МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД
“ДОНЕЦЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ”
Кафедра прикладної математики та інформатики

Методичні вказівки і завдання до практичних занять з дисципліни “ Організація зображень та мультимедіа, бази даних “

Для студентів освітнього ступеня «Бакалавр»,
що навчаються за спеціальністю
014.04 – Середня освіта (Математика)

ДРОГОБИЧ-2025

УДК 004.65+004.932+004.032.6](072)

М 54

Методичні вказівки і завдання до практичних занять з дисципліни «Організація зображень та мультимедіа, бази даних» для студентів освітнього ступеня «Бакалавр», що навчаються за спеціальністю 014.04 – Середня освіта (Математика) усіх форм навчання / уклад. М.О. Александров. – Дрогобич: ДонНТУ, 2025. – 128 с.

Охоплюються основні аспекти організації та обробки мультимедійних даних, а також роботу з базами даних у контексті зображень та мультимедіа. Завдання спрямовані на закріплення теоретичних знань і розвиток практичних навичок студентів, що вивчають цю дисципліну.

Укладач: Микита АЛЕКСАНДРОВ, д.ф., доцент кафедри ПМІ

Рецензент: Сергій КОВАЛЬОВ, к.т.н., доц., зав. каф. ЕТ і КІ

Відповідальний за випуск: Ярослав ДОРОГИЙ, д.т.н., проф., зав. каф. ПМІ

Затверджено навчально-методичним відділом ДВНЗ ДонНТУ,
протокол №5 від 25.03.2025 р.

Розглянуто на засіданні кафедри ПМІ, протокол № 2 від 21.02.2025р.

ЗМІСТ

ВСТУП

В умовах стрімкого розвитку технологій обробки мультимедійних даних та інтеграції різних типів медіафайлів у сучасні інформаційні системи, набуває важливості глибоке розуміння організації та зберігання мультимедіа. Це стосується не тільки зображень та відео, а й інших типів медіафайлів, таких як аудіофайли, що широко використовуються в освіті, науці та різних галузях.

Одним із ключових аспектів навчання в рамках дисципліни «Організація зображень та мультимедіа, бази даних» є освоєння методів обробки та зберігання мультимедійних даних у базах даних. Студенти ознайомлюються з основними підходами до роботи з різними типами медіа, такими як зображення, відео, аудіо, а також із принципами використання метаданих для опису та класифікації цих файлів.

Методичні вказівки та завдання практичних занять для студентів спеціальності 014.04 «Середня освіта (Математика)» спрямовані на розвиток навичок у роботі з мультимедійними даними через практичні завдання. У процесі виконання лабораторних занять студенти навчаться працювати з такими інструментами, як Google Sheets, Google Sites, а також API для автоматичної обробки файлів.

Ці матеріали мають на меті підготувати майбутніх фахівців, які здатні ефективно організовувати та обробляти мультимедійні дані, що є важливим аспектом сучасної освіти та науки.

Практичне заняття №1

«Створення та аналіз зображень»

Мета: навчитись основам роботи з графічними зображеннями, включаючи створення та редагування простих зображень, а також їх аналіз. Ознайомити з різними форматами зображень та інструментами для їх обробки, що дозволить отримати практичні навички в обробці мультимедійних даних.

Теоретичні відомості

1.1 Визначення

Зображення – це двовимірне представлення даних, яке складається з безлічі елементів, що відображають колір або відтінок на конкретній площині. Найбільш поширені типи зображень – растрові та векторні. Кожне зображення містить інформацію про його пікселі (у растрових зображеннях) або математичні елементи (у векторних зображеннях).

Зображення складаються з **пікселів** (скорочено від англ. "picture element"), кожен з яких має своє значення кольору та розташування в матриці. Кожен піксель відповідає певному кольору, що визначається різними кольоровими моделями, такими як RGB або CMYK.

Растрові зображення можна розглядати як **матрицю пікселів** (двовимірну таблицю), де кожен піксель має свої координати і значення. Кількість пікселів, що утворюють таке зображення, визначає його роздільну здатність, яка вимірюється в **пікселях на дюйм** (PPI або DPI). Кожен піксель може містити інформацію про колір, яскравість і прозорість, залежно від формату зображення.

Математично растрове зображення можна представити як матрицю, елементи якої є значеннями кольорів пікселів. Для кожного пікселя використовуються числові значення, які визначають інтенсивність червоного (R), зеленого (G) та синього (B) кольорів (модель RGB).

RGB модель для представлення кольору:

- Червоний (R): 0–255
- Зелений (G): 0–255
- Синій (B): 0–255

Таким чином, для одного пікселя зображення значення можуть бути представлені як трійка чисел (R,G,B)(R, G, B), де кожне значення визначає інтенсивність відповідного кольору в межах від 0 до 255. Таким чином, кольорові значення пікселя можуть бути, наприклад, (255,0,0)(255, 0, 0), що відповідає червоному кольору.

Формула для кольору пікселя: Колір пікселя $= (R, G, B)$

Векторно можна представити це через функції або рівняння, що описують функціональну залежність від положення пікселя на екрані.

Математично зображення розміром $w \times h$ можна уявити як двовимірну матрицю:

$$I = [I_{1,1}\, I_{1,2}\, \dots\, I_{1,w}\, I_{2,1}\, I_{2,2}\, \dots\, I_{2,w}\, \dots\, \dots\, \dots\, \dots\, I_{h,1}\, I_{h,2}\, \dots\, I_{h,w}]$$

де $I_{i,j}$ – це колір пікселя, що відповідає координатам (i,j).

Роздільна здатність зображення визначається числом пікселів на одиницю площі. Наприклад, зображення 1920x1080 пікселів має 1920 пікселів по горизонталі та 1080 по вертикалі.

Векторні зображення не містять пікселів, а використовують математичні рівняння для опису фігур. Вони описуються через геометричні елементи: **лінії**, **криві**, **кола**, **многокутники** та інші. Кожна фігура визначається набором математичних параметрів, таких як координати точок, радіус, кути, товщина лінії тощо.

Основною перевагою векторних зображень є те, що їх можна масштабувати без втрати якості. Це відбувається тому, що їх компоненти описуються математично, і при зміні розміру зображення розрахунки знову

генерують потрібні геометричні фігури без "розмиття" або пікселяції, що характерно для растрових зображень.

Математично, кожна фігура в векторному зображенні може бути описана через рівняння, які визначають її властивості. Наприклад:

Пряма лінія може бути описана рівнянням: $y = mx + b$, де m – це нахил лінії, а b – її перетин з віссю y.

Круг можна описати рівнянням: $(x - h)^2 + (y - k)^2 = r^2$ де (h, k) – координати центру кола, а r – його радіус.

Ці математичні рівняння дозволяють точно відтворювати фігури при будь-якому масштабуванні зображення.

Таблиця 1.1 Порівняння растрових та векторних зображень

Характеристика	**Растрові зображення**	**Векторні зображення**
Структура	Матриця пікселів	Геометричні елементи (лінії, криві)
Масштабованість	Втрата якості при збільшенні розміру	Без втрати якості при масштабуванні
Якість зображення	Визначається кількістю пікселів	Залишатиметься постійною при зміні розміру
Виготовлення	Фотографії, складні малюнки	Логотипи, діаграми, схеми
Використання	Фотографії, складні зображення	Логотипи, векторні ілюстрації

Растрові зображення використовуються для зображень з високою деталізацією, таких як фотографії, малюнки та складні графічні зображення, де необхідна велика кількість кольорів. Векторні зображення ідеальні для використання у графічних дизайнах, коли потрібно масштабувати зображення

до будь-якого розміру без втрати якості. Векторні формати використовуються в таких сферах, як створення логотипів, діаграм, ілюстрацій та схем.

1.2 Формати зображень

1.2.1 JPEG (.jpg, .jpeg)

JPEG (Joint Photographic Experts Group) – це формат стиснення з втратами, який був створений для збереження фотографічних зображень. Основною метою цього формату є досягнення високого рівня стиснення при збереженні прийнятної якості зображення, що робить його дуже популярним у веб-графіці, фотографії та цифрових медіа.

Алгоритм стиснення JPEG використовує методи перетворення зображення в частотну область за допомогою дискретного косинусного перетворення (DCT). Цей процес дозволяє перевести зображення у вигляді матриць пікселів у набір коефіцієнтів, що відповідають різним частотам (високим та низьким). Після цього знижуються коефіцієнти, що мають мале значення, що дозволяє зменшити обсяг даних. Це призводить до втрати певних дрібних деталей, що малопомітні для людського ока.

Формат був запропонований у 1992 році як результат співпраці між комітетом JPEG, який був утворений для стандартизації стиснення цифрових зображень. З того часу формат JPEG став основним стандартом для збереження та обміну зображеннями, завдяки високому стисненню і підтримці майже усіма пристроями та програмами.

1.2.2 PNG (.png)

PNG (Portable Network Graphics) – формат без втрат, розроблений для того, щоб стати вдосконаленим форматом заміни GIF з підтримкою прозорості. Основною перевагою PNG є здатність зберігати графіку з прозорими фонами, що стало важливим для сучасного веб-дизайну, а також здатність зберігати зображення без втрат якості.

PNG використовує метод стиснення, заснований на LZ77 (Lempel-Ziv 77) – алгоритмі без втрат, що є основою більшості сучасних методів стиснення. Алгоритм LZ77 замінює повторювані послідовності байтів на їх

короткі представлення. Внаслідок цього зображення можна зберігати з меншим розміром без втрати інформації.

PNG був розроблений в середині 1990-х років як відкрите і безплатне вдосконалення формату GIF, який був обмежений через патентні обмеження. Формат було введено в 1996 році як відкритий стандарт, що швидко отримав популярність завдяки підтримці прозорості та відсутності стиснення з втратами.

1.2.3 BMP (Bitmap)

BMP (Bitmap) – це простий формат для збереження растрових зображень, який зазвичай не використовує стиснення. Зображення в форматі BMP складається з пікселів, де кожен піксель зберігає інформацію про колір у вигляді RGB значень (червоний, зелений, синій).

Кожен піксель зображення зберігається в певному форматі кольору, зазвичай у 24 біт на піксель (8 біт на канал для червоного, зеленого та синього кольорів). Формат BMP не має компресії, що означає, що кожен піксель зберігається незалежно від сусідніх, що спричиняє великі розміри файлів. Зберігання кожного пікселя окремо робить цей формат ідеальним для використання в середовищах, де важлива максимальна точність без змін.

Формат BMP був розроблений компанією Microsoft в 1988 році як частина її Windows API для підтримки растрових зображень у операційних системах Windows. Його основною метою було забезпечити просте і швидке відображення зображень на екрані без необхідності в складних обчисленнях або стисненні.

1.2.4 GIF (.gif)

GIF (Graphics Interchange Format) – це формат для зображень, який підтримує до 256 кольорів і використовується для анімацій. Це один з найстаріших і найпопулярніших форматів, який активно використовувався для створення анімованих картинок в Інтернеті.

GIF використовує метод стиснення LZW (Lempel-Ziv-Welch), який є без втрат і здатен ефективно зберігати зображення з обмеженою палітрою

кольорів. Оскільки GIF підтримує лише 256 кольорів, він не підходить для збереження складних або кольорових зображень, але добре підходить для зберігання анімацій або малих графічних елементів з простими формами.

GIF був розроблений у 1987 році компанією CompuServe для передачі графічних зображень через Інтернет. В результаті популярності веб-графіки в 1990-х роках GIF став стандартом для створення анімацій в Інтернеті, однак через обмежену палітру кольорів його використання для фотографій було обмежено.

1.2.5 TIFF (.tif)

TIFF (Tagged Image File Format) – це формат для збереження високоякісних зображень, який часто використовується в професійному друку, скануванні і цифровій фотографії. TIFF підтримує стиснення без втрат і з втратами, а також може зберігати багатошарові зображення.

TIFF зберігає зображення у вигляді пікселів, причому кожен піксель може містити кілька кольорів або альфа-канал для прозорості. Формат підтримує використання різних алгоритмів стиснення, таких як LZW для без втрат і JPEG для стиснення з втратами. Це дозволяє вибирати оптимальні варіанти стиснення залежно від вимог до якості і розміру файлів.

TIFF був розроблений в 1986 році компанією Aldus (яка пізніше стала частиною Adobe) для збереження високоякісних зображень в публікаціях і цифрових архівах. Оскільки формат не має патентних обмежень, TIFF став популярним у професіональних сферах, де потрібні зображення високої якості, такі як в цифровій фотографії та друкованих виданнях.

1.3 Основні характеристики зображень

1.3.1 Роздільна здатність

Роздільна здатність зображення – це кількість пікселів, які складають зображення, і визначає, скільки окремих одиничних точок зображення можна побачити на екрані або в друкованому вигляді. Роздільна здатність зазвичай виражається як ширина та висота зображення в пікселях, наприклад,

1920x1080. Це означає, що зображення має 1920 пікселів по горизонталі і 1080 пікселів по вертикалі.

Якщо ми маємо зображення роздільною здатністю 1920x1080, то загальна кількість пікселів у зображенні визначається як:

$N_{pixels} = 1920 \times 1080 = 2{,}073{,}600$ пікселів.

Кожен піксель може бути окремо оброблений для зміни кольору чи яскравості, що дає можливість створювати чітке зображення.

Вища роздільна здатність означає, що зображення має більшу кількість пікселів, що дозволяє відображати більше деталей і чіткість. Проте збільшення роздільної здатності також веде до збільшення розміру файлу зображення. Наприклад, 4K роздільна здатність (3840x2160) містить близько 8,3 мільйонів пікселів, що майже в чотири рази більше, ніж стандартна Full HD роздільна здатність (1920x1080).

1.3.2 Колірна глибина

Колірна глибина визначає, скільки бітів відводиться на представлення кольору одного пікселя. Це прямо впливає на кількість кольорів, які можуть бути відображені в зображенні.

8 біт на піксель (256 кольорів):

У такому випадку кожен піксель зображення має 8 біт, що означає, що для кожного кольору може бути представлено 256 різних рівнів. Для кольорових зображень це зазвичай використовується для палітрових зображень, де кожен піксель відображається як індекс у палітрі з 256 кольорів.

Оскільки 8 біт – це 1 байт, ми можемо записати кількість можливих кольорів:

$$\text{Кількість кольорів} = 2^8 = 256.$$

24 біт на піксель (16,7 мільйона кольорів):

У стандартному кольоровому зображенні з **24 бітами на піксель** (RGB), 8 біт використовуються для кожного з трьох основних кольорів: червоний (R),

зелений (G) і синій (B). Кожен канал може представити 256 рівнів кольору, отже, кількість можливих кольорів:

$$\text{Кількість кольорів} = 256 \times 256 \times 256 = 16{,}777{,}216 \text{ кольорів.}$$

Це дозволяє створити зображення з високою кольоровою точністю та плавними переходами між відтінками.

32 біт на піксель (включаючи альфа-канал):

У разі 32-бітних зображень один з каналів може бути використаний для збереження прозорості (альфа-канал), що дозволяє зберігати не лише колір, а й інформацію про прозорість кожного пікселя.

1.3.3 Прозорість

Прозорість визначається наявністю **альфа-каналу**, що дозволяє частково або повністю робити фон зображення прозорим. Це особливо важливо для графічного дизайну, логотипів та веб-графіки, де зображення може бути накладено на різні фони або інші елементи дизайну без видимих кордонів.

Альфа-канал використовує додатковий біт для кожного пікселя для визначення прозорості. Значення альфа-каналу зазвичай варіюється від 0 (повна прозорість) до 255 (повна непрозорість). Таким чином, для кожного пікселя зображення з альфа-каналом маємо чотири значення – для червоного, зеленого, синього кольору та прозорості:

$$RGBA = (R, G, B, A).$$

Для зображення з альфа-каналом, де кожен канал (R, G, B) займає 8 біт, а альфа-канал також 8 біт, кількість можливих варіацій кольору для одного пікселя буде:

$$\text{Кількість кольорів з прозорістю} = 256 \times 256 \times 256 \times 256 = 4{,}294{,}967{,}296 \text{ варіантів.}$$

Це дозволяє створювати зображення, які можуть бути частково прозорими або мати складні градації прозорості, що широко використовується в дизайні та веб-розробці.

1.4 Інструменти для роботи з зображеннями

1.4.1 GIMP (GNU Image Manipulation Program)

GIMP - це потужний безкоштовний редактор растрових зображень, який використовується для створення та обробки фотографій і графічних елементів. Він має великий набір інструментів для ретушування, корекції кольору, роботи з шарами, а також дозволяє застосовувати різноманітні фільтри та ефекти.

Основні можливості GIMP:

Растрові інструменти: Обрізка, масштабування, зміна роздільної здатності, корекція кольору, фільтри для модифікації зображення.

Робота з шарами: GIMP підтримує роботу з багатошаровими зображеннями, що дозволяє редагувати окремі елементи без впливу на всю картину.

Маски: Використання масок дозволяє створювати складні ефекти, включаючи прозорість і градієнти.

Підтримка плагінів: GIMP дозволяє додавати нові інструменти через плагіни, що розширюють можливості програми.

Головні формати, що підтримує GIMP: JPEG, PNG, BMP, GIF, TIFF, PSD (формати Adobe Photoshop) та інші.

1.4.2 Inkscape

Inkscape - це **безкоштовний редактор векторної графіки**, який використовується для створення та редагування векторних зображень у форматі **SVG** (Scalable Vector Graphics). Векторна графіка описує зображення за допомогою математичних рівнянь (лінії, криві, кола, полігони), що дозволяє масштабувати зображення без втрати якості.

Основні можливості Inkscape:

Створення векторних малюнків: Векторні елементи можна масштабувати без втрат якості, що ідеально підходить для логотипів, схем, інфографіки та ілюстрацій.

Редагування SVG: Формат SVG є відкритим стандартом для веб-дизайну, і Inkscape дозволяє працювати з цими файлами ефективно.

Підтримка шрифтів: Inkscape підтримує як текстові елементи, так і шрифти векторних форматів, що дозволяє створювати стильні графічні дизайни з текстовими елементами.

Ефекти і фільтри: Програма має вбудовані фільтри для створення ефектів на векторних об'єктах, таких як тіні, градієнти та інші.

Головні формати, що підтримує Inkscape: SVG, EPS, PDF, DXF, AI, CDR (CorelDRAW), а також імпорт векторних форматів, таких як **PDF, EPS**, і **PostScript**.

1.4.3 Adobe Photoshop

Adobe Photoshop – це **професійний редактор растрових зображень**, який використовується для створення, редагування, ретушування та маніпулювання зображеннями. Це одна з найпопулярніших програм серед дизайнерів, фотографів і художників.

Основні можливості Photoshop:

Масштабування та обрізка зображень: Photoshop надає розширені інструменти для роботи з масштабуванням та обрізкою зображень без втрати якості.

Корекція кольору та освітленості: За допомогою інструментів корекції кольору, як-от рівні, криві, колірні фільтри, користувачі можуть точно налаштувати кожен аспект зображення.

Маски та шари: Використання масок дозволяє маніпулювати окремими частинами зображення, зберігаючи інші області незмінними. Шари дозволяють працювати з різними елементами зображення незалежно.

Фільтри та ефекти: Photoshop має велику бібліотеку фільтрів для створення різноманітних ефектів, таких як розмиття, текстури, налаштування освітлення.

Головні формати, що підтримує Photoshop: PSD (власний формат), **JPEG, PNG, GIF, TIFF, BMP** та інші.

1.4.4 Adobe Illustrator

Adobe Illustrator – це **професійний редактор векторних зображень**, що дозволяє створювати складні малюнки, ілюстрації, логотипи та типографіку. Він широко використовується в графічному дизайні, а також для створення веб-графіки та векторної анімації.

Основні можливості Illustrator:

Створення складних ілюстрацій: Illustrator дозволяє створювати високоякісні векторні ілюстрації, які не втрачають якості при зміні масштабу.

Типографіка: Програма має розвинуті можливості для роботи з шрифтами, даючи змогу створювати стильні текстові елементи.

Інтерактивні ефекти: Підтримка застосування різноманітних ефектів та стилів до векторних елементів, таких як градієнти, тіні, обводки.

Підтримка графічних стилів: Illustrator дозволяє створювати і застосовувати стилі для елементів, що використовуються повторно у дизайні.

Головні формати, що підтримує Illustrator: AI (власний формат), **SVG, EPS, PDF**.

Завдання

1. **Створення простого логотипу** за допомогою векторного редактора (Inkscape):

- Створіть базовий геометричний логотип (коло, квадрат, лінії, текст).
- Використовуйте різні інструменти для редагування форм і кольорів.
- Збережіть логотип у форматі SVG.

2. **Редагування растрового зображення**:

- Відкрийте довільну фотографію або малюнок у GIMP або Paint.
- Змініть розміри зображення та обріжте непотрібні частини.
- Використовуйте інструменти для корекції кольору (яскравість, контраст).
- Збережіть результат у форматі PNG.

