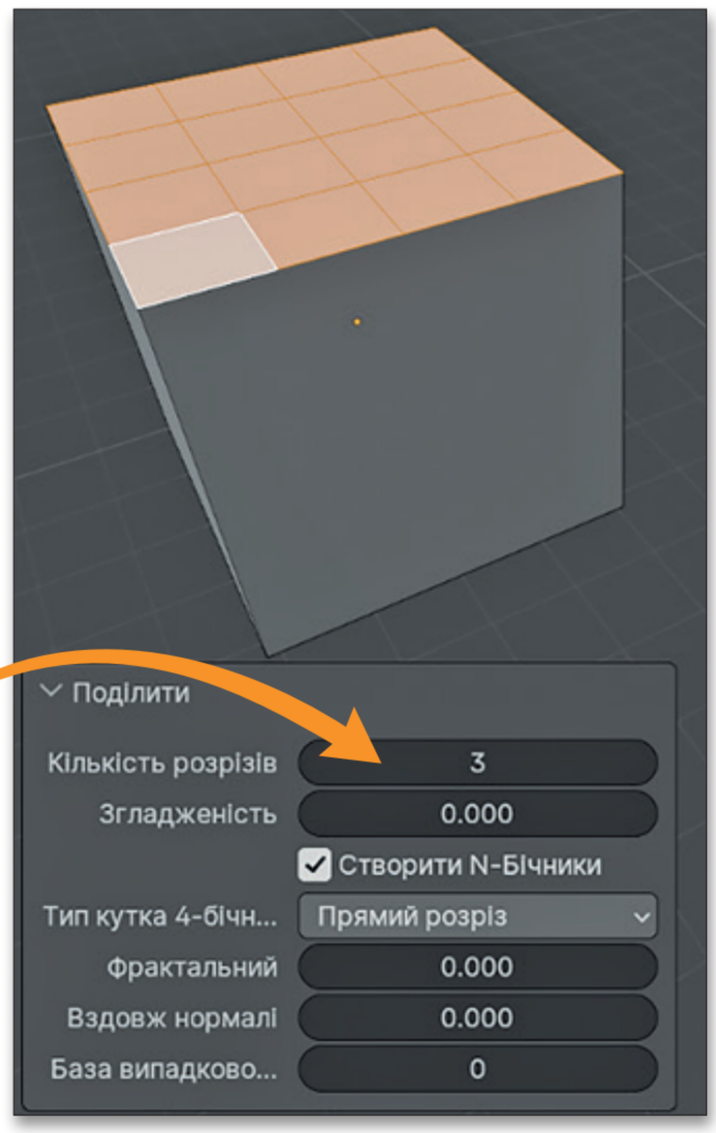
Команда Поділу дає змогу розподілити грань на рівномірні 4 осередки. Кількість та інші параметри можна налаштувати в палітрі останньої дії. Розгляньте дії команди на прикладі верхньої грані куба. Оберіть верхню грань, скористайтеся меню лівої клавіші миші для активації команди **Поділу**.



Алгоритм дії команди для поділу грані куба



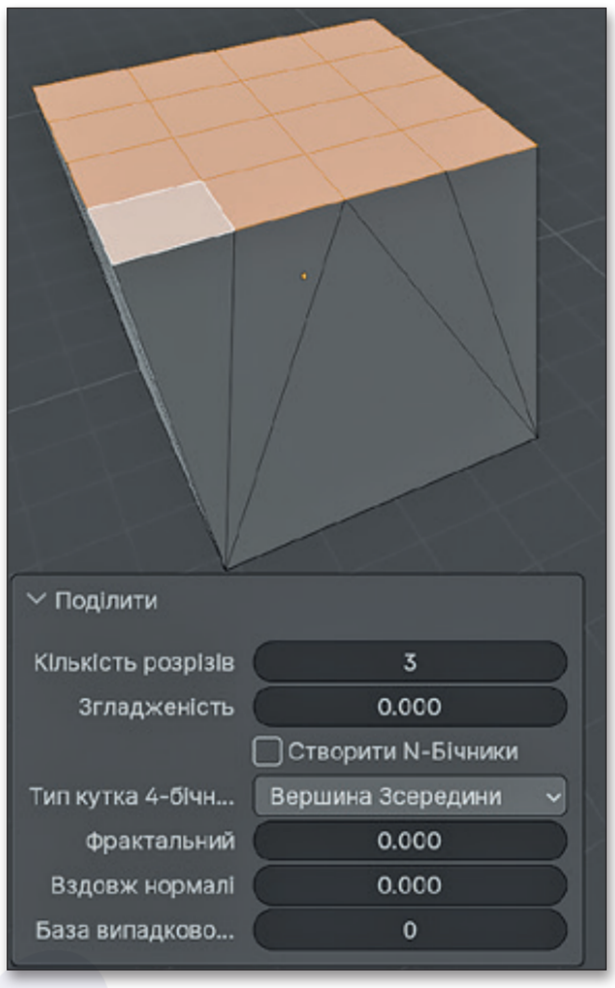




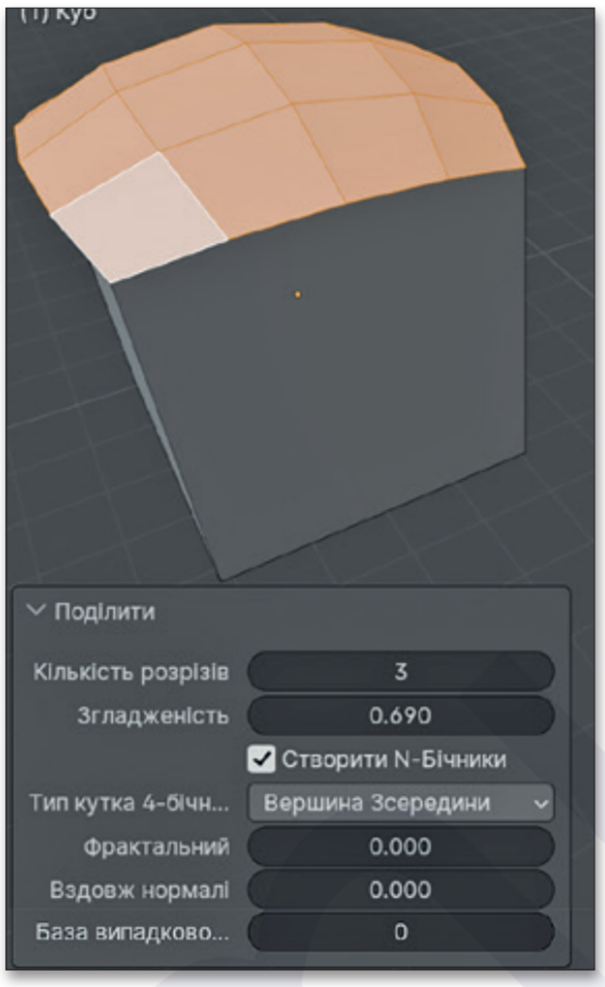
Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

За стандартним налаштуванням команда уможливлює отримання багатокутної геометрії у процесі застосування, але щоб уникнути многокутників, потрібно вимкнути відповідний прапорець у палітрі останньої дії. Для додавання викривлення поділеної поверхні варто задати показник **Згладженість**.

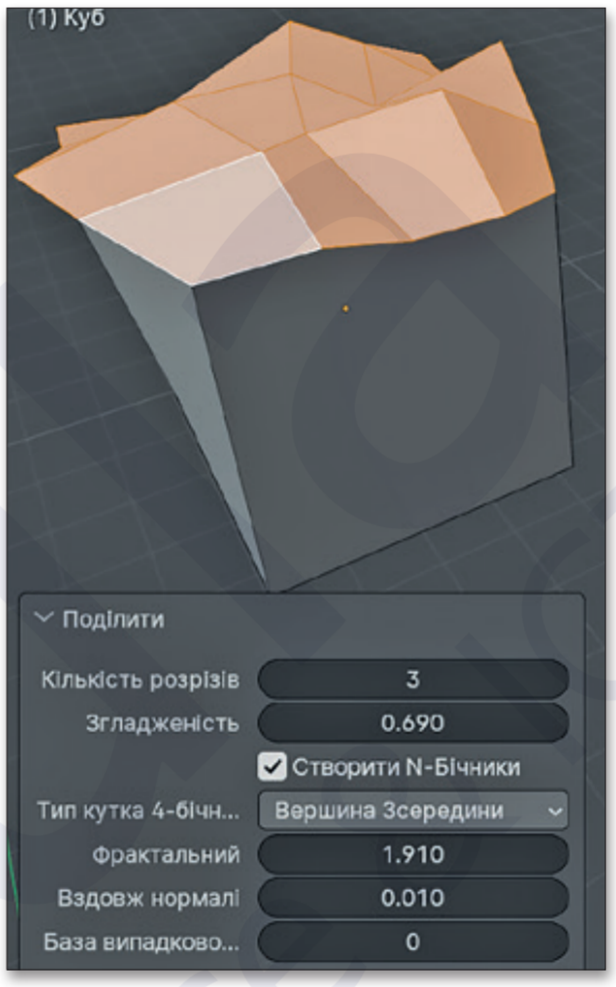
Уникання багатокутників



Викривлення форми



Додавання шумів



Переваги та недоліки полігонального моделювання у Blender

Аспект	Переваги	Недоліки
Універсальність / застосування	Підходить до ігрової розробки та відображення в реальному часі. Моделі легко експортуються	Складно створювати ідеально гладкі вигнуті поверхні та криві
Контроль та точність	Забезпечує повний контроль над кожною вершиною, ребром та гранню	Підтримка правильної топології є складною і вимагає постійної уваги
Продуктивність / оптимізація	Дає змогу створювати моделі з оптимальною кількістю полігонів для анімації та ігор	Для додавання дрібних деталей потрібна надмірна кількість полігонів, що є неефективним
Органічне моделювання	Можна використовувати для створення базових органічних форм (після чого застосовується скульптинг)	Дуже повільний та менш інтуїтивний метод для створення складних органічних форм (м'язи, анатомія)
Анімація	За правильної топології модель плавно деформується під час анімації	Не ефективна топологія призводить до заломів під час деформації моделі

Практичне завдання

Завантажте файл-заготовку зі сторінки інтернет-підтримки. Виконайте запропоноване завдання.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Тема 7
Основи візуалізації у Blender

Введення у візуалізацію

Спробуємо розглянути всесвіт, який створено не з фарб чи реальних матеріалів, а з чисел, координат і математичних формул. Саме такий вигляд має всесвіт 3D-графіки до того моменту, поки його не візуалізують – не перетворять на картинку, яку ми можемо побачити на екрані. Тривимірна візуалізація – це процес, який дає змогу показати глядачеві те, чого насправді не існує. Вона поєднує в собі науку, мистецтво та обчислювальну техніку.

У сфері 3D-графіки візуалізація (або рендеринг) – *це процес генерації фотореалістичного або стилізованого зображення із 3D-сцени, включаючи світло, матеріали й текстури.*



На практиці візуалізація – це ніби фотографування сцени, зібраної в 3D-програмі. Ми створюємо об'єкти, розміщуємо їх у просторі, налаштовуємо камеру, освітлення і натискаємо «рендер» – і насамкінець отримуємо зображення. Але все не так просто. Програма має прорахувати, як світло проходить у сцені, як воно відбивається від об'єктів, як проникає крізь прозорі матеріали, як утворюються тіні. Саме в цьому й полягає суть візуалізації: перетворити тривимірну сцену на двовимірне зображення, ураховуючи всі правила поведінки світла в реальному світі.

Один із найважливіших аспектів – відбиття променів. Промені світла не просто падають на об'єкт – вони відбиваються, заломлюються, створюють відблиски та тіні. Саме тому матеріал металу блищить, скло – прозоре, а пластик на вигляд матовий. Щоб досягти цього ефекту, рендер-рушії використовують спеціальні алгоритми, наприклад трасування променів (ray tracing). Вони імітують шлях кожного світлового променя, що взаємодіє з об'єктами сцени. Така імітація дуже точна, але й ресурсомістка – тому на високій якості рендеринг може тривати години.

Чому візуалізація важлива?

Візуалізація є критично важливою, оскільки людський мозок значно ефективніше обробляє візуальну інформацію, ніж текст або числа. Вона допомагає:

- швидко засвоювати інформацію;
- легше розуміти складне;
- покращувати пам'ять і запам'ятовування;
- приймати рішення на основі наочних доказів.

Схема відбиття променів

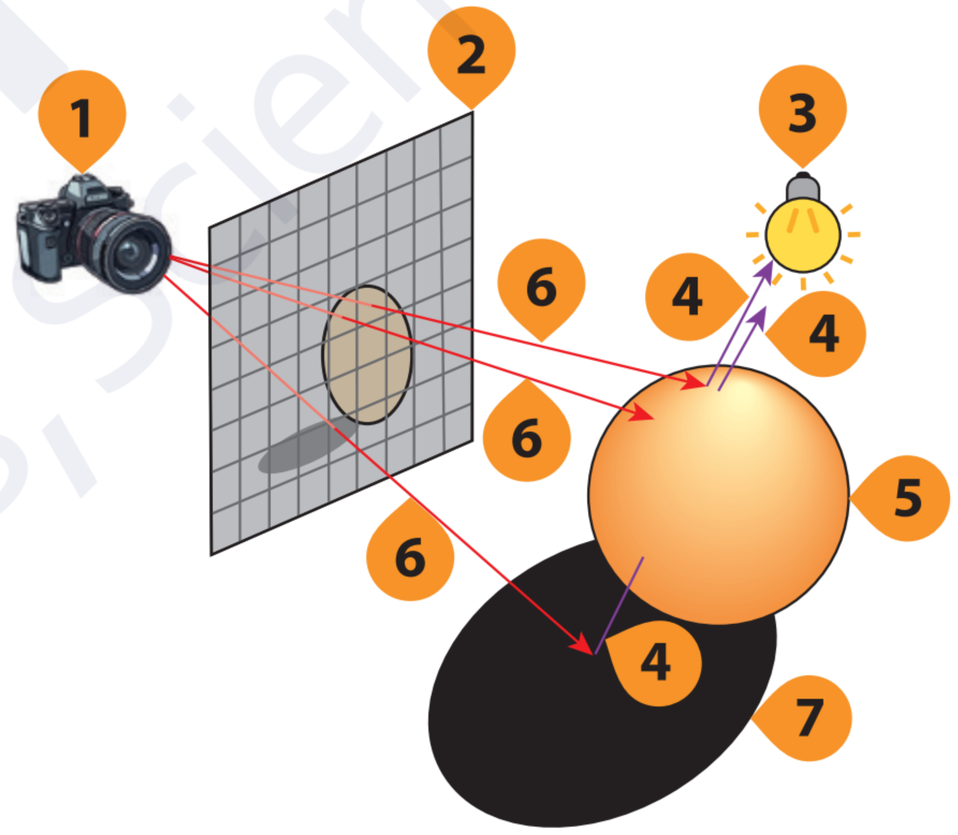


Схема відбиття променів

- Камера 1
- Зображення 2
- Джерело освітлення 3
- Промені тіні 4
- Об'єкт сцени 5
- Промені ракурсу 6
- Тінь від об'єкта 7

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Окремий аспект – це якість фінального зображення. У користувача є можливість налаштувати параметри рендерингу: кількість зразків освітлення, рівень згладжування країв, глибину відбиттів, розмір тіней тощо. Що вищі ці параметри – то реалістичніша картинка, але й більше часу займе її створення. Тому часто доводиться знаходити баланс між якістю та швидкістю.

У сфері дизайну, архітектури, ігор, анімації чи реклами тривимірна візуалізація – це фінальний акорд. Вона уможливлює презентацію ідеї ще до її втілення в реальному світі. Саме завдяки візуалізації ми можемо побачити, який вигляд матиме будинок, персонаж або новий продукт ще до початку виробництва.

Трасування променів (Ray tracing) *– це техніка рендерингу в комп'ютерній графіці, яка генерує зображення, симулюючи фізичну поведінку світла та його взаємодію з об'єктами в тривимірній сцені. Вона забезпечує високий ступінь фотореалізму, відстежуючи шлях окремих променів світла від уявної камери (глядацького ока) назад до джерел світла.*

Системи візуалізації у Blender

У Blender візуалізація (процес створення фінального зображення) здійснюється за допомогою двох основних систем рендерингу.



Основні системи рендерингу

Eevee (Рушій віртуального середовища)	Cycles (Цикли розрахунків)
Це система рендерингу в реальному часі (або майже реальному часі). Вона використовує технології растризації, щоб забезпечити швидку, інтерактивну візуалізацію.	Це фізично коректна система трасування променів (Ray tracing). Вона імітує реальну поведінку світла для досягнення фотореалістичних зображень, що вимагає більшого часу на обчислення.

Вибір системи рендерингу залежить від кінцевої мети проєкту

Сценарій	Рекомендований рендер	Чому?
Швидкі анімаційні відео	Eevee	Швидкий рендеринг дає змогу створювати велику кількість кадрів за короткий час
Архвіз (екстер'єри, інтер'єри)	Cycles	Потрібна висока точність світла, тіней та матеріалів для реалізму
Ігровий контент / активи	Eevee	Імітує графічні конвеєри, які використовуються в ігрових рушіях (Unity, Unreal)
Рендеринг продуктів (Product Viz)	Cycles	Необхідні точні відбиття та глянцеві поверхні
Стилізовані / мультяшні сцени	Eevee	Легко налаштувати нереалістичне освітлення та швидко досягти художнього ефекту швидко

ДОВІДКА

Екстер'єр – це моделювання та візуалізація зовнішнього вигляду об'єкта.
Інтер'єр – це моделювання та візуалізація внутрішнього простору об’єкта.
Архвіз – це моделювання та візуалізація архітектурного проєкту у всіх деталях.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Система візуалізації Eevee

Уявіть собі спритного художника, який малює швидкими, впевненими мазками. Він бачить результат своєї роботи миттєво, прямо на полотні, тому може експериментувати та змінювати задумане без затримок. Це – **Eevee**. Його ім'я нагадує щось швидке та живе, тож ідеально відображає його суть.

Система Eevee використовує метод, схожий на той, що робить відеоігри такими реалістичними, – растеризацію. Це дає змогу блискавично перетворювати складні 3D-об'єкти на пікселі на вашому екрані, водночас ефективно обробляючи світло й тіні. Так ви бачите зміну освітлення, текстуру матеріалів без жодних затримок. За стандартним налаштуванням ця система вже активна, отже саме вона відповідає за перший результат візуалізації.

У чому його перевага? Eevee – вдалий вибір, коли йдеться про швидкість. Час, потрібний для створення зображення, вимірюється секундами, а не хвилинами чи годинами. Це робить його ідеальним для швидкого прототипування, створення динамічних анімацій, де кожен кадр має значення, або для інтерактивних презентацій. Він здатен швидко перевіряти різні художні рішення та експериментувати зі стилем.

Яка ціна цієї швидкості? Хоча Eevee може створювати дивовижні візуалізації, він не є «фізично коректним» рушієм у повному розумінні. Це означає, що розрахунки світла, відбиття та заломлення є більш наближеними. Складні світлові ефекти, як-от каустика (коли світло фокусується, проходячи крізь скло), або дуже тонкі багаторазові відбиття, можуть бути на вигляд менш точними або вимагати додаткових «чаклунств».



Сильна сторона системи візуалізації Eevee *– у стилізації, у створенні мультяшної графіки, комікс-арту чи абстрактних візуалізацій, де не потрібна абсолютна фотографічна точність.*



Система візуалізації Cycles

Cycles – це система візуалізації, що забезпечує фотореалістичний рендеринг шляхом обчислень. Його функціонування базується на трасуванні променів (Ray tracing) та принципах фізично коректного рендерингу (PBR – physical based rendering), що дозволяє детально імітувати поширення світла з урахуванням законів фізики.

Cycles імітує, як світло поводиться в реальному світі. Він буквально відстежує шлях мільйонів світлових променів, що виходять з камери, а потім відбиваються від поверхонь, заломлюються крізь прозорі об'єкти та поглинаються матеріалами, створюючи тіні. Це забезпечує надзвичайно реалістичне освітлення та матеріали.

У чому його перевага? Головний козир Cycles – це якість зображення. Він здатен створювати фотореалістичні візуалізації з дивовижно м'якими та природними тінями, складними відбиттями, ідеальними заломленнями (наче справжнє скло чи вода), а також глобальним освітленням, що наповнює сцену життям та об'ємом.

Яка ціна цієї реалістичності? Час. Це його головний недолік порівняно з Eevee. Рендеринг у Cycles може тривати від кількох хвилин до кількох годин або навіть днів для особливо складних сцен, що вимагають надзвичайно детальних розрахунків світла. Тому він потребує значних обчислювальних ресурсів і терпіння. Проте сучасні технології значно прискорюють цей процес.

Cycles *є незамінним для архітектурних візуалізацій, рендерингу продуктів або будь-яких сцен, де прагнете до максимального наближення до реальності.*



Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Гібридний підхід у роботі

Досвідчені 3D-художники часто використовують обидва рушії в одному проєкті.

Гібридний підхід у роботі

1. **Eevee** використовується **як швидкий інструмент попереднього перегляду (Viewport Shading).** Це дає змогу налаштовувати освітлення, композицію та матеріали без затримки, значно економлячи час

2. Після завершення налаштування фінальний рендеринг для досягнення максимальної якості перемикається на **Cycles**

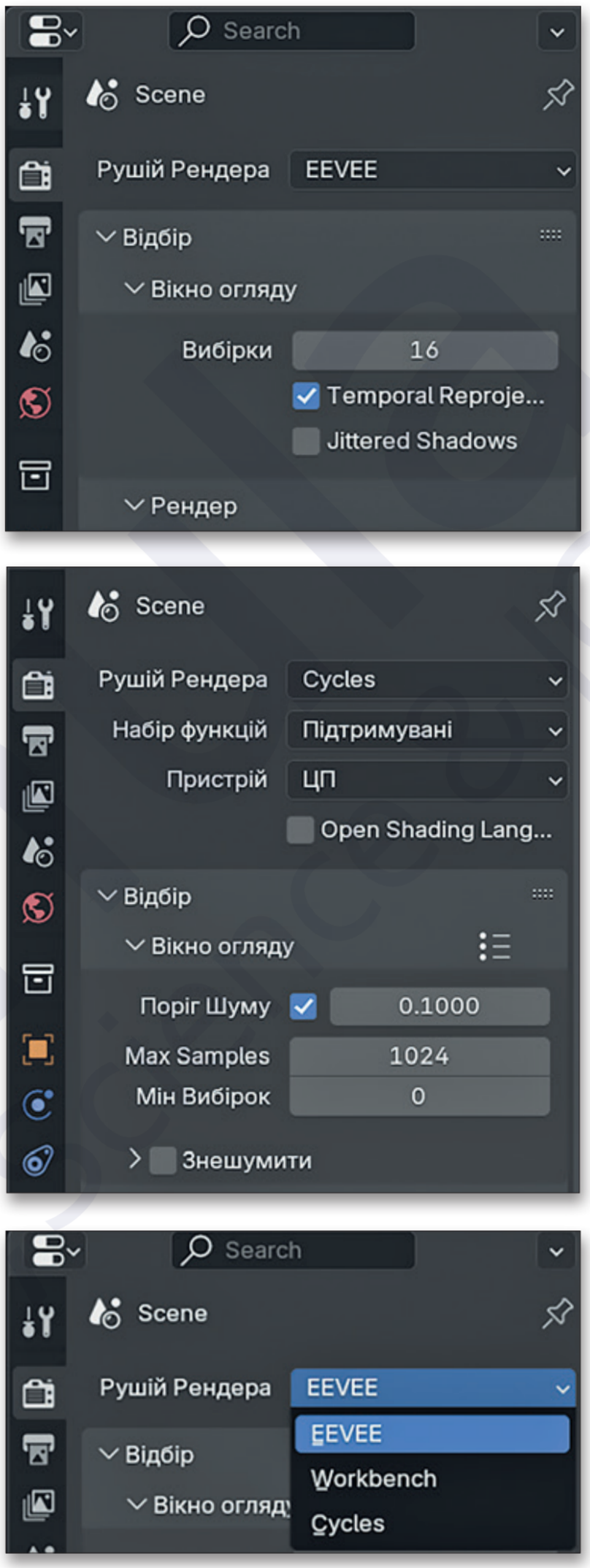
Отже, ці дві системи не конкурують, а **доповнюють** одна одну, забезпечуючи Blender гнучкість, необхідну як для швидкої розробки, так і для фотореалістичного фінального продукту.

Для того щоб побачити різницю між візуалізаціями в різних системах, налаштованих стандартно, розгляньте наступний приклад. На сцені є площина як підлога, одне джерело освітлення і декілька простих тривимірних об'єктів. Спочатку подивіться, який вигляд має візуалізація у перший системі (1), потім – у другій (2) та порівняйте отриманий результат.

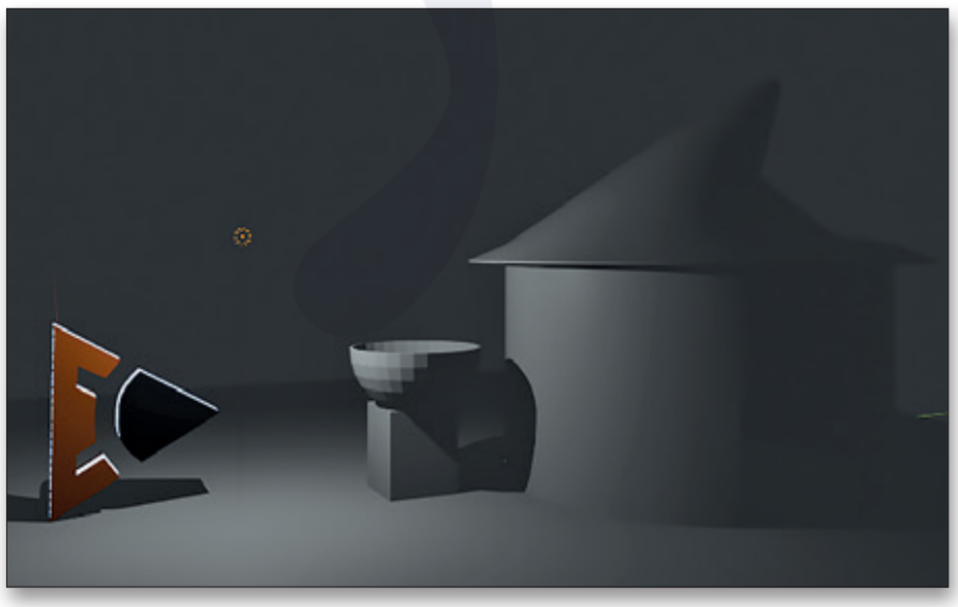
Для перемикання між системами використовуйте вкладку візуалізації (3).

***Eevee** дає швидкий попередній перегляд для налаштування, а **Cycles** – фінальний фотореалістичний рендер для якості.*

3 **Вкладка налаштування систем візуалізації**



1 **Візуалізація у системі Eevee**



2 **Візуалізація у системі Cycles**



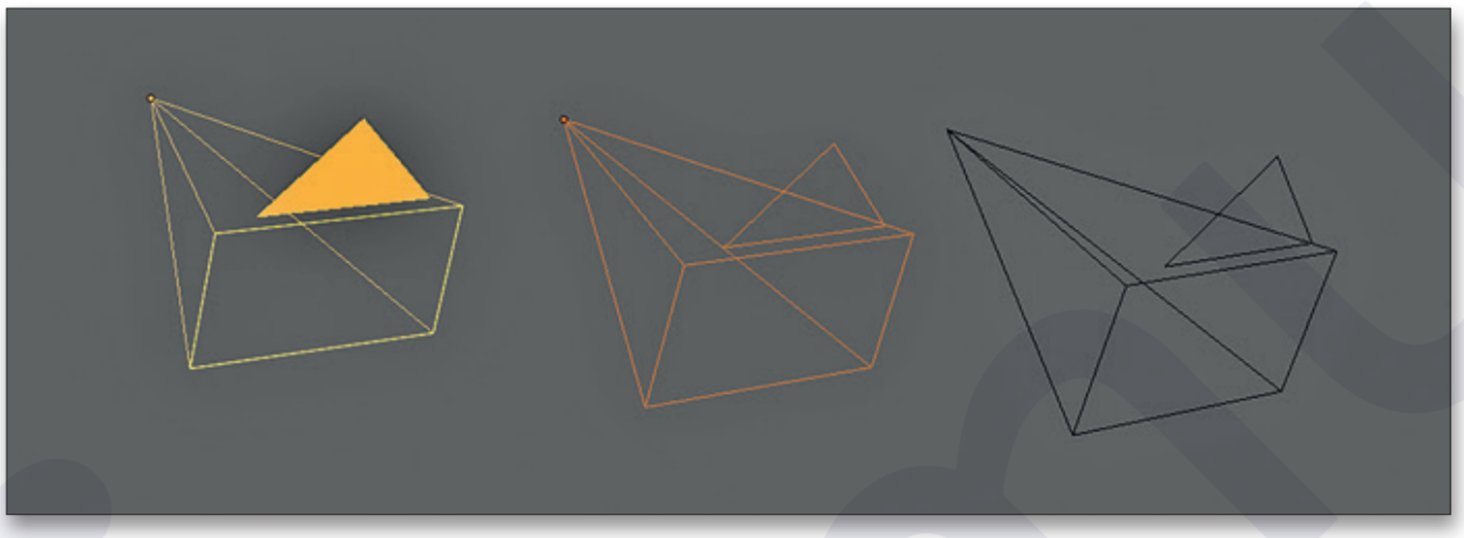
Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Камери: очі вашого віртуального світу



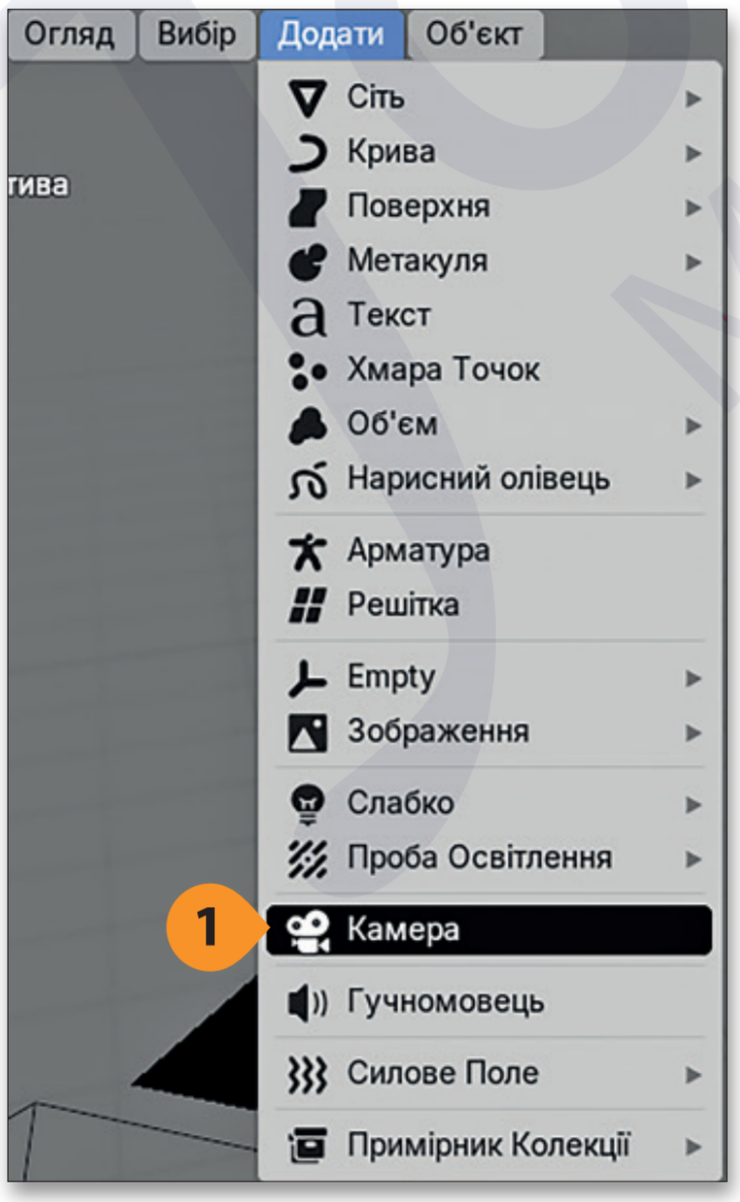
На просторах тривимірного моделювання, де ви створюєте власні світи, персонажів та історії, існує один надзвичайно важливий елемент, без якого ваші творіння так і залишаться невидимими для інших людей. Уявіть собі, що ви збудували дивовижний замок, але у вас немає вікна, щоб показати його красу. Саме таким «вікном» у Blender є камера.