3. **Конвертація між форматами**:

- Перетворіть створене векторне зображення (SVG) з завдання 1 у растровий формат (JPEG або PNG).
- Перетворіть растрове зображення (PNG) з завдання 2 у JPEG.
- Порівняйте розмір файлів та якість зображень після конвертації.

4. **Аналіз властивостей зображень**:

Для кожного з форматів (SVG, PNG, JPEG) перевірте такі характеристики:

- Роздільна здатність (на прикладі растрових зображень).
- Розмір файлу та його вплив на якість.
- Чи підтримує формат прозорість (для PNG та SVG).

5. **Висновки**.

Загальні рекомендації до виконання звіту

1. **Чіткість і структурованість**.

Опис кожного кроку має бути чітким і послідовним, з логічними поясненнями дій. Використовувати заголовки для кожного етапу роботи, щоб полегшити орієнтацію в звіті.

2. **Скріншоти**.

Для кожного етапу роботи додавати чіткі скріншоти з відповідними підписами. Підписи повинні пояснювати дії, що виконуються, та налаштування, що застосовуються.

3. **Аналіз і пояснення**.

У звіті повинно бути пояснення вибору кожного інструменту та налаштувань. Порівняти результати аналізу різних форматів зображень, зокрема, роздільну здатність, розмір файлу і прозорість.

4. **Оформлення**.

Використовувати чітку нумерацію та форматування для кожного розділу. Підписати всі скріншоти, що додаються до звіту.

5. **Висновки**.

Висновки мають бути логічними та чіткими, з підсумком виконаної роботи і порівнянням результатів.

Контрольні питання:

1. Що таке растрові зображення і як вони відрізняються від векторних?
2. Які основні формати використовуються для растрових зображень і які їхні особливості?
3. Що таке роздільна здатність зображення і як вона впливає на якість зображення?
4. Які методи стиснення зображень використовуються для зменшення розміру файлу?

5. Що таке прозорість зображення і в яких випадках це важлива властивість?

6. Як прозорість реалізується в різних форматах зображень, таких як PNG і GIF?

7. Чим відрізняються формати PNG та JPEG за якістю та ефективністю стиснення?

8. Що таке альфа-канал і як він використовується при створенні зображень з прозорим фоном?

9. Які інструменти редагування зображень використовувалися в рамках практичної роботи?

10. Як конвертація зображення з одного формату в інший може вплинути на якість та розмір файлу?

11. Що таке бітова глибина кольору і як вона впливає на якість зображення?

12. Чому важливо вибирати правильний формат зображення для конкретної задачі?

13. Які основні відмінності між форматами SVG і AI і коли кожен з них доцільно використовувати?

Література: [1-7]

Практичне заняття № 2

«Стиснення та оптимізація зображень»

Мета: ознайомлення з принципами стиснення зображень, а також освоєння методів оптимізації розміру файлів без значної втрати якості.

Теоретичні відомості

2.1. Пояснення принципів стиснення зображень

Стиснення зображень є важливим етапом оптимізації для зменшення розміру файлів без значних втрат якості. В основі стиснення зображень лежать різні методи, що використовуються для зменшення обсягу даних, необхідних для зберігання та передачі зображень. Вони можуть бути як **без втрат**, так і **з втратами**. Розуміння принципів цих методів дає змогу ефективно використовувати зображення в різних сферах, таких як веб-дизайн, цифрове мистецтво, фотографія, а також для економії простору на пристроях з обмеженим місцем для зберігання даних.

2.1.1. Растрові зображення та їх представлення

Растрові зображення складаються з пікселів – найменших одиниць зображення, кожен з яких має свій колір. Кожен піксель може бути описаний набором числових значень (наприклад, червоний, зелений і синій кольори – RGB). Стиснення зображень дозволяє зменшити кількість бітів, необхідних для представлення цих пікселів, не знижуючи значною мірою якість зображення.

Зображення може бути представлене у вигляді матриці чисел або бітів, що відображають пікселі. При стисненні намагаються знайти закономірності або надлишкові дані, які можна видалити, не втрачаючи важливої інформації.

2.1.2. Стиснення без втрат (lossless)

Методи стиснення без втрат зберігають всі дані зображення, і відновлене зображення є точною копією оригіналу. Вони використовують різні алгоритми для кодування і зберігання даних зображення, зазвичай з урахуванням

повторення пікселів або груп пікселів. Основною метою таких методів є досягнення максимально можливого зменшення розміру без втрати інформації.

Прикладами методів без втрат є:

- RLE (Run-Length Encoding) – стиснення за допомогою представлення повторюваних елементів (пікселів або груп пікселів) лише один раз з інформацією про кількість повторень.
- LZW (Lempel-Ziv-Welch) – широко використовуваний алгоритм для стиснення даних у форматах GIF і PNG, який також ґрунтується на пошуку повторюваних послідовностей байтів або символів.

2.1.3. Стиснення з втратами (lossy)

Методи стиснення з втратами, як і слідує з назви, зменшують розмір файлу, але за рахунок втрати деякої інформації з зображення. Це означає, що після стиснення та відновлення зображення може бути невелика втрата якості. Проте, такий підхід дозволяє досягти значного зменшення розміру файлу без суттєвих змін для людського ока.

Основні принципи стиснення з втратами:

- вибіркова втрата інформації, зазвичай на тих ділянках, де зміни неможливо помітити або де вони мають мінімальний вплив на сприйняття зображення.
- використання математичних перетворень, таких як перетворення косинусів, для усунення малопомітних частотних компонентів зображення.

Примітки про методи стиснення з втратами:

- JPEG (Joint Photographic Experts Group) – найпоширеніший формат для зображень з втратами, де застосовуються такі методи, як Дискретне косинусне перетворення (DCT).
- У форматах JPEG часто застосовуються алгоритми, що видаляють деталі, які людське око не може відразу сприйняти.

2.1.4. Алгоритми стиснення: основні підходи

1. **Розпізнавання та кодування повторюваних елементів (для lossless стиснення):** передбачає пошук послідовностей пікселів або груп пікселів, що повторюються, і заміну їх на коротші послідовності (наприклад, кодування повторів).

2. **Перетворення та вибіркова втрата (для lossy стиснення):** підхід включає застосування математичних методів для перетворення зображення в таку форму, де можна видаляти деякі менш важливі частини інформації (часто на основі сприйняття людиною).

3. **Представлення та кодування частотних компонентів (особливо для lossy).** Багато методів стиснення використовують частотний аналіз, щоб визначити, які компоненти частоти зображення можуть бути втрачені без значного впливу на сприйняття якості. Це включає перетворення, такі як дискретне косинусне перетворення (DCT) для JPEG.

2.1.5. Вибір методу стиснення

При виборі методу стиснення важливо враховувати такі фактори, як:

Тип зображення: Фотографії, графіки, діаграми або логотипи вимагають різних підходів.

Якість та обсяг: Користувач може вибирати між високою якістю та меншим розміром файлу.

Цільове застосування: Для веб-сайтів зазвичай використовують стиснення з втратами, щоб зменшити розмір файлів для швидшого завантаження сторінок, тоді як для архівування чи зберігання на носіях часто використовують стиснення без втрат.

2.2. Принцип роботи алгоритму RLE (Run-Length Encoding)

Алгоритм **Run-Length Encoding (RLE)** є одним із найпростіших і ефективних методів стиснення даних, який використовується для зображень, де значна кількість пікселів мають однакові значення або «продовження» (від цього й назва – «run»). Принцип роботи цього алгоритму полягає в тому, що

замість того, щоб зберігати кожен окремий піксель, зберігається лише його значення і кількість його повторів (довжина «продовження»).

2.2.1. Основні принципи RLE

У найпростішому вигляді алгоритм RLE працює наступним чином:

1. **Пошук груп однакових елементів**: алгоритм визначає послідовності однакових значень, які називаються "runs".
2. **Кодування кожної групи**: для кожної групи з однакових елементів алгоритм записує два значення:
   - Колір або значення елементів (наприклад, піксель).
   - Кількість повторів цього елементу.

Це дозволяє значно зменшити обсяг даних, особливо для зображень або даних, де багато повторюваних елементів (наприклад, великі ділянки однотонного фону).

2.2.2. Приклад роботи алгоритму RLE

Для кращого розуміння принципу роботи RLE, розглянемо простий приклад стиснення за допомогою цього методу.

Приклад 1: Нехай ми маємо рядок пікселів (або символів) у зображенні:

AAAAABBBCCCCCCDDDDDD

Замість того, щоб зберігати кожен піксель окремо, алгоритм RLE стискає дані так:

5A3B6C6D

- 5A означає, що символ "A" повторюється 5 разів.
- 3B означає, що символ "B" повторюється 3 рази.
- 6C означає, що символ "C" повторюється 6 разів.
- 6D означає, що символ "D" повторюється 6 разів.

Така компресія значно зменшує кількість збережених даних, оскільки замість того, щоб зберігати 18 символів, ми маємо лише 8 (5 символів для "A", 3 для "B", 6 для "C" і 6 для "D").

Приклад 2: Розглянемо зображення, де пікселі мають тільки два кольори:

RRRRRRRRGGGGGGGG

За допомогою RLE це буде стиснено до:

8R8G

Тут "R" позначає червоний піксель, а "G" – зелений піксель, і алгоритм записує кількість повторів кожного кольору.

2.2.3. Стиснення та декомпресія

- Стиснення: Алгоритм бере оригінальну послідовність даних і перетворює її на пару значень (символ або піксель, кількість повторів).
- Декомпресія: Для відновлення оригінальних даних алгоритм просто повторює символ або піксель відповідно до зазначеної кількості.

Наприклад, декомпресія для рядка "5A3B6C6D" поверне оригінальну послідовність:

AAAAABBBCCCCCCDDDDDD

2.2.4. Переваги алгоритму RLE

1. **Простота реалізації**: Алгоритм RLE дуже простий у реалізації та має низькі вимоги до обчислювальних ресурсів.
2. **Ефективність при наявності великої кількості повторів**: Якщо зображення містить багато однотонних ділянок або повторюваних елементів, то стиснення за допомогою RLE дає значне зменшення обсягу.
3. **Швидкість**: Алгоритм швидко працює як при стисненні, так і при відновленні даних.

2.2.5. Недоліки RLE

1. **Низька ефективність при складних зображеннях**: Якщо зображення має багато різних кольорів або дуже деталізоване, RLE не дасть суттєвого зменшення обсягу, оскільки будуть відсутні довгі послідовності однакових елементів.

2. **Збільшення розміру при відсутності повторів**: Якщо зображення містить мало повторів (наприклад, зображення з випадковими або деталізованими кольорами), то RLE може навіть збільшити розмір файлу.

2.2.6. Алгоритм RLE для стиснення зображень

У випадку зображень метод RLE використовується для стиснення кольорів, які повторюються в ряді пікселів. Наприклад, для стиснення зображення можна застосувати RLE до кожного ряду пікселів окремо або для всієї матриці пікселів. В разі використання в графічних форматах, таких як GIF, цей метод дозволяє значно зменшити розмір зображень, що містять багато однотонних областей.

2.2.7. Застосування RLE

Графічні формати: Формати, такі як GIF, часто використовують RLE для стиснення зображень, оскільки вони зазвичай мають великі однотонні ділянки.

Медичні зображення: Алгоритм RLE іноді використовується для стиснення медичних зображень, таких як зображення МРТ чи рентгенівські знімки, якщо зображення містить багато однотипних ділянок.

Факсимільні зображення: Стиснення факсимільних зображень для відправки факсами також використовує RLE для досягнення зменшення розміру файлів.

2.3 Принцип роботи алгоритму LZW (Lempel-Ziv-Welch)

LZW (Lempel-Ziv-Welch) – алгоритм стиснення без втрат, розроблений Террі Велчем у 1984 році як удосконалення алгоритмів LZ77 і LZ78. Він дозволяє ефективно зменшити обсяг даних, замінюючи повторювані послідовності символів на коротші коди. Алгоритм широко застосовується для стиснення текстових даних і зображень у форматах **GIF**, **TIFF**, а також у багатьох системах архівування.

2.3.1. Принцип роботи

Алгоритм працює на основі динамічного словника (таблиці), що поступово формується під час обробки даних. Основна ідея полягає в тому,

щоб замінити повторювані послідовності символів на унікальні коди зі словника.

Кроки роботи алгоритму LZW:

1. **Ініціалізація словника:**

На початку створюється словник, який містить усі можливі одиничні символи (наприклад, усі байти від 0 до 255 для зображень).

2. **Читання вхідних даних:**

Алгоритм читає послідовність символів і поступово формує більш довгі фрагменти.

3. **Додавання нових послідовностей:**

Якщо поточна послідовність символів відсутня у словнику, вона додається зі своїм унікальним кодом.

4. **Вихідні коди:**

Замість передачі початкових символів, алгоритм передає лише відповідні коди з таблиці.

2.3.2. Приклад роботи LZW

Вхідні дані: **ABABABA**

1. Ініціалізація словника:

A → 65

B → 66

2. Читання першого символу: **A → код 65**

3. Читання наступного символу: **B → код 66**

4. Читання послідовності **AB (не в таблиці) → додати до таблиці з новим кодом 256**

5. Читання наступного символу **A**, потім **BA (не в таблиці) → додати до таблиці з кодом 257**

Вихідні коди: **65, 66, 256, 257**

2.3.3. Математичне пояснення

Нехай $S = s_1, s_2, s_3, \dots, s_n$ – послідовність символів.

Початковий словник:

$$D_0 = \{s_1, s_2, s_3, \dots, s_m\}$$

де m – кількість унікальних символів.

1. На кожному кроці алгоритм шукає найдовшу послідовність w у словнику D.

2. Якщо послідовність w знайдена, вона замінюється кодом з таблиці, а нова послідовність додається:

$$D_{i+1} = D_i \cup \{w + s_{next}\}$$

2.3.4. Переваги та недоліки

Переваги:

- ефективне стиснення для текстів та зображень з великою кількістю повторів.
- простота реалізації.
- відсутність втрат якості.

Недоліки:

- не завжди забезпечує оптимальне стиснення для випадкових даних.
- підходить лише для структурованих даних.

2.3.5. Застосування

- Формат зображень **GIF** для анімацій.
- Формат **TIFF** у професійній обробці графіки.
- Системи архівування файлів, такі як UNIX-команди compress.

2.4 Дискретне косинусне перетворення (DCT)

Дискретне косинусне перетворення (DCT – Discrete Cosine Transform) є основним математичним методом у стисненні зображень формату JPEG, а також відеоформатів, таких як MPEG. DCT дозволяє перетворити сигнал із просторової області (де значення відповідають яскравості пікселів) у частотну область, де інформація розподіляється за амплітудами частотних компонент.

2.4.1. Математичне визначення DCT

Для одновимірного масиву довжиною N дискретне косинусне перетворення визначається як:

$$X_k = \alpha(k) \sum_{n=0}^{N-1} \quad x_n \cdot cos\left[\frac{\pi}{N} \cdot (n + \frac{1}{2})k\right]$$

де:

X_k – значення спектра для частоти k.

x_n – значення вхідного сигналу на позиції n.

$\alpha(k)$ – нормувальний коефіцієнт:

$$\alpha(k) = \{\sqrt{\frac{1}{N}} \text{ ,якщо } k = 0 \quad \sqrt{\frac{2}{N}} \text{ ,якщо } k > 0$$

2.4.2 Двовимірне дискретне косинусне перетворення

Для зображень використовують двовимірну версію DCT:

$$X_{u,v} = \alpha(u) \cdot \alpha(v) \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} \quad f(x,y) \quad \cdot cos\left[\frac{\pi}{N} \cdot (x + \frac{1}{2})n\right] \cdot cos\left[\frac{\pi}{N} \cdot (y + \frac{1}{2})v\right]$$

де:

- $X_{u,v}$ – коефіцієнти DCT для частот u та v.
- $f(x,y)$ – яскравість пікселя на координатах (x,y).
- $\alpha(u)\ i\ \alpha(v)$ – нормувальні коефіцієнти.

2.4.3 Принцип роботи DCT у стисненні зображень

1. Розбиття на блоки:

Зображення розбивається на блоки розміром 8x8 пікселів.

2. Перетворення значень яскравості:

Яскравість кожного пікселя віднімається від середнього значення (зазвичай 128), щоб центр спектра знаходився на нульовій частоті.

3. Застосування DCT:

До кожного блоку застосовується дискретне косинусне перетворення, що переводить просторові значення у частотну область.

4. Квантування:

Коефіцієнти DCT округлюються з метою зменшення точності високочастотних компонентів, які менш важливі для людського зору.

5. Кодування:

Після квантування використовуються методи ентропійного кодування (наприклад, алгоритм Хаффмана) для подальшого стиснення.

2.4.4 Переваги та недоліки DCT

Переваги:

- висока ефективність стиснення завдяки квантуванню частотних компонентів.
- здатність зберігати основну інформацію про зображення навіть після втрати деяких деталей.

Недоліки:

- втрата якості при сильному стисненні (поява артефактів блоків).
- важкість обробки для реального часу у великих зображеннях.

2.5. Вибір оптимального методу стиснення для конкретного випадку

2.5.1 Основні критерії вибору методу стиснення

Для вибору методу стиснення (lossless чи lossy) слід враховувати такі фактори:

Цільове використання зображення: друк, веб-публікація, архівування.

Вимоги до якості: допустимість втрати якості або її збереження на максимальному рівні.

Розмір файлу: важливість мінімізації розміру файлу для збереження чи передачі.

Чутливість до деталей: рівень важливості збереження тонких деталей зображення.

2.5.2 Випадки застосування lossless стиснення

Lossless стиснення зберігає всі дані зображення без втрати якості, але не завжди забезпечує значне зменшення розміру файлу.

Коли використовувати:

- **архівування важливих зображень** (збереження фотографій або графіки для подальшої обробки (файли у форматах tiff, png)).
- **наукові або медичні зображення** (рентгенівські знімки, супутникові знімки, де важлива точність).
- **графіка з чіткими межами та текстом** (логотипи, іконки, діаграми (png, gif)).
- **веб-графіка з прозорим фоном (**коли важлива підтримка прозорості (png)).

Приклади форматів:

PNG – підтримка прозорості, збереження якості, підходить для веб-графіки.

GIF – обмежена палітра кольорів, добре підходить для анімацій.

BMP (без стиснення) – зручний для внутрішніх обчислень, але громіздкий.

2.5.3 Випадки застосування lossy стиснення

Lossy стиснення дозволяє значно зменшити розмір файлу за рахунок втрати деяких деталей зображення.

Коли використовувати:

- фотографії для веб-ресурсів (коли потрібно мінімізувати розмір файлу для швидкого завантаження (JPEG, WebP)).
- зображення для соціальних мереж (де допустима незначна втрата якості).
- великі архіви зображень (для економії місця на диску (JPEG, HEIF)).
- відео кадри (використання DCT у відеокодеках (MPEG, H.264)).

Приклади форматів:

JPEG – оптимальний для фотографій із плавними переходами кольору.

WebP – сучасний формат для веб-графіки із кращим стисненням, ніж JPEG.

HEIF – новітній формат з високою ефективністю стиснення, використовується на iOS.

Таблиця 2.1 Порівняння прикладів

Критерій	**Lossless (PNG)**	**Lossy (JPEG)**
Якість	Без втрати	Часткова втрата
Розмір файлу	Великий	Маленький
Прозорість	Підтримується	Не підтримується
Швидкість обробки	Повільніше	Швидше
Використання	Логотипи, діаграми, архіви	Фотографії, веб-зображення

Таблиця 2.2 Рекомендації щодо вибору

Ситуація	**Рекомендований формат**	**Метод стиснення**
Логотипи та іконки	PNG, SVG	Lossless
Фотографії для друку	TIFF, JPEG	Lossless/Lossy
Веб-сайти	JPEG, WebP	Lossy
Архівування важливих даних	PNG, TIFF	Lossless
Анімації	GIF, APNG	Lossless

Вибір оптимального методу залежить від пріоритетів користувача – якщо важлива економія простору, перевага надається lossy-стисненню, якщо якість є критично важливою – використовуються lossless-алгоритми.

2.6 ImageMagick (CLI)

ImageMagick (CLI) – це набір інструментів для роботи з зображеннями, який працює через командний рядок (CLI – Command Line Interface). ImageMagick дозволяє конвертувати, редагувати, змінювати розмір, стискати та виконувати багато інших операцій з графічними файлами без використання графічного інтерфейсу.

Основні можливості ImageMagick (CLI):

- Конвертація між різними форматами зображень.
- Зміна розмірів та обтинання (crop).
- Застосування фільтрів та ефектів.
- Оптимізація та стиснення файлів.
- Робота з шарами та анімаціями (наприклад, GIF).
- Масова обробка зображень через скрипти.

2.6.1 Як встановити ImageMagick

URL: https://imagemagick.org/script/download.php

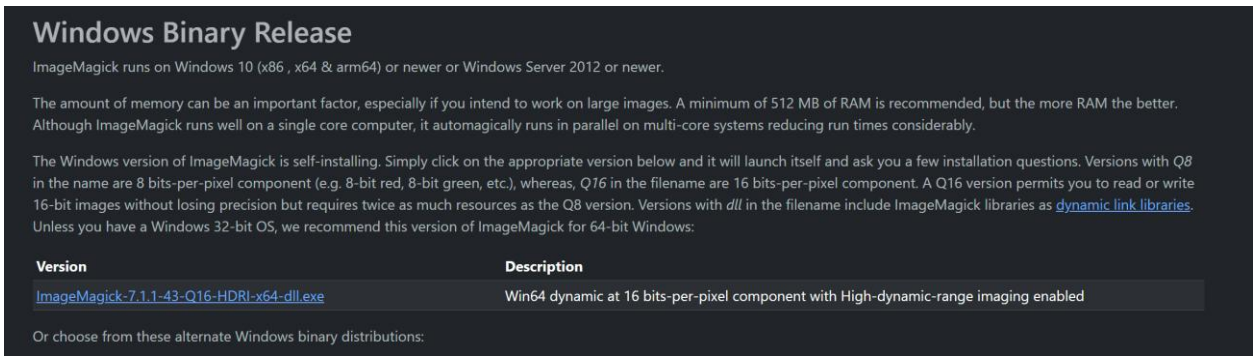


Рисунок 2.1 – Сторінка сайту ImageMagick для завантаження

Windows

1. Завантажити з офіційного сайту: https://imagemagick.org
2. Під час встановлення вибрати опцію "Add application directory to your system path".
3. Відкрити **Command Prompt (cmd)** та перевірити встановлення командою:

magick -version

Linux (Ubuntu/Debian)

sudo apt install imagemagick

macOS

brew install imagemagick

2.6.2 Робота з консоллю та приклад використання ImageMagick

Загальні поради:

- У **Windows** командний рядок відкривається через **Win + R → cmd → Enter**.
- У **Linux/macOS** потрібно відкрити **Термінал**.
- Якщо команда identify не працює, перевірте, чи встановлений **ImageMagick** (у Windows може знадобитися перевстановлення з опцією "Add to system path").

Для виконання лабораторної роботи потрібно перевіряти характеристики зображень за допомогою команди magick identify -verbose. Нижче описано, як це зробити правильно.

Щоб команда magick identify -verbose image.jpg працювала, необхідно:

Зображення має бути у тій самій папці, де виконується команда.

Наприклад, якщо зображення image.jpg збережене у папці C:\Images, потрібно відкрити командний рядок (**cmd** у Windows) або термінал (**Linux/macOS**) та перейти в цю папку:

Windows:

cd C:Images

identify -verbose image. jpg

Linux або macOS:

cd ~/Images

identify -verbose image.jpg

Якщо зображення знаходиться в іншій папці, необхідно вказати повний шлях.

Наприклад, якщо файл зображення знаходиться на робочому столі:

magick identify -verbose "C:\Users\Ім'я_користувача\Desktop\image.jpg"

Якщо потрібно перевірити всі зображення в папці, можна використати символ *.

magick identify -verbose *.jpg

Після виконання команд можна переглянути детальну інформацію про зображення: роздільну здатність, колірну глибину, формат стиснення тощо.

Стиснення за допомогою ImageMagick

Виконання стиснення з різними рівнями якості **JPEG**:

- magick image.png -quality 100 image_100.jpg
- magick image.png -quality 80 image_80.jpg

Роз’яснення:

Команда magick image.png -quality 100 image_100.jpg конвертує файл image.png в JPEG формат з максимальною якістю (100), при цьому вихідний файл буде називатися image_100.jpg.

Якщо формат файлу .jpg та його не треба змінювати, також необхідне стиснення 60, то команда буде виглядати так: magick image.jpg -quality 60 image_60.jpg

Виконання стиснення з різними рівнями якості **PNG**:

magick input_image.jpg -quality 9 -compress zip output_9.png

Роз’яснення:

Використовується команда для створення PNG з максимальним стисненням (найнижчою якістю стиснення) зі значенням compression level = 9.

- -quality: Цей параметр для PNG може використовуватися для контролю рівня стиснення (від 1 до 9). Чим вище значення, тим сильніше стиснення.
- -compress zip: вказує на використання алгоритму стиснення Zip для PNG.
- -compress lzw: використовує алгоритм **LZW** (Lempel-Ziv-Welch). Цей метод стиснення зберігає якість зображення без втрат і є ще одним варіантом для PNG.
- -compress deflate: це інший варіант для PNG, який використовує алгоритм **Deflate**, який є основою для ZIP.

Вимірювання розмірів файлів:

Windows: dir "C:\Users\Ім'я_користувача\Desktop\image_*.jpg"

Linux або macOS: ls -lh image_*.jpg

Завдання

Завдання 1. Початкове дослідження характеристик зображень.

1. Завантажити або створити три зображення у форматах **JPEG, PNG та GIF**.

2. Використовуючи **ImageMagick**, визначити основні характеристики кожного зображення:

- Розмір файлу (Filesize)
- Роздільну здатність (ширина × висота) (Geometry)
- Колірну глибину (Depth, Channels)
- Чи підтримує зображення прозорість (Alpha/ Transparency/ Matte)

Команди для виконання:

Щоб отримати інформацію про зображення, використати команду:

magick identify -verbose image.jpg (Замість image.jpg підставити реальне ім'я файлу.)

3. Оформити результати у вигляді таблиці.

Таблиця 1 – Основні характеристики зображень

Формат	**Розмір (KB)**	**Роздільна здатність**	**Колірна глибина**	**Прозорість (так/ні)**
JPEG	??? KB	??? × ??? px	??? біт	Ні
PNG	??? KB	??? × ??? px	??? біт	Так/Ні
GIF	??? KB	??? × ??? px	??? біт	Так/Ні

Завдання 2. Стиснення зображень та їх вплив на якість.

1. Взяти зображення у форматі PNG або JPEG (за варіантом).

2. Виконати його стиснення за допомогою ImageMagick із різними рівнями якості. (Табл. 2)

3. Порівняти результати та зробити висновки щодо впливу ступеня стиснення на якість.

4. Візуально оцінити отримані зображення

5. Виміряти розмір кожного отриманого файлу.

6. Порівняти якість зображень із різними рівнями стиснення.

7. Зробити висновки щодо вибору оптимального формату та рівня стиснення.

Таблиця 2.3 – Індивідуальні варіанти

Варіант	**Формат вихідного зображення**	**Формат після стиснення**	**Рівні якості (JPEG) / Стиснення (PNG)**
1	PNG	PNG	9, 5, 1
2	PNG	JPEG	100%, 80%, 60%, 40%
3	JPEG (90%)	JPEG	100%, 80%, 60%, 40%
4	PNG	PNG	9, 5, 1
5	PNG	JPEG	100%, 90%, 80%, 50%
6	JPEG (80%)	JPEG	90%, 70%, 50%, 30%
7	PNG	PNG	9, 7, 3
8	PNG	JPEG	100%, 85%, 65%, 45%
9	JPEG (70%)	JPEG	95%, 75%, 55%, 35%
10	PNG	PNG	9, 6, 2
11	PNG	JPEG	100%, 80%, 60%, 40%
12	JPEG (50%)	JPEG	90%, 70%, 50%, 30%

Завдання 3. Алгоритми стиснення.

1. **Виберіть зображення PNG** для стиснення.

2. **Використайте команду** magick для стиснення зображень різними алгоритмами (-compress zip, -compress deflate, -compress lzw.)

3. **Порівняйте розмір файлу для кожного варіанту стиснення.**

4. **Проаналізуйте якість зображення після стиснення (можна використовувати засоби порівняння зображень або просто візуально).**

5. **Зробіть висновки** щодо того, який метод стиснення дає найкращий баланс між якістю зображення та його розміром.

Загальні рекомендації до виконання звіту

1. **Чіткість і структурованість**.

Опис кожного кроку має бути чітким і послідовним, з логічними поясненнями дій. Використовувати заголовки для кожного етапу роботи, щоб полегшити орієнтацію в звіті.

2. **Скріншоти**.

Для кожного етапу роботи додавати чіткі скріншоти з відповідними підписами. Підписи повинні пояснювати дії, що виконуються, та налаштування, що застосовуються.

3. **Аналіз і пояснення**.

У звіті повинно бути пояснення вибору кожного інструменту та налаштувань. Порівняти результати аналізу різних форматів зображень, зокрема, роздільну здатність, розмір файлу і прозорість.

4. **Оформлення**.

Використовувати чітку нумерацію та форматування для кожного розділу. Підписати всі скріншоти, що додаються до звіту.

5. **Висновки**.

Висновки мають бути логічними та чіткими, з підсумком виконаної роботи і порівнянням результатів.

Контрольні питання:

1. Що таке стиснення зображень?
2. Які основні формати зображень використовуються для стиснення?
3. Які фактори впливають на якість стиснення зображень?
4. Що таке алгоритм стиснення ZIP, і як він працює?
5. Які переваги та недоліки форматів PNG та JPEG для зберігання зображень?
6. Як параметр -quality впливає на результат стиснення зображення у форматі JPEG?

Література: [1-7]

Практичне заняття № 3

«Основи роботи з аудіо»

Мета: ознайомитися з основними параметрами аудіофайлів (бітрейт, частота дискретизації, бітова глибина), навчитися конвертувати аудіофайли між форматами (MP3, WAV), дослідити, як зміна параметрів впливає на розмір та якість аудіофайлу.

Теоретичні відомості

Цифровий звук – це представлення аналогового звукового сигналу у вигляді цифрових значень, які зберігаються, передаються та обробляються за допомогою комп'ютерних систем. Основними параметрами цифрового аудіо є **частота дискретизації, бітова глибина та бітрейт**. Вони визначають якість та розмір аудіофайлу.

Одним із важливих аспектів цифрового аудіо є **формат файлу**. Він визначає спосіб зберігання та стиснення аудіоданих. Усі формати поділяються на 3 категорії, шо описані нижче.

Несжаті формати (наприклад, WAV) – містять аудіодані без змін, займають багато місця.

Стиснені формати без втрат (наприклад, FLAC) – зменшують розмір без втрати якості.

Стиснені формати з втратами (наприклад, MP3) – видаляють частину інформації, щоб суттєво зменшити розмір файлу.

3.1 Основні аудіоформати

3.1.1 MP3 (MPEG-1 Audio Layer 3)

MP3 – це формат стиснення аудіо з втратами, який значно зменшує розмір файлу, використовуючи **психоакустичне моделювання**.

Алгоритм стиснення MP3.

1. **Частотний аналіз**

Вхідний сигнал розбивається на **кадри** (фрагменти по 1152 семпли) і аналізується **перетворенням Фур'є** (FFT).

Застосовується **моделювання слухового сприйняття**, щоб виявити звуки, які людське вухо не чує.

2. **Психоакустичне кодування**

Видаляються непомітні звуки (**ефект маскування** – коли гучні звуки заглушають тихіші).

Зменшується точність представлення високочастотних компонентів (людина менше сприймає їхні деталі).

3. **Квантування та кодування**

Використовується **адаптивне квантування** – більш точне представлення важливих частот і менше точності для незначних частот.

Використовується **Huffman-кодування** для ефективного збереження даних.

4. **Формування вихідного файлу**

Дані упаковуються у структуру MP3, яка містить **заголовок**, **мета-дані** та **самі аудіодані**.

Формула бітрейту MP3

Розмір файлу MP3 залежить від бітрейту за формулою:

$$\text{Розмір файлу} = \frac{\text{Бітрейт} \times \text{Тривалість}}{8}$$

де:

Бітрейт – швидкість передачі даних у кбіт/с (наприклад, 128 кбіт/с).

Тривалість – довжина аудіо у секундах.

Приклад розрахунку

Припустимо, у нас є MP3-файл довжиною 3 хвилини (180 секунд) із бітрейтом 128 кбіт/с:

$$\frac{128 \times 180}{8} = 2880 \text{ КБ} \approx 2.8 \text{ МБ}$$

Переваги MP3:

- сильно зменшує розмір файлу;
- підтримується більшістю пристроїв;
- можливість регулювати якість через бітрейт.

Недоліки MP3:

- втрата якості через видалення частини звуку;
- менша точність у відтворенні високих частот;
- при багаторазовому перекодуванні якість значно погіршується.

1.2 WAV (Waveform Audio File Format)

WAV – це формат аудіофайлів, що містить **некомпресовані** (або стиснені без втрат) звукові дані. Використовується для **збереження звуку в оригінальній якості**.

Формат WAV базується на **RIFF (Resource Interchange File Format)**. Структура містить:

- **RIFF-заголовок** (ідентифікує, що файл є WAV).
- **fmt (format) секція** (містить інформацію про частоту дискретизації, бітову глибину, кількість каналів).
- **data секція** (містить необроблені аудіодані).

WAV-файл зберігає **амплітуду звуку** у вигляді цифрових значень.

Якщо f_s – частота дискретизації (Hz), а n – бітова глибина (bit), то **максимальна кількість можливих значень**: 2^n

Наприклад, при 16-бітному аудіо маємо 65 536 рівнів амплітуди (**від -32 768 до +32 767**).

Розмір файлу WAV визначається за формулою:

$$\text{Розмір} = \text{Частота дискретизації} \times \text{Кількість біт} \times \text{Кількість каналів} \times \text{Тривалість}/8$$

Наприклад, є стереофайл (2 канали), 44 100 Гц, 16 біт, тривалість – 1 хвилина:

$$44100 \times 16 \times 2 \times 60/8 = 10.6 \text{ МБ}$$

Переваги WAV:

- висока якість звуку (відсутність стиснення);
- використовується у професійному записі.

Недоліки WAV:

- великий розмір файлів;
- не підтримує вбудоване стиснення.

3.1.3 FLAC (Free Lossless Audio Codec)

FLAC – це формат **стиснення аудіо без втрат**, який **зменшує розмір файлу** в середньому на **40-60%** без втрати якості.

Алгоритм стиснення FLAC

1. **Частотний аналіз.** Аналізуються звукові дані та усуваються **повторювані** (надлишкові) фрагменти.
   - **Прогнозування.** Використовуються **математичні моделі**, щоб передбачити наступні значення амплітуди. Наприклад, якщо попередні значення **10, 11, 12**, можна передбачити **13**.
   - **Кодування залишкових помилок.** Кодується **відхилення** від прогнозованих значень (залишкова інформація).
   - **Стиснення та упаковка.** Використовується **Huffman-кодування** для ефективного зберігання.

Формула розміру FLAC-файлу:

Оскільки FLAC стискає WAV на **40-60%**, то розмір можна оцінити наступним чином.

$$\text{Розмір } FLAC = \text{Розмір } WAV \times (0.4 - 0.6)$$

Наприклад, WAV-файл 50 МБ після стиснення FLAC стане **20-30 МБ**.

Переваги FLAC:

- зберігає оригінальну якість звуку;
- менший розмір, ніж WAV;
- підтримує **теги** (інформацію про виконавця, альбом).

Недоліки FLAC:

- не всі пристрої підтримують FLAC;
- більший розмір у порівнянні з MP3.

3.2 Ключові параметри аудіофайлів

При цифровому збереженні звуку якість аудіофайлу залежить від **трьох основних параметрів**:

- **Частота дискретизації (Sample Rate)**
- **Бітова глибина (Bit Depth)**
- **Бітрейт (Bitrate)**

3.2.1 Частота дискретизації (Sample Rate)

Частота дискретизації – це **кількість вимірювань звукової хвилі за секунду**. Вимірюється в герцах (**Hz**).

Чим вища частота дискретизації, тим точніше цифровий сигнал відтворює аналоговий звук.

Таблиця 3.1 – Основні значення частоти дискретизації

Частота (Hz)	**Де використовується**
8 000 Hz	Телефонні дзвінки
22 050 Hz	Радіотрансляції, низькоякісне аудіо
44 100 Hz	Стандарт для CD-дисків
48 000 Hz	Відео та професійний звук
96 000 Hz	Студійний звук, Hi-Res Audio
192 000 Hz	Аудіо найвищої точності (використовується рідко)

Теорема Найквіста-Шеннона

Для коректного відтворення аналогового сигналу необхідно, щоб частота дискретизації була **вдвічі більшою**, ніж максимальна частота звуку в записі.

$$f_s \geq 2f_{max}$$

Де: f_s - частота дискретизації, f_{max} - максимальна частота аудіосигналу.

Оскільки **людське вухо чує звуки до 20 000 Hz**, то **CD-стандарт 44 100 Hz обраний якраз відповідно до теореми Найквіста**.

Приклад дискретизації

Якщо частота дискретизації **низька**, то отримані дані будуть приблизними, що призводить до втрати якості звуку.

Приклад:

Якщо записати звук із частотою дискретизації **8 000 Hz**, високі частоти обріжуться – голос звучатиме спотворено.

Для якісного музичного запису використовується **44 100 Hz або вище**.

Недоліки високої частоти дискретизації:

- Збільшення розміру файлу
- Потрібно більше ресурсів для обробки

3.2.2 Бітова глибина (bit depth)

Бітова глибина (bit depth) – це кількість бітів, що використовується для збереження кожного семплу звуку. Визначає точність відображення амплітуди аудіосигналу.

Чим більше бітів, тим ширший динамічний діапазон і тим точніше оригінальний аналоговий сигнал передається в цифровому вигляді.

Таблиця 2 – Основні значення бітової глибини

Бітова глибина	**Кількість рівнів гучності**	**Динамічний діапазон (dB)**	**Де використовується**
8 біт	256 рівнів	48 dB	Старі комп’ютерні аудіофайли, телефонія
16 біт	65 536 рівнів	96 dB	CD-аудіо

24 біт	16 777 216 рівнів	144 dB	Студійний звук
32 біт (float)	∞ (плаваюча точка)	~168 dB	Професійний звук

1) **8 біт** – звук із помітним шумом
2) **16 біт** – стандартний CD-звук
3) **24 біт** – професійне аудіо
4) **32 біт float** – не обмежений жорсткими значеннями

Бітова глибина визначає **роздільну здатність амплітуди** звукового сигналу.

Формула для обчислення кількості рівнів гучності:

$$L = 2^n L$$

Де:

- L – кількість можливих рівнів гучності
- n – бітова глибина

Приклад:

- **8-бітне аудіо** має $2^8 = 256$ рівнів амплітуди.
- **16-бітне аудіо** має $2^{16} = 65536$ рівнів, що забезпечує плавніший і точніший звук.

Динамічний діапазон і шум квантування

Динамічний діапазон – це різниця між **найтихішим** і **найгучнішим** звуком, який можна закодувати.

Формула для обчислення динамічного діапазону:

$$D = 6.02 \cdot n + 1.76\ (dB)$$

Де:

- D – динамічний діапазон (в децибелах);
- n – бітова глибина.

Приклад:

Для **16 біт**: $6.02 \times 16 + 1.76 = 96.32$

Для **24 біт**: $6.02 \times 24 + 1.76 = 144.24$

3.2.3 Бітрейт

Бітрейт (bitrate) – це **кількість бітів, що використовується для збереження аудіо за одиницю часу**.

Вимірюється в **кілобітах за секунду (kbps)** або **бітів за секунду (bps)**.

Види бітрейту

CBR (Constant Bitrate) – Постійний бітрейт

- Кожна секунда аудіо має однаковий бітрейт.
- Забезпечує передбачуваний розмір файлу, але менш ефективний.
- Використовується для потокового аудіо та радіо.

VBR (Variable Bitrate) – Змінний бітрейт

- Бітрейт змінюється залежно від складності звуку.
- Висока якість при меншому розмірі файлу.
- Використовується в музичних файлах (MP3 VBR, AAC VBR).

ABR (Average Bitrate) – Середній бітрейт

- Компроміс між CBR та VBR: середнє значення фіксоване, але можливі коливання.
- Застосовується у стрімінгових сервісах.

Формула розрахунку розміру аудіофайлу:

$$\text{Розмір } (MB) = \frac{\text{бітрейт } (kbps) \times \text{тривалість (сек)}}{8 \times 1024}$$

Приклад:

Аудіо довжиною **3 хвилини (180 секунд)** у форматі **MP3 320 kbps**:

$$\frac{320 \times 180}{8 \times 1024} \approx 7\ MB$$

Якщо зменшити бітрейт до **128 kbps**, розмір складе лише **2.8 MB**.

Таблиця 2 - Стандартні значення бітрейту

Бітрейт (kbps)	**Якість**	**Де використовується**
32 kbps	Дуже низька	Мовлення, телефонія
96 kbps	Низька	Подкасти, радіо

128 kbps	Середня	Інтернет-радіо, стандарт MP3
192 kbps	Хороша	Базова музична якість
256 kbps	Висока	Apple Music, Spotify Premium
320 kbps	Дуже висока	MP3 максимальної якості
~1000 kbps	Lossless (FLAC, ALAC)	Аудіофіли, студії
~4608 kbps	Студійна якість	24-біт WAV, FLAC

Вплив бітрейту на якість

Високий бітрейт (320 kbps, FLAC, WAV):

- максимальна якість;
- великий розмір файлу;
- використовується для студійної роботи.

Низький бітрейт (128 kbps, 64 kbps):

- стиснення призводить до втрат деталізації;
- помітні артефакти компресії (металевий звук, втрата високих частот);
- використовується для телефонії, аудіокниг.