Без використання камери отримати фінальну візуалізацію вашого тривимірного шедевра просто неможливо. Вона є тим єдиним оком, через яке Blender «бачить» сцену, а потім перетворює її на двовимірне зображення – саме те, що ми бачимо на екрані. Камера фіксує ракурс, композицію, освітлення та всі деталі, які ви хочете показати у своєму фінальному рендері. Вона – ваш віртуальний фотограф або оператор.



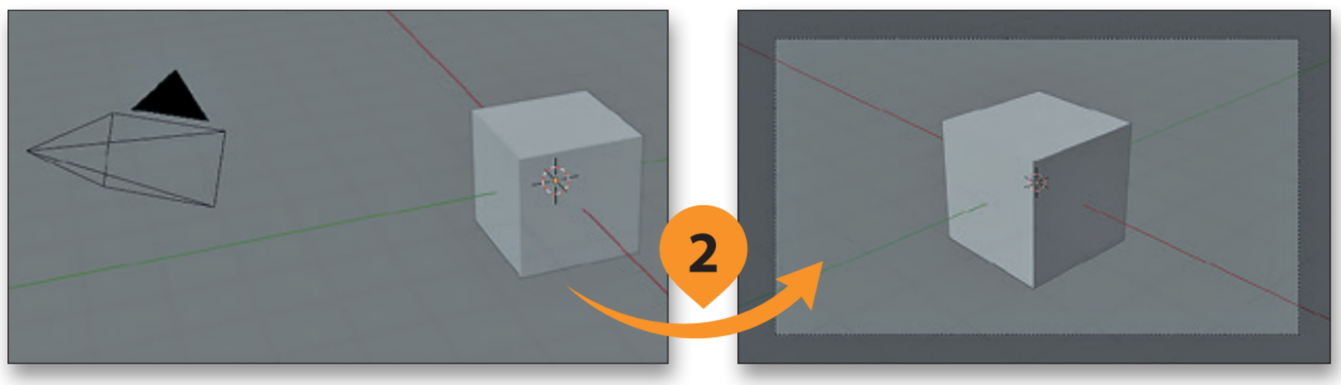
Камера у Blender *– це об'єкт, який визначає точку зору для фінального рендерингу (візуалізації) сцени. Вона є віртуальним аналогом справжньої фото- або відеокамери. Без доданої та активної камери Blender не зможе створити фінального зображення, оскільки рендеринг завжди відбувається через об'єктив активної камери.*

Найпростіший спосіб додати камеру до сцени – це скористатися магічним поєднанням клавіш. Натисніть **Shift** + **A** *(відкриває меню для додавання об'єктів) і перед вами з'явиться цілий список того, що можна створити. Серед нього шукайте пункт* ***Камера****. Клацніть на нього – і ось вона! Нова камера з'явилася у центрі вашої сцени, можливо, трохи невдало розташована, але вона вже тут.*



Якщо ж ви хочете бути більш точними, то можете знайти пункт **Додати** у верхньому меню екрана моделювання та обрати **Камера** (1) там. Результат буде той самий.

Камера вже є, але як побачити світ її очима? Це просто! Натисніть на цифровій клавіатурі клавішу **0** *– і ви миттєво переміщуєтеся з власного вільного огляду в точку огляду* (2)*. Тепер те, що ви бачите у вікні перегляду, є саме тим, що потрапить до фінального рендера.*



Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Налаштування точки огляду: магія композиції

Коли ви перебуваєте в режимі камери, ви можете вільно переміщати її, щоб знайти потрібний ракурс. Для цього вам знадобиться функція **Камера за оглядом (Заблокувати камеру до вигляду)**. Це як приклеїти камеру до вашої голови, і куди ви дивитеся, туди дивиться й вона.

Тепер, коли ви обертаєтеся, наближаєтесь або віддаляєтесь у своєму вільному режимі перегляду (використовуючи колесо миші для наближення / віддалення та середню кнопку миші для обертання), камера буде рухатися разом з вами! Це надзвичайно зручно для швидкого пошуку ідеального кадру.

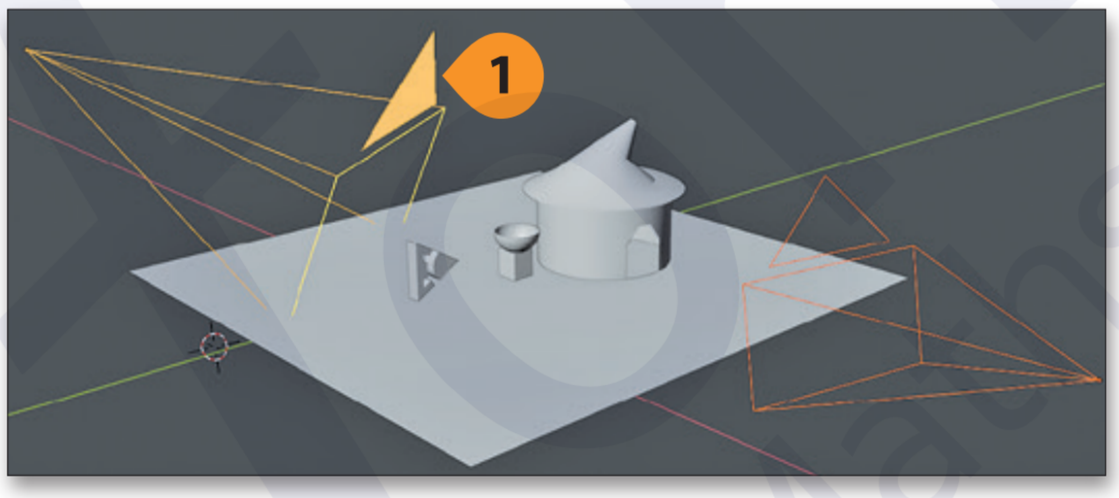
Ви можете додавати декілька камер, щоб отримувати візуалізації різних ракурсів тривимірної сцени. Налаштування вигляду робиться для активної камери. Активна камера помічена заповненим помаранчевим трикутником (1).

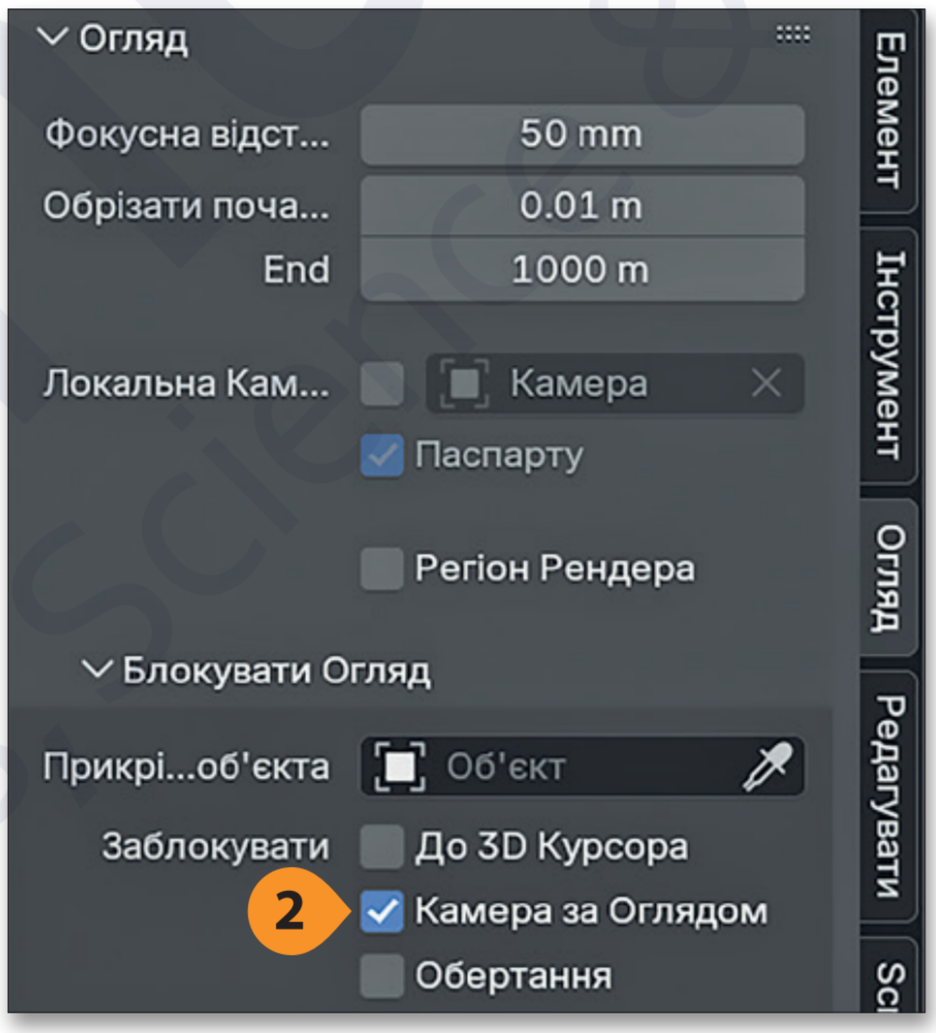
Щоб знайти потрібний ракурс у режимі камери

- натисніть клавішу **N** – збоку з'явиться панель **Бічна панель**;
- перейдіть до вкладки **Огляд (View)**;
- знайдіть пункт **Камера за оглядом (Lock camera to view)** (2) і встановіть прапорець;
- коли ви знайшли потрібний ракурс, зніміть галочку з **Камера за оглядом** (2), щоб камера зафіксувала своє положення, а ви могли вільно переміщатися по сцені, не зрушуючи її.

Щоб зробити камеру активною:

- виділіть потрібну камеру;
- оберіть із контекстного меню команду **Задати активну камеру** або скористайтеся «гарячими клавішами» клавіатури **Ctrl** + **0**.





Вплив фокусних відстаней на сприйняття 3D-сцени

Фокусна відстань	Категорія об'єктива	Сприйняття простору та глибини	Типове використання
10 мм – 35 мм	Ширококутний	Створює сильне відчуття глибини та ширини. Лінії дуже сходяться	Пейзажі, архітектурні інтер'єри, динамічні / драматичні кадри
35 мм – 55 мм	Нормальний	Зберігає природні пропорції та відстані між об'єктами	Загальні плани, стандартні композиції, сцени з нейтральним тоном
70 мм – 200+ мм	Телеоб'єктив	Зменшує відчуття глибини; робить сцену пласкішою та сфокусованою	Портрети крупним планом, ізоляція об'єктів, деталізація, віддалені об'єкти

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Освітлення у сцені

У віртуальному світі Blender кожен об'єкт, кожна сцена залишатиметься лише сірою тінню, доки ви не вдихнете в неї світло. **Освітлення** – це не просто додавання яскравості; це «душа» вашої сцени, яка визначає її настрій, об'єм, деталі та емоції. Без світла ваш шедевр ніколи не оживе.

Уявіть, що ви художник, який малює картину. Кольори, тіні, відблиски – усе це створюється завдяки світлу. У Blender ви маєте повний контроль над цим віртуальним світлом, використовуючи різні його джерела, кожне з яких має свій унікальний характер і призначення.

Освітлення, або світло у Blender – *це ключовий елемент 3D-сцени, який імітує фізичну поведінку світла для створення візуальної атмосфери, підкреслення форми об'єктів (через тіні та відблиски) і забезпечення фінального реалізму під час рендерингу.*

Можливості налаштування освітлення у сцені

- Налаштування параметрів фону
- Установлення та налаштування джерел освітлення
- Використання HDRI-карт

Керування фоном сцени

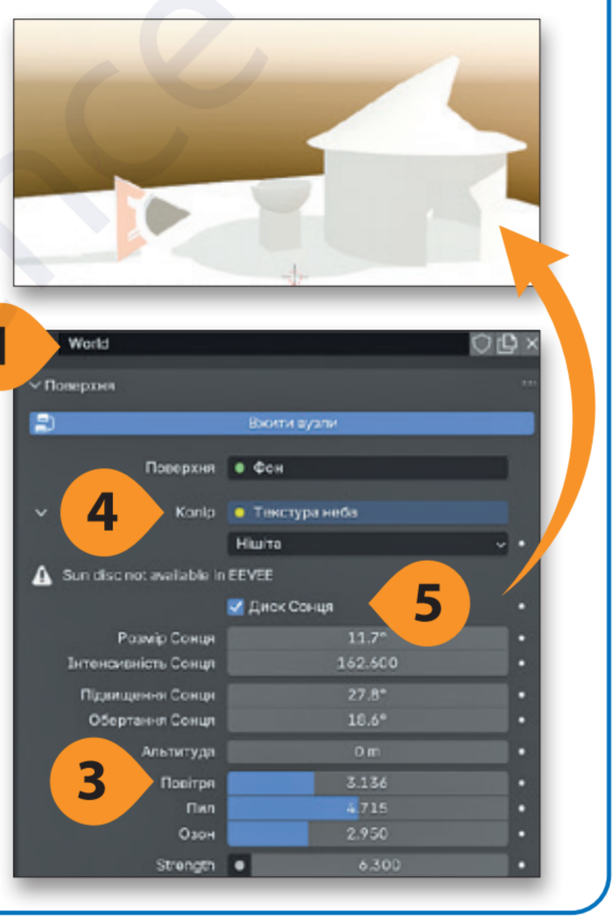
Уявіть, що ваша сцена – це театр під відкритим небом. Перше, що ви можете контролювати, це саме небо, навколишнє середовище, яке обгортає все. Це називається **фоном сцени**, або **Світ (World)** (1).

Ви можете просто обрати один колір для всього оточення. Світло, що виходить з цього кольору, буде рівномірно освітлювати всі об'єкти, немов вони розташовані у великій кольоровій кулі. Змінюючи колір на блакитний, ви отримаєте відчуття дня, на темно-синій – ночі, на помаранчевий – заходу сонця.

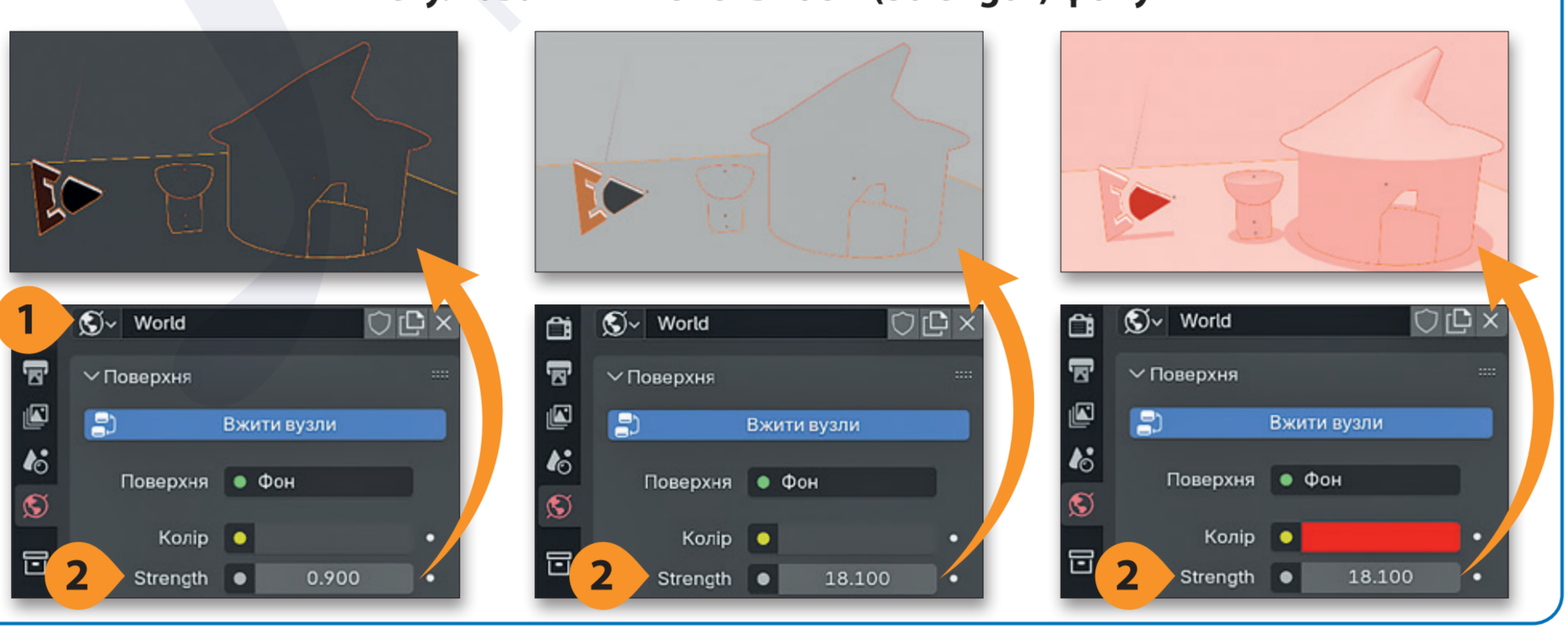
Регулюючи **Інтенсивність (Strength)** (2), ви робите цей фон яскравішим або тьмянішим, ніби крутите ручку регулювання загального освітлення. Це найпростіший спосіб задати загальну атмосферу та глобальне освітлення.

Крім фону, також можна використовувати вже налаштоване середовище оточення (3) або текстуру неба (4). Регулюючи відповідні параметри **Сонця та оточення** (5), можна отримувати цікаві результати.

Регулювання параметрів Сонця та оточення



Регулювання Інтенсивності (Strength) фону

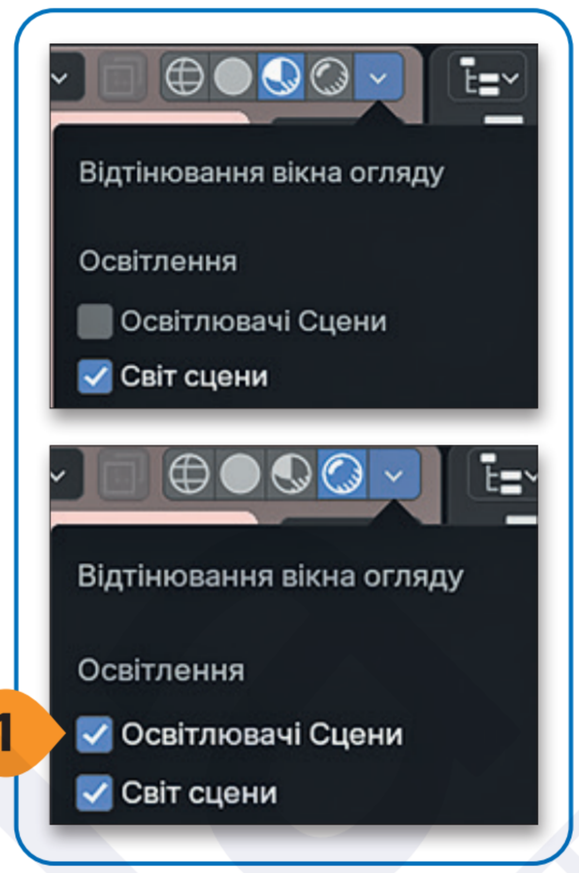


Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Джерела освітлення у сцені

Якщо фон сцени задає загальний настрій, то окремі джерела освітлення – це актори, які виступають на авансцені, виділяючи головних героїв і створюючи драматичні тіні.

Існує 4 джерела освітлення, які ми можемо використовувати. Це точкове, сонце, прожектор та площа. Розглянемо їхню дію, але спочатку нагадаємо, що відображення результатів дії променів можна побачити лише в режимах відображення візуалізації з увімкненим відповідним перемикачем (1).



Джерела освітлення у Blender

- Точкове освітлення
- Сонце
- Прожектор
- Площа

Точкове світло (Point light)

Це як одна єдина лампочка в кімнаті. Додайте її, і ви отримаєте світло, що розходиться в усі боки з однієї точки. Переміщуйте її – і вона буде акцентувати різні частини об'єкта, створюючи різкі тіні, що розсіюються.

Щоб додати точкове світло (Point light):

- скористайтеся «гарячими» клавішами клавіатури та оберіть відповідні пункти в меню (2)

Shift + **A** → **Слабко** → **Точкове**

Сонце (Sun light)

Додайте віртуальне сонце, і ви отримаєте паралельні промені, що падають на всю сцену. Це ідеально для імітації денного світла. Обертайте **Сонце**, щоб змінити час доби, – і тіні будуть видовжуватися чи скорочуватися, як у реальному житті.

Щоб додати сонце (Sun light):

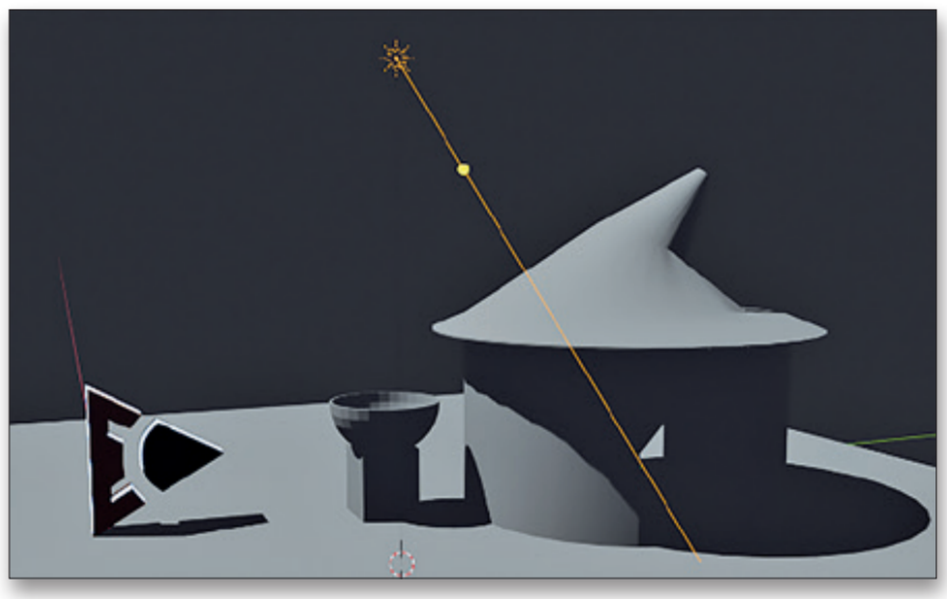
- скористайтеся «гарячими» клавішами клавіатури та оберіть відповідні пункти в меню (3)

Shift + **A** → **Слабко** → **Сонце**

Точкове світло (Point light)



Сонце (Sun light)



Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Прожектор (Spot light)

Це ваш театральний прожектор. Він випромінює світло конусом, і ви можете спрямувати його на конкретний об'єкт, регулюючи розмір цього конуса та м'якість його країв. Ідеально для виділення деталей або створення драматичних ефектів.

Площа (Area light)

Це як велике студійне джерело світла або вікно. Додайте його, і ви отримаєте м'яке, розсіяне світло з великої площі. Що більша площа, то м'якшими будуть тіні. Це незамінний інструмент для створення природного освітлення, особливо для інтер'єрів або портретів.