Приклад: спробуйте слухати одну й ту саму пісню у **128 kbps** та **320 kbps** – у нижчій якості буде гірше чути деталі, інструменти звучатимуть менш чітко.

3.3 Конвертація аудіоформатів

Конвертація аудіо – це процес зміни формату файлу без зміни змісту звукової інформації.

Причини конвертації:

- зменшення розміру файлу (наприклад, WAV → MP3).
- збереження високої якості (MP3 → FLAC немає сенсу, але WAV → FLAC зручно для архівування).
- сумісність з програмами або пристроями.

Існують 3 основні способи конвертації:

- використання **онлайн-сервісів** (Online Audio Converter);

- **програми з графічним інтерфейсом** (Audacity, VLC);
- **командний рядок (FFmpeg)** – потужний спосіб для автоматизації.

3.3.1 Конвертація за допомогою FFmpeg

FFmpeg – це консольна утиліта для роботи з мультимедіа. Вона дозволяє конвертувати аудіофайли за допомогою простих команд.

3.3.1.1 Завантаження FFmpeg

Офіційний сайт: https://ffmpeg.org/download.html

Для різних операційних систем потрібно виконати різні кроки.

3.1.1.1 Встановлення на Windows

1) Завантаження архіву

Перейдіть на сайт https://ffmpeg.org/download.html.

Виберіть розділ **Windows** та натисніть **Windows builds by gyan.dev** (або використовуйте альтернативні збірки).

Завантажте **Full Build** (ZIP-архів).

2) Розпакування та налаштування змінних середовища

Розпакуйте архів у зручне місце, наприклад, `C:\ffmpeg`.

У папці `C:\ffmpeg\bin` знаходиться файл `ffmpeg.exe` – саме його будемо використовувати.

3) Додавання FFmpeg у змінну середовища PATH

Щоб можна було використовувати FFmpeg у будь-якому місці командного рядка:

Натисніть **Win + R**, введіть sysdm.cpl і натисніть **Enter**.

Перейдіть у вкладку **Додатково → Змінні середовища**.

У розділі **Системні змінні** знайдіть **Path** і натисніть **Змінити**.

Натисніть **Новий** і додайте шлях: `C:\ffmpeg\bin`.

Натисніть **ОК → ОК → Закрити**.

Можна працювати без додавання в **Path. Для цього постійно вказуйте повний шлях до** ffmpeg.exe (наприклад, "D:\ffmpeg\bin\ffmpeg.exe")

4) Перевірка встановлення

Відкрийте **Командний рядок (Win + R → cmd)** і введіть:

```
ffmpeg -version
```

3.1.1.2 Встановлення на macOS

Найпростіший спосіб встановлення FFmpeg на macOS – використання Homebrew.

Якщо у вас ще немає Homebrew, встановіть його за допомогою команди в **Терміналі**:

```
/bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"
```

Після встановлення Homebrew введіть команду:

```
brew install ffmpeg
```

Перевірка встановлення:

```
ffmpeg -version
```

3.3.1.3 Конвертація

Документація: https://ffmpeg.org/ffmpeg.html

Конвертація MP3 у WAV (без втрати якості):

```
ffmpeg -i input.mp3 output.wav
```

input.mp3 – вхідний файл

output.wav – вихідний файл

Приклад: файл `music.mp3` конвертується у `music.wav`

```
ffmpeg -i music.mp3 music.wav
```

Конвертація WAV у MP3 (з компресією, бітрейт 192 kbps):

```
ffmpeg -i input.wav -b:a 192k output.mp3
```

-b:a 192k – встановлює бітрейт у 192 kbps

Приклад: конвертуємо `record.wav` у `record.mp3` з бітрейтом 192 kbps

ffmpeg -i record.wav -b:a 192k record.mp3

Конвертація WAV у FLAC (без втрати якості):

ffmpeg -i input.wav output.flac

FLAC стискає без втрати якості (~50% економії місця).

Приклад: song.wav → song.flac:

ffmpeg -i song.wav song.flac

Налаштування параметрів аудіо при конвертації.

Зміна частоти дискретизації (48 кГц → 44.1 кГц):

ffmpeg -i input.wav -ar 44100 output.wav

-ar 44100 – встановлює частоту дискретизації у 44.1 кГц

Зміна бітової глибини (16-біт → 24-біт):

ffmpeg -i input.wav -sample_fmt s24 output.wav

-sample_fmt s24 – змінює бітову глибину на 24 біти

Зміна бітрейту MP3 (128 kbps):

ffmpeg -i input.wav -b:a 128k output.mp3

-b:a 128k – встановлює бітрейт 128 kbps

3.3.2 Графічні програми для конвертації

Audacity – це безкоштовний аудіоредактор з відкритим кодом, який дозволяє записувати, редагувати та експортувати аудіофайли в різні формати, зокрема MP3, WAV, FLAC, OGG.

Можливості Audacity:

- запис аудіо з мікрофона або системного звуку.
- редагування аудіофайлів (обрізка, нормалізація, шумозаглушення).
- конвертація між форматами (MP3, WAV, FLAC, OGG).

- зміна бітрейту, частоти дискретизації та глибини звуку.

Як конвертувати аудіофайл у Audacity:

1) Відкрити файл у програмі (**File → Open**).

2) При необхідності відредагувати звук.

3) Обрати формат експорту (**File → Export →** вибрати MP3, WAV або FLAC).

4) Налаштувати параметри файлу (бітрейт, якість).

5) Натиснути **"Save"** – файл буде збережено у вибраному форматі.

При експорті у MP3 можна вибрати якість:

- 128 kbps – середня якість
- 192 kbps – хороша якість
- 320 kbps – максимальна якість

VLC Media Player - це популярний медіаплеєр, який підтримує не лише відтворення, а й конвертацію аудіофайлів.

Можливості VLC:

- відтворення аудіо та відео будь-якого формату.
- конвертація аудіофайлів (MP3, WAV, FLAC, OGG).
- виділення аудіо з відеофайлів.
- пакетна обробка кількох файлів.

Як конвертувати аудіофайл у VLC:

1) Відкрити VLC → **Media → Convert/Save**.

2) Натиснути **"Add"** та обрати аудіофайл.

3) Натиснути **"Convert/Save"** та вибрати формат (MP3, WAV, FLAC).

4) Вказати шлях для збереження та натиснути **"Start"**.

Приклад конвертації WAV у MP3:

1. **Media → Convert/Save.**
2. **Add →** вибрати WAV-файл.

3. **Convert → Profile →** вибрати MP3.

4. **Start** – файл буде збережено у MP3.

Online Audio Converter – це веб-сервіс, який дозволяє швидко змінювати формат аудіофайлів без встановлення програм.

Можливості сервісу:

- підтримка форматів MP3, WAV, FLAC, M4A, OGG.
- налаштування якості звуку (бітрейт, частота дискретизації).
- видалення шуму та нормалізація гучності.
- пакетна конвертація кількох файлів.

Як конвертувати файл в Online Audio Converter:

1) Завантажити файл на сайт (натиснути **"Upload"**).
2) Вибрати формат для конвертації (MP3, WAV, FLAC).
3) Налаштувати якість файлу (128 kbps, 192 kbps, 320 kbps).
4) Натиснути **"Convert"** → після обробки завантажити готовий файл.

Приклад зміни формату MP3 у WAV:

1. Відкрити сайт.
2. Завантажити MP3-файл.
3. Вибрати вихідний формат WAV.
4. Натиснути "Convert".
5. Завантажити готовий файл.

Завдання

Завдання 1. Конвертація аудіо у формат MP3 та WAV

1. Завантажити аудіофайл у форматі **WAV**.
2. Використовуючи FFmpeg (або будь-який інший зручний для вас засіб конвертації та вибудовування інформації про аудіофайл), конвертувати його у **MP3**.
3. Перевірити характеристики включаючи розмір файлів до і після конвертації (надати скріншоти з видобутою інформацією).

Завдання 2. Налаштування параметрів аудіофайлу

1. Конвертувати WAV → MP3 із різними бітрейтами (128 kbps, 256 kbps, 320 kbps).
2. Визначити різницю в розмірі та якості файлів.
3. Провести зворотну конвертацію (MP3 → WAV) і перевірити, чи повертається оригінальна якість.

Завдання 3. Дослідження впливу частоти дискретизації

1. Завантажити файл із частотою **44100 Hz**.
2. Конвертувати його у варіанти **22050 Hz** та **48000 Hz**.
3. Прослухати файли та зробити висновки щодо якості та доцільності конвертування.

Загальні рекомендації до виконання звіту

1. **Чіткість і структурованість**.

Опис кожного кроку має бути чітким і послідовним, з логічними поясненнями дій. Використовувати заголовки для кожного етапу роботи, щоб полегшити орієнтацію в звіті.

2. **Скріншоти**.

Для кожного етапу роботи додавати чіткі скріншоти з відповідними підписами. Підписи повинні пояснювати дії, що виконуються, та налаштування, що застосовуються.

3. **Аналіз і пояснення**.

У звіті повинно бути пояснення вибору кожного інструменту та налаштувань. Порівняти результати аналізу різних форматів зображень, зокрема, роздільну здатність, розмір файлу і прозорість.

4. **Оформлення**.

Використовувати чітку нумерацію та форматування для кожного розділу. Підписати всі скріншоти, що додаються до звіту.

5. **Висновки**.

Висновки мають бути логічними та чіткими, з підсумком виконаної роботи і порівнянням результатів.

Контрольні питання:

1. Яка різниця між MP3 і WAV?
2. Чому бітрейт впливає на якість звуку?
3. Як змінюється розмір файлу при зміні частоти дискретизації?
4. Чи можна повністю відновити якість після конвертації MP3 у WAV?
5. Які програми використовуються для конвертації аудіофайлів?

Література: [8-11]

Практичне заняття № 4

«Обробка відеофайлів»

Мета: ознайомитись з основними методами обробки відеофайлів, зокрема, з процесами конвертації форматів відео та витягування кадрів з відео, вивчити використання інструменту для перетворення відеофайлів у різні формати, зміни параметрів відео (роздільної здатності, бітрейту) та витягування окремих кадрів із відеофайлу, розвинути навички роботи з мультимедійними даними, застосовуючи отримані знання для реальних задач з обробки відео.

Теоретичні відомості

4.1 Конвертація відеофайлів.

Конвертація відеофайлів – це процес перетворення одного відеоформату в інший. Це може бути необхідно для забезпечення сумісності відеофайлів з різними пристроями, операційними системами, медіаплеєрами або програмами для обробки та редагування відео. Конвертація дозволяє адаптувати відеофайли до потрібних вимог, таких як зміна якості, роздільної здатності або підтримка специфічних кодеків.

Відеофайли зазвичай зберігаються у різних форматах, кожен з яких має свої переваги та недоліки. Наприклад, формат **MP4** з кодеком **H.264** є популярним завдяки своїй універсальності та високій якості при невеликому розмірі файлу. Однак не всі пристрої підтримують цей формат, або відеофайл може бути занадто великим для певного застосування (наприклад, при передачі через інтернет). У таких випадках може виникнути потреба в конвертації відео у формат, який підтримується цільовим пристроєм чи програмою.

Навіщо конвертувати відео:

1. **Сумісність з пристроями та програмами.**

Різні пристрої (мобільні телефони, планшети, телевізори, комп'ютери) можуть підтримувати різні формати відео. Наприклад, старі пристрої можуть не підтримувати сучасні формати або кодеки, такі як HEVC (H.265).

2. **Оптимізація для Інтернету.**

Для швидшої передачі відео через Інтернет або зменшення обсягу для збереження на пристрої можна перетворити відеофайл у формат з більш стиснутими даними, наприклад, MP4, щоб забезпечити баланс між якістю та розміром.

3. **Полегшення обробки відео.**

При редагуванні відеофайлів може бути необхідно перетворити його в інший формат або кодек для зручності роботи в певному відеоредакторі.

4. **Покращення якості.**

Іноді конвертація може бути використана для покращення якості відео або адаптації до специфічних вимог (зміна роздільної здатності, бітрейту, частоти кадрів).

Типові операції при конвертації відео:

1) **Зміна формату контейнера:** перетворення з одного контейнера (наприклад, AVI) на інший (наприклад, MP4).

2) **Зміна кодека:** перекодування відео з одного кодека (наприклад, VP8) в інший (наприклад, H.264).

3) **Зміна параметрів відео:** можна зменшити роздільну здатність, змінити частоту кадрів або бітрейт для покращення сумісності або зменшення розміру файлу.

4) **Зміна аудіо параметрів:** при необхідності можна також змінювати аудіоформат, частоту дискретизації або бітрейт.

4.2 Відеоформати

Відеоформати визначають спосіб зберігання відеоданих у файлі, а також тип використовуваного контейнера і кодека для стиснення та декодування відео і аудіо. Залежно від формату відео можна оптимізувати його для різних пристроїв, вимог якості або розміру файлу.

MP4 (або MPEG-4 Part 14) є одним з найпоширеніших форматів відео в Інтернеті завдяки своїй високій ефективності стиснення та широкій підтримці на різних платформах і пристроях.

Переваги:

- Висока сумісність з більшістю пристроїв та медіаплеєрів.
- Добре стискає відео при збереженні гарної якості.
- Може містити відео, аудіо, текстові субтитри та зображення.

Недоліки:

- Може мати обмеження щодо розширених функцій, таких як обробка 3D або HDR, у порівнянні з іншими форматами.

Типовий кодек: H.264 (для відео), AAC (для аудіо).

Приклад використання: MP4 широко використовується для зберігання відео на смартфонах, комп'ютерах, телевізорах і в потокових сервісах (YouTube, Netflix)

AVI (Audio Video Interleave) – один з найстаріших форматів відеофайлів, розроблений Microsoft. Використовується для зберігання відео та аудіо даних у одному файлі.

Переваги:

- Підтримує великі обсяги відео і високі якості.
- Легко редагується в багатьох відеоредакторах.

Недоліки:

- Великий розмір файлу при високій якості, що ускладнює зберігання та передачу.
- Менша ефективність стиснення порівняно з іншими форматами, такими як MP4 або MKV.

Типовий кодек: Часто використовує кодек DivX або XviD.

Приклад використання: AVI використовувався в основному для зберігання відеофайлів на старих комп'ютерах і для відеоредагування в професійному середовищі.

MKV (Matroska Video) – це відкритий формат контейнера, який дозволяє зберігати відео, аудіо, субтитри та інші дані в одному файлі.

Переваги:

- Підтримка декількох відео та аудіотреків у одному файлі.
- Без втрат якості відео при конвертації.
- Може зберігати субтитри і метадані.

Недоліки:

- Може не підтримуватись старими медіаплеєрами або деякими мобільними пристроями.

Типовий кодек: H.264 або H.265 для відео, AAC або DTS для аудіо.

Приклад використання: MKV часто використовується для зберігання високоякісних фільмів і серіалів, оскільки підтримує декілька мовних треків, субтитри та інші мультимедійні елементи.

MOV – це відеоформат, розроблений компанією Apple, і є основним форматом для відео у середовищі QuickTime.

Переваги:

- Висока якість зображення.
- Широка підтримка на платформах Apple та високоякісне стиснення.

Недоліки:

- Великий розмір файлу у порівнянні з іншими форматами.
- Може не підтримуватись на деяких пристроях або операційних системах без спеціальних плагінів.

Типовий кодек: H.264, ProRes.

Приклад використання: MOV часто використовується професіоналами для зйомки та редагування відео в індустрії кіно та відеопродукції.

FLV (Flash Video) – це формат, розроблений для відтворення відео через Adobe Flash Player.

Переваги:

- Добре стиснутий, що дозволяє зменшити розмір файлу.
- Широко підтримується на вебсайтах і у потокових сервісах.

Недоліки:

- Потребує Flash Player для відтворення, що обмежує підтримку на нових пристроях та браузерах.
- Низька якість порівняно з іншими сучасними форматами (такими як MP4).

Типовий кодек: H.264 або VP6 для відео.

Приклад використання: FLV широко використовувався для потокового відео на сайтах (наприклад, YouTube до 2015 року).

WEBM – це відкритий формат для відео і аудіо, розроблений спеціально для використання в Інтернеті. Він підтримується більшість сучасних браузерів і є оптимальним для онлайн-платформ.

Переваги:

- Висока якість стиснення без великих втрат.
- Підтримується у більшості сучасних браузерів (Chrome, Firefox, Opera).
- Відкритий стандарт без патентних обмежень.

Недоліки:

- Не підтримується на старих браузерах або старих мобільних пристроях.

Типовий кодек: VP8 або VP9 для відео, Vorbis або Opus для аудіо.

Приклад використання: WEBM часто використовується для зберігання відео на вебсайтах і в потокових сервісах завдяки своїй оптимізації для інтернету.

4.3 Приклад виконання практичного завдання.

Конвертація відео з формату .mp4 у формат .avi.

Для цього використовуємо команду FFmpeg, яка дозволяє швидко конвертувати відеофайли між різними форматами.

Команда: `ffmpeg -i input.mp4 output.avi`

`-i input.mp4` вказує вхідний файл (в нашому випадку, відео у форматі .mp4).

`output.avi` – це бажаний формат виводу.

FFmpeg автоматично визначить параметри відео та конвертує його у формат .avi, зберігаючи базову якість відео.

Далі налаштуємо параметри конвертації: зменшимо бітрейт і роздільну здатність.

Для зменшення бітрейту відео до 1000 кбіт/с та зниження роздільної здатності до 640x360 пікселів використовуємо додаткові параметри FFmpeg.

Команда: `ffmpeg -i input.mp4 -b:v 1000k -s 640x360 output.avi`

(Можете користуватись повними шляхами: `"D:\ffmpeg\bin\ffmpeg.exe" -i "C:\Users\User\Desktop\input.mp4" -b:v 1000k -s 640x360 "C:\Users\ User\Desktop\output.avi"`)

`-b:v 1000k` – задає бітрейт відео (1000 кбіт/с).

`-s 640x360` – змінює роздільну здатність на 640x360 пікселів.

`output.avi` – вихідний файл у форматі .avi.

Бітрейт контролює кількість даних, що використовуються для збереження кожного кадру відео, знижуючи його ми отримуємо менший розмір файлу.

Зменшення роздільної здатності також допомагає зменшити розмір файлу, але впливає на якість зображення.

Перевіримо результат конвертації.

Після завершення процесу конвертації перевірте файл, відкривши його в медіаплеєрі або перевіривши параметри файлу **командою:**

`ffmpeg -i output.avi`

Це дозволить вам побачити, чи правильний формат, бітрейт та роздільна здатність файлу. Далі спробуємо витягнути окремі кадри з відео на конкретних

відрізках часу для подальшого використання, наприклад, як зображення для створення ескізів або кадрів для прев'ю.

Витягуємо кадри кожні 10 секунд і зберігаємо їх у форматі .jpg.

Команда:

```
ffmpeg -i input.mp4 -vf "fps=1/10" frame_%03d.jpg
```

`-vf "fps=1/10"` – задає частоту кадрів: витягати один кадр кожні 10 секунд (1/10 кадра на секунду).

`frame_%03d.jpg` – зберігає кожен кадр як зображення з іменем "frame_001.jpg", "frame_002.jpg" і т.д.

FFmpeg автоматично генерує зображення з відеофайлу кожні 10 секунд і зберігає їх у вказаному форматі (у нашому випадку це .jpg). Число `%03d` означає, що кожен кадр буде мати тризначний номер.

Витягуємо кадри з конкретних точок відео (наприклад, на 5-й, 10-й та 15-й секунді).

Команда: `ffmpeg -i input.mp4 -vf "select='eq(n\,5)+eq(n\,10)+eq(n\,15)'" -vsync vfr frame_%03d.jpg`

`select='eq(n\,5)+eq(n\,10)+eq(n\,15)'` – команда, яка вибирає кадри на 5-й, 10-й та 15-й секунді відео.

`-vsync vfr` – опція для синхронізації відео без фіксованої частоти кадрів.

Цей метод дозволяє вибирати кадри, що знаходяться на конкретних відрізках часу (5, 10, 15 секунд). Ви можете змінити час у команді, щоб вибрати інші кадри.

Перевіряємо коректність збережених зображень.

Перевірте, чи були правильно витягнуті кадри, відкривши їх у папці з файлами. Переконайтеся, що зображення відповідають точкам часу, які ви вказали (5, 10, 15 секунд).

Завдання

Завдання 1: Конвертація формату відео:

1. Використовуючи **FFmpeg** (чи інший зручний для вас засіб конвертації), конвертувати відеофайл з формату .mp4 у формат .avi.
2. Налаштувати параметри конвертації: зменшити бітрейт відео до ***N*** кбіт/с і знизити роздільну здатність відео до ***N*** пікселів.
3. Перевірити результат конвертації.

Завдання 2: Витяг кадрів із відеофайлу:

1. Використовуючи **FFmpeg** (чи інший зручний для вас засіб конвертації), витягнути кадри з відеофайлу .mp4 кожні ***N*** секунд і зберегти їх у форматі .jpg.
2. Використати команду FFmpeg для витягування кадрів із різних точок відео (наприклад, на ***N*** -й, ***N*** -й та ***N*** -й секунді).
3. Перевірити коректність збережених зображень.

Таблиця 4.1 – Варіанти завдань

Варіант	**Бітрейт (кбіт/с)**	**Роздільна здатність (px)**	**Інтервал витягування кадрів (сек)**	**Часові мітки кадрів (сек)**
1	1000	650x370	11	4, 8, 12
2	1200	800x450	8	4, 12, 20
3	1500	1024x576	5	3, 9, 18
4	800	480x270	15	7, 14, 21
5	2000	1280x720	6	2, 8, 16
6	500	320x180	12	6, 13, 19
7	1800	960x540	7	3, 11, 22
8	2500	1920x1080	4	1, 5, 10
9	1400	854x480	9	4, 10, 17
10	600	400x225	14	5, 15, 25

Загальні рекомендації до виконання звіту

1. **Чіткість і структурованість**.

Опис кожного кроку має бути чітким і послідовним, з логічними поясненнями дій. Використовувати заголовки для кожного етапу роботи, щоб полегшити орієнтацію в звіті.

2. **Скріншоти**.

Для кожного етапу роботи додавати чіткі скріншоти з відповідними підписами. Підписи повинні пояснювати дії, що виконуються, та налаштування, що застосовуються.

3. **Аналіз і пояснення**.

У звіті повинно бути пояснення вибору кожного інструменту та налаштувань. Порівняти результати аналізу різних форматів зображень, зокрема, роздільну здатність, розмір файлу і прозорість.

4. **Оформлення**.

Використовувати чітку нумерацію та форматування для кожного розділу. Підписати всі скріншоти, що додаються до звіту.

5. **Висновки**.

Висновки мають бути логічними та чіткими, з підсумком виконаної роботи і порівнянням результатів.

Контрольні питання:

1. Що таке конвертація відеоформатів і чому вона може бути необхідною?
2. Які основні формати відеофайлів підтримуються у сучасних відеопрограмах і додатках?
3. Що таке **FFmpeg** і як цей інструмент може допомогти в обробці відео?
4. Які параметри відео можна змінювати під час конвертації файлів? Наведіть приклади.

5. Як можна витягнути окремі кадри з відео, і в яких випадках це може бути корисно?

6. Які формати зображень підтримуються для збереження витягнутих кадрів із відео?

7. Як за допомогою **FFmpeg** налаштувати частоту витягування кадрів (наприклад, кожні 5 або 10 секунд)?

8. Чому важливо правильно налаштовувати параметри конвертації відеофайлів для збереження якості?

Література: [8, 11]

Практичне заняття № 5

«Робота з метаданими мультимедіа»

Мета: ознайомитись з поняттям метаданих у мультимедіа, їх типами та стандартами, вивчити методи перегляду, редагування та вбудовування метаданих у зображення, відео та аудіофайли та набути практичних навичок роботи з метаданими за допомогою вбудованих інструментів операційної системи та програмних засобів.

Теоретичні відомості

5.1 Визначення метаданих мультимедіа.

Метадані – це дані, які описують інші дані. Вони надають контекст, структуру та додаткову інформацію про дані, що зберігаються або передаються. У випадку мультимедійних файлів, метадані містять інформацію про властивості контенту, таку як дата створення, автор, розмір, формат, кодек, роздільна здатність, геолокація, а також можуть включати текстові описи, ключові слова або ліцензійні дані.

Метадані можуть бути вбудовані безпосередньо в сам файл (наприклад, у форматі EXIF для зображень або ID3 для аудіофайлів) або зберігатися окремо в базах даних або зовнішніх файлах, пов'язаних з мультимедійним контентом. Вони значно покращують організацію, пошук і обробку мультимедійних ресурсів, а також дозволяють здійснювати автоматичні процеси (наприклад, індексацію або категоризацію).

Метадані можна класифікувати на два **основні види**: **структуровані** та **неструктуровані**.

Структуровані метадані мають чітко визначену структуру або формат, за яким організована інформація. Вони можуть бути представлені у вигляді таблиць або записів в базах даних, що містять конкретні поля для зберігання певних характеристик. Прикладом можуть бути метадані в форматах XML,

JSON або базах даних, де є чітко визначені поля для кожного елементу, наприклад, дата створення файлу, автор, розмір файлу, роздільна здатність зображення, тривалість аудіо чи відео тощо. Структуровані метадані дозволяють швидко здійснювати пошук, фільтрацію, аналіз і управління даними через їх чітке організоване представлення. Наприклад, у базі даних для відео можна мати такі поля, як "Тривалість", "Дата випуску", "Режисер".

Неструктуровані метадані не мають чіткої структури та не обов'язково зберігаються у стандартних формах, таких як таблиці або записи. Вони можуть бути представлені у вигляді текстових описів, коментарів, тегів або інших неструктурованих даних, що описують зміст файлу. Наприклад, метаданими можуть бути текстові описи до відео, вільно написані теги для зображень чи додаткові анотації. Неструктуровані метадані дають більше свободи для опису контенту, але їх обробка та пошук можуть бути більш складними, оскільки відсутня чітка організація. Наприклад, до того ж відео можна додати тег "пригоди" або текстовий опис сюжету, що є менш формалізованим.

Метадані відіграють важливу роль у контексті мультимедіа, оскільки вони дозволяють ефективно організовувати, зберігати, шукати та управляти мультимедійними файлами. Ось кілька **основних функцій метаданих** у мультимедіа:

1. **Ідентифікація та класифікація контенту**.

Метадані забезпечують можливість точно визначити вміст мультимедійного файлу. Це може бути заголовок, опис, ключові слова, автор, дата створення тощо. Вони дозволяють класифікувати контент та роблять його доступним для пошуку та навігації.

2. **Пошук та фільтрація**.

Використання метаданих дозволяє швидко знаходити потрібні мультимедійні файли, навіть якщо їх кількість є дуже великою. Наприклад, можна шукати за датою, автором, категорією або навіть за ключовими словами, що описують вміст файлу.

3. **Управління правами доступу**.

Метадані можуть містити інформацію про права доступу до мультимедійного контенту, що дозволяє контролювати, хто має доступ до файлів. Це важливо для захисту авторських прав і забезпечення безпеки.

4. **Опис та контекст контенту**.

Метадані забезпечують додаткову інформацію про мультимедійний контент, допомагаючи користувачам зрозуміти, що міститься в файлі. Наприклад, метадані відеофайлу можуть включати опис сюжету, жанр, акторів, режисера, що дозволяє краще зрозуміти, чого очікувати від файлу перед його переглядом.

5. **Сумісність і стандартизація**.

Метадані допомагають забезпечити сумісність мультимедійних файлів із різними програмами та платформами. Вони стандартизують формат файлів та полегшують їх використання на різних пристроях і в різних середовищах.

6. **Аналіз та звітність.**

Метадані використовуються для аналізу використання мультимедійного контенту, наприклад, для статистики переглядів, завантажень, часу перегляду, що є корисним для аналітики та покращення контенту.

7. **Автоматизація та інтеграція**.

Завдяки метаданим мультимедійні файли можуть автоматично інтегруватися з іншими системами або додатками. Це дозволяє автоматично додавати нові файли в каталоги, оновлювати інформацію про них та здійснювати інші автоматизовані операції.

Для фотографій метадані можуть включати інформацію про камеру, експозицію, географічне місце зйомки (GPS-координати), що дозволяє класифікувати фотографії за різними параметрами, шукати їх за географічним місцем або типом камери та ефективно організовувати фотографії в базі даних.

Типи метаданих мультимедіа:

- Описовий (Content-based) тип метаданих.
- Адміністративний (Administrative) тип метаданих.

- Технічний тип метаданих.
- Системний тип метаданих.

Описовий тип метаданих забезпечує інформацію про вміст мультимедійного файлу. Це може включати текстові описи, ключові слова, категорії, жанри, авторів, титули, а також інші елементи, що допомагають класифікувати і ідентифікувати контент. Описові метадані дозволяють користувачам шукати, знаходити і оцінювати мультимедійні файли за змістом. **Приклад**: для відеофайлу описові метадані можуть включати жанр (комедія, драма), сюжетний опис, режисера, акторів та ключові слова (наприклад, "екшн", "пригоди").

Адміністративні метадані пов’язані з управлінням і контролем мультимедійних файлів. Вони включають інформацію про права доступу, авторські права, ліцензії, а також дані, що визначають, хто створив, опублікував або змінив файл. Цей тип метаданих також може включати дату створення, дату останнього доступу, ідентифікатори файлів тощо. **Приклад**: Метаінформація про автора файлу, дату створення, тип ліцензії (наприклад, "вільне використання", "тільки для особистого використання").

Технічні метадані надають інформацію про формат і властивості мультимедійного файлу. Це включає такі дані, як роздільна здатність зображень, тривалість відео, частота кадрів, розмір файлу, кодек та інші технічні характеристики, необхідні для обробки або відтворення файлу. Вони допомагають програмам та пристроям коректно обробляти та відтворювати мультимедійний контент. **Приклад**: Для відеофайлу технічні метадані можуть включати роздільну здатність (1920x1080), частоту кадрів (30 fps), використовуваний відеокодек (H.264).

Системні метадані зберігають інформацію, яка забезпечує функціонування файлової системи або бази даних. Вони містять дані про файл, які автоматично генеруються під час створення, зміни або зберігання файлів. Системні метадані включають дані про файли, такі як їхній шлях, розмір, тип,

ідентифікатор версії та іншу технічну інформацію, яка допомагає системам управляти файлами. **Приклад**: Для файлу це можуть бути метадані, які вказують його місцезнаходження в системі (шлях до файлу), дата останньої модифікації або версія файлу.

5.2 Структури метаданих для мультимедіа і їх інтеграція у мультимедійні файли

EXIF (Exchangeable Image File Format): стандарт для зберігання метаданих у зображеннях, що зберігаються в форматах JPEG, TIFF і інших. Вони включають технічну інформацію про фотоапарат, параметри зйомки, такі як експозиція, баланс білого, роздільна здатність, час зйомки та географічні координати (якщо вбудовано GPS).

Приклад: у зображенні, зробленому камерою, EXIF-метадані можуть містити такі відомості, як марка та модель камери, налаштування діафрагми та витримки, дата й час зйомки, географічні координати місця зйомки.

IPTC (International Press Telecommunications Council) є стандартом метаданих, широко використовуваним у журналістиці та медіа, і визначає формат для зберігання текстової інформації про зображення. Це включає в себе опис, авторські права, категорії, ключові слова, заголовки та інші описові дані. IPTC дозволяє організовувати та класифікувати медіафайли для полегшення їх пошуку.

Приклад: у новинному зображенні IPTC може включати такі метадані, як ім’я фотографа, дата публікації, заголовок (наприклад, "Протест у місті X") та опис ("Демонстрація на центральній площі міста X").

XMP (Extensible Metadata Platform) – це стандарт метаданих, розроблений Adobe, що дозволяє зберігати метадані в різних типах медіафайлів. XMP є більш гнучким і розширюваним, ніж EXIF і IPTC, і дозволяє зберігати як стандартні, так і користувацькі метадані, включаючи відомості про автора, ліцензію, права доступу та інші. XMP також підтримує інтеграцію з іншими форматами, такими як XML.

Приклад: у фотографії XMP-метадані можуть містити додаткову інформацію про обробку зображення (наприклад, використання конкретних фільтрів або корекцій), а також права на використання та ліцензію.

EXIF в основному використовує технічні метадані, зокрема інформацію про параметри камери під час зйомки. **IPTC** надає більше можливостей для організації та класифікації, акцентуючи увагу на описових даних, таких як автор, заголовки та категорії. **XMP** є більш універсальним і розширюваним стандартом, що дозволяє зберігати не лише стандартні, але й користувацькі метадані.

XML є стандартом для зберігання та обміну структурованими даними у вигляді текстових файлів. Він використовується для зберігання метаданих через свою гнучкість і можливість описувати складні структури даних за допомогою тегів.

Переваги:

- Читається людиною, має зрозумілу структуру.
- Підтримує вкладеність ієрархічних даних (можливість створювати вкладені теги).
- Стандартизована підтримка на багатьох платформах.

Приклад:

```
<image>
  <title>Sunset over the mountains</title>
  <author>John Doe</author>
  <date>2023-05-01</date>
  <description>A beautiful sunset captured in the mountains.</description>
</image>
```

Тут XML використовується для зберігання метаданих зображення, таких як заголовок, автор і дата.

JSON – це легкий формат для обміну даними, який зазвичай використовується в веб-розробці. JSON має більш компактний вигляд порівняно з XML і є популярним через зручність обробки в JavaScript та інших мовах програмування.

Переваги:

- Менший розмір файлів у порівнянні з XML.
- Легше обробляється програмно, зокрема в JavaScript.
- Підтримується більшістю сучасних мов програмування.

Приклад:

```
{
  "image": {
    "title": "Sunset over the mountains",
    "author": "John Doe",
    "date": "2023-05-01",
    "description": "A beautiful sunset captured in the mountains."
  }
}
```

У JSON метадані зображення зберігаються у вигляді об'єкта, де кожне поле має своє значення.

Вибір між XML та JSON залежить від конкретних вимог до метаданих. Якщо потрібно зберігати складні ієрархії та працювати з великими даними, XML може бути кращим вибором. Якщо важлива компактність, швидкість і сумісність з веб-технологіями, то JSON буде оптимальним варіантом.

Вбудовування метаданих у зображення здійснюється через стандартні формати, такі як EXIF (Exchangeable Image File Format) і XMP (Extensible Metadata Platform).

Для **відеофайлів** використовують такі стандарти як MPEG-7 і XMP для вбудовування метаданих. MPEG-7 спеціалізується на описі аудіо- та відеоінформації, що дозволяє класифікувати відео за різними

характеристиками. XMP, як і у випадку зображень, дозволяє зберігати метадані для відеофайлів, наприклад, авторські права, дату зйомки тощо.

У випадку з **аудіофайлами** популярними є формати MP3, FLAC і WAV, де метадані зберігаються за допомогою ID3 тегів. ID3 містить інформацію про назву треку, виконавця, альбом, рік випуску, жанр і навіть обкладинку альбому. Для більш гнучкого зберігання можуть використовуватися формати, такі як XMP або XML.

Техніки вбудовування метаданих у файли:

1. **Вбудовування метаданих через теги.**

У мультимедійних файлах (зображеннях, відео та аудіо) метадані часто вбудовуються за допомогою спеціальних тегів або структур, які інтегруються безпосередньо в сам файл. Це дозволяє зберігати метадані разом із контентом і зберігати цілісність даних при їх переміщенні чи копіюванні.

2. **Використання контейнерних форматів**

В багатьох випадках мультимедійні файли можуть використовувати контейнерні формати, які дозволяють вбудовувати різні типи даних, включаючи метадані. Наприклад, формати MP4 або MKV дозволяють вбудовувати метадані безпосередньо в структуру контейнера.

3. **Структуровані формати метаданих**

Окрім вбудовування метаданих в форматах, таких як EXIF або ID3, використовуються структуровані формати, такі як XML або JSON, для збереження більш складної та багатошарової інформації. Вони дозволяють зберігати метадані, організовані за допомогою тегів, які можуть описувати більше атрибутів мультимедійного файлу.

4. **Вбудовування за допомогою спеціальних програмних інтерфейсів (API)**

Для автоматизованого вбудовування метаданих в файли використовуються програмні інтерфейси, такі як FFmpeg для відео та аудіофайлів або бібліотеки для роботи з зображеннями, наприклад, PIL (Python Imaging Library). Ці інтерфейси дозволяють програмістам додавати, змінювати

або видаляти метадані в різних типах файлів без необхідності вручну редагувати кожен файл.

5. **Вбудовування метаданих у "порожні" частини файлів**

Деякі типи файлів дозволяють використовувати неактивні ділянки даних, наприклад, порожні байти в кінці файлів для зберігання метаданих. Це може бути корисно для великих мультимедійних файлів, де основні дані зображень, аудіо чи відео не мають бути змінені, але метадані все одно повинні бути збережені.

6. **Інтеграція через хмарні платформи**

Деякі хмарні сервіси, такі як Amazon S3 або Google Cloud Storage, дозволяють додавати метадані до мультимедійних файлів на сервері, зберігаючи їх разом з файлами в хмарі. Це дозволяє централізовано управляти метаданими і полегшує доступ до них через API.

Ці техніки вбудовування метаданих дозволяють ефективно організовувати та управляти мультимедійними даними, забезпечуючи додаткову інформацію для кращого пошуку, категоризації та використання контенту.

5.3. Приклади керування метаданими

Параметр `-metadata` у FFmpeg використовується для **додавання, редагування або видалення** метаданих у мультимедійних файлах. Метадані можуть містити інформацію про назву файлу, автора, мову, дату створення тощо.

Синтаксис: `-metadata[:metadata_specifier] key=value`

`key=value` – визначає поле метаданих і його значення (наприклад, `title="Мій відеофайл"`).

`metadata_specifier` (необов’язково) – вказує, до якої частини файлу застосовуються метадані (потік, глава, програма).

Приклад: встановлення заголовка (title).

Щоб задати назву відеофайлу, використовуйте команду:

```
ffmpeg -i input.avi -metadata title="Мій заголовок" output.flv
```

Ця команда бере файл `input.avi`, встановлює його заголовок як "Мій заголовок" і зберігає вихідний файл як `output.flv`.

Додати власний коментар до файлу:

```
ffmpeg -i input.mp4 -metadata comment="Це тестовий файл для навчання" output.mp4
```

Додати одразу кілька метаданих у відеофайл:

Додавання назви `"Демо-фільм"`, автора `"Студія А"`, рік `"2024"`.

```
ffmpeg -i input.mp4 -metadata title="Демо-фільм" -metadata author="Студія А" -metadata year="2024" output.mp4
```

Додавання метаданих до певних потоків.

Можна додати метадані до конкретного потоку (наприклад, аудіо). Щоб встановити мову для першого аудіопотоку:

```
ffmpeg -i input.mp4 -metadata:s:a:0 language=eng output.mp4
```

Розбір команди:

`-metadata:s:a:0` – вказує, що змінюється **аудіопотік** (`a`), **перший за порядком** (`0`).

`language=eng` – встановлює мову як **англійську**.

Перезапис та видалення метаданих.

Якщо одне й те саме поле метаданих використовується кілька разів, останнє значення **перезапише** попереднє.

Щоб **видалити метадані**, потрібно задати порожнє значення:

```
ffmpeg -i input.mp4 -metadata title="" output.mp4
```

Це видалить поле `title`.

Завдання

Завдання 1: Аналіз метаданих зображення через властивості файлу

1. Завантажити будь-яке зображення.
2. Відкрити його метадані через **ПКМ → Властивості → Докладно**.
3. Визначити, які типи метаданих (описові, адміністративні, технічні, системні) присутні.
4. Заповнити (або змінити) деякі поля метаданих вручну (наприклад, автор, назва, коментар).
5. Записати короткий опис метаданих файлу.

Завдання 2: Використання FFmpeg для роботи з метаданими мультимедіа.

4. Вибрати зображення, аудіо або відеофайл.
5. Використати FFmpeg для перегляду метаданих (`ffmpeg -i файл`).
6. Використайте FFmpeg, щоб змінити назву файлу на "`<Група> <Прізвище> <Поточний рік>`".
7. Додати власний коментар до файлу "`Я знаходжусь у команді прихильників котів/собак`" (або будь-який інший коментар).
8. Додати одразу кілька метаданих у файл (від 3 на власний розсуд).

Загальні рекомендації до виконання звіту

1. **Чіткість і структурованість**.

Опис кожного кроку має бути чітким і послідовним, з логічними поясненнями дій. Використовувати заголовки для кожного етапу роботи, щоб полегшити орієнтацію в звіті.

2. **Скріншоти**.

Для кожного етапу роботи додавати чіткі скріншоти з відповідними підписами. Підписи повинні пояснювати дії, що виконуються, та налаштування, що застосовуються.

3. **Аналіз і пояснення**.

У звіті повинно бути пояснення вибору кожного інструменту та налаштувань. Порівняти результати аналізу різних форматів зображень, зокрема, роздільну здатність, розмір файлу і прозорість.

4. **Оформлення**.

Використовувати чітку нумерацію та форматування для кожного розділу. Підписати всі скріншоти, що додаються до звіту.

5. **Висновки**.

Висновки мають бути логічними та чіткими, з підсумком виконаної роботи і порівнянням результатів.

Контрольні питання:

1. Що таке метадані, і яку роль вони відіграють у мультимедіа?
2. Які існують типи метаданих? Наведіть приклади.
3. У чому різниця між описовими, адміністративними, технічними та системними метаданими?
4. Які стандартні формати метаданих використовуються у мультимедійних файлах?
5. Для чого використовуються формати EXIF, IPTC, XMP?
6. Чим відрізняються структуровані та неструктуровані метадані?

7. Як забезпечується збереження метаданих при редагуванні файлів?

8. Чи можна змінювати метадані без зміни самого мультимедійного файлу? Якщо так, то як?

9. Як переглянути метадані зображення у Windows без сторонніх програм?