Щоб додати прожектор (Spot light):

- скористайтеся «гарячими» клавішами клавіатури та оберіть відповідні пункти в меню (1)

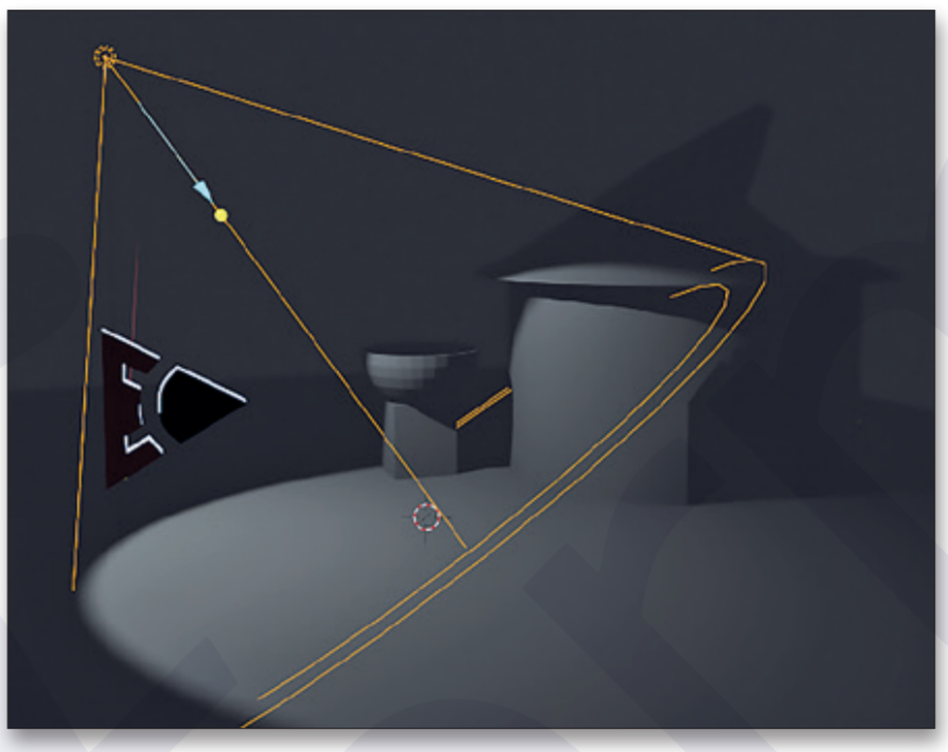
Shift + **A** → **Слабко** → **Прожектор**

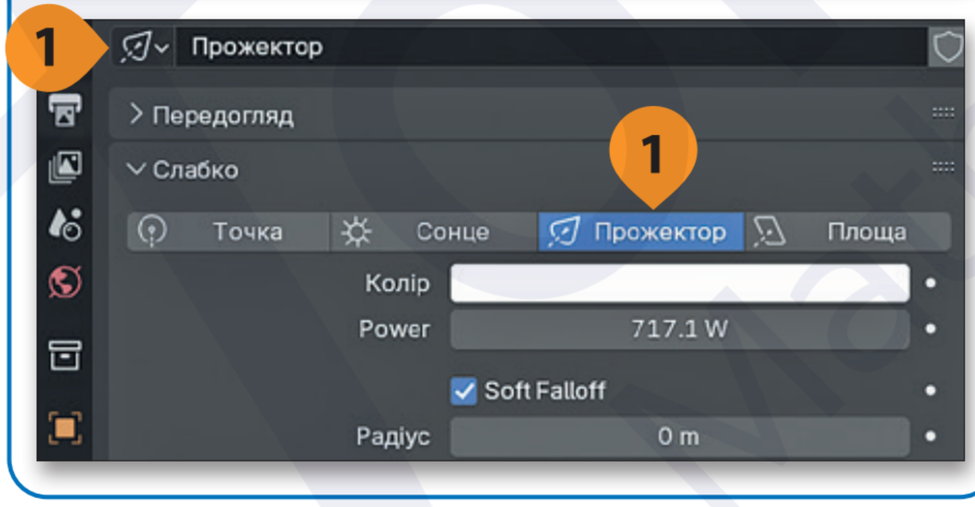
Щоб додати площу (Area light):

- скористайтеся «гарячими» клавішами клавіатури та оберіть відповідні пункти в меню (2)

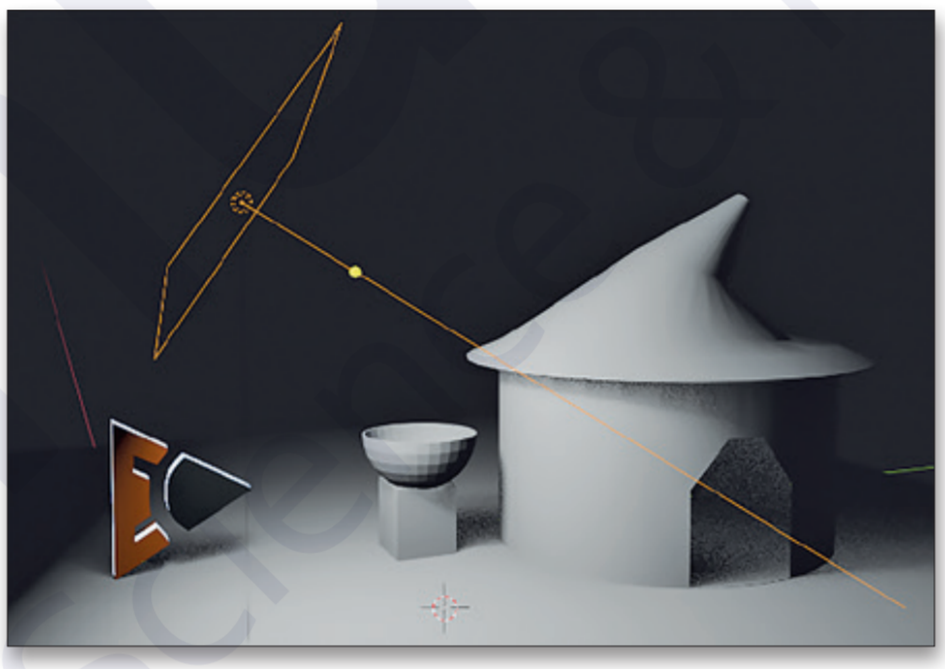
Shift + **A** → **Слабко** → **Площа**

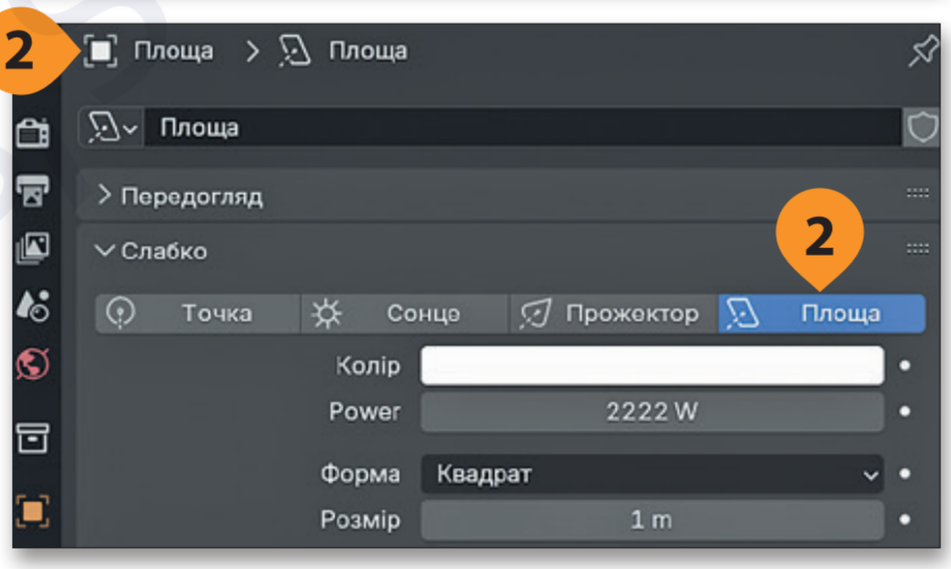
Прожектор (Spot light)





Площа (Area light)





Комбінуючи ці джерела світла, ви можете створити складні схеми:

Ключове світло	Заповнююче світло	Контурове світло
Для головного освітлення	Для пом'якшення тіней	Для підсвітки контуру об'єкта

ДОВІДКА

Комбінування ключового (основного), заповнюючого (пом'якшення тіней) та контурового (підсвічування контуру) світла дає змогу створити об'ємну та драматичну світлову схему, ідеально підкреслюючи форму об'єкта.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

HDRI-карти

А тепер – справжня магія, яка здатна перетворити вашу сцену на частинку реального світу: **HDRI-карти (High Dynamic Range Imaging)**. Уявіть, що ви не просто обираєте колір для фону, а буквально «вставляєте» у свою сцену фотографію освітлення.

HDRI-карта фіксує не просто кольори, а й інтенсивність світла від кожної точки реального світу. Це означає, що вона «запам'ятовує» положення сонця, колір неба, світло від хмар, відбиття від будівель – усе, що впливає на освітлення в природному середовищі.

Як це працює у **Blender**? Ви не додаєте **HDRI-карту** як звичайну текстуру на об'єкт. Ви завантажуєте її у розділ **Властивості світу**, замінюючи простий колір фону на текстуру оточення. Далі обираєте відповідну карту, натиснувши на кнопку **Відкрити**. **Blender** тоді використовує інформацію з цієї карти, щоб освітлювати вашу сцену з усіх боків.

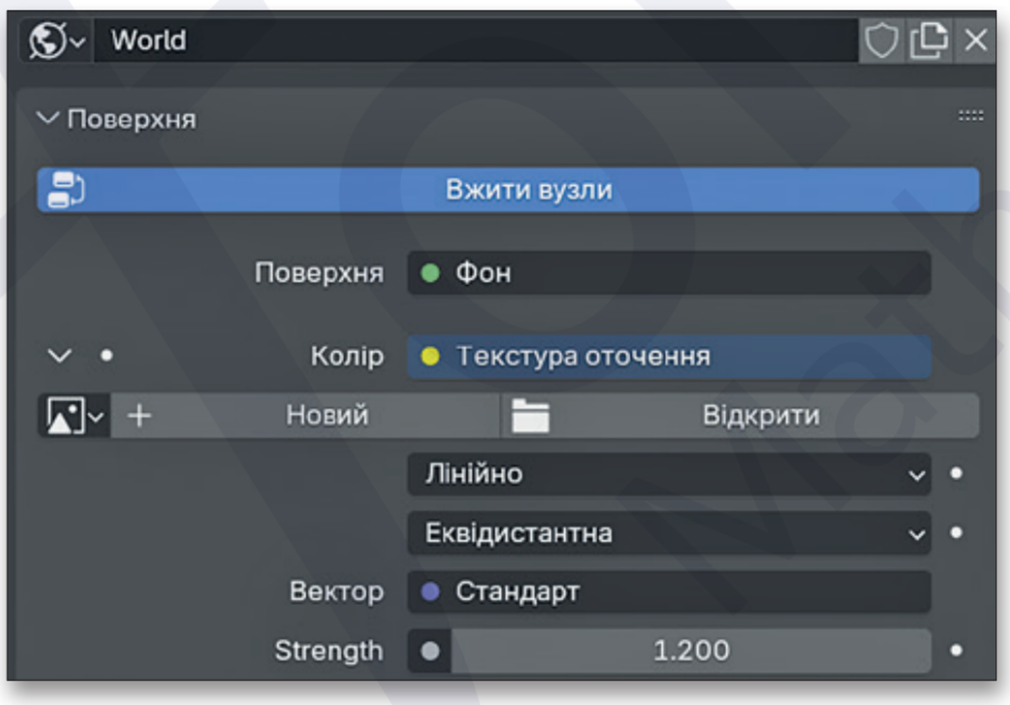
Результат? Напрочуд реалістичне освітлення, тіні та відблиски, які ідеально відповідають середовищу, зображеному на **HDRI-карті**. Якщо на карті сонячний день у лісі, ваша сцена матиме такий вигляд, ніби вона стоїть у тому лісі, пустелі або іншій локації.

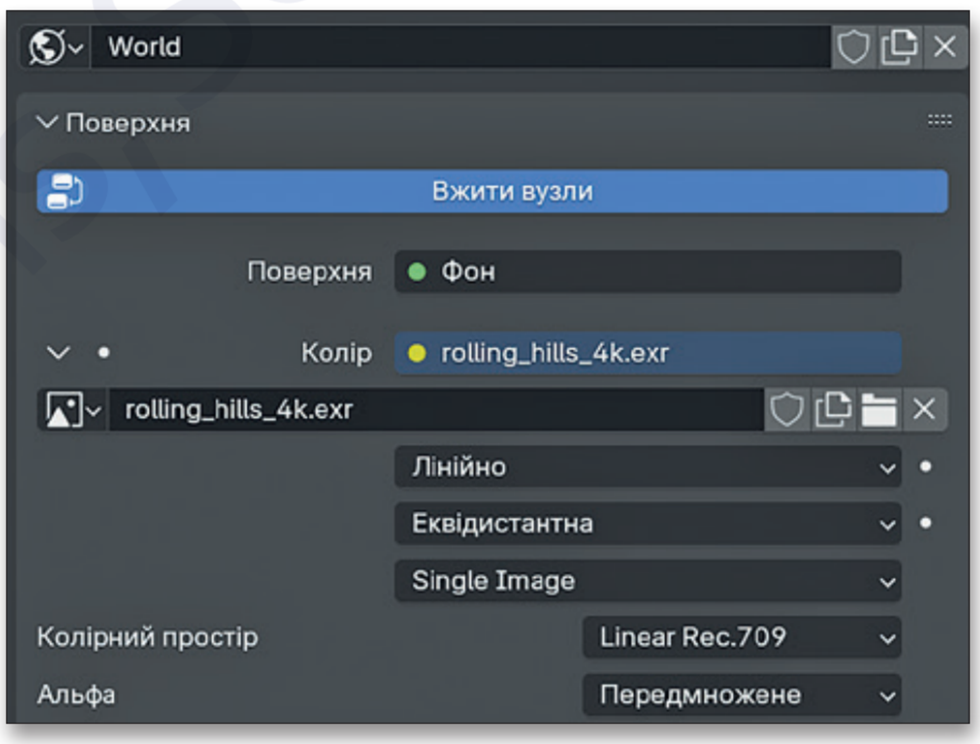
HDRI-карта – *це панорамне зображення, яке містить надзвичайно широкий діапазон світлових значень.*









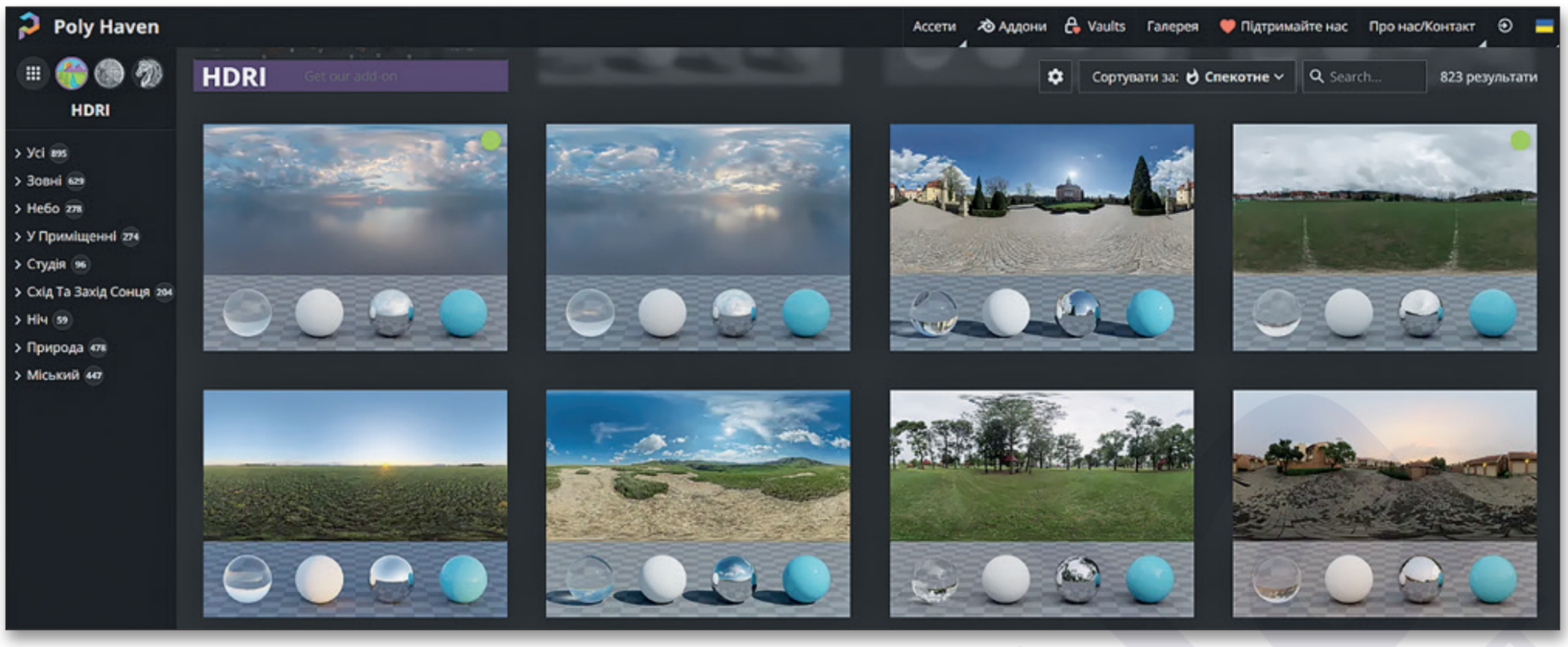


Щоб додати HDRI-карту у Blender:

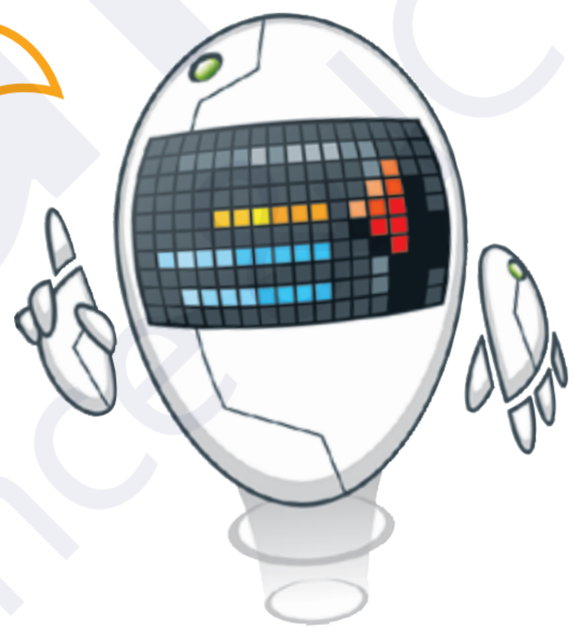
- завантажте її у розділ **Властивості світу**;
- оберіть відповідну карту, натиснувши на кнопку **Відкрити**.

*Що вища роздільна здатність текстури (або карти), то більше даних потрібно завантажити в оперативну та відеопам'ять, і ще більше часу може знадобитися для її обробки рушієм рендерингу. На слабких комп'ютерах або в роботі з дуже високою роздільною здатністю (наприклад, 32K+ на відеокарті з 4 ГБ пам'яті) це може призвести до уповільнення роботи в'юпорту та збільшення часу рендерингу, особливо якщо ви використовуєте рушій **Cycles**.*

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.



*Якщо ви зацікавились і вже шукаєте місце, де можна знайти безліч якісних **HDRI-карт** абсолютно безкоштовно, тоді сайт **Poly Haven** – це вдалий вибір. Він схожий на величезний архів, де зібрані панорамні зображення з найрізноманітніших куточків світу та студій: від яскравих сонячних днів на пляжі до похмурих вечірніх вулиць міста, від затишних інтер'єрів до космічних просторів.*



Завжди звертайте увагу на ліцензію використання **HDRI-карти**. Хоча **Poly Haven** пропонує вільні ліцензії (CC0), інші сайти можуть мати обмеження, особливо якщо ви плануєте використовувати свої рендери для комерційних проєктів. Краще перестрахуватися, ніж потім мати проблеми.

HDRI-карти є одним з найпотужніших інструментів для досягнення фотореалізму, зокрема в рушіях, подібних до **Cycles**. Вони миттєво додають сцені глибини, правдоподібності та оточення, що було б дуже важко відтворити за допомогою лише окремих джерел світла.

*Для чорнових рендерів, швидкого тестування в **Eevee** або на слабкому ПК достатньо 2K або 4K **HDRI-карти**. Вони забезпечать загальну атмосферу, але не будуть надто навантажувати систему.*

Матеріали

За стандартним налаштуванням усі об'єкти сцени, які ви «ліпите» з полігонів або сплайнів, спочатку мають непоказний вигляд – сірі, безликі, без власної ідентичності. Але є художній етап, на якому ви можете надати їм текстури, кольору, блиску, шорсткості, – це використання матеріалів. Уявіть собі, що ви не просто створюєте форму, а «надягаєте» на неї віртуальну шкіру, яка визначає, як об'єкт взаємодіятиме зі світлом.

Кожен предмет у реальному світі має свої унікальні властивості поверхні: дерево на вигляд інакше, ніж метал, а скло – інакше, ніж тканина. Матеріали у **Blender** допомагають імітувати ці властивості, наближаючи вигляд віртуальних об'єктів до їхніх справжніх аналогів. Без належних матеріалів навіть найскладніша модель залишиться лише сірою масою.

Коли ви додаєте новий об'єкт до сцени у **Blender** і застосовуєте до нього матеріал (або якщо він уже має матеріал за стандартним налаштуванням), ви майже завжди бачите перед собою **Принциповий BSDF (Principled BSDF)**. Це не просто стандартний матеріал, а універсальний, який вміє імітувати величезну кількість різних поверхонь, просто змінюючи кілька своїх параметрів.

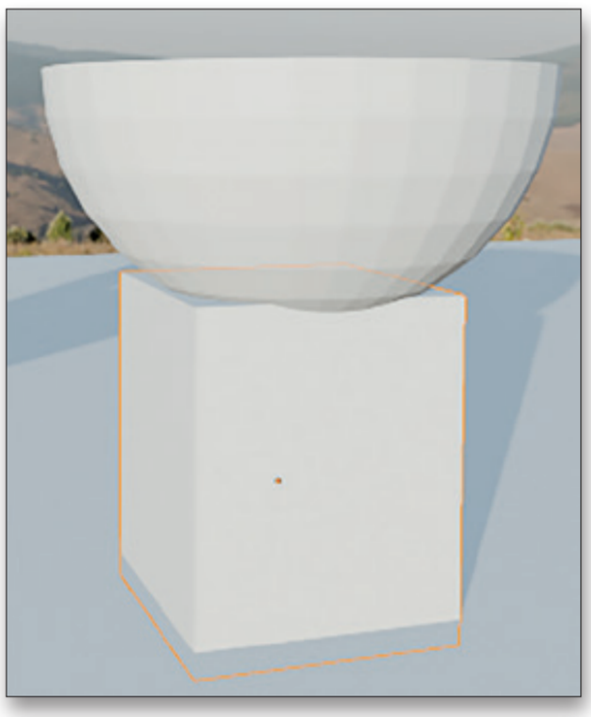
*Акронім **BSDF** розшифровується як **Bidirectional Scattering Distribution Function (двонаправлена функція розподілу розсіяння)**. Звучить складно, але насправді це означає, що цей матеріал точно описує, як світло падає на поверхню, розсіюється, відбивається чи поглинається, а потім повертається до камери.*

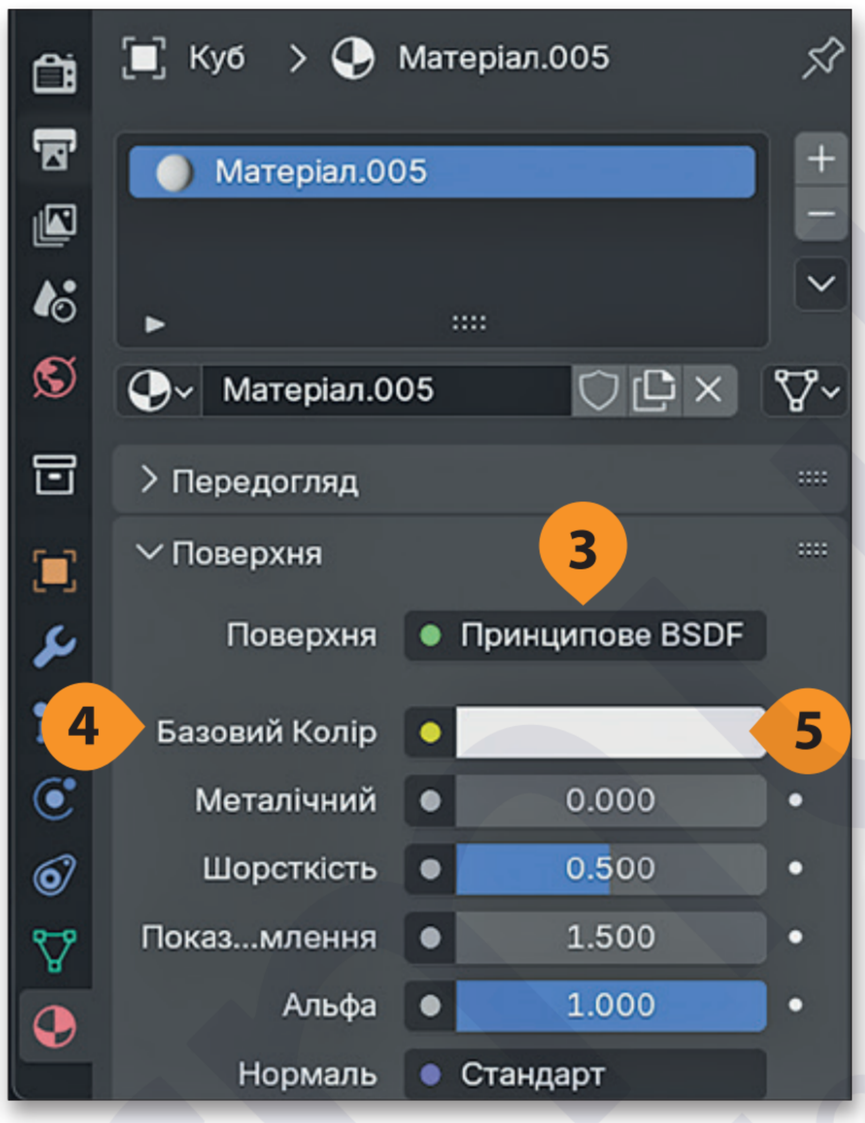
Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Спробуйте просто пофарбувати коробку у сцені. Для цього виділіть коробку й у бічний панелі перейдіть до вкладки матеріалів. Початково матеріалу у створеного об'єкта не буде. Тому додайте новий матеріал, натискаючи на кнопку з **плюсом** або на кнопку **Новий**. Буде додано матеріал **Принципове BSDF** (3).

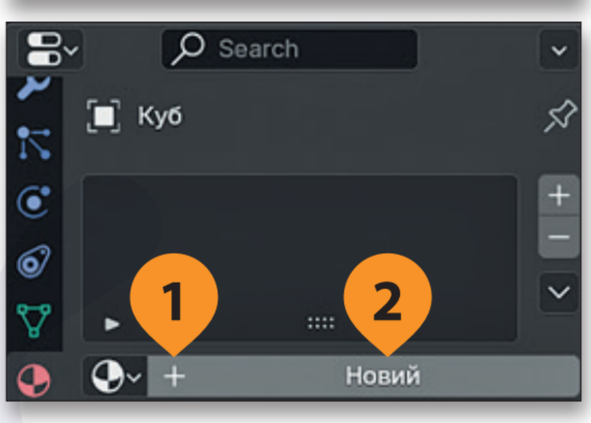
Щоб додати новий матеріал у Blender:

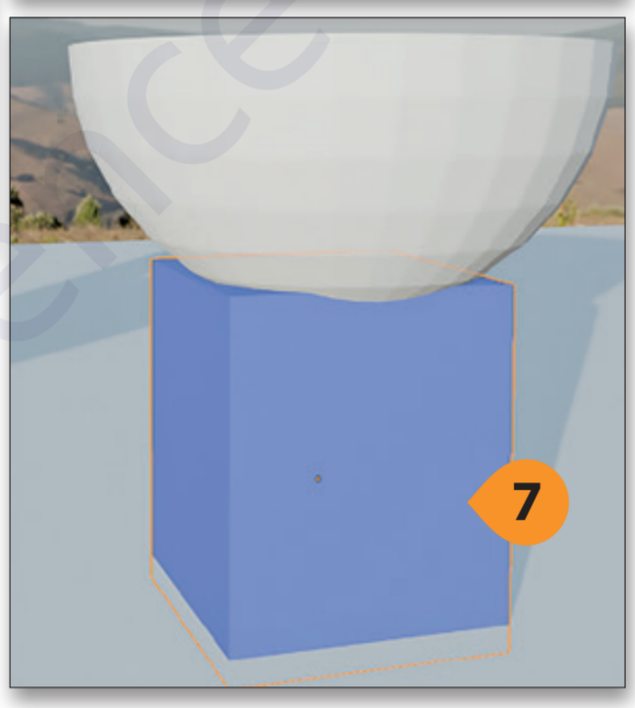
натисніть на кнопку з плюсом (1) або на кнопку **Новий** (2).







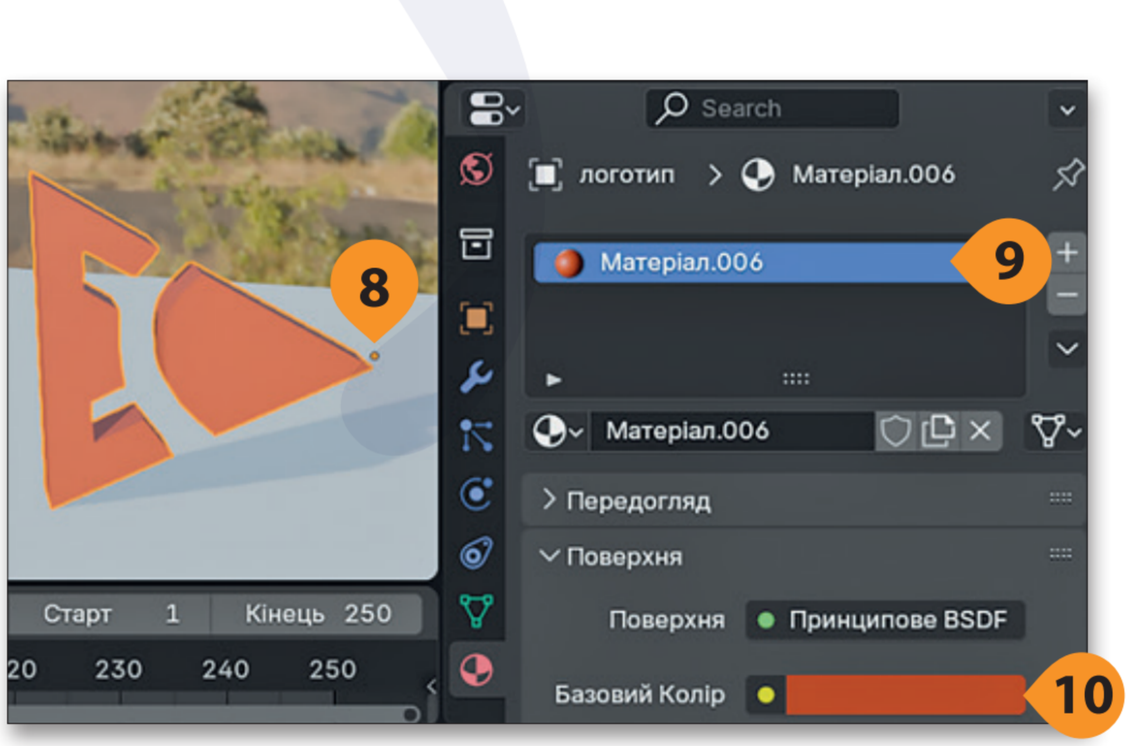


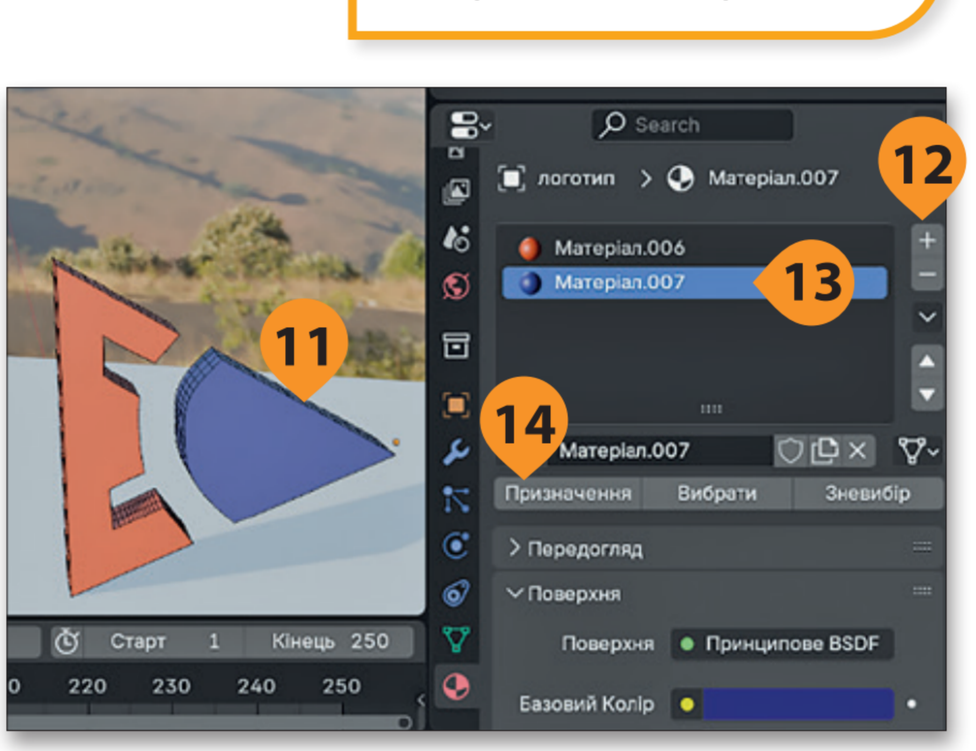


Для того щоб змінити колір, скористайтеся пунктом **Базовий колір** (4), де відкриється вікно кольорової палітри (5), і тоді оберіть потрібний колір (6). Якщо в оглядовому вікні обрано режим відображення кольорів чи візуалізації – зміна кольору буде відбуватись у реальному часі наочно (7).