10. Що відбудеться, якщо спробувати записати нове значення у вже існуюче поле метаданих?

11. Як додати кілька метаданих одночасно у файл за допомогою FFmpeg?

Література: [11-13]

Практичне заняття № 6

«Зберігання мультимедіа у базах даних»

Мета: ознайомитись з основами зберігання мультимедійних даних у базах даних шляхом створення умовної структури бази на основі Google Таблиць, навчитися впорядковувати, описувати та систематизувати мультимедійний контент (зображення, відео, аудіо) за допомогою метаданих і посилань на файли, збережені у Google Drive.

Теоретичні відомості

У межах трьох послідовних практичних занять пропонується реалізувати єдиний міні-проєкт, пов'язаний із організацією, обробкою та поданням мультимедійних об'єктів. Метою є ознайомлення студентів із базовими принципами роботи з мультимедійними файлами, базами даних і засобами їх візуального представлення, без необхідності програмування.

Практичні роботи реалізуються за допомогою Google-сервісів – зокрема Google Таблиць, Google Диску, Google Сайтів, Google Apps Script та додаткових онлайн-інструментів.

Проєкт поділяється на три логічні етапи:

1. «Зберігання мультимедіа у базах даних». Студенти створюють таблицю (умовну базу даних) мультимедійних об'єктів з основними полями: назва, тип, посилання, опис, дата, автор тощо. Дані зберігаються в Google Таблиці, а самі мультимедійні файли – на Google Диску.

2. «Автоматична обробка мультимедіа». Використовуючи функції Google Таблиць або сторонні сервіси (за бажанням), студенти здійснюють базову автоматичну обробку медіа (наприклад, генерація мініатюр, визначення типу файлу, автоматичне заповнення полів).

3. «Створення мультимедійного каталогу». На основі даних із Таблиці створюється візуальний мультимедійний каталог – наприклад, у

вигляді сайту на Google Sites. У ньому реалізується перегляд медіа, фільтрація за типом, короткий опис тощо.

У межах даної практичної роботи студенти знайомляться з основами зберігання мультимедійних даних у таблицях, які виконують роль простої бази даних. Головна мета – навчитися впорядковувати, систематизувати та описувати мультимедійний контент (зображення, відео, аудіо) за допомогою Google Таблиць, використовуючи метадані для полегшеного пошуку та навігації.

Це дозволяє сформувати повноцінну структуру "каталогу мультимедіа", який у подальших практичних роботах буде розширюватися автоматичною обробкою (наприклад, зменшення розміру зображень, конвертація файлів) та візуальною презентацією на Google Сайті. Таким чином, ця робота є фундаментом для подальших етапів побудови простого мультимедійного інформаційного сервісу.

6.1 Основні визначення

База даних (БД) – це впорядкований набір даних, які зберігаються в електронному вигляді та призначені для зручного збереження, пошуку, оновлення й управління інформацією. База даних дозволяє уникнути дублювання даних, забезпечити їх цілісність і підтримувати структуру, що спрощує обробку.

Основні компоненти бази даних:

- **Дані** – інформація, що зберігається у вигляді таблиць, документів або інших структур.
- **Схема** – опис структури бази даних (таблиці, поля, зв’язки).
- **Система управління базами даних (СУБД)** – програмне забезпечення, що забезпечує взаємодію користувача з базою даних (наприклад, MySQL, PostgreSQL, MongoDB, Google Sheets як спрощений варіант).

Реляційні бази даних

- Організовані у вигляді **таблиць** (рядки – записи, стовпці – поля).
- Дані між таблицями можуть бути зв’язані за допомогою **ключів**.
- Підтримують мову SQL (Structured Query Language).
- Приклади: **MySQL**, **PostgreSQL**, **SQLite**.
- У реляційних БД мультимедіа зазвичай зберігають як **BLOB** або через **посилання на зовнішні файли**.

Нереляційні бази даних (NoSQL)

- Не мають строгої табличної структури.
- Дані можуть зберігатися у вигляді **документів**, **ключ-значення**, **графів** або **колонок**.
- Гнучкіші в плані структури даних.
- Приклади: **MongoDB** (документна БД), **Redis** (ключ-значення), **Neo4j** (графова).
- Часто використовуються для зберігання великих обсягів мультимедійної інформації з гнучкою структурою метаданих.

Потреба в зберіганні мультимедійних даних у базах даних

Мультимедійні дані (зображення, відео, аудіо, документи) відіграють важливу роль у сучасних інформаційних системах, вебсайтах, цифрових архівах, освітніх платформах тощо. Зберігання таких даних у базах даних має низку переваг:

1. **Систематизація** – зручно організовувати медіа за категоріями, темами, типами тощо.
2. **Пошук** – швидкий пошук потрібного об'єкта за ключовими словами, автором, датою тощо.
3. **Метадані** – можливість зберігати додаткову інформацію (розмір, розширення, мова, опис).
4. **Цілісність** – зв’язок медіафайлів з іншими об'єктами бази даних (наприклад, з користувачами чи статтями).

5. **Автоматизація** – бази даних легко інтегруються з вебсайтами та іншими сервісами, що дозволяє створювати мультимедійні каталоги, галереї, бібліотеки тощо.

Особливо актуальним є використання **хмарних платформ** (наприклад, Google Таблиць і Google Диску), що дозволяють легко реалізувати мультимедійну базу даних без спеціального програмного забезпечення та знань програмування.

Реляційні бази даних (РБД) – це найпоширеніший тип баз даних, який зберігає інформацію у вигляді взаємопов'язаних таблиць. Такий підхід добре підходить для структурованих даних і дозволяє будувати зв'язки між різними типами об'єктів, включаючи мультимедіа.

Таблиці, поля, зв'язки

У реляційній БД основною структурною одиницею є **таблиця**. Кожна таблиця складається з **полів** (стовпців), які визначають тип даних, та **записів** (рядків), які містять самі дані.

- **Поле** – це окрема характеристика об'єкта (наприклад, назва, тип файлу, посилання).
- **Запис** – це повний набір характеристик конкретного об'єкта.
- **Зв'язки** між таблицями забезпечують логічну цілісність даних (наприклад, файл може бути пов'язаний з конкретним користувачем або категорією).

Для зберігання мультимедійних даних зазвичай створюється окрема таблиця, яка містить інформацію про файли, а також пов'язані таблиці для авторів, категорій, тегів тощо (якщо проєкт масштабний).

Використання BLOB (Binary Large Object)

BLOB – це спеціальний тип поля, призначений для зберігання **великих об'єктів**, таких як:

- Зображення (JPEG, PNG),
- Відео (MP4, AVI),

- Аудіо (MP3, WAV),
- Документи (PDF, DOCX).

Файл можна повністю зберігати в полі типу BLOB. Це зручно, коли потрібен високий рівень безпеки та контроль за цілісністю даних, однак має недоліки:

- Займає багато місця в самій базі.
- Уповільнює роботу бази при великій кількості даних.
- Погано масштабується у вебсередовищі.

Тому часто використовують **гібридний підхід**: сам файл зберігається у файловій системі або хмарному сховищі (наприклад, Google Диск), а в БД зберігається лише посилання на нього.

Нереляційні бази даних (NoSQL)

NoSQL (від *Not Only SQL*) – це тип баз даних, який не використовує традиційну табличну модель. Замість цього дані зберігаються у гнучкіших структурах: документах, графах, парах ключ-значення тощо. Такі бази особливо добре підходять для **великих обсягів мультимедійних даних**, що мають складну або змінну структуру.

Переваги для зберігання мультимедійних даних

1. **Масштабованість** – легке горизонтальне масштабування дозволяє працювати з терабайтами і петабайтами даних.
2. **Гнучкість структури** – можна зберігати записи з різною кількістю полів або вкладеними структурами (наприклад, JSON).
3. **Швидкість доступу** – оптимізовані для швидкого читання/запису, що важливо при роботі з відео чи зображеннями.
4. **Вбудована підтримка BLOB-даних** – можливість зберігати файли напряму або у зв’язці з файловими системами (наприклад, GridFS у MongoDB).

Типи NoSQL-баз і їхнє застосування

1. **Документні бази** (наприклад, MongoDB, CouchDB)

- Дані зберігаються у вигляді документів (часто у форматі JSON або BSON).
- Кожен документ може містити як посилання на файл, так і метадані.
- Підходить для створення мультимедійних каталогів із динамічною структурою.

2. **Ключ-значення бази** (Redis, Amazon DynamoDB)

- Кожен файл зберігається як значення, прив’язане до унікального ключа (наприклад, ID).
- Добре підходить для кешування зображень, мініатюр або швидкого доступу до медіа за ключем.

3. **Графові бази** (Neo4j)

- Дані подаються у вигляді вузлів та зв’язків.
- Застосовується для складних взаємозв’язків між файлами, наприклад, у медіаархівах або рекомендаційних системах.

Залежно від типу бази даних, обсягу файлів, цілей системи та обмежень – існує кілька методів зберігання мультимедійної інформації: від прямого зберігання файлів до зберігання лише їх опису.

Бінарне зберігання (BLOB)

BLOB (*Binary Large Object*) – це тип поля у базах даних, що використовується для збереження великих бінарних файлів, зокрема зображень, аудіо, відео, PDF-документів тощо.

Переваги:

- Уся інформація зберігається централізовано в базі даних.
- Простота резервного копіювання та захисту даних.

Недоліки:

- Зменшення продуктивності бази при зберіганні великих файлів.
- Ускладнення масштабування та розподілу навантаження.

Коли використовується:

- У внутрішніх системах, де важлива цілісність даних.
- У корпоративних застосунках з обмеженим обсягом файлів.

Посилання на файли

Замість зберігання файлів у базі, зберігаються лише **метадані** та **URL-посилання** до файлів, що зберігаються на зовнішніх ресурсах, таких як:

- Локальна або мережева файлова система
- Хмарні сервіси (Google Drive, Dropbox, Amazon S3)

Переваги:

- База даних залишається "легкою" та продуктивною.
- Можливість інтеграції з хмарними платформами.

Недоліки:

- Необхідність окремо захищати файли.
- Можливе порушення зв’язку між метаданими та файлами.

Коли використовується:

- У вебдодатках, портфоліо, фотогалереях.
- У навчальних мультимедійних проєктах (наприклад, Google Таблиці + Drive).

Гібридний підхід

Поєднання обох методів:

- Частина файлів (наприклад, мініатюри, обкладинки) зберігається напряму в БД (через BLOB).
- Основні великі файли – у зовнішніх джерелах, із відповідними посиланнями в базі.

Переваги:

- Баланс між продуктивністю і зручністю доступу.
- Можливість кешування або попереднього перегляду прямо з бази.

Недоліки: ускладнення реалізації (необхідно слідкувати за узгодженістю між файлами).

6.2 Приклад виконання

Створити нову Google Таблицю з назвою «Мультимедійна база даних – <група> – <ПІБ> – <поточний рік>»

1. Відкрити Google Таблиці: https://sheets.google.com
2. Натиснути «Пуста таблиця»
3. Зберегти таблицю з новою назвою: натиснути у верхньому лівому куті на назву файлу (наприклад, "Електронна таблиця без назви") → ввести «Мультимедійна база даних – <група> – <ПІБ> – <поточний рік>» → натиснути Enter (рис. 6.1).

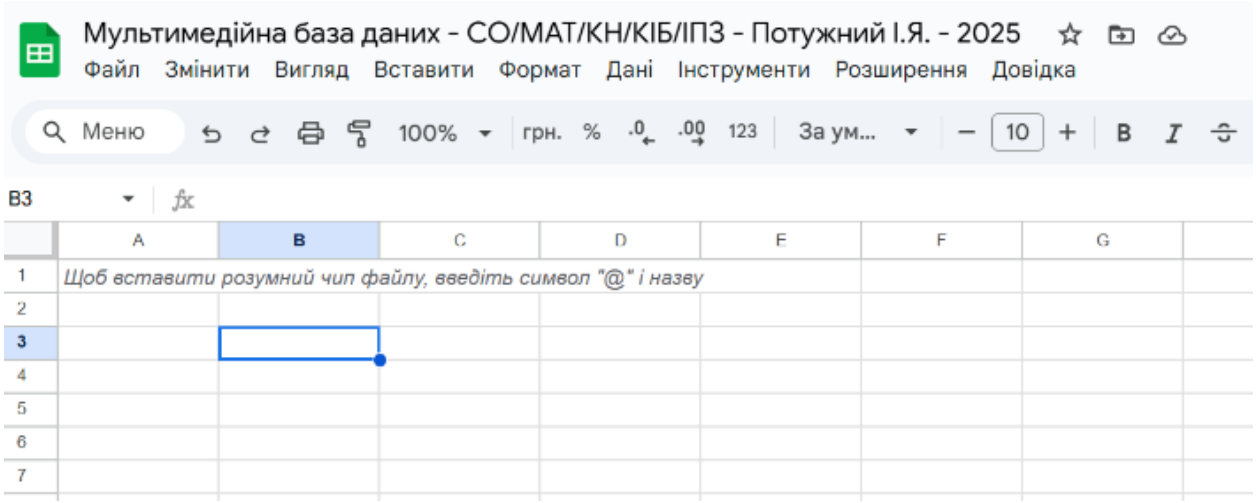


Рисунок 6.1 – Створена таблиця з назвою

Створити структуру таблиці

1. У першому аркуші (Sheet1) ввести заголовки в першому рядку:
   - ID
   - Назва файлу
   - Тип медіа
   - URL або шлях до файлу
   - Опис
   - Тривалість / Розмір

- Дата додавання
- Автор / Джерело
- Теги
- Статус

2. За потреби, розширити стовпці для зручного перегляду
3. Заморозити перший рядок (Вигляд → Закріпити → 1 рядок)

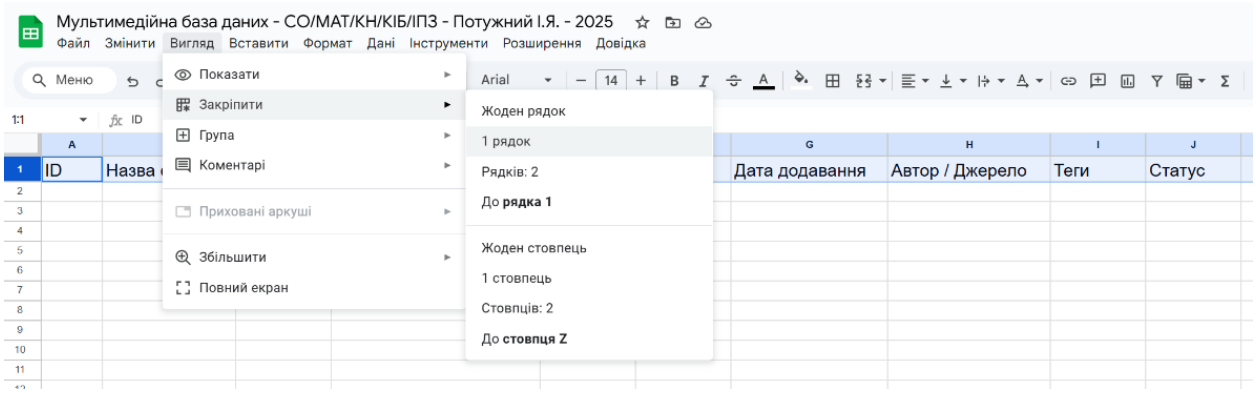


Рисунок 6.2 – Демонстрація закріплення рядка

Додати приклади мультимедійних об'єктів

1. Додати щонайменше 5 записів у таблицю, заповнивши кожен стовпець:

- ID: 1, 2, 3, …
- Назва файлу: назви медіа (наприклад, «video_about_animals.mp4»)
- Тип медіа: Відео / Аудіо / Зображення
- URL або шлях: вставити посилання на Google Диск або інший ресурс
- Опис: коротко про що контент
- Тривалість / Розмір: наприклад, «3:45» або «2.1 MB»
- Дата додавання: обрати дату
- Автор / Джерело: «YouTube», «Pixabay», власний запис тощо
- Теги: через кому – «тварини, природа, документальний»
- Статус: Наприклад, «Активний», «В архіві»

Відформатувати таблицю

1. Виділити заголовки першого рядка → натиснути «Жирний»
2. Встановити кольорове тло для заголовків (Формат → Заливка)
3. Установити відповідні формати для стовпців:
   - Для дати: Формат → Число → Дата
   - Для тривалості можна використати формат тексту
4. За бажанням, застосувати умовне форматування до «Статусу»

- Формат → Умовне форматування (рис. 6.4)
- Якщо текст дорівнює «Активний» → зелений фон
- Якщо «В архіві» → сірий фон

Або можна зробити іншим чином – через спадне меню (рис. 6.3) → «Редагувати» → виставити кольори (рис. 6.5).

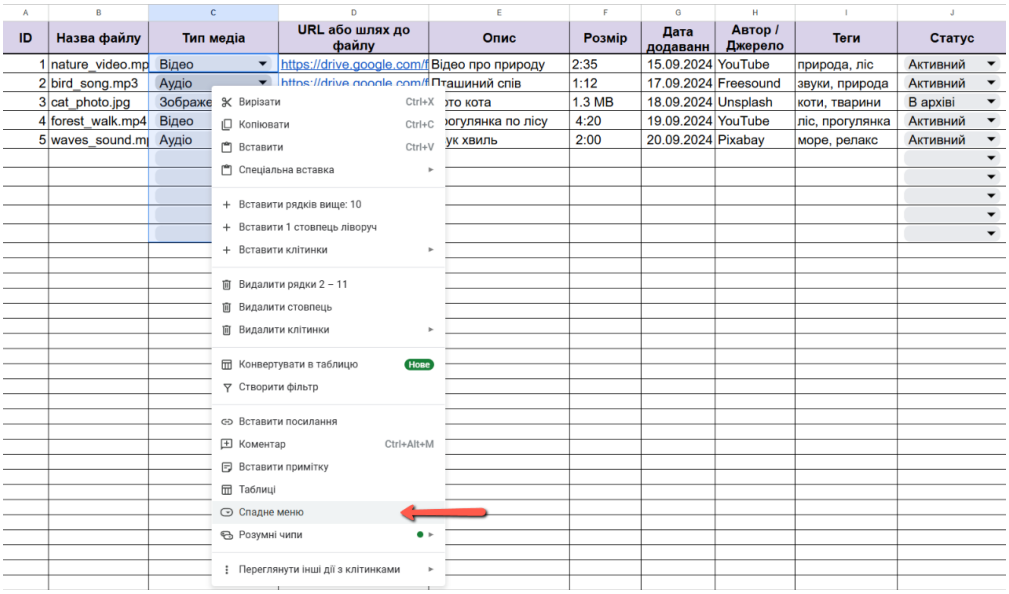


Рисунок 6.3 – Використання спадного меню

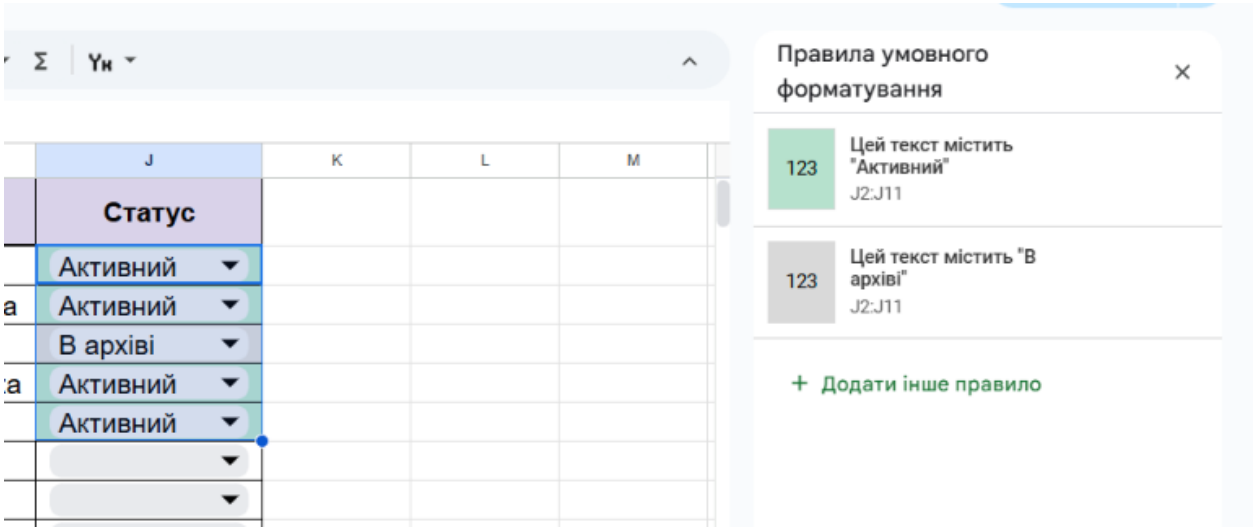


Рисунок 6.4 – Умовне форматування

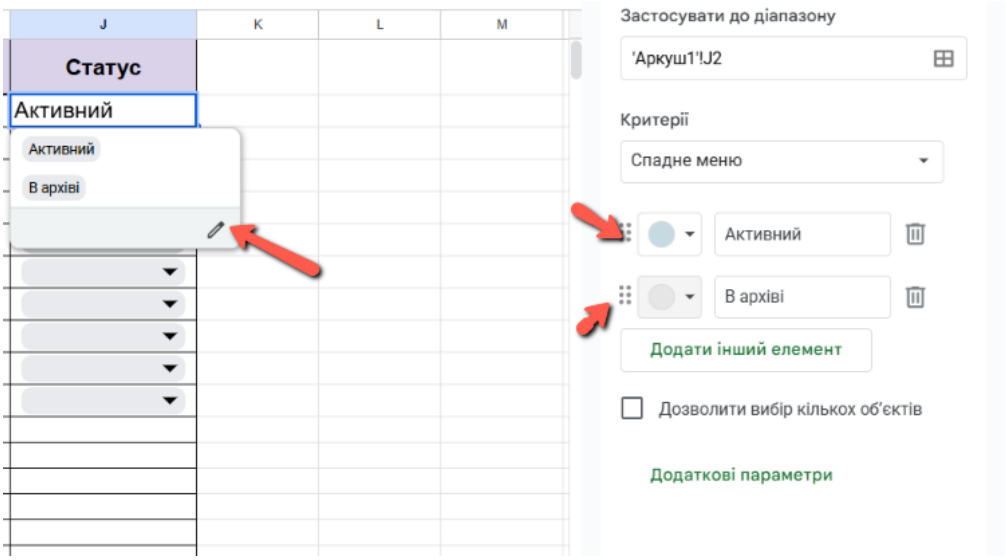


Рисунок 6.5 – Зміна кольору через редагування спадного меню

Використати фільтри

1. Виділити перший рядок із заголовками
2. Натиснути «Дані» → «Створити фільтр»
3. Тепер у кожному стовпці є піктограма фільтра – протестувати:

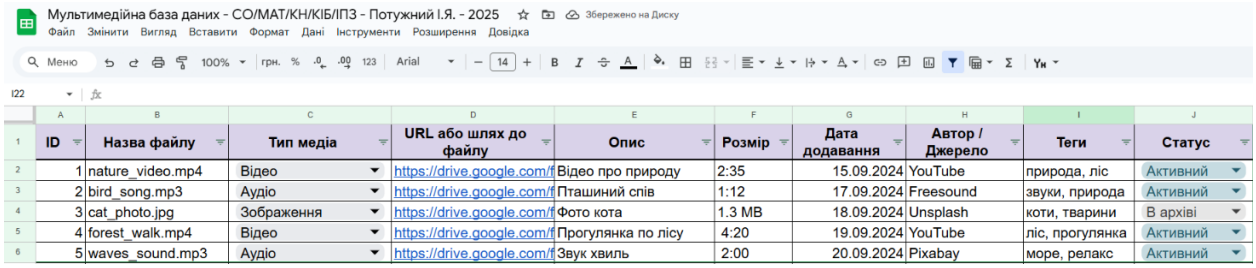
- Відфільтрувати лише «Відео»
- Відфільтрувати за «Автор / Джерело» = YouTube

Поділитися доступом

1. Натиснути кнопку «Поділитися» (у верхньому правому куті)

2. Додати адреси користувачів або обрати «Будь-хто з посиланням» → «Редактор»

3. Скопіювати посилання і вставити у звіт.



	A	B	C	D	E	F	G	H	I	J
1	ID	Назва файлу	Тип медіа	URL або шлях до файлу	Опис	Розмір	Дата додавання	Автор / Джерело	Теги	Статус
2	1	nature_video.mp4	Відео	https://drive.google.com/f	Відео про природу	2:35	15.09.2024	YouTube	природа, ліс	Активний
3	2	bird_song.mp3	Аудіо	https://drive.google.com/f	Пташиний спів	1:12	17.09.2024	Freesound	звуки, природа	Активний
4	3	cat_photo.jpg	Зображення	https://drive.google.com/f	Фото кота	1.3 MB	18.09.2024	Unsplash	коти, тварини	В архіві
5	4	forest_walk.mp4	Відео	https://drive.google.com/f	Прогулянка по лісу	4:20	19.09.2024	YouTube	ліс, прогулянка	Активний
6	5	waves_sound.mp3	Аудіо	https://drive.google.com/f	Звук хвиль	2:00	20.09.2024	Pixabay	море, релакс	Активний

Рисунок 6.5 – Готова таблиця

Завдання

Завдання 1: Створення структури бази:

- Створіть Google Таблицю.
- Розробіть структуру бази даних для зберігання медіа-об’єктів. Обов’язкові поля:

1. ID
2. Назва
3. Тип файлу (зображення / відео / аудіо / документ)
4. Посилання на файл у Google Drive
5. Короткий опис
6. Категорія / Тема
7. Дата завантаження
8. Автор / джерело
9. Ключові слова (теги)
10. Описові метадані
11. Технічні характеристики (розширення, розмір, тривалість / роздільна здатність тощо)
12. Мова / формат

Завдання 2: Завантаження медіафайлів:

1. Створіть окрему папку у Google Drive.
2. Завантажте туди 10-15 різнотипних медіа-файлів (аудіо, зображення, відео, документи).
3. Додайте до кожного публічне посилання для перегляду.

Завдання 3: Заповнення бази даних:

1. Внесіть дані про кожен файл у таблицю.
2. У полі «Посилання» вставте публічне посилання з Google Drive.
3. У полях «Описові метадані» та «Технічні характеристики» заповніть відповідну інформацію вручну або через властивості файлу.

Завдання 4: Систематизація:

1. Додайте можливість сортування / фільтрації за категорією, типом, датою, автором тощо.

2. Налаштуйте умовне форматування (наприклад, різні кольори для типів медіа або за категорією).

Загальні рекомендації до виконання звіту

1. **Чіткість і структурованість**.

Опис кожного кроку має бути чітким і послідовним, з логічними поясненнями дій. Використовувати заголовки для кожного етапу роботи, щоб полегшити орієнтацію в звіті.

2. **Скріншоти**.

Для кожного етапу роботи додавати чіткі скріншоти з відповідними підписами. Підписи повинні пояснювати дії, що виконуються, та налаштування, що застосовуються.

3. **Аналіз і пояснення**.

У звіті повинно бути пояснення вибору кожного інструменту та налаштувань. Порівняти результати аналізу різних форматів зображень, зокрема, роздільну здатність, розмір файлу і прозорість.

4. **Оформлення**.

Використовувати чітку нумерацію та форматування для кожного розділу. Підписати всі скріншоти, що додаються до звіту.

5. **Висновки**.

Висновки мають бути логічними та чіткими, з підсумком виконаної роботи і порівнянням результатів.

Контрольні питання:

1. Що таке база даних і які її основні характеристики?

2. У чому різниця між реляційними та нереляційними базами даних?

3. Які переваги використання реляційних баз даних для зберігання мультимедіа?

4. Що таке BLOB і в яких випадках його доцільно використовувати?

5. Які є основні методи зберігання мультимедійних даних у базах даних?

6. У чому полягають переваги та недоліки зберігання файлів за посиланнями?

7. Що таке гібридний підхід до зберігання мультимедіа?

8. Які поля повинна містити таблиця для зберігання мультимедійних об’єктів?

9. Які технічні характеристики файлів доцільно фіксувати в таблиці?

11. Яким чином Google Таблиця може виконувати роль умовної бази даних?

12. Як створити публічне посилання на файл у Google Drive?

13. Як організувати фільтрацію та сортування даних у Google Таблиці?

Література: [14-16]

Практичне заняття № 7

«Автоматична обробка мультимедіа»

Мета: навчитись автоматизувати обробку мультимедійних файлів (зображень, аудіо тощо) за допомогою Google Таблиць, скриптів Google Apps Script та зовнішніх API (наприклад, CloudConvert), а також ознайомитись з принципами роботи із запитами, обробкою відповіді та інтеграцією сторонніх сервісів без потреби програмування на складному рівні.

Теоретичні відомості

7.1 Визначення

Автоматизація в інформаційних технологіях – це процес заміни ручної роботи комп'ютеризованими засобами, який дозволяє виконувати повторювані або складні завдання без безпосереднього втручання людини. В основі автоматизації лежить створення алгоритмів і систем, що здатні самостійно приймати рішення або виконувати дії за заданим сценарієм.

У контексті IT автоматизація означає, що дії, які раніше виконувались вручну (наприклад, копіювання файлів, запуск програм, перевірка статусу даних), виконуються автоматично – за допомогою скриптів, макросів, сервісів або спеціального програмного забезпечення.

Переваги автоматизованих процесів

1. **Швидкість.** Автоматизовані процеси працюють значно швидше за ручні дії. Наприклад, скрипт може обробити десятки файлів за секунди, тоді як вручну це зайняло б хвилини або години.
2. **Масштабованість.** Рішення, яке працює для одного файлу, можна легко масштабувати на сотні або тисячі файлів без значного збільшення часу чи витрат.
3. **Зниження ризику помилок.** Ручна робота завжди супроводжується ризиком помилок через неуважність або втому.

Автоматизація зменшує ці ризики, оскільки процеси виконуються точно за заданою логікою.

4. **Економія ресурсів.** Звільняє час працівників, які можуть зосередитися на більш складних або творчих завданнях, замість повторення однакових дій.

5. **Стабільність і відтворюваність.** Автоматизований процес можна запустити багаторазово з однаковим результатом. Це особливо важливо у виробничих або тестувальних середовищах.

Автоматизація в роботі з мультимедійними файлами є важливим аспектом сучасної обробки даних, особливо коли йдеться про великі обсяги інформації. Типові завдання, які потребують автоматизації, включають конвертацію форматів, стиснення, зміну розмірів та оптимізацію якості. У всіх цих випадках автоматизація дозволяє виконувати завдання без втручання людини, зберігаючи при цьому високий рівень точності та ефективності.

Один із найпоширеніших процесів – це **конвертація форматів** файлів. Наприклад, зображення можуть бути перетворені з формату PNG в JPEG або зображення у форматі TIFF можуть бути конвертовані в більш стиснуті формати, такі як WebP для веб-використання. Відео також часто потребують конвертації, щоб змінити формат на більш підходящий для конкретного пристрою чи платформи. Ці операції можуть займати багато часу, якщо виконувати їх вручну для великої кількості файлів, тому автоматизація цього процесу дозволяє значно скоротити час обробки.

Іншим важливим завданням є **стискання файлів**, що особливо актуально для веб-ресурсів, де швидкість завантаження зображень і відео має критичне значення. Автоматизація стиснення зображень або відео дозволяє зберігати якість при значному зменшенні розміру файлів, що полегшує їх завантаження і покращує досвід користувачів. Наприклад, для зображень може бути автоматично застосовано стиснення, яке не призводить до помітної втрати якості, що особливо важливо для дизайну веб-сайтів.

Також важливою є задача **зміни розмірів файлів**, особливо коли потрібно адаптувати зображення чи відео до конкретних вимог. Наприклад, для соціальних мереж або мобільних додатків зображення мають бути певного розміру або формату. За допомогою автоматизації можна налаштувати процес змінювання розмірів без необхідності вручну обробляти кожне зображення, що значно знижує час на виконання завдання.

Проблеми ручної обробки великої кількості мультимедійних файлів є очевидними. Один із найбільших недоліків – це **часозатратність**. Коли мова йде про обробку десятків або сотень файлів, кожен етап процесу, будь то конвертація, стиснення чи зміна розмірів, стає важким і займає багато часу. Крім того, ручне виконання таких операцій підвищує ймовірність помилок. Люди можуть не помітити неточності в налаштуваннях, що призводить до непередбачуваних результатів. Потрібно постійно контролювати кожен крок, що значно збільшує обсяг роботи і навантаження.

Саме тому автоматизація є вирішенням цих проблем. Автоматизація дозволяє здійснювати **масове конвертування** файлів, обробляючи їх одночасно без необхідності вручну виконувати кожну операцію. Це дозволяє значно зменшити час на обробку великої кількості медіафайлів, а також знижує ймовірність помилок. Окрім цього, автоматизація забезпечує **масштабованість** процесів. Якщо раніше потрібно було виконувати обробку файлів вручну або за допомогою програмного забезпечення з обмеженими можливостями, то сучасні автоматизовані системи дозволяють обробляти тисячі файлів одночасно, завдяки використанню хмарних сервісів або високопродуктивних серверів.

Автоматизація також дозволяє легко **оптимізувати файли** для різних платформ, що важливо для медіа-ресурсів, таких як вебсайти, соціальні мережі та мобільні додатки. Налаштування автоматизованих процесів дає змогу зберігати високий рівень якості та одночасно знижувати розмір файлів, що зменшує навантаження на сервери та покращує швидкість завантаження сторінок.

Завдяки інтеграції з такими сервісами, як **CloudConvert** або **Imgix**, можна автоматизувати процеси через API, що значно спрощує роботу з медіафайлами, забезпечує їх обробку безпосередньо з хмарних сховищ, таких як Google Drive. Це дозволяє здійснювати **масову обробку** файлів без необхідності вручну завантажувати їх у програмне забезпечення для обробки, що скорочує час і трудовитрати.

Отже, автоматизація в роботі з мультимедіа не тільки значно покращує ефективність обробки файлів, але й дозволяє знизити ризик помилок, забезпечити стабільність і масштабованість процесів. Це особливо важливо для проектів, що вимагають обробки великої кількості медіафайлів у короткий час.

Хмарні сервіси для обробки медіа

Сучасні хмарні сервіси для обробки мультимедійних файлів є потужними інструментами для автоматизації багатьох процесів, таких як конвертація, стискання, зміна розмірів, оптимізація та інші операції з медіа. Використання хмарних API надає значні переваги, оскільки дозволяє швидко та ефективно виконувати складні операції без необхідності встановлення додаткового програмного забезпечення або спеціалізованого обладнання.

Переваги використання хмарних API

Однією з головних переваг є **підтримка багатьох форматів без необхідності встановлення програм**. Хмарні сервіси, такі як CloudConvert, дозволяють працювати з різноманітними форматами медіафайлів (зображення, аудіо, відео, документи тощо) без необхідності завантажувати і налаштовувати спеціалізовані програми на комп'ютері. Це особливо зручно для тих, хто працює з великими обсягами файлів або регулярно має справу з різними форматами, оскільки зберігається велика кількість ресурсів на локальних машинах.

Іншою важливою перевагою є **висока швидкість та доступність**. Хмарні сервіси забезпечують швидку обробку файлів завдяки використанню потужних серверів, що дозволяє обробляти великі обсяги даних у короткий

час. Крім того, вони доступні 24/7, що дозволяє працювати з медіафайлами в будь-який час і з будь-якого місця, де є доступ до Інтернету.

Ще одним значним плюсом є **інтеграція з іншими інструментами**, такими як Google Sheets, Google Drive та іншими хмарними сервісами. Це дозволяє автоматизувати обробку медіафайлів прямо в рамках робочих процесів, що використовують ці сервіси. Наприклад, за допомогою API можна автоматично отримувати медіафайли з Google Drive, обробляти їх і зберігати результати в тому ж хмарному сховищі або навіть створювати звіти в Google Sheets про виконану обробку.

Приклад API: CloudConvert

CloudConvert – це хмарний сервіс для конвертації файлів, який підтримує більше 200 форматів файлів. Він надає API, що дозволяє інтегрувати функції конвертації безпосередньо в інші програми та сервіси, автоматизуючи таким чином процеси обробки файлів.

Основна ідея роботи з CloudConvert полягає в передачі медіафайлу на сервер для подальшої обробки, а потім отриманні обробленого файлу після завершення всіх операцій. Це означає, що всі складні обчислювальні процеси відбуваються на сервері CloudConvert, що значно зменшує навантаження на локальні ресурси.

Типи операцій в CloudConvert

CloudConvert надає кілька типів операцій для роботи з медіафайлами:

1. **Convert** (Конвертація) – основна операція, яка дозволяє змінювати формати медіафайлів. Наприклад, конвертувати зображення з формату PNG у JPEG або відео з формату AVI в MP4.
2. **Optimize** (Оптимізація) – ця операція включає зменшення розміру файлів, зберігаючи при цьому високу якість. Це особливо важливо для зображень і відео, що використовуються в Інтернеті.
3. **Merge** (Об'єднання) – дозволяє об'єднати кілька медіафайлів (наприклад, зображень або відео) в один файл. Це корисно при створенні мультимедійних проектів, де необхідно об'єднати кілька частин в один файл.

4. **Split** (Розбиття) – дозволяє розділяти великі файли на кілька менших частин. Наприклад, відеофайл можна розділити на окремі частини для полегшення зберігання чи поширення.

5. **Extract** (Вилучення) – ця операція дозволяє витягувати певні частини з медіафайлів, наприклад, отримувати лише аудіо з відеофайлів або окремі зображення з PDF-документів.

Модель роботи з CloudConvert

Процес роботи з CloudConvert можна розглядати в кілька етапів:

1. **Створення job** (створення завдання) – на першому етапі користувач ініціює завдання на сервері CloudConvert, де вказується, які операції потрібно виконати з медіафайлом, а також деталі для кожної операції.

2. **Перевірка статусу** – після створення завдання необхідно перевірити статус його виконання. CloudConvert надає можливість перевіряти, чи завершено обробку файлів, і чи готовий результат до завантаження.

3. **Завантаження результату** – після того як операція завершена, користувач отримує доступ до оброблених файлів, які можна завантажити або перенаправити на інший хмарний сервіс (наприклад, Google Drive) для подальшої роботи.

Завдяки таким етапам, CloudConvert дозволяє автоматизувати процеси обробки файлів без участі людини, значно скорочуючи час та зусилля на виконання рутинних завдань.

Інтеграція CloudConvert API з Google Apps Script

Google Apps Script – це хмарна платформа, яка дозволяє автоматизувати завдання в рамках Google Workspace (Google Таблиці, Google Документи, Gmail та інші сервіси Google). Це потужний інструмент, який надає можливість створювати сценарії для автоматизації різних процесів, таких як обробка даних, взаємодія з іншими сервісами через API, обробка подій та багато іншого.

Можливості Google Apps Script

1. **Зчитування даних з Google Таблиць**: Google Apps Script дозволяє легко отримувати, редагувати та маніпулювати даними в Google Таблицях. Ви можете використовувати скрипт для доступу до конкретних клітинок, стовпців або навіть весь діапазон даних.

2. **Надсилання HTTP-запитів**: Одна з головних можливостей – надсилання HTTP-запитів через метод `UrlFetchApp`. Це дозволяє інтегрувати Google Apps Script з різноманітними API, такими як CloudConvert, для виконання операцій з медіафайлами. За допомогою цієї функціональності можна взаємодіяти з зовнішніми сервісами та отримувати потрібні результати.

3. **Реакція на натискання кнопок або запуск за розкладом**: Google Apps Script підтримує створення користувацьких інтерфейсів, де можна додавати кнопки, що активують певні дії (наприклад, запуск скрипта для обробки файлів). Крім того, ви можете налаштувати запуск скрипта за розкладом, наприклад, щоб перевіряти статус обробки файлів щогодини.

Приклад сценарію

Уявімо, що в Google Таблиці зберігаються дані про мультимедійні файли, які потрібно обробити за допомогою CloudConvert. Таблиця містить стовпці для посилання на файл, типу обробки, статусу та посилання на результат. Завдяки Google Apps Script, ви можете автоматизувати цей процес за допомогою інтеграції з CloudConvert API.

1. **Зчитування даних з Google Таблиці**: Спочатку скрипт зчитує дані з таблиці. Наприклад, отримує посилання на файли та типи обробки для кожного медіафайлу.

```
function readData() {
  var sheet = SpreadsheetApp.getActiveSpreadsheet().getSheetByName("MediaFiles");
  var data = sheet.getDataRange().getValues();
  return data;
}
```

2. **Надсилання запиту до CloudConvert API**: Для кожного файлу скрипт формує запит до CloudConvert API, передаючи необхідні параметри, такі як посилання на файл і тип обробки (наприклад, конвертація чи стиснення). Запит надсилається через `UrlFetchApp.fetch`.

```
function sendRequestToCloudConvert(fileUrl, operation) {
  var apiKey = 'YOUR_CLOUDCONVERT_API_KEY';
  var url = 'https://api.cloudconvert.com/v2/convert';

  var payload = {
    "input": "url",
    "file": fileUrl,
    "operation": operation
  };

  var options = {
    'method': 'post',
    'payload': JSON.stringify(payload),
    'headers': {
      'Authorization': 'Bearer ' + apiKey,
      'Content-Type': 'application/json'
    }
  };

  var response = UrlFetchApp.fetch(url, options);
  return JSON.parse(response.getContentText());
}
```

3. **Вставлення результату у таблицю**: Після того, як CloudConvert обробить файл, скрипт вставляє посилання на результат обробки в таблицю. Це дозволяє легко відслідковувати статус обробки медіафайлів та отримувати посилання на готові файли.

```
function updateTableWithResult(fileId, resultLink) {
  var sheet = SpreadsheetApp.getActiveSpreadsheet().getSheetByName("MediaFiles");
  var data = sheet.getDataRange().getValues();

  for (var i = 0; i < data.length; i++) {
    if (data[i][0] == fileId) {  // припускаємо, що fileId знаходиться в першому стовпці
```

```
            sheet.getRange(i + 1, 6).setValue(resultLink);  // припускаємо, що результат знаходиться в 6-му стовпці
            sheet.getRange(i + 1, 5).setValue("Готово");  // оновлення статусу на "Готово"
          }
        }
      }
```

4. **Перевірка статусу обробки**: Окремо можна налаштувати скрипт для періодичної перевірки статусу обробки файлів. Наприклад, можна використовувати функцію для перевірки статусу завдання в CloudConvert за допомогою API.

```
function checkJobStatus(jobId) {
  var apiKey = 'YOUR_CLOUDCONVERT_API_KEY';
  var url = 'https://api.cloudconvert.com/v2/jobs/' + jobId;

  var options = {
    'method': 'get',
    'headers': {
      'Authorization': 'Bearer ' + apiKey
    }
  };

  var response = UrlFetchApp.fetch(url, options);
  var result = JSON.parse(response.getContentText());

  if (result.data.status === 'finished') {
    return result.data.files[0].url;  // посилання на готовий файл
  } else {
    return null;  // файл ще не готовий
  }
}
```

Як це працює в реальному часі:

- Ви натискаєте кнопку в Google Таблиці (яка викликає скрипт).
- Скрипт зчитує дані з таблиці, створює запит до CloudConvert API для обробки медіафайлів.
- Після обробки файлів результат (посилання на готові файли) вставляється назад у таблицю.