А тепер завдання ускладнюється. Наприклад, логотип розфарбуйте кількома кольорами. Виділіть логотип (8), додайте матеріал (9) і змініть колір на помаранчевий (10). Після чого активуйте режим редагування і виділіть частку, яку треба розфарбувати в інший колір (11). Щоб додати наступний колір, натисніть **плюс** (12) у палітрі матеріалів. Виберіть темно-блакитний колір (13) та натисніть кнопку **Призначення** (14).

Таким способом можна один об'єкт розфарбувати різними кольорами, які налаштовуємо в окремих матеріалах.





Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Матеріалу притаманна велика кількість властивостей, які варто враховувати під час налаштувань для відтворення реалістичного вигляду поверхні об'єкта.

Правильне поєднання цих параметрів є основою для створення будь-якого матеріалу – від іржавого металу до матового пластику чи вологого бетону.

Властивості матеріалу

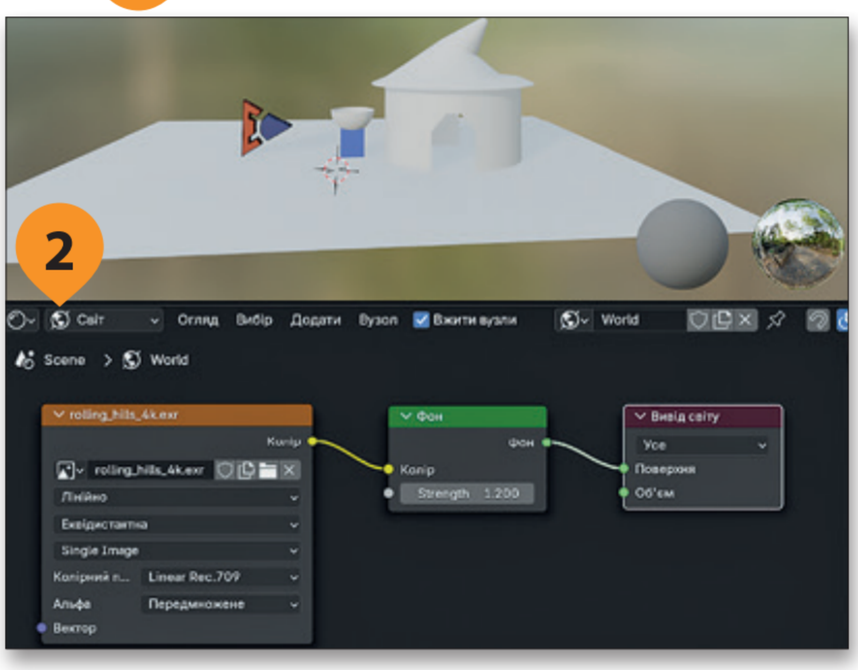
Властивість (Property)	Призначення	Вплив на реалізм
Базовий колір (Base Color)	Визначає основний колір об'єкта	Фундаментальний колір, який відбивається від об'єкта
Металевість (Metallic)	Визначає, чи є матеріал металом (1.0) або діелектриком (0.0)	Ключовий параметр: метали не мають дифузного кольору, їхній колір визначається відбиттям
Шорсткість (Roughness)	Контролює, наскільки розсіяним є світло, що відбивається від поверхні	Визначає гостроту відблисків. Низька шорсткість → гладка, дзеркальна поверхня Висока шорсткість → матова поверхня
Дзеркальність (Specular)	Контролює загальну інтенсивність дзеркального відбиття світла (відблисків)	Визначає, наскільки сильно поверхня відбиває світло
Прозорість (Transmission)	Визначає, наскільки світло проходить крізь об'єкт	Використовується для створення скла, води та інших прозорих матеріалів
Емісія (Emission)	Дає змогу матеріалу самостійно випромінювати світло (незалежно від зовнішніх джерел)	Створює ефект свічення, наприклад, для екранів або ламп
Нормалі (Normal Map)	Використовує текстуру для імітації дрібних деталей поверхні (зморшок, пор, подряпин) без додавання реальної геометрії	Додає реалістичну деталізацію та ілюзію об'єму під час попадання світла

Вузельна система

Усі ці параметри налаштування матеріалів, а також їхнє поєднання, контролюються через **Вузельну систему (Node editor) Blender**. Уявіть собі схему, де кожен блок **(Вузол)** виконує певну функцію (наприклад, додає колір, змішує текстури, застосовує шорсткість), а лінії з'єднують їх, показуючи потік інформації. **Принциповий BSDF** – це один великий вузол з усіма цими входами. Ви можете підключати до нього текстури, процедурні генератори або інші вузли, щоб зробити матеріал ще складнішим та унікальним.

Наприклад, ми можемо перемкнутись у режим редактора **матеріалів (Шейдерів)**, який має назву **Шейдинг (Shading)** (1). Й одразу побачимо матеріал оточення, який застосовано до **Світу сцени** (2). У попередньому зразку як фон використовується карта оточення.

1 Шейдинг (Shading)



Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

На перший погляд, **Вузельний редактор** може здатися трохи заплутаним: купа коробочок і ліній. Але насправді це інтуїтивно зрозумілий і надзвичайно логічний спосіб мислення.

Саме тут ми радимо вам створювати вигляд поверхонь об'єктів. Замість того, щоб просто обирати колір, ви можете створити складну мережу вузлів, яка визначатиме, як світло взаємодіє з поверхнею, які текстури на ній відображаються, наскільки вона блискуча чи шорстка, чи є на ній рельєф і так далі.

Для утворення реалістичного матеріалу, як-от цегли чи дошок, треба мати відповідно налаштований матеріал з потрібною **Текстурою**. **Текстура** досить часто розробляється на базі фотографії поверхні.

Уявіть, що кожен ***Вузол (Node)*** *– це маленька фабрика або трансформатор. Він отримує якусь інформацію на вході, обробляє її, а потім видає результат на виході.*

Лінії (Links), *що з'єднують вузли, є шляхами, якими ця інформація рухається.*



Фото



Текстура



Чим текстура відрізняється від звичайного фото поверхні?
Текстура *– це завжди плоске зображення, частіше карта або відповідно підготовлене якісне деталізоване зображення певного матеріалу, на який ми дивимося під кутом 90°.*

Уявіть, що ви бажаєте покрити величезну стіну цеглою. Якщо ви просто візьмете одну фотографію цеглини й повторите її багато разів, ви побачите чіткі межі, де одна картинка закінчується і починається інша. Це має дуже неприродний вигляд і видає, що це комп'ютерна графіка.

Отже, щоб уникнути таких виглядів поверхонь, використовуються **Безшовні текстури**.

Безшовні текстури створюються або вручну (що вимагає великої майстерності у фоторедакторах), або за допомогою спеціальних програм, які аналізують краї зображення і «зшивають» їх, або шляхом фотографування матеріалів у контрольованих умовах з подальшою обробкою.

Безшовна текстура *– це зображення, яке спеціально підготовлене так, що коли ви розміщуєте його поруч із самим собою (повторюєте його), краї зображення ідеально збігаються. У такому разі немає видимих швів, ліній або розривів у візерунку. Це створює ілюзію нескінченної, безперервної поверхні.*

Фото



Текстура



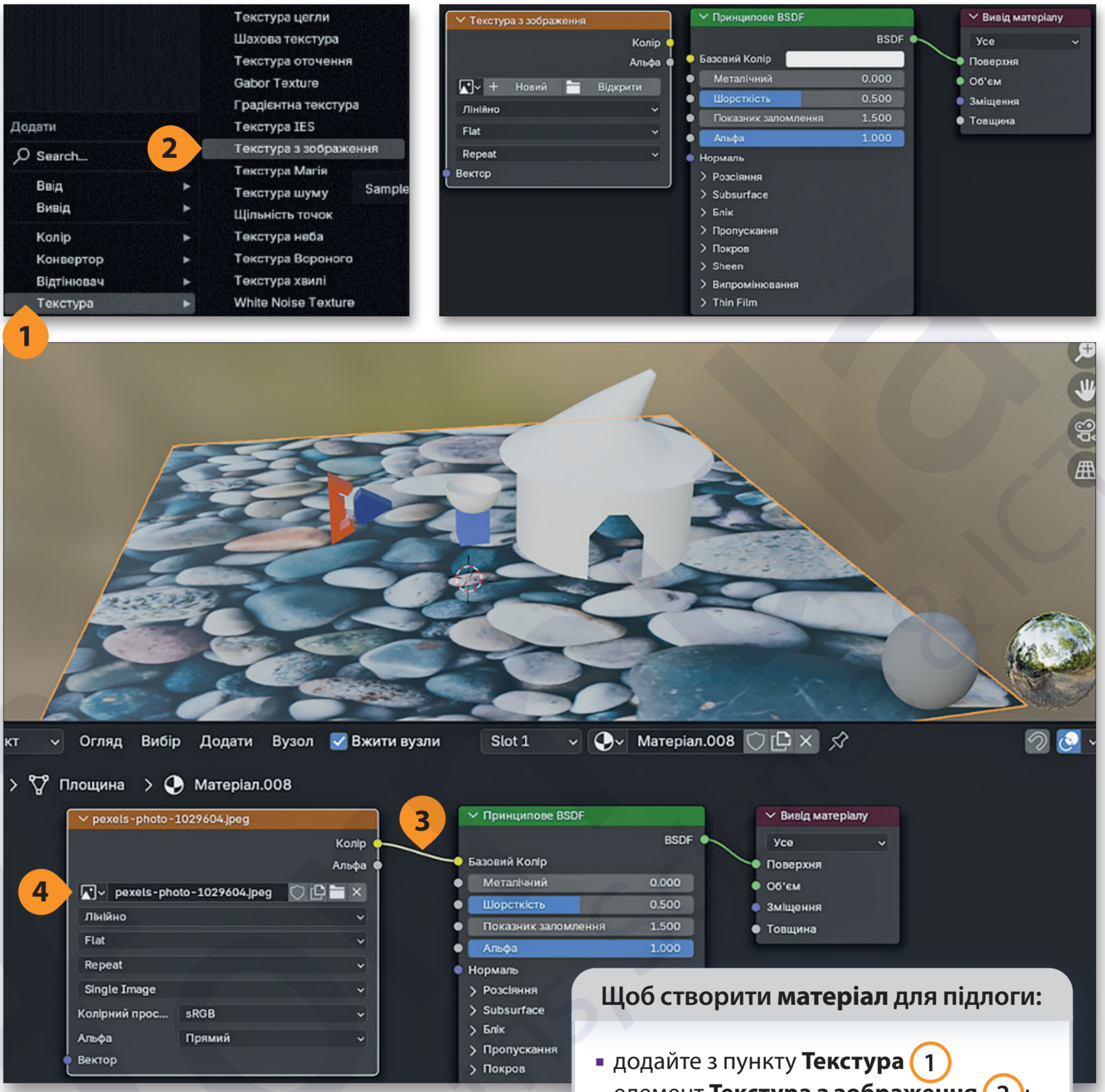
Фото



Текстура



Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.



Спробуйте додати на підлогу локації **Текстуру каміння**. Спочатку на відповідному ресурсі шукайте **Безшовну текстуру каміння**, або **Матеріал**. Потім створюйте матеріал для підлоги.

Щоб створити матеріал для підлоги:

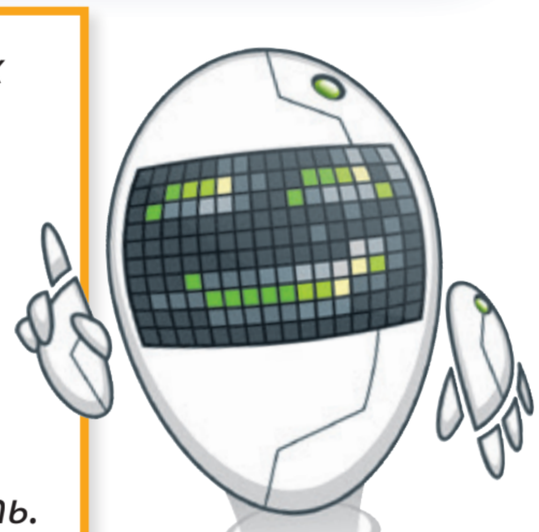
- додайте з пункту **Текстура** (1) елемент **Текстура з зображення** (2);
- поєднайте колір в елементі **Текстура** з кольором **матеріалу BSDF** (3);
- завантажте **Текстуру каміння** в **елемент Текстура** (4), натиснувши на кнопку меню **Відкрити**.

Для пошуку високоякісних текстур для використання в 3D-програмах існує багато онлайн-ресурсів. Ці сайти пропонують як безкоштовні, так і платні текстури, часто в комплекті з усіма потрібними картами.

***Poly Haven** – велика бібліотека високоякісних текстур, моделей та HDRI-зображень. Усі матеріали безкоштовні під ліцензією, що уможливлює їхнє використання у будь-яких комерційних проєктах.*

***Quixel Megascans** – одна з найбільших і найякісніших бібліотек фотограмметрії у світі. Текстури часто мають найвищу реалістичність.*

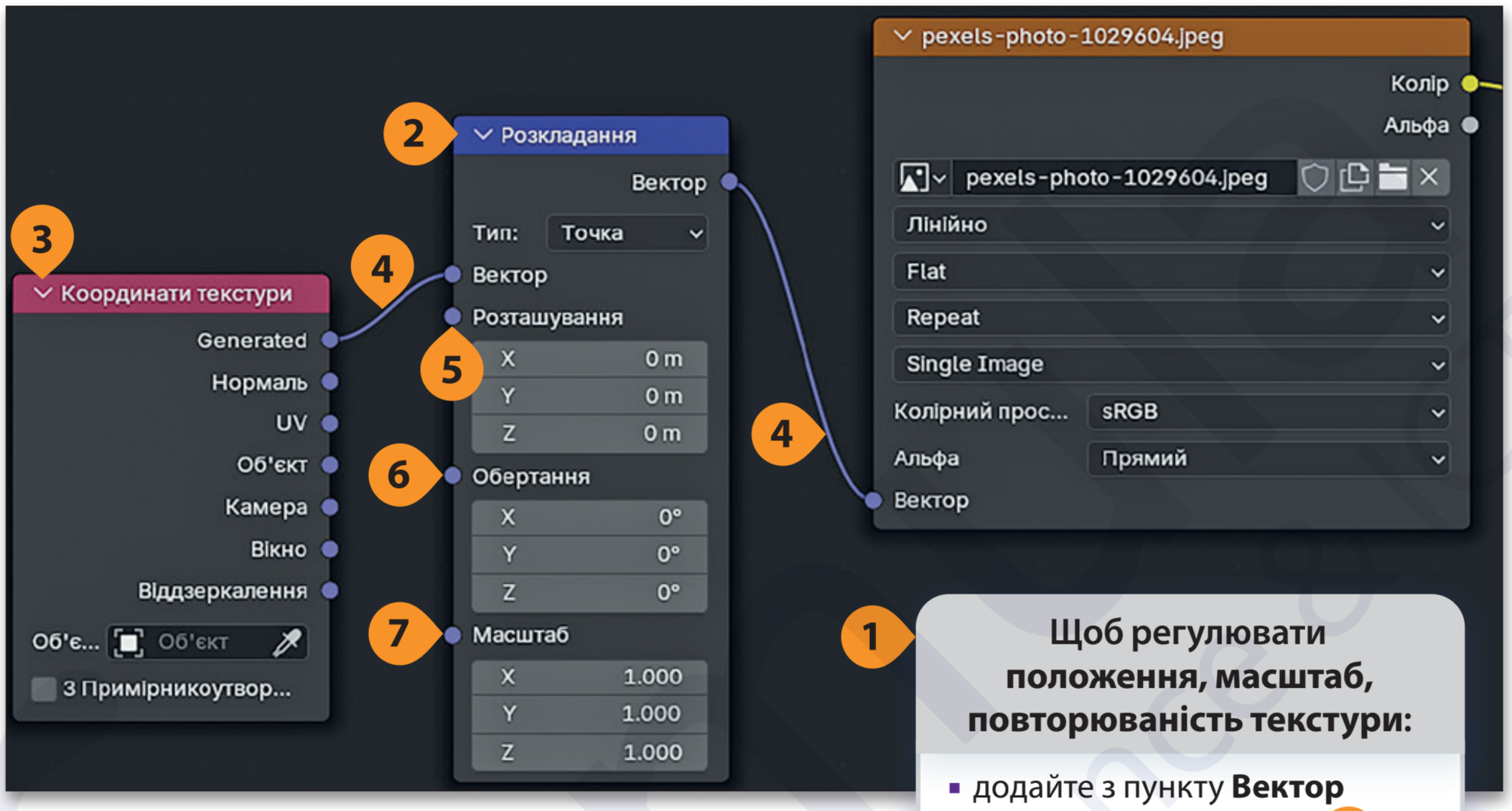
***GameTextures.com** – сфокусований на PBR-текстурах для розробки ігор.*



Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

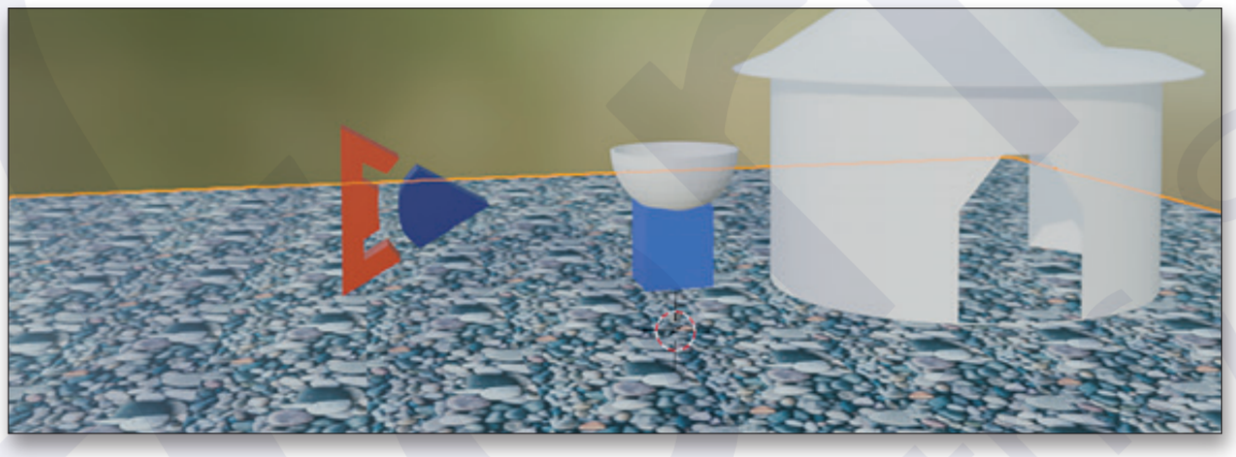
Таким способом **Текстура каміння** застосувалась до поверхні на підлозі. Для того щоб регулювати положення, масштаб, повторюваність текстури на об'єкті, треба зробити ще 2 кроки (1).

Отже, елемент **Розкладання** (2) дає змогу зміщати текстуру за пунктом **Розташування** (5), обертати за пунктом **Обертання** (6) та повторювати за пунктом **Масштаб** (7). Зменште каміння на підлозі у 5 разів, відповідно змінивши значення у пункті **Масштаб**.



1 Щоб регулювати положення, масштаб, повторюваність текстури:

- додайте з пункту **Вектор** елемент **Розкладання** (2). Він відповідає за розташування. Але текстура більше не відображається, тому що зникла розгортка;
- додайте елемент **Координати текстури** (3) з пункту **Ввід**. Усе поєднується через вхід з назвою **Вектор** (4).

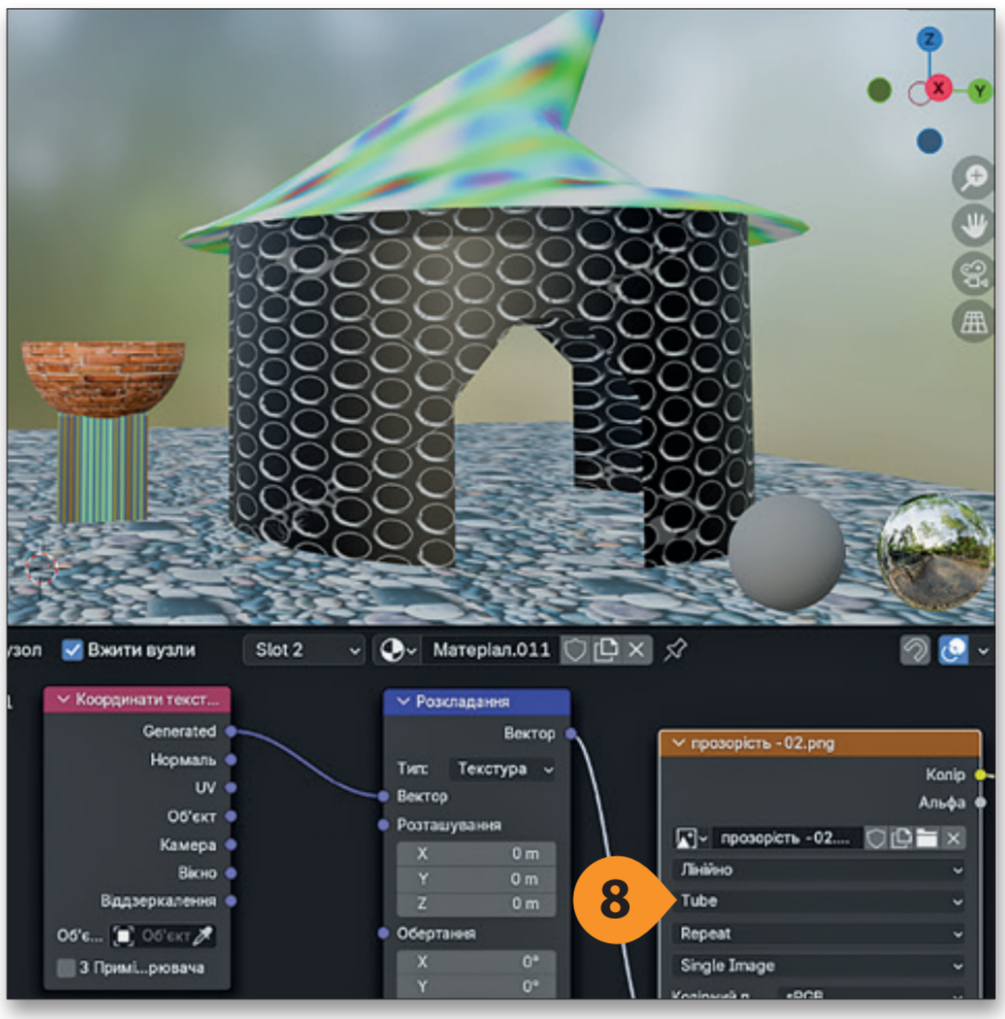


Налаштування прозорості

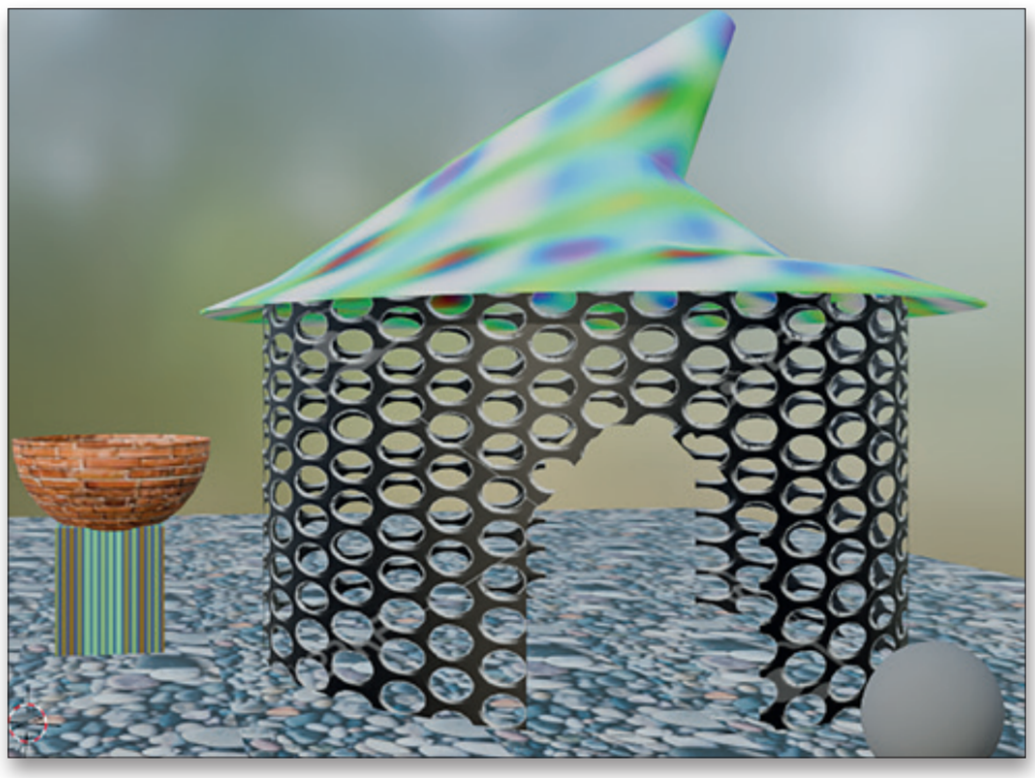
Спробуте налаштувати точкову прозорість, яка має назву **Альфа-канал**, або просто **Альфа**. Узагалі-то прозорість підтримувати може не кожний растровий документ. Ви можете використовувати формати **PNG, TIF, GIF** із найбільш розповсюджених форматів, що підтримують прозорість. Отже, знайдіть або розробіть у растровому редакторі текстуру з прозорістю. Збережіть у форматі **PNG** та накладіть цю текстуру на альтанку.

У налаштуваннях текстури тип розподілу зображення по поверхні треба обрати **Труба (TUBE)** (8), інакше розкладення не буде збігатися із формою об'єкта.

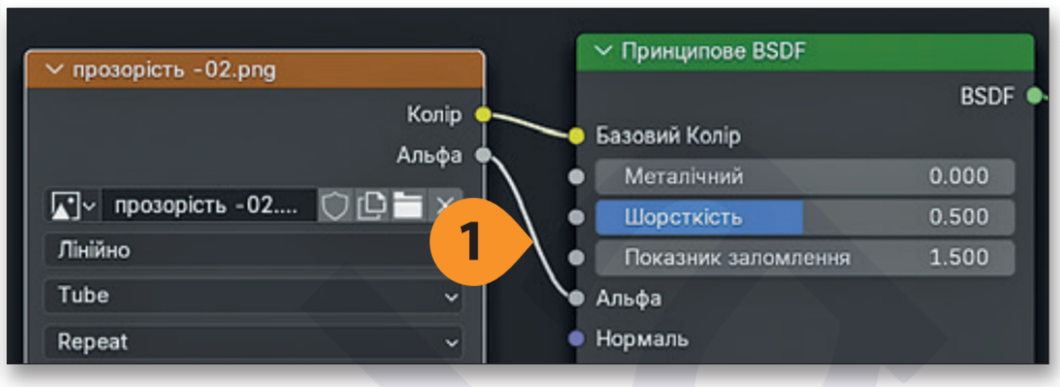
Далі зменште масштаб текстури, за потреби змініть розгортки та оберніть чи перетягніть текстуру на об'єкті до потрібного варіанта.



Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.



Підключивши альфа-текстури до параметра **матеріалу Альфа BSDF** (1), отримайте отвори у прозорих частинах. Отже, використання прозорості під час візуалізації дає змогу не моделювати повторювані отвори та інші види прозорих або напівпрозорих елементів.



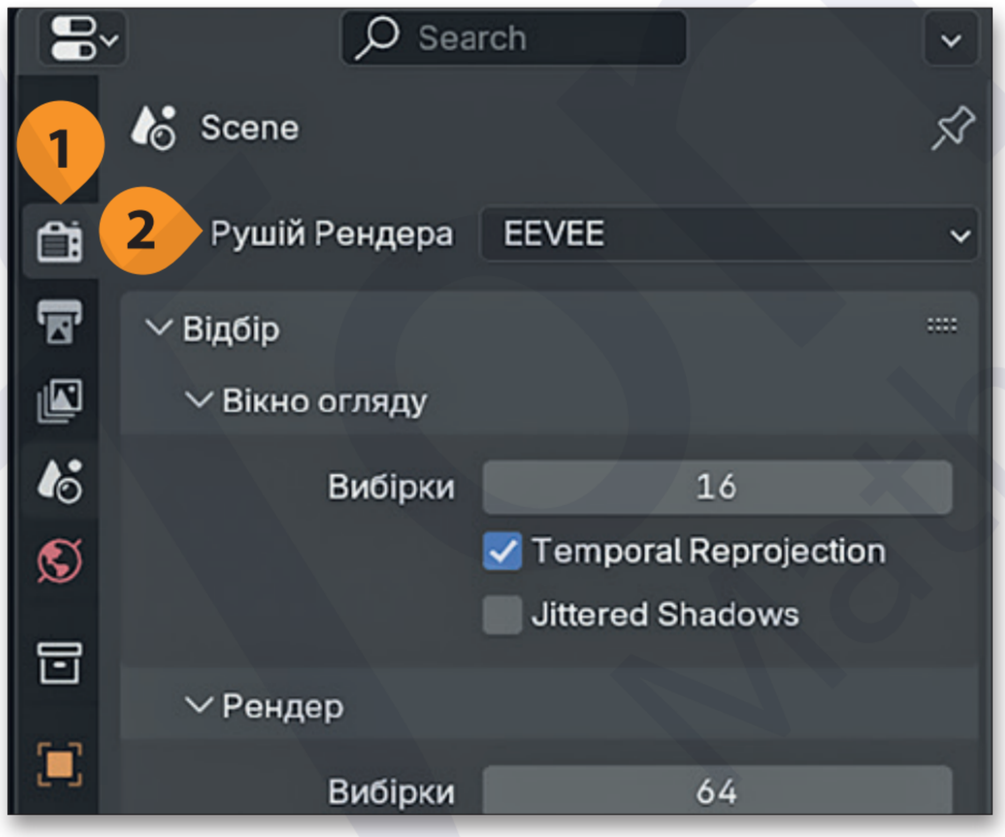
Рендер зображення

Рендер – це як фотолабораторія, де виявляються усі дрібні деталі та кольори, щоб отримати готове фото.

Перший і, можливо, найважливіший крок у цьому процесі – це вибір **Візуалізатора (Render engine)**. У **Blender** їх декілька, але ми вже знайомі з двома головними: **Eevee** та **Cycles**.

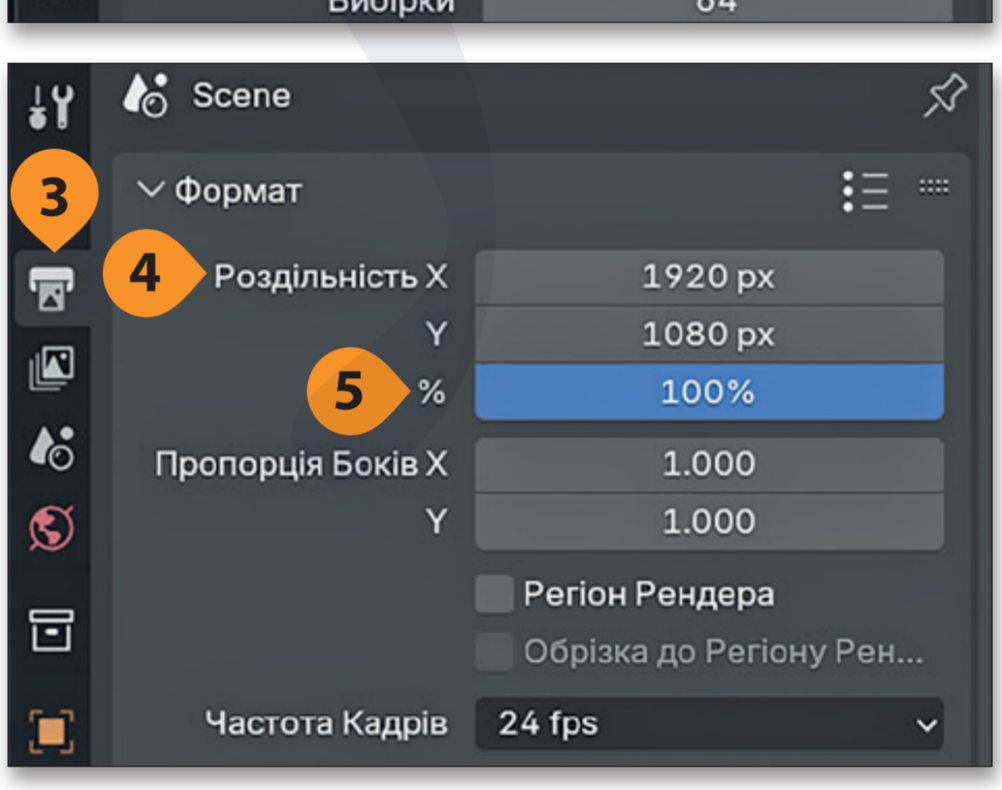
Після вибору візуалізатора наступний крок – визначити **розміри фінального зображення**. Це як обрати формат фотографії: буде це маленька світлина для паспорта чи величезне полотно для галереї.

***Рендер** – це не просто натискання кнопки. Це складний процес обчислень, під час якого програма збирає всю інформацію про ваші об'єкти, матеріали, світло, камеру та оточення, а потім розраховує, як кожен промінь світла взаємодіє з кожною поверхнею, створюючи остаточну картинку.*



Щоб обрати візуалізатор (Render engine):

- перейдіть до вкладки **Render (Властивості рендера)** (1) у вашій панелі властивостей праворуч (іконка, схожа на фотоапарат);
- угорі цієї вкладки ви побачите випадний список **Рушій рендера** (2), де й можна обрати **Eevee**, **Cycles** або **Workbench** (який використовується для технічного перегляду моделей).



Щоб визначити розміри фінального зображення:

- перейдіть до вкладки **Output (Властивості виводу)** у тій же панелі властивостей (іконка, схожа на принтер). Розділ **Формат** (3) – це ваше місце призначення;
- **Роздільність X** та **Y** (4) – це ширина та висота вашого зображення в пікселях;
- **Відсоток** (5) – коефіцієнт масштабування рендерингу.

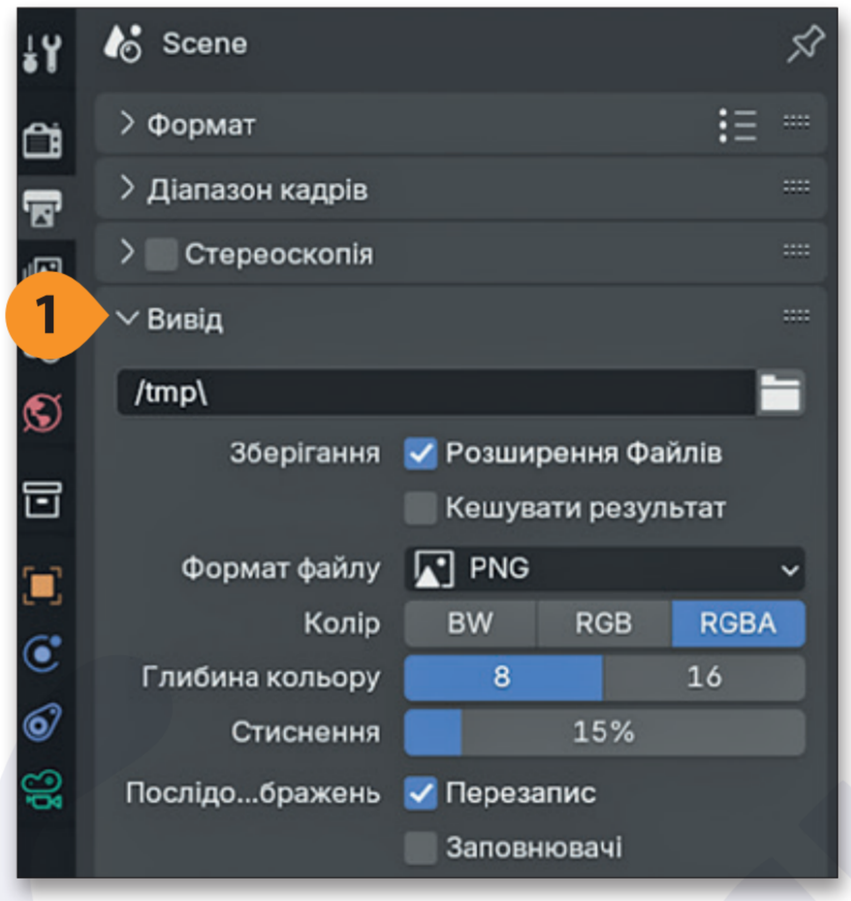
Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

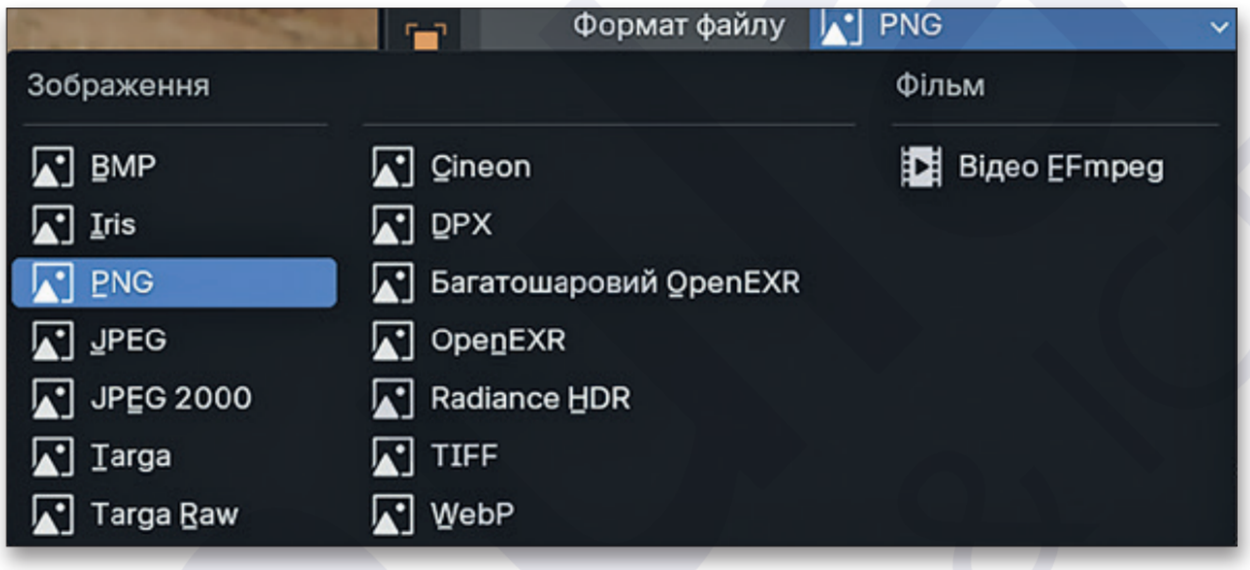
Що більші значення **Роздільність X** та **Y**, то якіснішим і детальнішим буде зображення, але й більше часу знадобиться для рендерингу. Стандартні значення для **Full HD** – **1920x1080**, для **4K** – **3840x2160** пікселів.

Збереження зображення

Ви зробили знімок, але в якому форматі його зберегти? Як і звичайні фотографії, рендери можуть бути збережені в різних форматах, кожен зі своїми особливостями. Прокрутіть вікно вниз до розділу **Вивід** (1).

Відсоток *– дуже корисний повзунок! Ви можете залишити* ***роздільну здатність*** *1920x1080, але встановити* ***Відсоток*** *на 50 %. Тоді Blender відрендерить зображення розміром* ***960x540****. Це ідеально для швидких тестових рендерів, щоб перевірити освітлення чи композицію, не чекаючи довгого фінального рендера.*





Коли всі налаштування зроблені, залишається останній, найприємніший крок. Перейдіть до верхнього меню **Blender** і натисніть **Рендер**, а потім оберіть **Рендер зображення** або просто натисніть клавішу **F12**.

Формати збереження файлу у Blender

PNG

Це, мабуть, найпопулярніший вибір для більшості 3D-рендерів. Він підтримує прозорість (Alpha Channel), що дає змогу зберігати об'єкти без фону (наприклад, якщо ви хочете потім вставити модель на інше зображення). Якість JPEG, але без втрат.

JPEG

Чудовий вибір для вебграфіки, адже він забезпечує невеликий розмір файлу. Однак він не підтримує прозорість і є «стисненням із втратами», що означає деяке погіршення якості під час збереження.

OpenEXR

Це «професійний» формат для рендерів. Він підтримує високий динамічний діапазон (що важливо для постобробки), а також різні «рендеринг-паси» (окремі шари інформації про тіні, світло, відбиття тощо). Файли OpenEXR дуже великі, але дають максимальну гнучкість під час роботи в програмах для композитингу, таких як Blender Compositor або Adobe After Effects.

FFmpeg Video

Якщо ви рендерите анімацію, цей формат дає вам змогу зберегти її як відеофайл (наприклад, MP4), обираючи відповідні кодеки (наприклад, H.264).

Targa (TGA)

Растровий формат файлів зображень, який є одним із найстаріших та найбільш поширених форматів у комп'ютерній графіці та 3D-індустрії.

TIFF

Формат файлів зображень, який широко використовується в професійній графіці, поліграфії та високоякісній обробці зображень.

Закінчення списку на наступній сторінці →

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Формати збереження файлу у Blender *(закінчення списку)*

BMP

Формат файлів растрових зображень, розроблений компанією Microsoft. Він є одним із найстаріших і найпростіших форматів для зберігання цифрових зображень.

Cineon (DPX)

Спеціалізований формат файлів зображень, який доступний у Blender і широко використовується в професійній індустрії кіно та телебачення.

Iris

Формат файлів растрових зображень, який доступний для рендерингу та збереження у Blender.

Radiance HDR

Формат файлів зображень, який доступний для рендерингу та збереження у Blender і має здатність зберігати широкий динамічний діапазон інформації про освітлення.

Blender *«оживе», почнуться обчислення (можливо, ви побачите, як промені світла «будують» зображення, якщо використовуєте* ***Cycles****).*

І ось, через кілька секунд, хвилин або годин – залежно від складності сцени та обраного візуалізатора – ваш віртуальний світ буде закарбований у фінальному прекрасному зображенні.

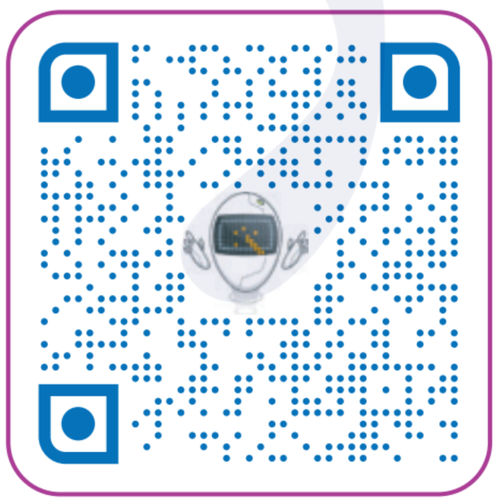
Це кульмінація вашої творчості та технічної майстерності – мить, коли невидиме стає видимим, а уявне – реальним!



Практичне завдання

Завантажте файл-заготовку зі сторінки інтернет-підтримки. Виконайте запропоноване завдання.

https://formula.education/online-ict-9-klas/





Інші платформи



Проєкт



Підсумки



Глосарій

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

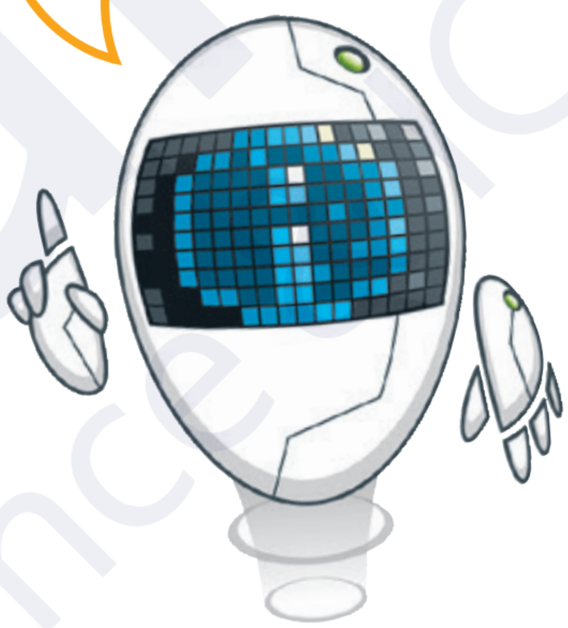
Додаткова тема
Основи 3D-моделювання

Що таке CAD-системи

CAD-система – це спеціальна програма, у якій інженери, архітектори, дизайнери або навіть школярі можуть проєктувати різні вироби: від простих деталей до складних машин, будівель або ігрових персонажів.

Замість того щоб креслити на папері, люди працюють у комп'ютерному середовищі, де можна легко змінювати розміри, форму, кольори та перевіряти, який вигляд матиме готовий виріб.

***CAD (англ. Computer-Aided Design** – комп'ютерне проєктування) – це технологія, яка дає змогу створювати креслення та 3D-моделі за допомогою комп'ютера.*



Основні можливості CAD-систем

1. Створення 2D-креслень і 3D-моделей
У CAD можна креслити у двовимірному (2D) режимі, як на аркуші паперу, або створювати об'ємні (3D) моделі, які можна обертати й розглядати з усіх боків

2. Параметричне моделювання
Це означає, що модель «пам'ятає» свої параметри – розміри, кути, відстані. Якщо змінити один розмір, інші автоматично підлаштовуються. Наприклад, якщо збільшити довжину крила літака, CAD-система автоматично змінить розміри закрилків і корпусу, щоб усе залишалося узгодженим

3. Збірки
Можна об'єднувати окремі деталі в одну конструкцію. Наприклад, створити коліщатко, вісь і корпус, а потім зібрати з них працездатний механізм

4. Аналіз конструкцій
Деякі CAD-системи можуть проводити симуляції – перевіряти, чи витримає деталь навантаження, як вона нагрівається, або як рухатимуться її частини. Це допомагає знайти помилки ще до виготовлення реального виробу

Популярні CAD-системи

Autodesk AutoCAD – одна з найвідоміших програм для створення креслень і 3D-моделей.

Autodesk Fusion – сучасна хмарна система, у якій можна моделювати, збирати, тестувати й готувати вироби до 3D-друку.

SolidWorks – зручна програма для машинобудування та інженерії.

CATIA – використовується у великих компаніях, наприклад, у виробництві літаків та автомобілів.

PTC Creo – потужна система для точного моделювання складних технічних виробів.

*Отже, **CAD-системи** – це незамінний інструмент сучасного конструктора, інженера чи дизайнера. Вони допомагають втілювати ідеї в точні, зручні й реалістичні моделі, які можна використовувати у навчанні, виробництві чи творчості.*

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Огляд Autodesk Fusion

Autodesk Fusion 360 – це сучасна програма для 3D-моделювання, проєктування та виробництва.

Ця платформа підходить і для початківців, які лише знайомляться з 3D-моделюванням, і для професіоналів, що працюють над реальними інженерними проєктами.

Fusion – *це універсальний інструмент, у якому можна придумати, спроєктувати, протестувати та навіть підготувати до виготовлення будь-який виріб – усе в одній програмі.*

Autodesk Fusion 360 поєднує одразу кілька напрямів:

CAD (Computer-Aided Design)	CAM (Computer-Aided Manufacturing)	CAE (Computer-Aided Engineering)
Створення креслень і 3D-моделей.	Підготовка моделей до виготовлення на верстатах або 3D-принтерах.	Проведення інженерних аналізів і перевірок.

Основні переваги Autodesk Fusion

Інтуїтивний інтерфейс
Програма має зручне меню, зрозумілі інструменти та підказки. Навіть початківець швидко навчиться малювати прості фігури, з'єднувати їх і створювати об'ємні моделі

Хмарне збереження проєктів
Усі робóти можна зберігати у хмарі, тобто на сервері в інтернеті. Це дає змогу відкривати свої моделі з будь-якого комп'ютера та не боятися втратити файли

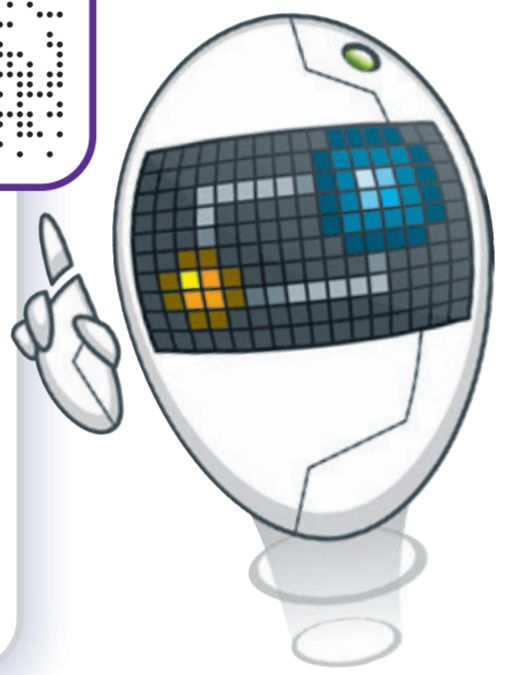
Створення 3D-збірок
Програма дозволяє поєднувати кілька окремих деталей у єдиний виріб, щоб перевірити, як вони взаємодіють між собою (наприклад, як рухаються шестерні або відкривається кришка)

Безкоштовна версія для навчання та хобі
Учні, студенти, викладачі й початківці можуть безкоштовно користуватися програмою. Це чудовий варіант для тих, хто тільки починає знайомитися зі світом 3D-дизайну

Реєстрація та встановлення (персональна ліцензія)

Щоб зареєструватись і встановити Autodesk Fusion 360:

- перейдіть за посиланням на офіційний сайт https://www.autodesk.com/products/fusion-360/personal та завантажте інсталятор (це сторінка безкоштовної особистої версії **Autodesk Fusion 360**);
- натисніть кнопку **Download free trial**;
- створіть **обліковий запис Autodesk**, заповнивши всі потрібні поля;
- виберіть тип використання: наприклад, **Для особистого (hobby) використання** чи **Стартап**. Серед варіантів є **безкоштовна особиста ліцензія (Personal Use)** після пробної версії;
- завантажте інсталятор: виберіть вашу **операційну систему (Windows, MacOS)**. Запустіть **інсталятор** і пройдіть кроки установки на екрані;
- після встановлення запустіть програму й увійдіть під своїм **обліковим записом Autodesk**.



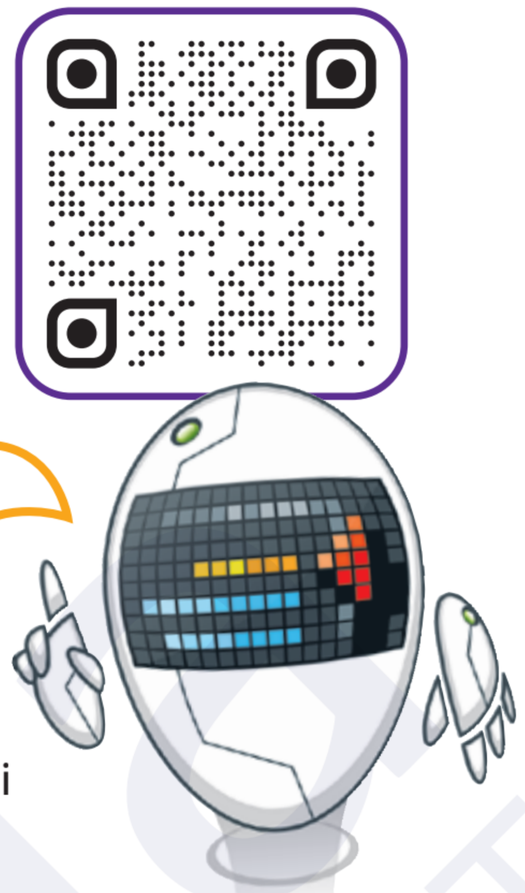
Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Деякі важливі нюанси та поради

Версія для особистого, некомерційного використання доступна, але має обмеження після пробної повної версії. Для учасників і учасниць освітнього процесу також існує ***безкоштовна навчальна ліцензія****.*

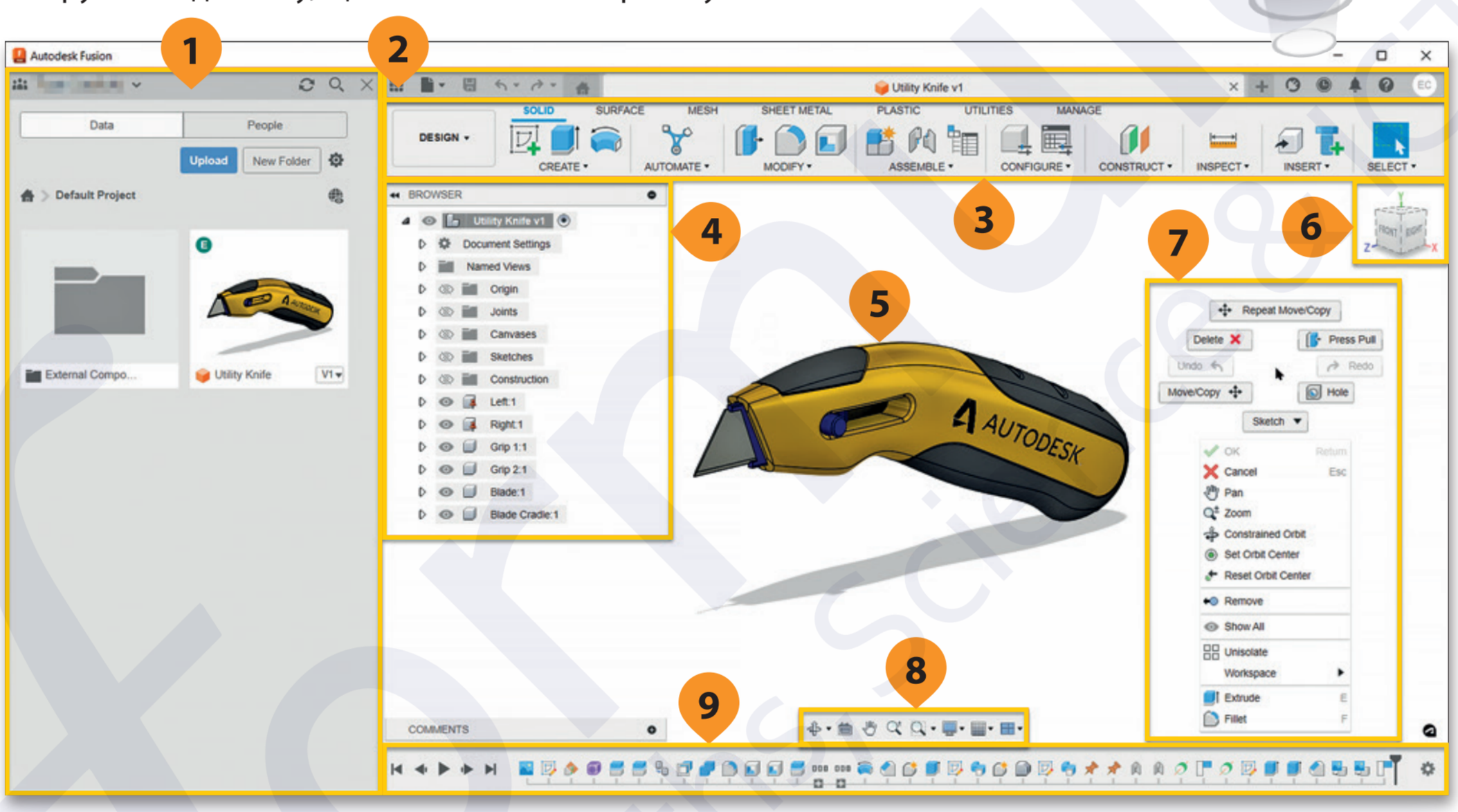
Переконайтесь, що ваш комп'ютер відповідає системним вимогам програми перед установкою:
https://www.autodesk.com/support/system-requirements/overview.

Під час встановлення, можливо, буде потрібен дозвіл від адміністратора або інтернет-з'єднання, тому краще робити це вдома чи в шкільному комп'ютерному класі разом з учителем / учителькою.



Інтерфейс Autodesk Fusio

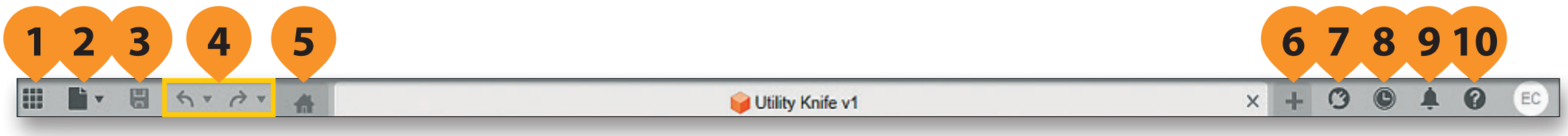
Ознайомтеся з інтерфейсом Fusion, щоб дізнатися, де розташовані всі інструменти дизайну, щойно ви почнете проєктувати.



Інтерфейс Autodesk Fusion:

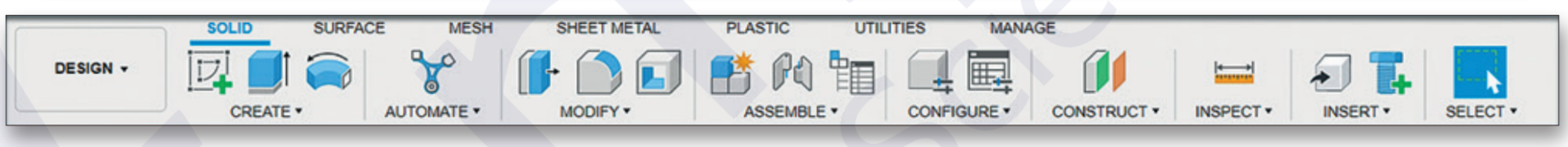
- **Панель даних** (1): демонструє доступ до центрів, проєктів і дизайнів, дає змогу керувати даними дизайну та співпрацювати з іншими користувачами з панелі даних;
- **Панель програми** (2);
- **Панель інструментів** (3);
- **Браузер** (4): відображає об'єкти у збірці та допомагає керувати видимістю об'єктів;
- **Дизайн або збірка** (5): модель, над якою ми працюємо зараз;
- **ViewCube** (6): функція обертання по орбіті або для перегляду з різних позицій;
- **Контекстне меню** (7): відкриває контекстне меню, яке містить команди, що часто використовуються, та додаткові команди;
- **Панель навігації** (8): містить команди, що використовуються для масштабування, панорамування та переміщення вашого проєкту по орбіті, а також налаштування відображення для керування зовнішнім виглядом інтерфейсу та відображенням збірок проєкту;
- **Часова шкала** (9): містить список операцій, що виконуються у вашому проєкті. Клацніть правою кнопкою миші на операціях на часовій шкалі, щоб внести зміни. Перетягніть операції, щоб змінити порядок їх обчислення.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.



Панель програми Autodesk Fusion:

- **Панель даних** (1): відкриває або приховує панель даних;
- **Меню «Файл»** (2): дає змогу створювати, відкривати, завантажувати, зберігати, експортувати, відновлювати, друкувати на 3D-принтері та ділитися проєктами;
- **Зберегти** (3): зберігає зміни до дизайну;
- **Скасувати / Повторити** (4): допомагає скасувати або повторити останню операцію;
- **Home** (5): закриває поточну робочу вкладку (Design, Render, Manufacture тощо) і показує список усіх збережених або хмарних проєктів, папок, командних просторів, а також посилання на навчальні матеріали, форум та перелік доповнень (додатків) від Autodesk;
- **Новий дизайн** (6): створює новий дизайн;
- **Розширення** (7): керування доступом до розширень Fusion;
- **Стан завдання** (8): перегляд стану завдання, стану оновлення Fusion та стану онлайн / офлайн;
- **Центр сповіщень** (9): перегляд сповіщень про оновлення, фонові завдання та поради;
- **Довідка** (10): доступ до навчального контенту, документації, швидкого налаштування, ресурсів спільноти, технічної підтримки, інструментів діагностики, перевірки оновлень, доступу до останньої публікації блогу «Що нового» та інформації про встановлення Fusion.



Панель інструментів Autodesk Fusion

Робочі простори
Fusion використовує робочі простори для керування доступними командами та типом даних, які ви створюєте

Меню робочого простору
Дає змогу обрати робочий простір для переходу

Вкладки
Кожне робоче середовище організовано у вкладки, які містять логічні групи пов'язаних інструментів

Контекстні вкладки
Деякі команди активують контекстну вкладку, наприклад **Ескіз**. Контекстна вкладка відображається на **Панелі інструментів** поряд з іншими вкладками в робочому просторі, поки контекстні інструменти активні

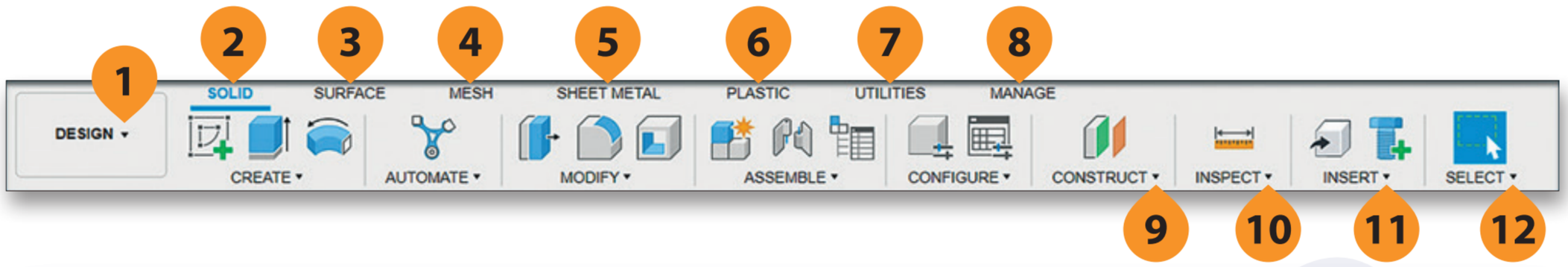
Контекстні середовища
Деякі команди активують контекстне середовище, наприклад **Форма**. Контекстне середовище відображається на **Панелі інструментів**, а його вкладки замінюють стандартні вкладки у поточному робочому просторі, доки ви не вийдете з контекстного середовища

ДОВІДКА

Autodesk Fusion 360 було офіційно представлено **24 вересня 2013 року** компанією **Autodesk**. Fusion 360 є хмарною системою автоматизованого проєктування, яка багато в чому наслідує і замінює попередній продукт компанії – **Inventor Fusion**.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Панель інструментів Autodesk Fusion (показана нижче на зображенні) належить до середовища **Design** (1) і складається з кількох основних вкладок та груп інструментів.

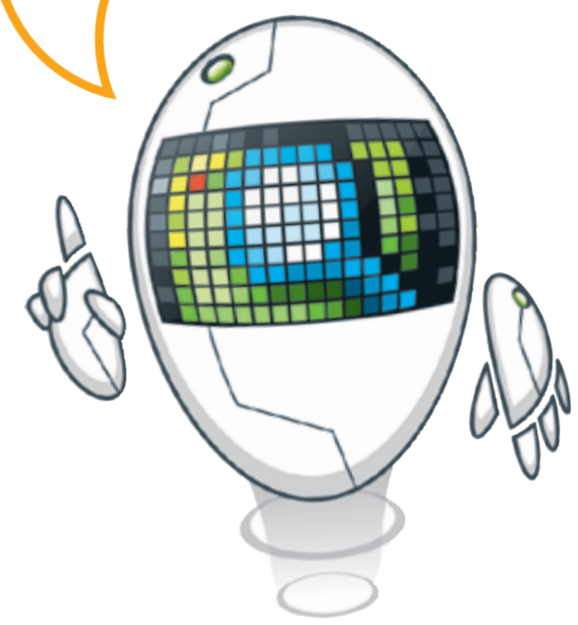


Основні вкладки панелі інструментів та команд Autodesk Fusion:

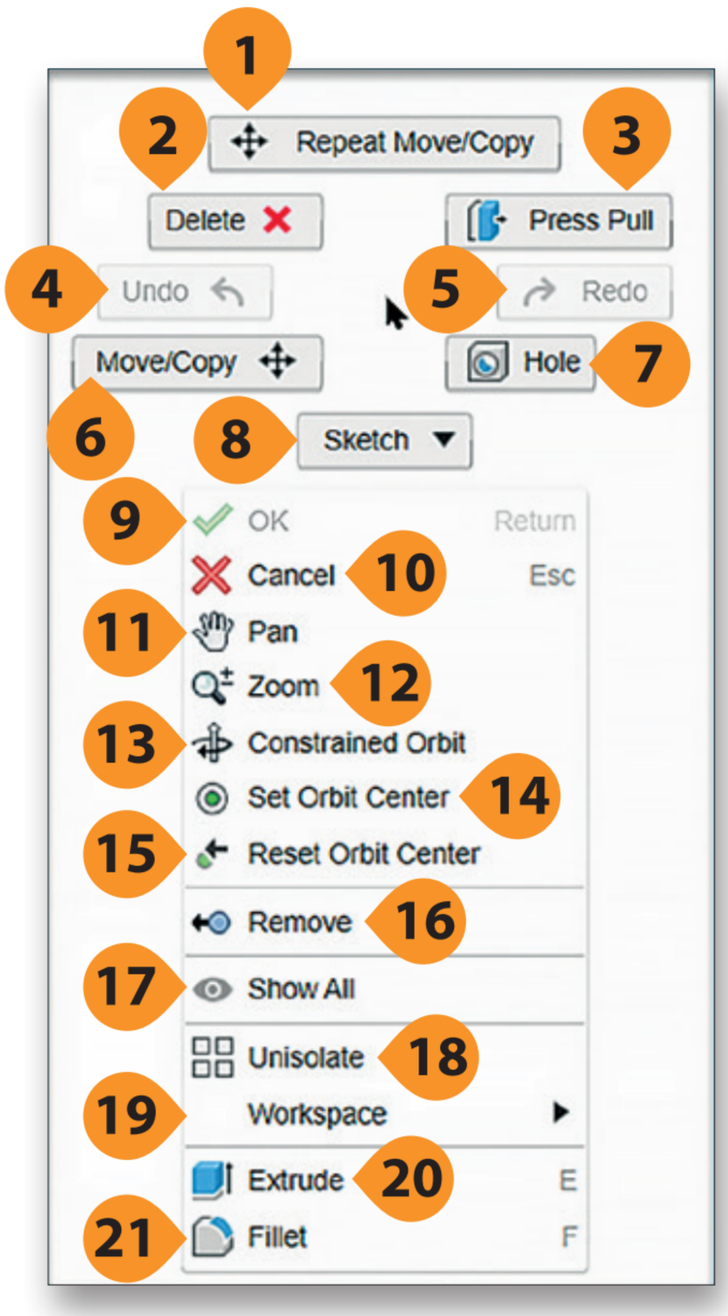
- **SOLID** (2) – операції зі створення та редагування твердотільних моделей. Інструменти для побудови примітивів, екструзії, вирізання, об'єднання тіл тощо;
- **SURFACE** (3) – робота з поверхнями, дає змогу створювати та модифікувати поверхні, які використовуються для складного моделювання;
- **MESH** (4) – операції з полігональними сітками, імпорт 3D-моделей, створення та обробка сіткових об'єктів;
- **SHEET METAL** (5) – проєктування виробів з листового металу, згини, розгортки, вкладання деталей;
- **PLASTIC** (6) – інструменти для компонентів із пластику, оптимізація геометрії для лиття під тиском;
- **UTILITIES** (7) – додаткові функції: аналіз моделі, вимірювання, додавання залежностей тощо;
- **MANAGE** (8) – керування компонентами, версіями, історією змін та організацією моделі;
- **CONSTRUCT** (9) – інструменти допоміжної геометрії для створення допоміжних площин, осей, точок;
- **INSPECT** (10) – інструмент для аналізу моделі, вимірювання відстаней, кутів, створення розрізу моделі, перевірки перетинів компонентів;
- **INSERT** (11) – додавання зовнішніх елементів, імпорту 3D-моделей, ескізів, зображень, нанесення зображення (текстури) на поверхню;
- **SELECT** (12) – інструменти вибору елементів, фільтрації за типом об'єкта (тіло, грань, край, ескіз тощо).



Кнопки (команди) у вкладках панелі інструментів Autodesk Fusion 360 можуть мінятися залежно від вибору активного робочого середовища.



Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.



Контекстне меню програми Autodesk Fusion – Команди вищого рівня

Ці елементи зазвичай відображаються у вигляді підписів у радіальному меню жестів:

- **Repeat... (Повторити...)** (1): повторює останню команду, яку ви щойно виконали;
- **Delete (Видалити)** (2): видаляє вибраний об'єкт (тіло, грань, ескіз тощо);
- **Press Pull (Натиснути / Витягнути)** (3): загальна команда для зміни розмірів граней або профілів;
- **Undo (Скасувати)** (4): скасовує останню дію;
- **Redo (Повторити)** (5): повторює скасовану дію;
- **Move/Copy (Перемістити / Копіювати)** (6): інструмент для переміщення або створення копії об'єктів;
- **Hole (Отвір)** (7): інструмент для створення отворів;
- **Sketch (Ескіз)** (8): швидкий доступ до інструментів створення ескізу.

Контекстне меню програми Autodesk Fusion – Управління командою та видом :

- **OK** (9): підтверджує та завершує активну команду (якщо ви її виконували);
- **Cancel** (10): скасовує активну команду або виходить із режиму, не зберігаючи змін.

  Клавіша Ctrl;
- **Pan** (11): тимчасово активує інструмент **Панорамування (переміщення виду)**;
- **Zoom** (12): тимчасово активує інструмент **Масштабування**;
- **Constrained Orbit** (13): дає змогу обертати модель навколо осей (обмежена орбіта);
- **Set Orbit Center** (14): допомагає встановити нову центральну точку, навколо якої буде обертатися модель при використанні Orbit;
- **Reset Orbit Center** (15): скидає центр обертання до стандартного.

Контекстне меню програми Autodesk Fusion – Відображення та робоча область:

- **Remove** (16): використовується для видалення об'єкта або елемента (наприклад, видалити грань, об'єкт з виділення);
- **Show All** (17): робить видимими всі приховані об'єкти (ескізи, тіла, компоненти), які були приховані раніше;
- **Unisolate** (18): скасовує режим **Isolate (Ізолювати)**, показуючи всі об'єкти, які були приховані для фокусування;
- **Workspace (Робочий простір)** (19): відкриває підменю для перемикання між різними робочими просторами **Fusion (Design, Render, Simulation, Manufacturing, Drawing)**;
- **Extrude (Видавлювання)** (20): запускає команду **Видавлювання (Extrude)**, щоб створити 3D-тіло з 2D-профілю.

  Клавіша N;
- **Fillet (Скруглення)** (21): запускає команду **Скруглення (Fillet)**, щоб округлити краї або кути моделі.

  Клавіша F.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.



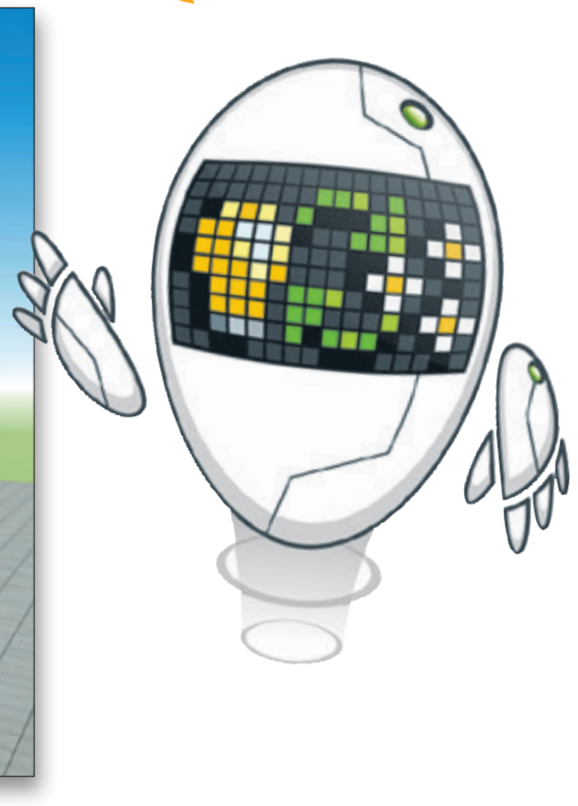
Панель навігації програми Autodesk Fusion:

- **Orbit (Орбіта)** (1): дає змогу обертати вид навколо центру моделі;
  Можна вільно змінювати кут огляду.
  Часто використовується як **Free Orbit (Вільна орбіта)** або доступна через клавішу Shift + **середня кнопка миші**;
- **Look At (Подивитися на)** (2): переорієнтовує вид, щоб нормаль до вибраної площини, грані або осі була прямо перед вами. Ідеально підходить для початку ескізу;
- **Pan (Панорамування)** (3): дає змогу переміщувати вид горизонтально та вертикально, не змінюючи кута огляду чи збільшення. Це «перетягування» сцени. Доступне через **середню кнопку миші**;
- **Zoom (Масштабування)** (4): допомагає збільшувати або зменшувати масштаб виду моделі;
- **Zoom Window** (5): масштабування, яке дозволяє вам збільшити конкретну область вашої моделі, яку ви обираєте, малюючи прямокутник (вікно);
- **Display Settings (Параметри відображення)** (6): відкриває меню для налаштування: візуального стилю (наприклад, каркас, затінений), сітки, ефектів (тіні, віддзеркалення), середовища тощо;
- **Grid and Snaps (Сітка та прив'язки)** (7): кнопка відповідає за прив'язку моделі до сітки;
- **Viewports (Вікна перегляду)** (8): може змінити розкладку графічної області, що є особливо корисним для огляду моделі з різних стандартних точок зору одночасно.

Тож інтерфейс Autodesk Fusion продуманий так, щоб навіть початківцям було легко розібратися в його роботі. Усі інструменти логічно розташовані, а панелі та робочі області допомагають швидко переходити від ескізу до готової 3D-моделі. Завдяки зручній навігації, таймлайну та хмарному збереженню проєктів ви можете експериментувати, виправляти помилки та бачити результат своєї роботи одразу на екрані.

*Коли ви наводите курсор миші на будь-яку кнопку на панелі інструментів (наприклад, **Extrude**, **Revolve** або **Line**), з'являється невелике вікно з описом команди.*
У кінці цього опису зазвичай вказана відповідна «гаряча» клавіша, наприклад, E *– для Extrude або* L *– для Line.*





Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Практикум: створення гвинта та гайки в Autodesk Fusion

Під час цього практикуму ми створимо дві прості, але важливі деталі – гвинт і гайку. Вони широко використовуються у техніці, адже завдяки різьбі дають змогу надійно з'єднувати частини механізмів.

Для виконання цієї роботи використовуйте **Середовище Design**, яке відповідає за побудову та **Редагування тривимірних тіл (Solid Modeling)**.

У цьому середовищі можна створювати **Ескізи (Sketches)**, виконувати **Об'ємні операції (Extrude, Revolve, Sweep тощо)** та з'єднувати кілька деталей у **Збірки (Assemblies)**.

Мета роботи:
*ознайомитися з принципами параметричного 3D-моделювання та навчитися створювати деталі з різьбовим з'єднанням (гвинт і гайка) у середовищі **Design** програми **Autodesk Fusion.***

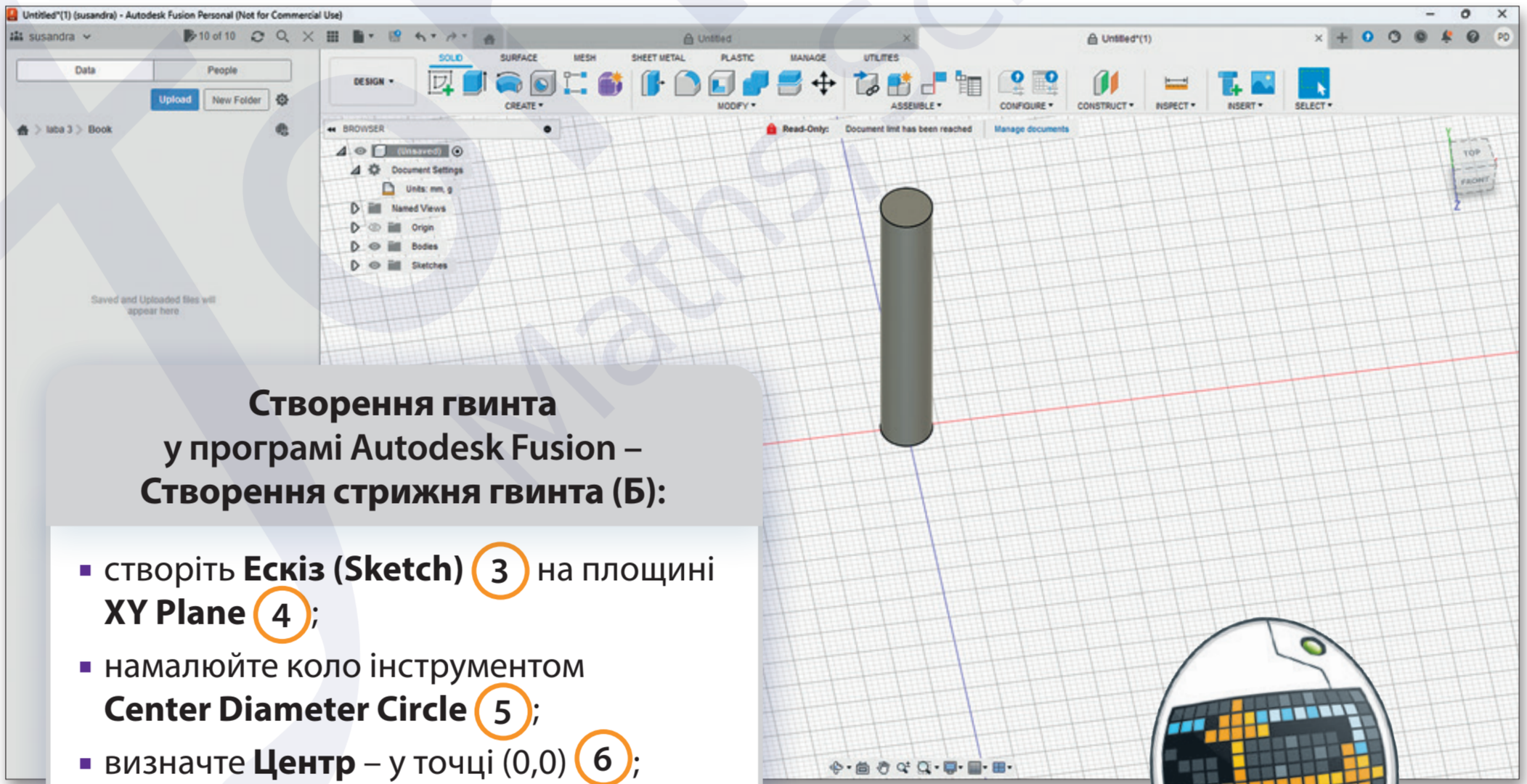
Створюючи гвинт і гайку у Fusion, ви навчитеся:

- Працювати з ескізами та об'ємними тілами
- Застосовувати операції **Extrude**, **Thread**, **Join**
- Налаштовувати параметри для точного моделювання деталей
- Готувати моделі до 3D-друку

Створення гвинта

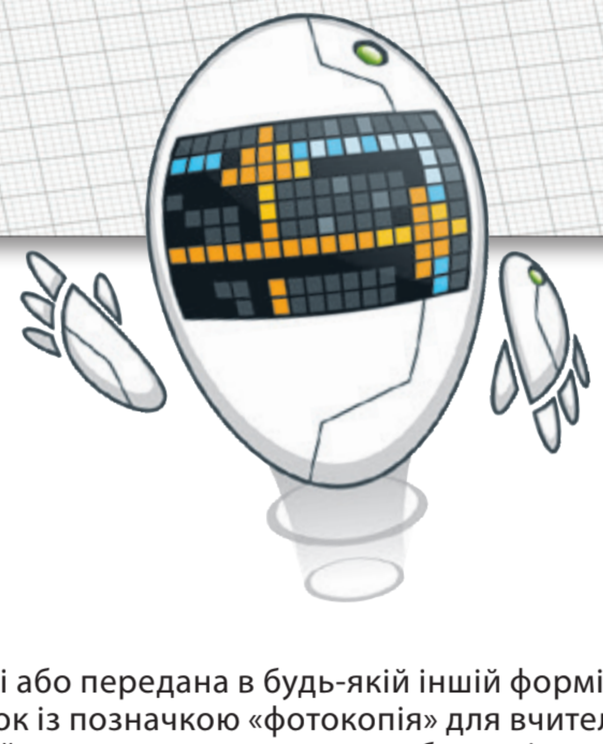
Створення гвинта у програмі Autodesk Fusion – Підготовка документа (А):

- відкрийте програму Fusion. Створіть новий дизайн: **File → New Design** (1);
- установіть одиниці вимірювання в міліметрах: **Browser → Document Settings → Units → Millimeters (mm)** (2). Це потрібно для точності всіх розмірів.



Створення гвинта у програмі Autodesk Fusion – Створення стрижня гвинта (Б):

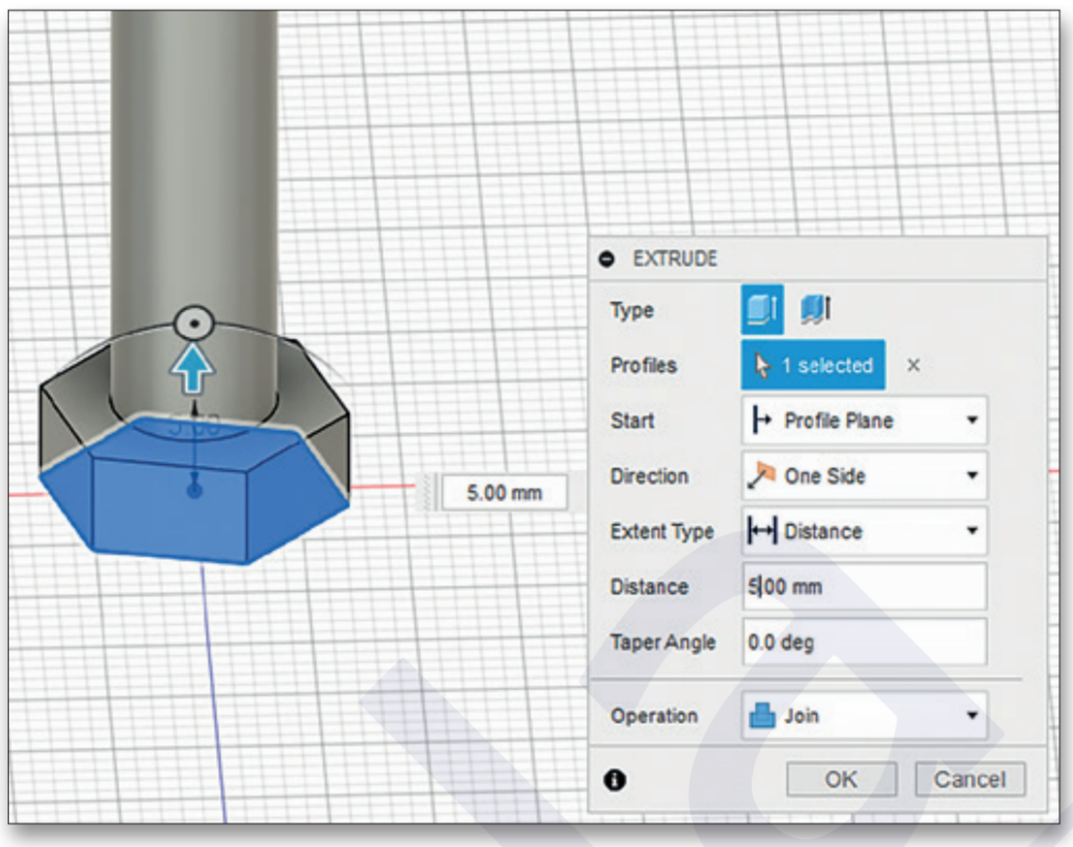
- створіть **Ескіз (Sketch)** (3) на площині **XY Plane** (4);
- намалюйте коло інструментом **Center Diameter Circle** (5);
- визначте **Центр** – у точці (0,0) (6);
- установіть **Діаметр** – 8 мм (7);
- натисніть **Завершити ескіз (Finish Sketch)** (8) і виконайте **Extrude (E)** (9) на довжину 50 мм. Це тіло гвинта.



Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

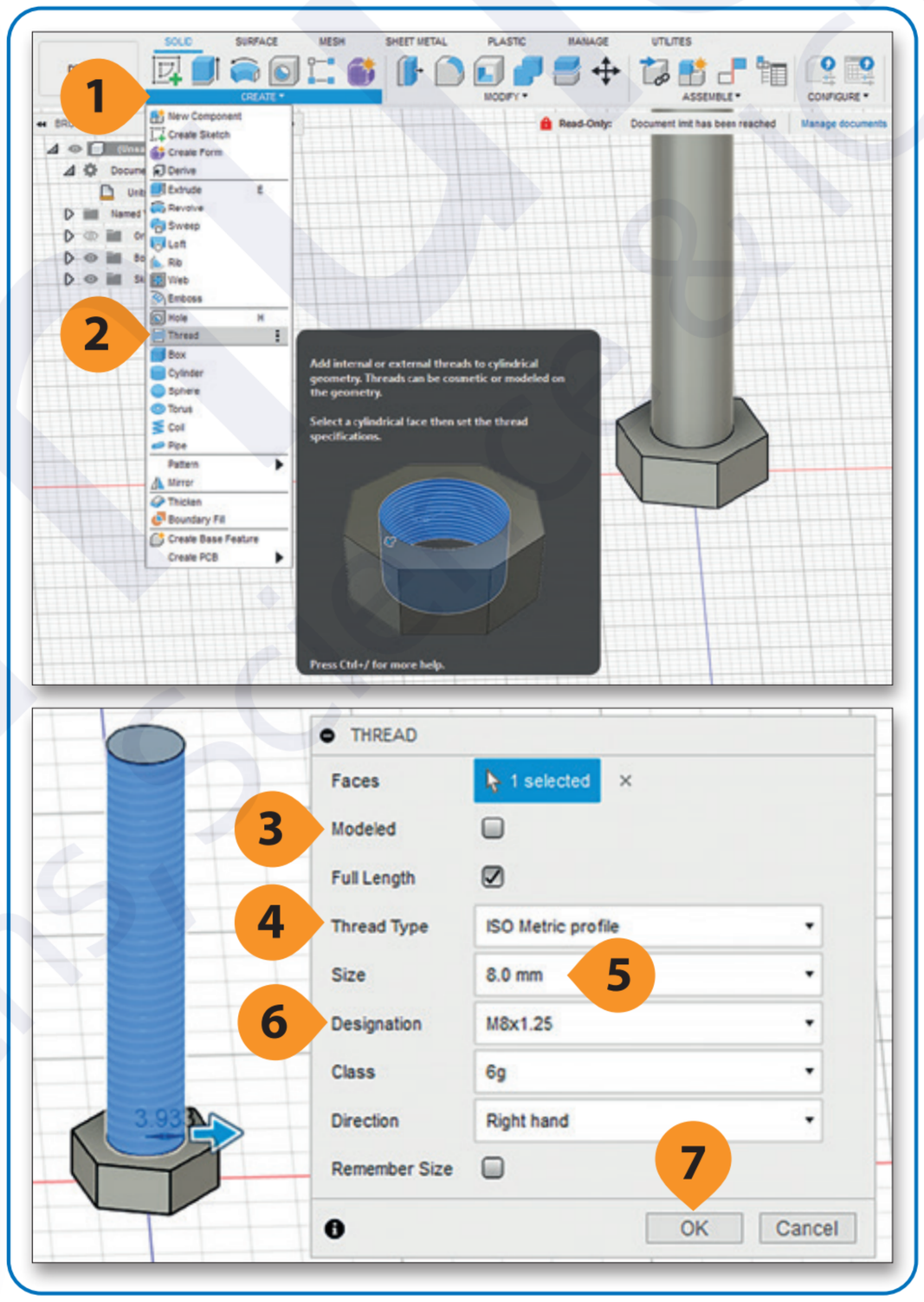
Створення гвинта у програмі Autodesk Fusion – Створення головки гвинта (B):

- створіть новий **Ескіз (Sketch)** на верхній поверхні;
- оберіть **Create** → **Polygon** → → **Circumscribed Polygon**;
- установіть 6 сторін, діаметр описаного кола – 13 мм;
- виконайте **Extrude** угору на 5 мм, тип операції – **Join**.



Створення гвинта у програмі Autodesk Fusion – Додавання різьби (Г):

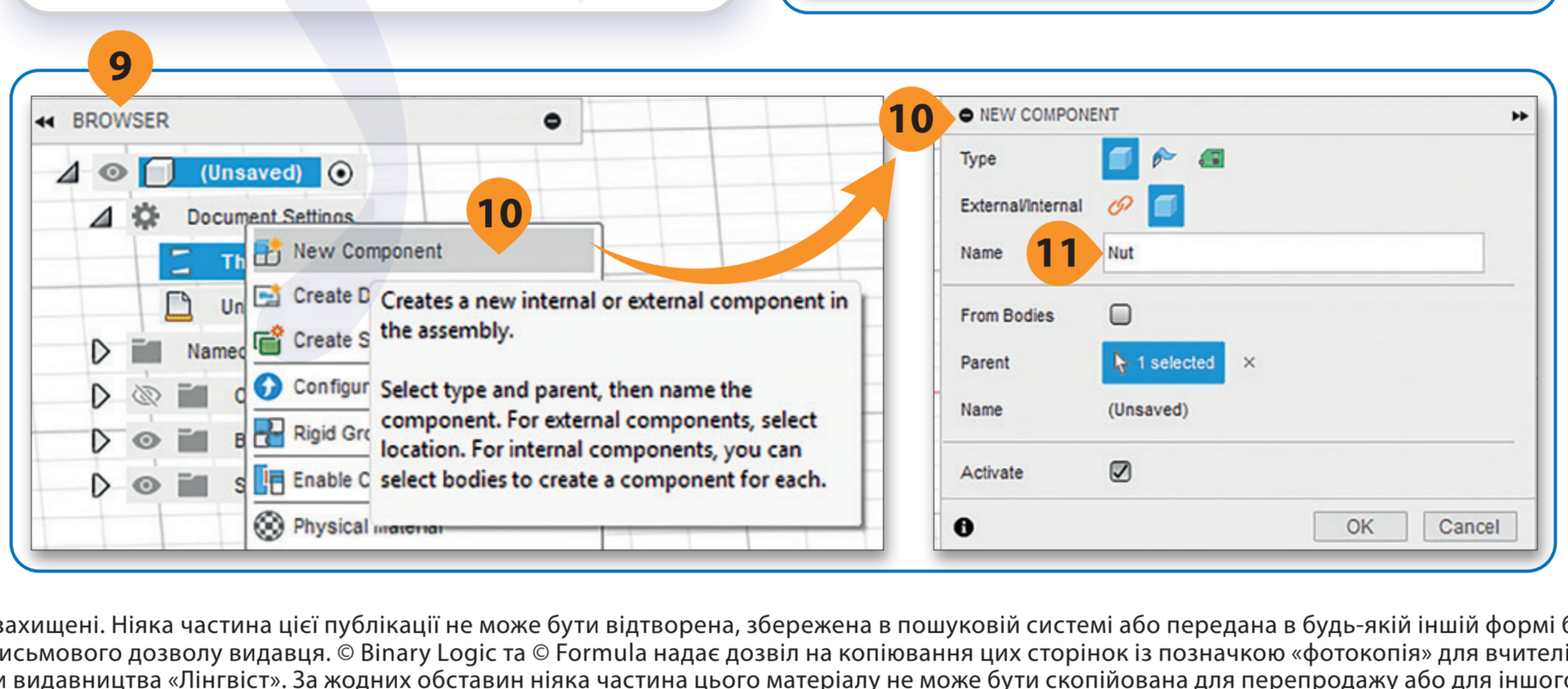
- оберіть **Create** (1) → **Thread** (2);
- клацніть на бічній поверхні циліндра, установіть параметри:
  - **Modeled** (3) – увімкнено;
  - **Thread Type:** (4)**:** ISO Metric Profile;
  - **Size** (5)**:** 8 mm;
  - **Designation** (6)**:** M8 × 1.25;
- натисніть **OK** (7) – гвинт готовий.



Створення гайки

Створення гайки у програмі Autodesk Fusion – Новий компонент (A):

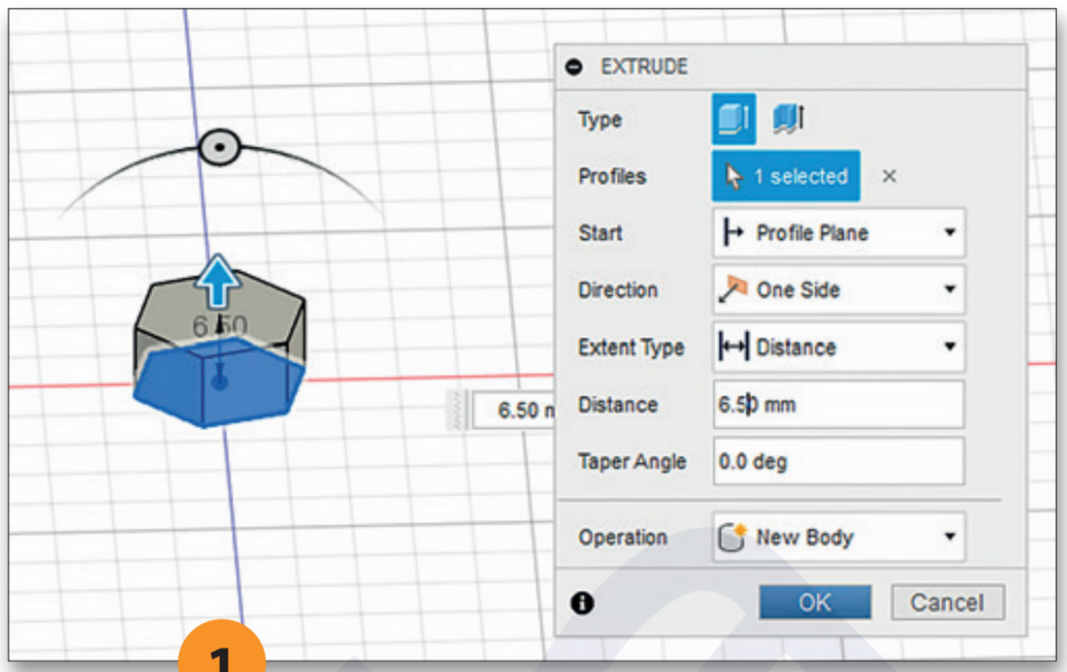
- у **Browser** (9) клацніть правою кнопкою, далі виберіть → **New Component** (10);
- назвіть його **Nut** (11).



Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

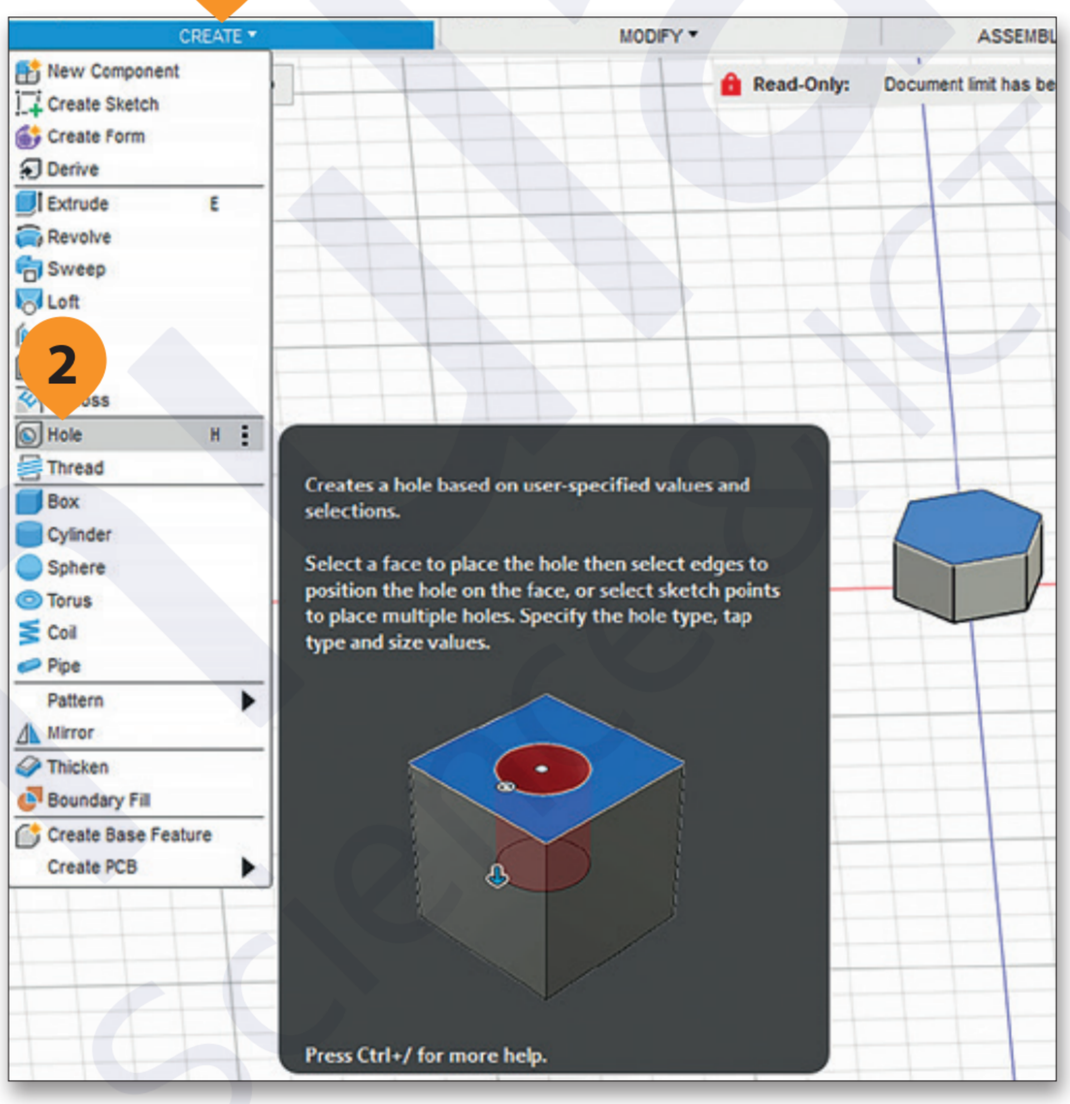
Створення гайки у програмі Autodesk Fusion – Моделювання гайки *(операції ідентичні до створення гвинта)* (Б):

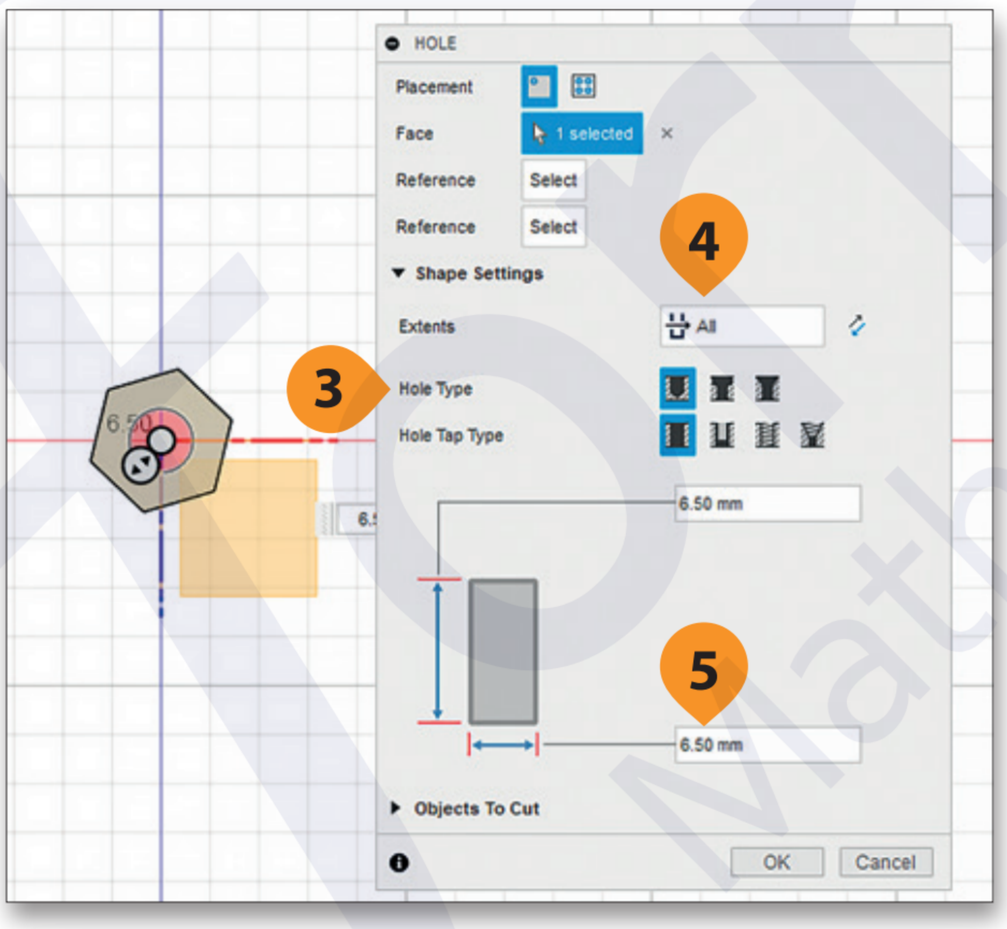
- створіть **Ескіз (Sketch)** на площині **XY Plane**;
- створіть шестикутник діаметром 13 мм;
- виконайте **Extrude (E)** на 6.5 мм → → **New Body**.

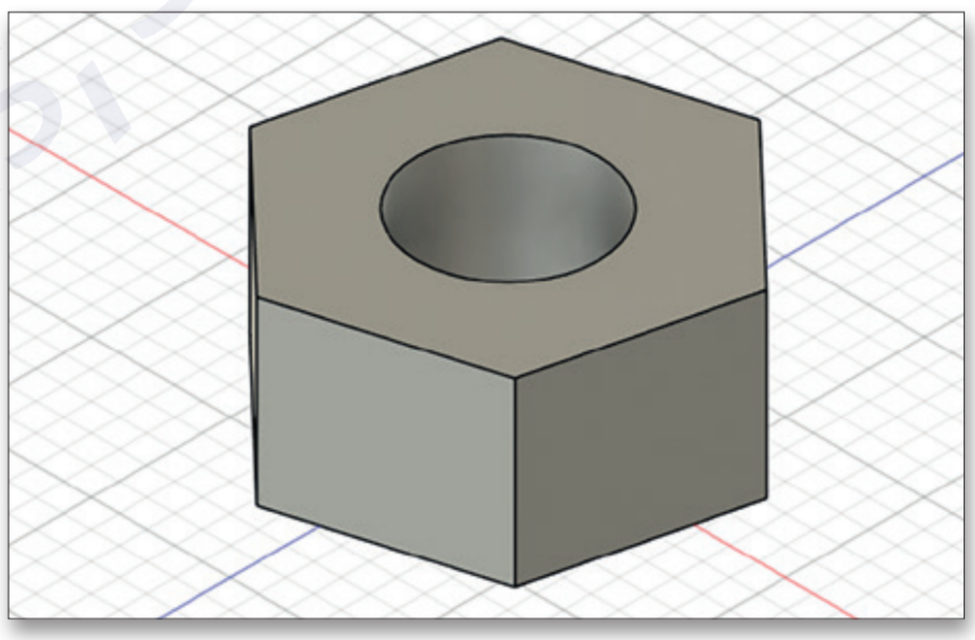


Створення гайки у програмі Autodesk Fusion – Отвір (В):

- покроково натисніть **Create** (1) → → **Hole** (2) → **Placement: At Point** → у центрі шестикутника;
- установіть параметри:
  - **Hole Type** (3): Simple;
  - **Extents** (4): All;
  - **Diameter** (5): 6.50 мм.

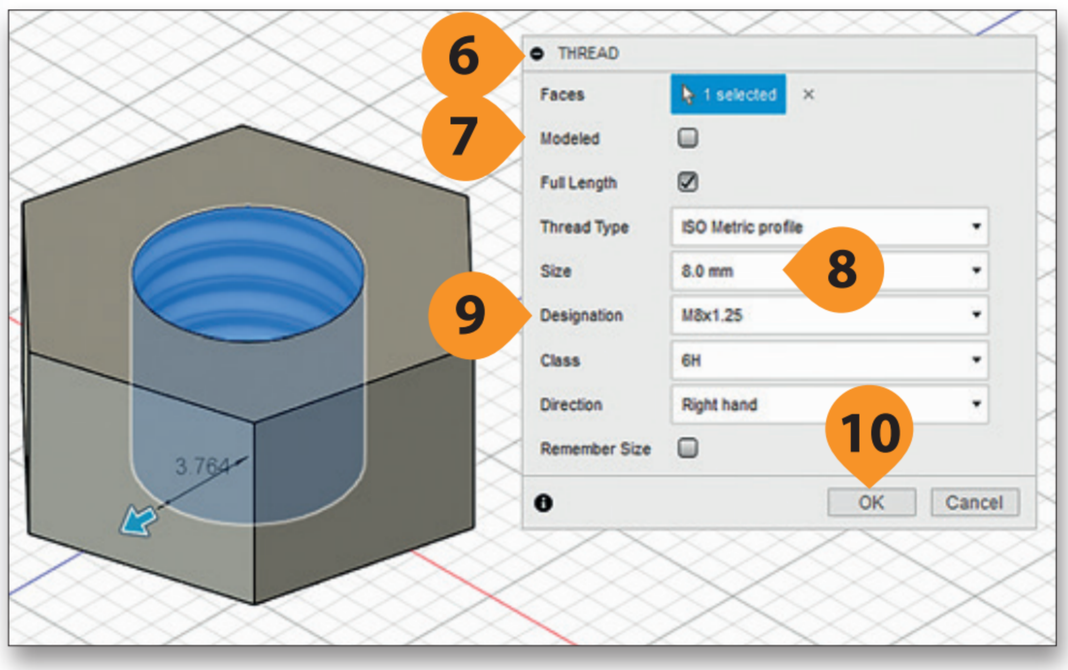






Створення гайки у програмі Autodesk Fusion – Внутрішня різьба (Г):

- оберіть **Create** → **Thread** (6), клацніть на внутрішній поверхні;
- активуйте **Modeled** (7) (клацніть на моделі вказівником мишки);
- установіть **Size** (8): 8 mm.
- визначте **Designation** (9): M8 × 1.25;
- натисніть **OK** (10) – гайка готова.



Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Збереження та експорт результату

Збереження файлів у програмі Autodesk Fusion:

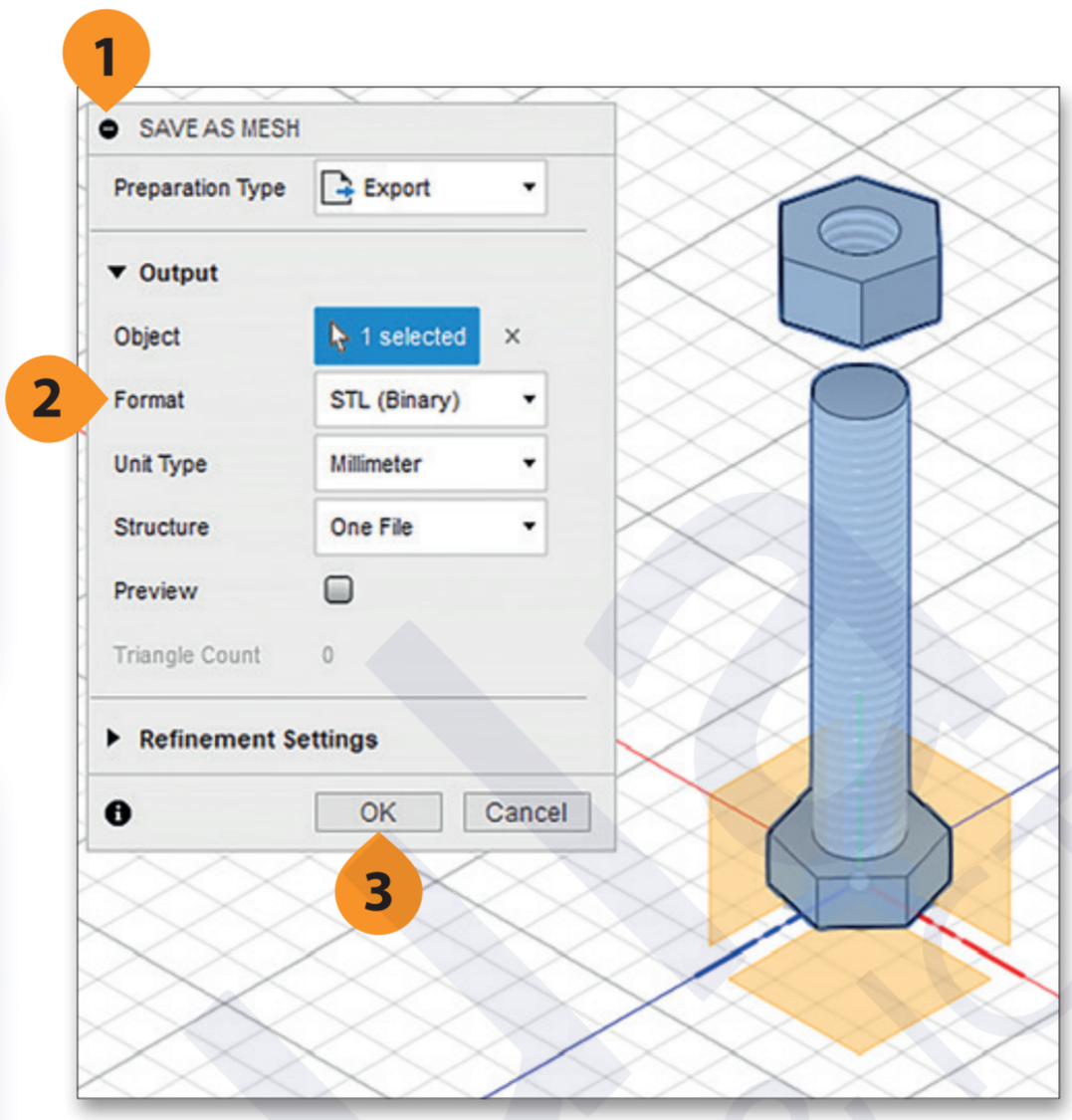
- натисніть **Save** або скористайтеся «гарячими» клавішами клавіатури

  Ctrl + S;
- назвіть проєкт **Bolt and Nut**;
- модель буде збережена у **Хмарному просторі Autodesk**.

Експорт файлів для майбутнього 3D-друку у програмі Autodesk Fusion:

- клацніть правою кнопкою на компоненті **Save as Mesh** (1);
- установіть **Формат файлу:** STL (.stl) (2);
- натисніть **OK** (3).

  Файл готовий для подальшої адаптації під конкретний верстат чи 3D-принтер та виготовлення матеріальної моделі.



ДОВІДКА

У безкоштовній версії застосунку **Autodesk Fusion** у хмарному сховищі можна зберігати до 10 проєктів.

Висновки

Після виконання цієї роботи ви навчилися створювати деталі з різьбовим з'єднанням, використовувати параметричні інструменти Fusion та готувати модель до подальшого виробництва. Ці навички є базовими для роботи у сфері технічного моделювання, інженерії та проєктування.

На завершення цієї теми варто підкреслити, що 3D-моделювання та використання CAD-систем є одним із ключових напрямів сучасного інженерного проєктування і дизайну. Здобуті знання про основи тривимірного моделювання, ознайомлення із CAD-системами, зокрема Autodesk Fusion 360, відкривають широкі можливості для створення точних і складних деталей, оптимізованих для виробництва та 3D-друку.

Ця тема допомогла засвоїти фундаментальні принципи роботи у Fusion – від реєстрації і встановлення програми до створення базових моделей гвинта й гайки.

Подальший розвиток навичок у 3D-моделюванні дасть змогу не лише розширити спектр складності проєктів, а й ефективно впроваджувати інновації у різних сферах інженерії, промислового дизайну та виробництва. Бажаємо успіхів у подальшому навчанні та творчому застосуванні цих технологій!



Дотримання технології моделювання і підготовки файлів є гарантією високої якості кінцевих виробів та успішної роботи із сучасними адитивними технологіями.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Додаткова тема
3D-друк

Вступ

3D-друк – це інноваційна технологія створення об'єктів поетапно, шар за шаром, безпосередньо з комп'ютерної 3D-моделі. Він трансформує будь-яку ідею в реальний фізичний предмет за короткий час.

Ця технологія застосовується у багатьох сферах: від медицини та архітектури до хобі й освіти. 3D-друк відкриває нові можливості для творчості, розробки власних винаходів і розуміння сучасних технологічних процесів.



Застосування технології 3D-друку

Друк у космосі, на Місяці та Марсі
Космічні агентства, як-от NASA, активно досліджують 3D-друк для створення запасних частин безпосередньо в космосі. Це дає змогу астронавтам ремонтувати обладнання, не чекаючи доставки із Землі

Біодрук та о́ргани
У галузі біодруку (bioprinting) вчені використовують 3D-принтери для створення структур, що імітують людські тканини та о́ргани, використовуючи біочорнила (клітини). Хоча друк повноцінного функціонального о́ргана все ще є викликом, але вже використовується для тестування ліків

«Друковані» будівлі
У будівельній галузі 3D-друк використовується для швидкого зведення цілих будинків зі спеціального бетонного складу. Це потенційно може прискорити будівництво доступного житла

*Термін **3D-друк (3D printing)** є популярним, але більш технічно коректною є назва **Адитивне виробництво (Additive Manufacturing, AM)**. Ця назва підкреслює його відмінність від традиційних методів, таких як фрезерування чи точіння, які є **субтрактивним виробництвом** (віднімання матеріалу).*



ІСТОРІЯ

Перший запатентований метод 3D-друку, відомий як **стереолітографія (SLA)**, був винайдений американським інженером **Чаком Халлом (Chuck Hull)** ще у 1986 році. Він же став співзасновником компанії 3D Systems. Справжній бум 3D-друку для широкого загалу розпочався приблизно у **2009 році**, коли закінчився термін дії патенту на ключову технологію **FDM (Fused Deposition Modeling)**, що зробило її доступною для розробки недорогих домашніх принтерів.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Для чого використовується 3D-друк

3D-друк використовують для:

- **Виробництва прототипів** – це допомагає швидко створювати зразки нових виробів і тестувати їх
- **Малосерійного виробництва складних деталей**, які важко зробити традиційними методами
- **Виготовлення медичних імплантів**, протезів та моделей о́рганів
- **Виготовлення унікальних предметів**, наприклад дизайн-аксесуарів
- **Створення архітектурних макетів** для точного уявлення про будівлі

Ефективність 3D-друку в цих сферах базується на трьох ключових теоретичних перевагах над традиційним (субтрактивним) виробництвом.



Ключові переваги ефективності 3D-друку над традиційним (субтрактивним) виробництвом:

Теорія швидкого прототипування (Rapid Prototyping)

Це є основою прискореної розробки продукту. Замість тривалого виготовлення зразка традиційними методами, 3D-друк дає змогу отримати фізичний прототип за години. Це значно скорочує цикл **«Дизайн – Тест – Ітерація»**, мінімізуючи час виходу продукту на ринок та зменшуючи витрати на ранніх етапах проєкту.

Принцип геометричної складності (Design Freedom)

На відміну від фрезерування, вартість виготовлення деталі на 3D-принтері не залежить від її геометричної складності. Це допомагає створювати органічні форми, порожнисті структури та використовувати оптимізацію топології, де матеріал розташований лише там, де він необхідний. Як наслідок – створення легших, міцніших та менш матеріалоємних деталей.

Ідея персоналізації та кастомізації (Mass Customization)

Завдяки 3D-друку можна перейти від моделі масового виробництва (ідентичні товари) до масової кастомізації (унікальні товари у великих обсягах). Оскільки налаштування принтера мінімально залежать від індивідуальних параметрів моделі, кожен виготовлений об'єкт може бути унікальним (наприклад, протез, створений за сканом конкретної людини) без суттєвого збільшення вартості одиниці продукції.

Технології 3D-друку

Існує кілька основних технологій, але важливо зосередитися на тих, які найбільш доступні й поширені для домашнього використання.

Технології 3D-друку

- **FDM (Fused Deposition Modeling)** – найпопулярніша технологія в домашніх 3D-принтерах. Принтер нагріває пластиковий філамент і шар за шаром розкладає його на платформі. Підходить для друку простих і міцних об'єктів із різних видів пластику
- **SLA** – використовує лазер для затвердіння рідкої фотополімерної смоли. Забезпечує дуже високу детальність, але вимагає дорожчого обладнання і матеріалів
- **DLP і MSLA** – аналогічні SLA технології, але використовують світлодіоди і LCD-екрани для полімеризації шару смоли. Ці принтери більш доступні, ніж SLA, і теж використовуються для точного друку
- **SLS (Selective Laser Sintering)** – лазер спікає порошкові матеріали (пластик, метал). Технологія для промислового використання, рідко застосовується в домашніх умовах через дороговизну

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Порівняльні характеристики технологій 3D-друку

Тип принтера	Технологія друку	Основні матеріали	Переваги	Недоліки	Домашнє використання
FDM/FFF	Наплавлення розплавленого пластику	PLA, ABS, PETG, TPU	Дешеві, прості у використанні, широкий вибір матеріалів	Низька деталізація порівняно з SLA	Широко використовується
SLA	Лазерне затвердіння смоли	Фотополімерні смоли	Висока якість друку, деталі з високою роздільною здатністю	Дорожчі матеріали, складний догляд	Рідше, переважно досвідчені користувачі
DLP/MSLA	Світлодіодна полімеризація	Фотополімерні смоли	Відносно низька ціна порівняно з SLA, висока деталізація	Обмежений розмір друку, складність у догляді	Достатньо популярні в домашніх умовах
SLS	Лазерне спікання порошку	Пластик, металевий порошок	Висока міцність виробів, немає потреби у підтримках	Дуже дорога техніка і матеріали	Переважно промислове використання

FDM-друк

Теорія та застосування

Принцип роботи FDM-принтера

Подача та нагрівання
Філамент (спеціальна термопластична нитка) подається з котушки до друкувальної головки – екструдера. Всередині екструдера розташований нагрівальний елемент, який плавить матеріал до напіврідкого стану

Екструзія
Розігрітий матеріал видавлюється (екструдується) через сопло (форсунку) дуже малого діаметра (зазвичай 0.4 мм), утворюючи тонкий, контрольований потік

Пошарове наплавлення
Екструдер рухається у горизонтальній площині (осі X та Y) згідно з траєкторією, визначеною 3D-моделлю, накладаючи матеріал на друковану платформу. Після завершення шару платформа опускається (або екструдер піднімається) по осі Z, і процес повторюється, доки об'єкт не буде завершено

З'єднання
Матеріал швидко застигає та сплавляється з попереднім шаром завдяки залишковому теплу та тиску, що забезпечує міцність готової деталі

***FDM (Fused Deposition Modeling)**, або **Моделювання методом пошарового наплавлення**, є найпоширенішою та найдоступнішою технологією 3D-друку, яка використовується як у промисловості, так і в домашніх умовах.*



Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Філаменти для 3D-друку: види та характеристики

Функціональність та якість об'єкта, надрукованого за технологією FDM, безпосередньо залежать від властивостей витратного матеріалу. Саме термопластична нитка, відома як **філамент**, визначає механічну міцність, термічну стійкість, гнучкість, а також потребує відповідних температурних умов для екструзії та друку.



***Філамент** – це серце **FDM-технології**. Він є однорідним матеріалом, намотаним на котушку, який повинен мати точний, калібрований діаметр (зазвичай 1.75 мм або 2.85 мм) для коректної роботи екструдера.*



***Вибір філаменту** є критичним кроком, оскільки його характеристики диктують, для якої мети може бути використана надрукована деталь.*

*Також **філамент може мати різне забарвлення**, як постійне, так і таке, що змінюється градієнтом, і навіть таке, що світиться в темряві.*

Порівняльна таблиця типів філаменту

Філамент	Особливості	Переваги	Недоліки	Застосування
PLA (Полілактидна кислота)	Біорозкладний, виготовляється з природних матеріалів	Легкий друк, без запаху, екологічний	Низька термостійкість, крихкий	Прототипування, декоративні вироби
ABS (Акрилонітрил-бутадієн-стирол)	Висока міцність, термостійкість	Стійкий до ударів, гнучкий	Виділяє запах під час друку, складніше друкувати	Функціональні деталі, корпуси
PETG	Стійкий до хімікатів і вологи	Комбінує міцність ABS і легкість PLA	Може бути гігроскопічним (вбирає вологу)	Контейнери, технічні деталі
TPU (Термопластичний поліуретан)	Гнучкий, еластичний	Амортизатори, деталі, що піддаються деформації	Складний у друці	Амортизація, прокладки, елементи одягу та взуття
Нейлон (PA)	Дуже міцний, зносостійкий	Висока міцність і гнучкість	Поглинає вологу, складний друк	Механічні деталі, зубчасті колеса

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

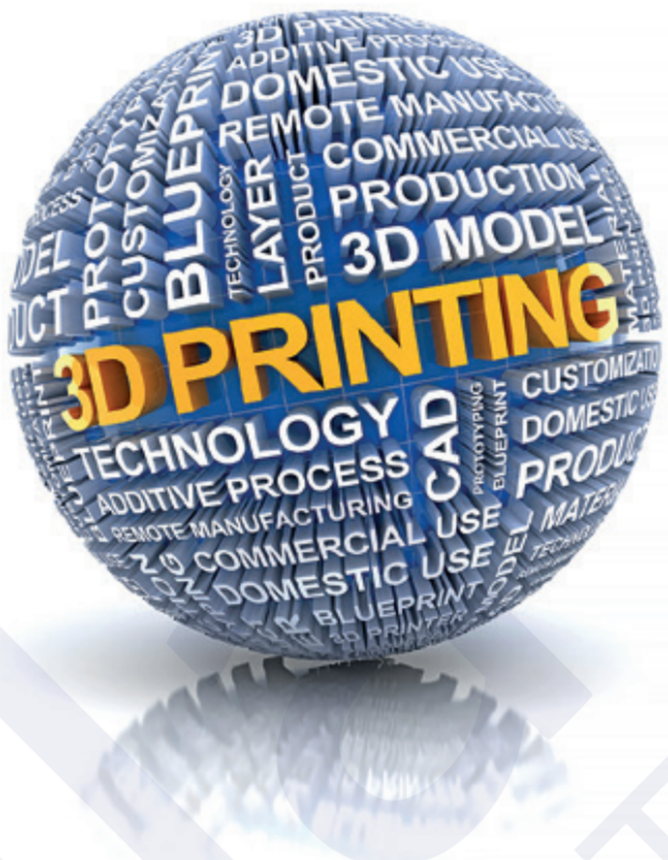
Бібліотеки готових 3D-моделей

Бібліотеки готових 3D-моделей є основним джерелом контенту для користувачів 3D-принтерів, які не володіють навичками 3D-моделювання або шукають об'єкти для негайного друку. Ці онлайн-платформи функціонують як величезні репозиторії, де дизайнери та ентузіасти з усього світу діляться своїми файлами, зазвичай у форматі **.STL (Standard Tessellation Language**).

Вони пропонують моделі для будь-яких цілей: від практичних побутових інструментів (тримачі, органайзери) і запасних частин до художніх об'єктів (фігурки, мініатюри) та освітніх посібників. Ефективність 3D-друку значною мірою залежить від цих бібліотек, оскільки вони демократизують процес, допомагаючи кожному, хто має принтер, легко знаходити, завантажувати та відтворювати різноманітні об'єкти.

Ці платформи різняться за своєю бізнес-моделлю та підходом до якості. Деякі зосереджені на масовості та безкоштовному доступі, покладаючись на саморегуляцію спільноти. Інші функціонують як маркетплейси або «кураторські» бібліотеки, де моделі перед публікацією проходять перевірку якості або продаються за невелику плату, забезпечуючи вищу деталізацію та гарантовану готовність до друку.

Користувачам важливо звертати увагу не тільки на естетику моделі, а й на супутні ліцензії, які визначають, чи можна використовувати файл лише для особистого користування, чи дозволено його модифікувати та комерційно розповсюджувати. Отже, вибір платформи залежить від потреб користувача: чи потрібна йому безкоштовна, але масова бібліотека, чи він готовий заплатити за перевірену і високоякісну модель.



Файл STL *– це універсальна мова для опису форми будь-якого об'єкта як набору з'єднаних трикутників, що дає змогу програмам-слайсерам точно визначити, де і як принтер має побудувати об'єкт шар за шаром.*

Thingiverse

Thingiverse є найбільшим і найвідомішим онлайн-репозиторієм для 3D-моделей серед платформ 3D-друку. Належність до компанії MakerBot сприяла його швидкому зростанню та накопиченню мільйонів безкоштовних файлів у форматі .STL, що охоплюють практично всі категорії – від іграшок та мистецтва до корисних побутових інструментів та складних механічних деталей.

Його головна перевага полягає у величезному обсязі контенту та активній спільноті, яка залишає коментарі та ділиться фотографіями своїх виробів. Однак через відкритість платформи якість деяких моделей може бути нерівномірною, і користувачам інколи доводиться витрачати час на пошук якісних, добре оптимізованих для друку файлів.

Для ознайомлення з бібліотекою **Thingiverse** перейдіть за посиланням
https://www.thingiverse.com/

Printables

Платформа **Printables**, створена компанією Prusa Research, швидко здобула популярність як високоякісна альтернатива Thingiverse. Її головна перевага – це акцент на надійності та якості моделей. Дизайнери активно заохочуються до завантаження файлів з детальною документацією, рекомендаціями щодо друку та фотографіями готових виробів, що значно підвищує ймовірність успішного друку для кінцевого користувача. Printables також підтримує високу активність спільноти, регулярно проводячи конкурси дизайну та надаючи винагороди авторам за їхні популярні моделі.

Для ознайомлення з бібліотекою **Printables** перейдіть за посиланням
https://www.printables.com/

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

MyMiniFactory

MyMiniFactory позиціонує себе як «кураторська» бібліотека, де головним пріоритетом є гарантована друковність. На відміну від відкритих платформ, кожна модель, перш ніж стати доступною для завантаження, проходить перевірку адміністраторами або спільнотою на відсутність помилок та придатність до друку.

Цей підхід мінімізує ризик невдалих спроб користувачів. Платформа спеціалізується на високодеталізованих моделях, особливо у сферах мініатюр, ігор, косплею та колекційних фігурок. MyMiniFactory працює як гібрид, пропонуючи значну кількість безкоштовного контенту, а також підтримуючи незалежних дизайнерів через продаж преміум-файлів.

Для ознайомлення з бібліотекою **MyMiniFactory** перейдіть за посиланням https://www.myminifactory.com/

Cults3D

Cults3D – це преміум-маркетплейс, який приваблює користувачів, що шукають унікальні, естетичні та інноваційні 3D-моделі високого рівня дизайну. Платформа є популярною для незалежних дизайнерів, які пропонують свої роботи, часто виділяючись естетичною складністю або оригінальністю концепції.

Тут можна знайти вишукані об'єкти для декору, ювелірні вироби та функціональні предмети з елегантним дизайном, які рідко трапляються на безкоштовних ресурсах. Хоча більшість контенту є платним, що забезпечує пряму підтримку авторам, Cults3D також пропонує відбіркові безкоштовні файли як промо-матеріал.

Для ознайомлення з бібліотекою **Cults3D** перейдіть за посиланням https://cults3d.com/

Порівняння бібліотек 3D-моделей

Платформа	Домінантний тип контенту	Цінова модель	Якість та призначення
Thingiverse	Масовий, аматорський контент	100 % безкоштовно. Усі файли доступні для вільного завантаження	Різна якість. Ідеально підходить для експериментів та пошуку простих рішень
Printables	Якісний, перевірений контент	100 % безкоштовно. Усі файли доступні для вільного завантаження	Висока якість, з фокусом на друковності та детальних інструкціях
MyMiniFactory	Кураторський контент (мініатюри, фігурки)	Гібрид: безкоштовно + преміум. Значна кількість безкоштовних файлів, але багато унікальних, високодеталізованих моделей продаються	Висока, перевірена якість з акцентом на колекціонуванні та іграх
Cults3D	Дизайнерський, естетичний контент	Маркетплейс: переважно платно. Моделі продаються незалежними дизайнерами за індивідуальною ціною	Унікальний, естетичний дизайн; ідеально для декору, моди та вишуканих об'єктів

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Ліцензія

Коли дизайнер публікує свою модель у бібліотеці, він обирає конкретний вид ліцензії, яка чітко визначає, що саме дозволено робити користувачам, які завантажують цей файл. Вона встановлює межі вашого права: чи можете ви просто надрукувати об'єкт для особистого користування, чи дозволено вам вносити зміни у його дизайн, і найголовніше – чи маєте ви право продавати надруковані копії або змінені версії моделі.

Отже, ліцензія є юридичним інструментом, що захищає права автора, водночас надаючи користувачам легальні умови для взаємодії з його твором. Найпоширенішою системою ліцензування в спільноті 3D-друку є Creative Commons (CC).

Ліцензія *– це офіційний дозвіл від автора (правовласника) на використання його інтелектуальної власності, у цьому випадку – цифрової 3D-моделі.*

Види ліцензій Creative Commons для 3D-моделей

Код ліцензії	Назва ліцензії	Визначення	Дозволені дії	Обмеження
CC0	Public Domain Dedication	Модель передана у суспільне надбання	Будь-яке використання, зміна, продаж – без обмежень	Немає. Це найвільніша ліцензія
CC BY	Attribution (із зазначенням авторства)	Вимагає зазначення імені (нікнейму) оригінального автора	Використання, зміна, комерційний продаж – дозволено	Обов'язкове зазначення автора
CC BY-SA	Attribution-ShareAlike (із зазначенням авторства, поширення на тих самих умовах)	Вимагає зазначення автора, а також поширення похідних робіт (модифікацій) під тією самою ліцензією (SA)	Використання, зміна, комерційний продаж – дозволено	Обов'язкове зазначення автора; похідні роботи мусять мати CC BY-SA
CC BY-ND	Attribution-NoDerivatives (із зазначенням авторства, без похідних робіт)	Дозволяє іншим лише друкувати модель, але забороняє її змінювати (створювати похідні)	Використання, комерційний продаж – дозволено	Обов'язкове зазначення автора; заборона на зміну моделі (дизайну)
CC BY-NC	Attribution-NonCommercial (із зазначенням авторства, некомерційне використання)	Дозволяє використання та зміну, але забороняє будь-яке комерційне використання (продаж друкованих виробів)	Використання, зміна – дозволено	Обов'язкове зазначення автора; заборона на комерційний продаж
CC BY-NC-SA	Attribution-NonCommercial-ShareAlike	Найбільш обмежувальна з популярних ліцензій CC	Використання, зміна – дозволено	Обов'язкове зазначення автора; заборона на комерційний продаж; похідні роботи мусять мати CC BY-NC-SA

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Процес завантаження та підготовки 3D-моделі до друку

Процес перетворення цифрового файлу на фізичний об'єкт складається з трьох основних кроків:

Завантаження → **Слайсинг** → **Передача на принтер**

Завантаження (Download)

Щоб завантажити 3D-модель:

- на обраній платформі **знайдіть потрібну модель.** Зверніть увагу на коментарі користувачів, які підтверджують успішність друку, а також на ліцензію (щоб переконатися, що ваше використання є дозволеним);
- натисніть кнопку **Download** або **Get this file**.
  Модель зазвичай завантажується у вигляді одного або кількох файлів із розширенням **.STL (Standard Tessellation Language)** або іноді **.OBJ** чи **.3MF**.

Слайсинг *– це найважливіший етап підготовки, під час якого програма-слайсер (наприклад,* ***Cura****,* ***PrusaSlicer*** *або* ***Simplify3D****) перетворює 3D-модель на серію пошарових інструкцій, які розуміє 3D-принтер.*

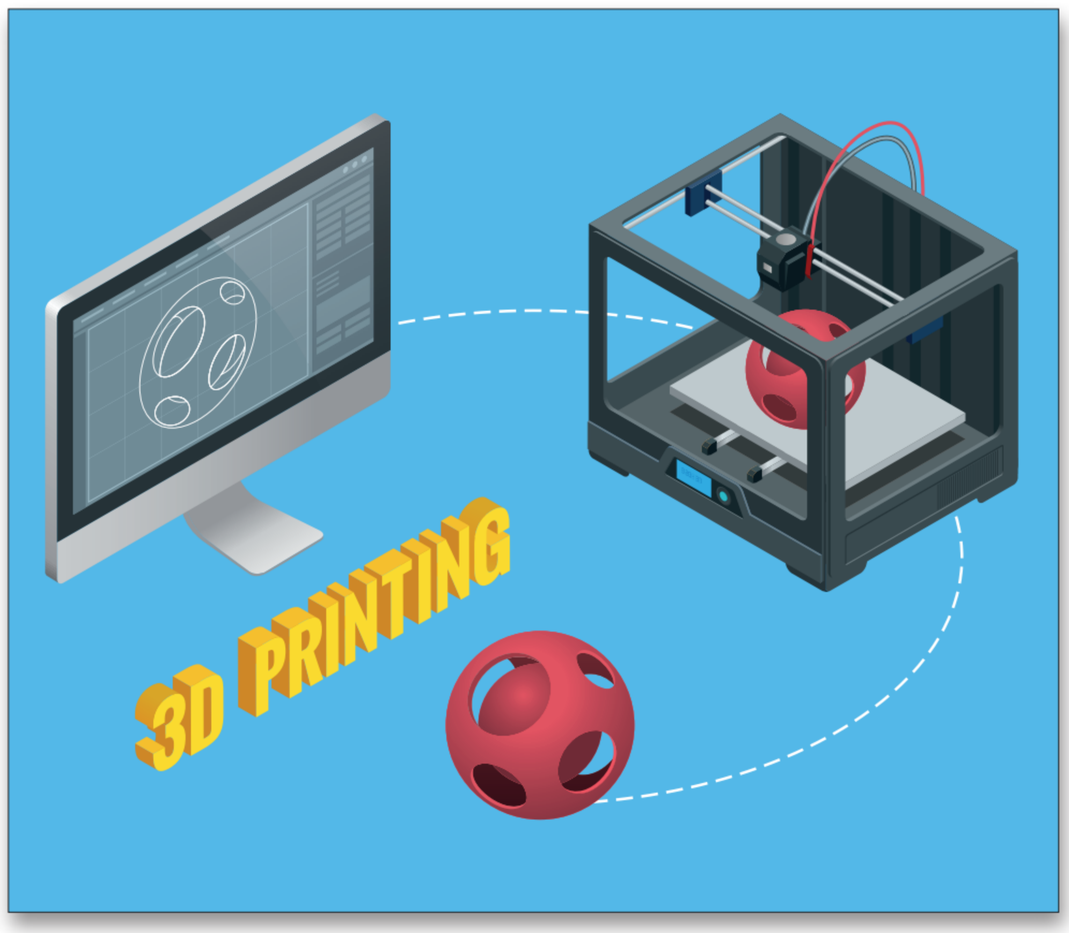


Слайсинг (Slicing)

Щоб перетворити 3D-модель на серію пошарових інструкцій для 3D-принтера:

- **відкрийте програму-слайсер** та **імпортуйте завантажений файл .STL.** Модель з'явиться на віртуальній робочій платформі;
- **налаштуйте орієнтацію:** за потреби поверніть модель, щоб мінімізувати необхідність використання опор або забезпечити максимальний контакт основи моделі з робочою поверхнею принтера;
- **оберіть профіль друку (Settings)**: установіть ключові параметри, які визначають якість і час друку:
  - **висота шару (Layer height)**: впливає на деталізацію (0.2 мм – стандарт, 0.1 мм – висока якість);
  - **заповнення (Infill)**: визначає внутрішню щільність моделі у відсотках (зазвичай 15–20 % для міцності);
  - **опори (Supports)**: потрібні для друку нависаючих частин моделі. Слайсер автоматично генерує їх;
  - **прилипання до столу (Adhesion/Raft/Brim)**: допомагає моделі не відклеїтися під час друку;
- після налаштування всіх параметрів програма виконує «слайсинг» і **генерує файл із розширенням .GCODE**.
  Цей файл містить тисячі команд, які точно вказують принтеру, куди рухати друкувальну голівку, коли подавати пластик і на якій швидкості.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.



Передача на принтер

Щоб надрукувати 3D-модель:

- збережіть згенерований файл **.GCODE** на SD-карту, USB-накопичувач або передайте його на принтер через Wi-Fi / мережу (залежно від моделі принтера).
- уставте носій даних у принтер, виберіть файл .GCODE у меню та **запустіть друк**. Принтер почне пошарово створювати об'єкт, керуючись інструкціями з файлу G-code.

Етапи постобробки 3D-моделей

Після того, як 3D-моделі надруковані, вони зазвичай потребують низки дій, які називаються **постобробкою (post-processing)**. Ці кроки необхідні для видалення допоміжних матеріалів, покращення зовнішнього вигляду та підвищення міцності виробу.

Основні кроки постобробки залежать від використаної технології друку та матеріалу. Нижче наведено покрокову постобробку для найпоширенішої технології FDM (пошарове наплавлення).



Крок 1: Очищення від допоміжних структур

Першим кроком після завершення друку є вивільнення моделі та її очищення від усіх допоміжних елементів. Об'єкт потрібно обережно зняти з робочої платформи, часто після її охолодження, щоб уникнути пошкодження основи. Найважливішим етапом є видалення опор – структур, які були надруковані для підтримки нависаючих частин моделі. Вони акуратно відламуються або відрізаються за допомогою кусачок та ножів.



Крок 2: Шліфування та вирівнювання поверхні

Після видалення опор поверхня моделі зазвичай залишається шорсткою та має видимі лінії шарів. Для досягнення гладкості проводиться шліфування. Цей процес починається з використання грубого наждачного паперу для усунення великих нерівностей і сліду від опор, поступово переходячи до паперу з дрібною зернистістю для повного згладжування поверхні. Для деяких матеріалів, як-от ABS, можна застосувати хімічну обробку. Цей метод розм'якшує зовнішній шар пластику, фактично усуваючи лінії шарів і надаючи виробу глянцевого вигляду без потреби тривалого ручного шліфування.



Крок 3: Фінішне оздоблення та складання

Завершальні етапи зосереджені на естетиці та функціональності. Якщо модель буде фарбуватися, спочатку наноситься ґрунтовка, яка покращує адгезію фарби та забезпечує рівномірне покриття. Після висихання модель фарбується акриловими або модельними фарбами. Фінальний шар лаку наноситься для захисту фарби та надання остаточного вигляду. Якщо модель була розроблена як збірна, окремі компоненти склеюються між собою або з'єднуються за допомогою кріпильних елементів, щоб утворити єдиний функціональний виріб.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

3D-Друк: революція, що змінює світ

Машина часу для дизайну

3D-друк, або адитивне виробництво, – це не просто технологія, це революційний спосіб мислення про те, як створюються речі. Уявіть, що ви маєте машину часу, яка може перетворити ідею у вашій голові на фізичний об'єкт лише за кілька годин. Саме це і робить 3D-друк!

Якщо раніше, щоб створити щось складне, потрібні були величезні заводи, верстати та місяці робóти, то сьогодні достатньо комп'ютера й настільного 3D-принтера. Ця технологія дала нам змогу перейти від масового виробництва (виробництва мільйонів однакових виробів) до масової кастомізації (виробництва мільйонів унікальних виробів).



Чому 3D-друк важливий?

Ключові переваги 3D-друку

Необмежена свобода дизайну

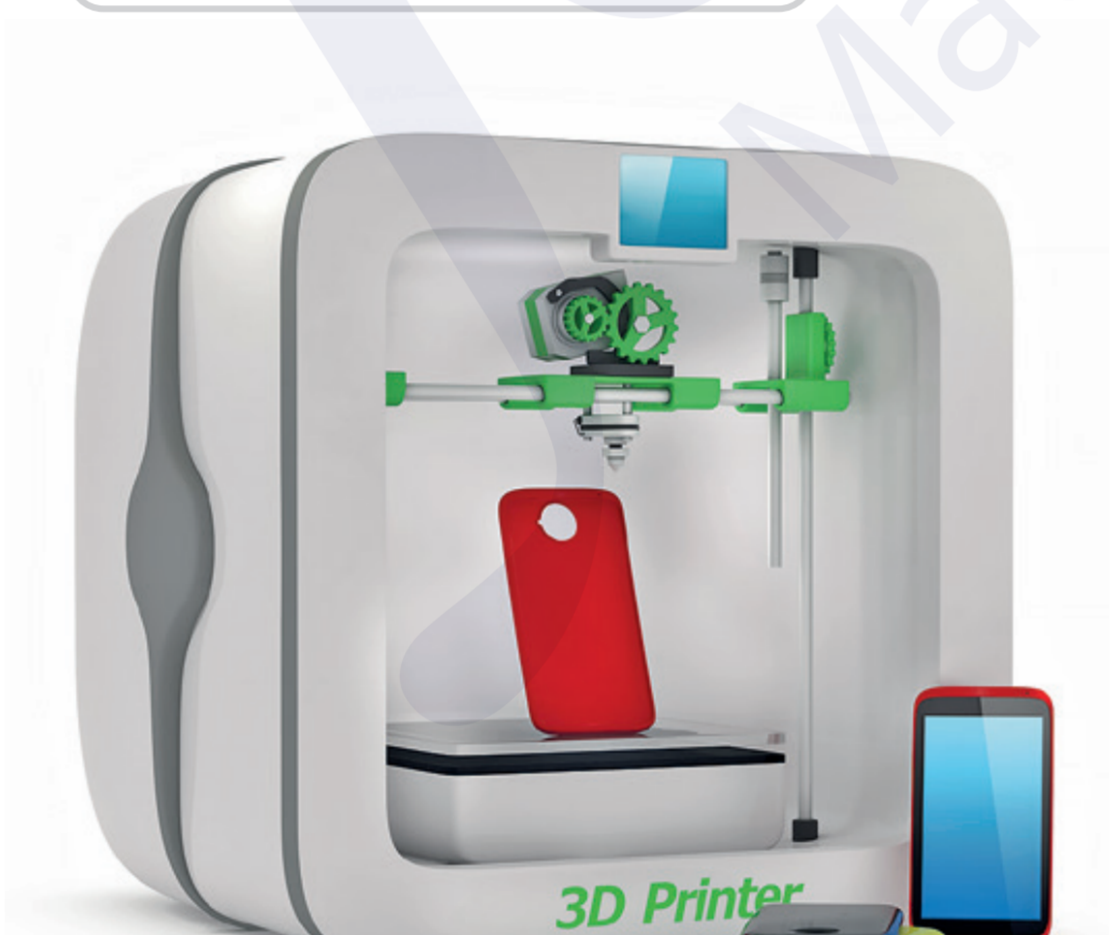
Звичайні методи виробництва (фрезерування, лиття) мають багато обмежень. Вони не можуть створити об'єкти з порожнинами, які неможливо дістати інструментом, або дуже складні сітчасті структури. 3D-друк знімає ці обмеження й уможливлює створення інженерами легших, міцніших та ефективніших деталей, які мають органічні форми, подібні до тих, що трапляються в природі.

Швидкість та економія

У світі технологій швидкість є вирішальною. Завдяки швидкому прототипуванню, компанії можуть створювати, тестувати, вдосконалювати та випускати новий продукт не за рік, а за кілька місяців. Це економить мільярди доларів і прискорює інновації.

Персоналізація майбутнього

3D-друк є основою для справжньої індивідуальної медицини. Сьогодні лікарі друкують протези, які ідеально підходять конкретному пацієнту, та хірургічні інструменти, розроблені під одну унікальну операцію. Завтра це може бути друк біотканин або навіть їжі, адаптованої під ваш персональний метаболізм.



3D-друк – це не просто друк пластиком. Це ключ до розуміння того, як працює сучасний світ проєктування та виробництва. Освоївши програми для CAD-моделювання, ви отримуєте суперсилу – здатність перетворювати нематеріальні дані (цифрову модель) на матеріальну реальність.

У майбутньому адитивне виробництво використовуватимуть в інженерній, дизайнерській, лікарській та навіть продовольчій галузях. 3D-друк переносить фабрику з великих промислових зон прямо на ваш стіл.

3D-друк – це технологія,
яка вже змінила правила гри.
Яку річ ви надрукуєте першою,
щоб змінити свій світ?

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

Навчальне видання

Джон Ендрю **Біос**

ІНФОРМАТИКА

Підручник для 9 класу закладів загальної середньої освіти

Автор українського контенту ***Ілля Вернер***

Головна редакторка ***Дарія Орловцева***
Редакторки-методистки: ***Олена Вернигора, Катерина Мищенко***
Дизайнери/ки: ***Вікторія Субочева, Рустам Керімов, Надія Коровянська***
Коректорка ***Ольга Пластун***
Перекладач ***Ростислав Вернигора***

Формат 60×90/8. Ум. друк. арк. 26. Обл.-вид. арк. 23,4.
Наклад 0000 прим. Зам. № 000000.

ТОВ «Видавництво "Лінгвіст"». 04208, м. Київ, просп. Василя Порика, 9-Г, к. 58.
Адреса редакції: 04071, м. Київ, вул. Нижній Вал, 17/8. E-mail: publishing@linguist.ua. Тел. (044) 599-32-28.
Свідоцтво про внесення суб'єкта видавничої справи до Державного реєстру видавців,
виготівників і розповсюджувачів видавничої продукції серія ДК № 6168 від 03.05.2018 р.

Віддруковано у

Please note: This book contains links to websites that are not maintained by Binary Logic. Although we make every effort to ensure these links are accurate, up-to-date and appropriate, Binary Logic cannot take responsibility for the content of any external websites.
Trademark notice: Product or corporate names mentioned herein may be trademarks or registered trademarks and are used only for identification and explanation without intent to infringe. Binary Logic disclaims any affiliation, sponsorship, or endorsement by the respective trademark owners. Microsoft, Windows, Windows Live, Office, Office 365, Access, Excel, PowerPoint, Outlook, Teams, OneDrive, OneNote, Bing, Edge, Skype, and Kodu are trademarks or registered trademarks of Microsoft Corporation. Google, Gmail, Chrome, Google Docs, Google Drive, Android, Blogger, Google Maps, Google Translate, and YouTube are trademarks or registered trademarks of Google Inc. Apple, iPad, iPhone, Pages, Numbers, Keynote, iCloud, and Safari are registered trademarks of Apple Inc. LibreOffice is a registered trademark of the Document Foundation. Facebook, Messenger, Instagram, and WhatsApp are trademarks owned by Facebook and its affiliate companies. Twitter is a trademark of Twitter, Inc. Zoom is a trademark of Zoom Video Communications, Inc. Cisco WebEx is a trademark of Cisco Systems, Inc. The above companies or organizations do not sponsor, authorize, or endorse this book.
The publisher has made every effort to trace all copyright holders, but if they have inadvertently overlooked any they will be pleased to make the necessary arrangements at the first opportunity.

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.

НОВА УКРАЇНСЬКА ШКОЛА

Джон Ендрю Біос

ІНФОРМАТИКА 9

ЦИФРОВІ ПІДЛІТКИ

- Виклад навчального матеріалу представлений в доступній, чіткій та ілюстративній формі відповідно до вікових особливостей учнів і учениць.
- Підручник створено з урахуванням компетентнісного підходу в освіті.
- Відповідає міжнародним стандартам навчання ІКТ грамотності та цифрових компетентностей DigComp.
- Містить цифровий контент з додатковими інтерактивними вправами та файлами-заготовками до практичних робіт.

ISBN 978-617-8146-65-8

Усі права захищені. Ніяка частина цієї публікації не може бути відтворена, збережена в пошуковій системі або передана в будь-якій іншій формі будь-якими способами без письмового дозволу видавця. © Binary Logic та © Formula надає дозвіл на копіювання цих сторінок із позначкою «фотокопія» для вчителів, які працюють за виданнями видавництва «Лінгвіст». За жодних обставин ніяка частина цього матеріалу не може бути скопійована для перепродажу або для іншого використання.