- Якщо обробка займає деякий час, скрипт може періодично перевіряти статус завдання в CloudConvert і оновлювати таблицю, коли файл буде готовий.

Завдяки такій інтеграції можна легко автоматизувати процес обробки медіафайлів і значно знизити час та зусилля, які витрачаються на виконання рутинних завдань.

7.2 Приклад виконання

Отримання API-ключа CloudConvert:

- Перейдіть на сайт **CloudConvert**: https://cloudconvert.com.
- Клікніть на кнопку **Sign Up** (або **Log In**, якщо у вас вже є обліковий запис).
- Заповніть форму реєстрації, використовуючи електронну пошту або зареєструйтеся через Google або Facebook.
- Після реєстрації увійдіть в свій обліковий запис на CloudConvert.
- Перейдіть на сторінку **API**: https://cloudconvert.com/api.
- На цій сторінці буде написано: **create an API key** (створити ваш API-ключ). Клікніть на посилання **create an API Key** (рис.7.1).
- У відкритій сторінці клікніть на кнопку «Create New API key».
- У полі **Name** введіть щось на кшталт MediaProcessingScript.
- task.read
- task.write
- user.read
- Натисніть Create – з’явиться ваш API-ключ. СКОПІЮЙТЕ ЙОГО (рис. 7.2-7.3).

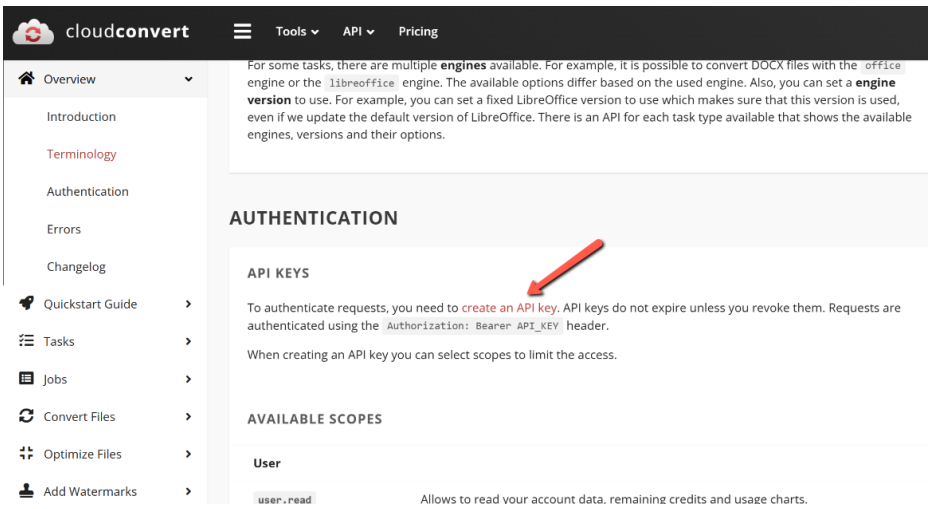


Рисунок 7.1 – Перехід на сторінку створення API-ключа

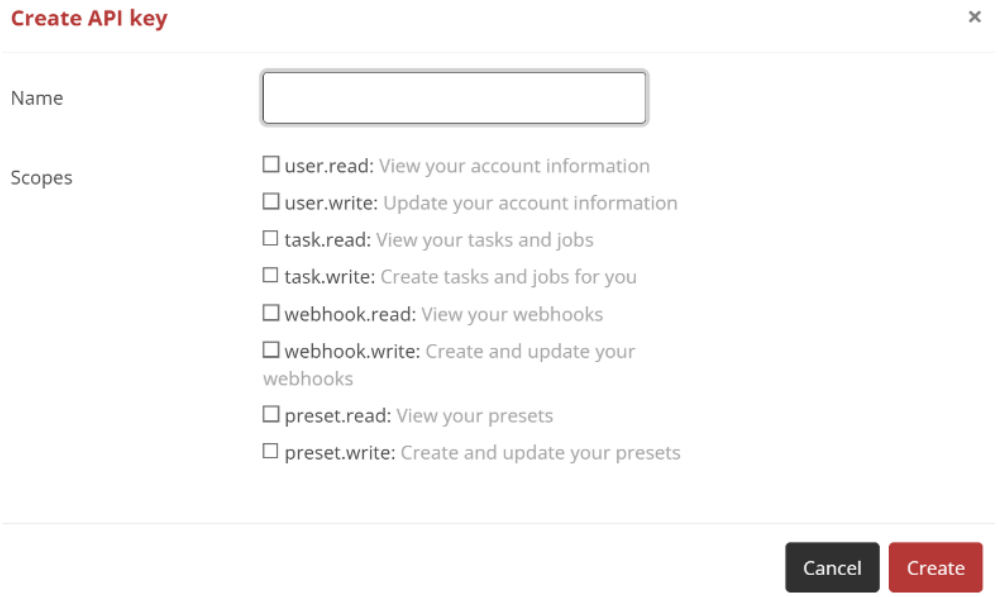


Рисунок 7.2 – Модальне вікно створення API-ключа

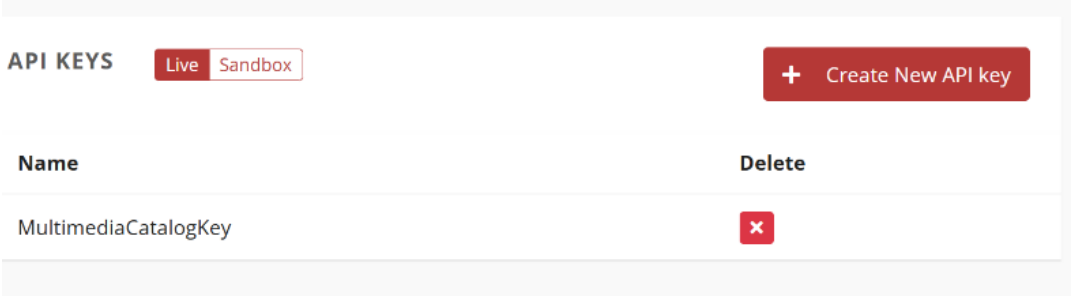


Рисунок 7.3 – Створений API-ключ

Підготовка таблиці

1. Відкрийте **Google Таблицю**. Створіть новий аркуш або використайте той, що вже є.

2. Створіть наступні заголовки колонок:

- Назва
- Посилання на файл (переконайтесь, що встановлений відкритий доступ)
- Тип обробки (compress / resize / convert)
- Стан обробки
- Посилання на результат
- *(Додатково, за бажанням: Розмір до / після)*

3. Заповніть таблицю даними для 5–10 медіафайлів (зображення, аудіо). Посилання мають бути **публічними** (Google Drive або інтернет).

Важливо:

Щоб **URL дійсно працював**, потрібно замінити його з формату:

https://drive.google.com/file/d/1rOqybanTq57lNrr-qR56fP2L5k0-xjaQ/view?usp=drive_link

на такий:

https://drive.google.com/uc?export=download&id=1rOqybanTq57lNrr-qR56fP2L5k0-xjaQ

Тобто треба витягнути ID і вставити у такий шаблон:

https://drive.google.com/uc?export=download&id=FILE_ID

Або скористатись скриптом:

```
function convertDriveLinks() {
  const sheet = SpreadsheetApp.getActiveSpreadsheet().getActiveSheet();
  const data = sheet.getDataRange().getValues();

  for (let i = 1; i < data.length; i++) {
    let originalLink = data[i][3]; // Колонка D (індекс 3)

    // Перевірка, чи це посилання на Google Drive у старому форматі
    const match = originalLink.match(/https:\/\/drive\.google\.com\/file\/d\/([^/]+)\/view/);
```

```
      if (match && match[1]) {
        const fileId = match[1];
        const directLink = `https://drive.google.com/uc?export=download&id=${fileId}`;
        sheet.getRange(i + 1, 4).setValue(directLink); // записуємо нове посилання
      }
    }

    SpreadsheetApp.getUi().alert("Посилання оновлено!");
}
```

Скрипт треба прив’язати до кнопки:

Як додати кнопку в Google Таблиці

Крок 1. Створіть малюнок (кнопку)

1. У Google Таблиці перейдіть на потрібний аркуш.
2. Зверху у меню натисни: **Вставка → Малюнок → Новий**.
3. У вікні редактора: намалюйте просту кнопку (наприклад, прямокутник із написом "Оновити посилання").
4. Натиснити **Зберегти та закрити** – кнопка з'явиться в таблиці.

Крок 2. Призначте скрипт

1. Клікніть **один раз на кнопку** (вона виділиться).
2. Натисніть на три крапки у верхньому правому куті кнопки (або правою кнопкою миші).
3. Оберіть **Призначити сценарій** (*Assign script*).
4. У вікні введи точну назву функції без дужок:

convertDriveLinks

5. Натисніть **OK**.

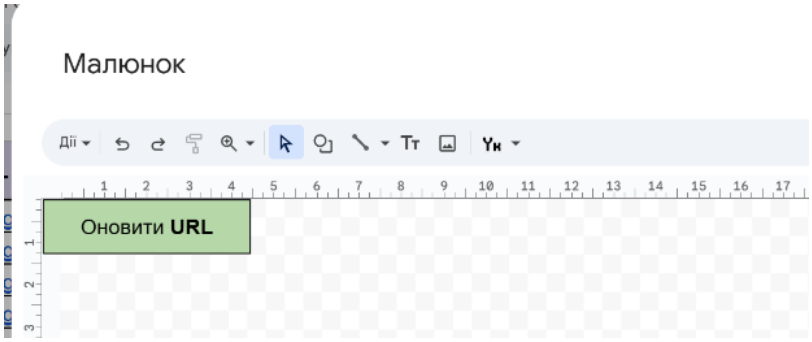


Рисунок 7.4 – Малювання кнопки

Написання Google Apps Script

1. У Google Таблиці відкрийте:

`Розширення → Apps Script`

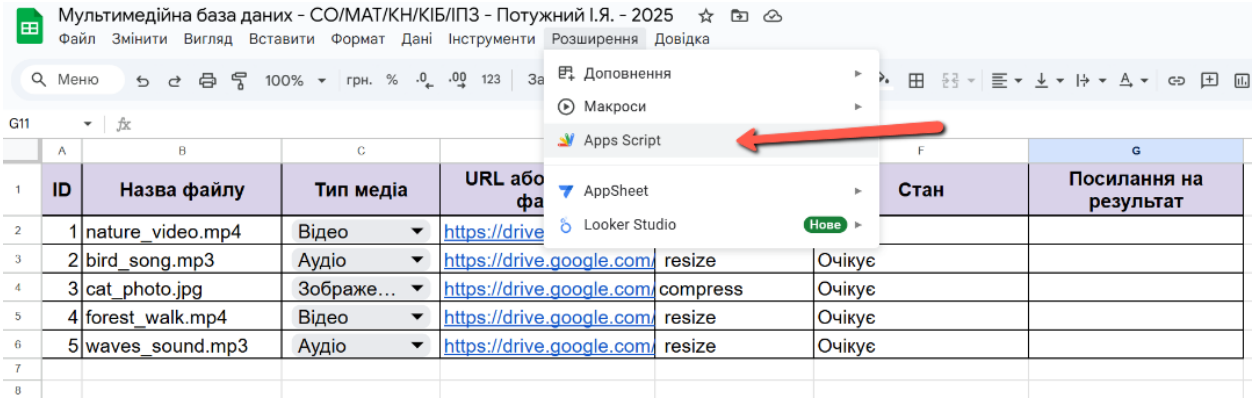


Рисунок 7.5 – Перехід на розширення Apps Script

2. Замініть вміст на наступний код:

```
function processMediaV2() {
  const apiKey = 'Вставьте свій ключ';
  const sheet =
SpreadsheetApp.getActiveSpreadsheet().getActiveSheet();
  const data = sheet.getDataRange().getValues();

  for (let i = 1; i < data.length; i++) {
    const status = data[i][5];
    if (status !== 'Очікує') continue;

    const url = data[i][3];
    const fileName = data[i][1];
    const processType = data[i][4];

    const jobPayload = {
      "tasks": {
        "import": {
          "operation": "import/url",
          "url": url
        },
        "convert": {
          "operation": "convert",
```

```
          "input": "import",
          "output_format": "jpg",  // змінити залежно від
обробки
          "engine": "imagemagick",
          "engine_version": "7.0.11",
          "options": processType === "compress" ? {
"quality": 60 } : {}
        },
        "export": {
          "operation": "export/url",
          "input": "convert"
        }
      }
    };

    try {
      const response =
UrlFetchApp.fetch('https://api.cloudconvert.com/v2/jobs',
{
        method: 'post',
        contentType: 'application/json',
        headers: {
          Authorization: 'Bearer ' + apiKey
        },
        payload: JSON.stringify(jobPayload)
      });

      const job = JSON.parse(response.getContentText());
      const exportTask = job.data.tasks.find(t => t.name
=== "export");

      // CloudConvert виконує завдання асинхронно, треба
чекати завершення (або перевіряти пізніше)
      sheet.getRange(i + 1, 6).setValue("У процесі");
      sheet.getRange(i + 1, 7).setValue("Job ID: " +
job.data.id);

    } catch (e) {
      sheet.getRange(i + 1, 6).setValue("Помилка");
      sheet.getRange(i + 1, 7).setValue(e.message);
    }
  }
}
```

Запуск скрипта

1. Збережіть скрипт.
2. Натисніть **"Запустити"**
3. Дайте дозвіл (уперше система попросить авторизацію).
4. Після виконання: у стовпці “Стан” буде **У процесі** (з Job ID в посиланні на результат) або **Помилка.**

Для перевірки статусу створіть новий файл скрипту:

```
function checkJobStatus() {
  const apiKey ='Вставьте свій ключ';  // API ключ
  const sheet =
SpreadsheetApp.getActiveSpreadsheet().getActiveSheet();
  const data = sheet.getDataRange().getValues();

  for (let i = 1; i < data.length; i++) {
    const status = data[i][5];
    const resultCell = data[i][6];

    if (status !== "У процесі" || !resultCell.includes("Job
ID:")) continue;

    const jobId = resultCell.replace("Job ID: ", "").trim();

    try {
      const response =
UrlFetchApp.fetch(`https://api.cloudconvert.com/v2/jobs/${jo
bId}?include=tasks`, {
        method: 'get',
        headers: {
          Authorization: 'Bearer ' + apiKey
        }
      });

      const jobData = JSON.parse(response.getContentText());
      const job = jobData.data;

      if (job.status === "finished") {
        // Шукаємо експортне посилання
        const exportTask = job.tasks.find(t => t.name ===
"export" || t.operation === "export/url");
        const fileUrl = exportTask.result?.files?.[0]?.url;
```

```
        if (fileUrl) {
          sheet.getRange(i + 1, 6).setValue("Готово");
          sheet.getRange(i + 1, 7).setValue(fileUrl);
        } else {
          sheet.getRange(i + 1, 6).setValue("Помилка");
          sheet.getRange(i + 1, 7).setValue("Файл не
знайдено");
        }

      } else if (job.status === "error") {
      sheet.getRange(i + 1, 6).setValue("Помилка");
      sheet.getRange(i + 1, 7).setValue(JSON.stringify(job,
null, 2));
      }

    } catch (e) {
      sheet.getRange(i + 1, 6).setValue("Помилка");
      sheet.getRange(i + 1, 7).setValue(e.message);
    }
  }
}
```

Завдання

Завдання 1: Підготовка структури даних

1. **Створіть Google Таблицю** або додайте новий аркуш до таблиці з попередньої практичної роботи.

2. **Додайте такі стовпці** (назви мають бути саме такими, для коректної роботи скриптів):

- ID
- Назва файлу
- Тип медіа (наприклад: image, audio, video)
- URL або шлях до файлу (посилання на файл у Google Drive або з інтернету)
- Тип обробки (наприклад: convert, compress)
- Стан (наприклад: Очікує, У процесі, Готово, Помилка)
- Посилання на результат

3. **Заповніть таблицю** даними про 5–10 мультимедійних файлів (наприклад, зображення з відкритим доступом до перегляду через URL). Для кожного вкажіть:

- Назву,
- Тип (наприклад, image),
- Посилання на файл,
- Тип обробки (наприклад, convert),
- У стовпці Стан введіть: Очікує.

Завдання 2: Ознайомлення з API

1. **Перейдіть на сайт CloudConvert API**.

2. Ознайомтесь з особливостями конверсії медіафайлів, зокрема:

- які формати підтримуються (наприклад: .png, .jpg, .mp3, .mp4, .pdf),

- які параметри можна використовувати (output_format, engine, quality тощо),
- які конверсії не підтримуються напряму (наприклад, .html → .jpg – не працює).

Завдання 3: Написання скрипту

1. **Напишіть три Google Apps Script-функції:**

3.1 Кнопка запуску:

Створіть кнопку в Google Sheets і зв’яжіть її з функцією runProcessingFromButton():

```
function runProcessingFromButton() {
  processMedia(); // Основна функція обробки
}
```

3.2 Основна обробка:

- Напишіть функцію processMedia(), яка:
  - Зчитує записи зі статусом "Очікує",
  - Перевіряє тип медіа й тип обробки,
  - Формує й надсилає запит до CloudConvert API (через UrlFetchApp.fetch()),
  - Зберігає ID завдання (jobId) у таблицю (можна створити прихований стовпець),
  - Оновлює статус на "У процесі".

3.3 Перевірка статусу:

Додайте окрему функцію checkJobStatus(), яка:

- Перевіряє, чи завершено обробку,
- Якщо успішно – додає посилання на результат і змінює статус на "Готово",
- Якщо сталася помилка – оновлює статус на "Помилка".

Завдання 4: Тестування

1. **Запустіть скрипт через кнопку.**

2. Перевірте:

- Чи змінився статус на "У процесі",
- Чи з’явилось посилання на результат після деякого часу (після запуску checkJobStatus()).

3. Додайте до таблиці нові стовпці: Розмір до, Розмір після – можете обчислити за допомогою скрипту або вручну.

4. Зафіксуйте зміни формату, розміру та якості файлу.

Загальні рекомендації до виконання звіту

1. **Чіткість і структурованість**.

Опис кожного кроку має бути чітким і послідовним, з логічними поясненнями дій. Використовувати заголовки для кожного етапу роботи, щоб полегшити орієнтацію в звіті.

2. **Скріншоти**.

Для кожного етапу роботи додавати чіткі скріншоти з відповідними підписами. Підписи повинні пояснювати дії, що виконуються, та налаштування, що застосовуються.

3. **Аналіз і пояснення**.

У звіті повинно бути пояснення вибору кожного інструменту та налаштувань. Порівняти результати аналізу різних форматів зображень, зокрема, роздільну здатність, розмір файлу і прозорість.

4. **Оформлення**.

Використовувати чітку нумерацію та форматування для кожного розділу. Підписати всі скріншоти, що додаються до звіту.

5. **Висновки**.

Висновки мають бути логічними та чіткими, з підсумком виконаної роботи і порівнянням результатів.

Контрольні питання:

1. Що таке метадані, і яку роль вони відіграють у мультимедіа?
2. Які існують типи метаданих? Наведіть приклади.
3. У чому різниця між описовими, адміністративними, технічними та системними метаданими?
4. Які стандартні формати метаданих використовуються у мультимедійних файлах?
5. Для чого використовуються формати EXIF, IPTC, XMP?
6. Чим відрізняються структуровані та неструктуровані метадані?
7. Як забезпечується збереження метаданих при редагуванні файлів?

8. Чи можна змінювати метадані без зміни самого мультимедійного файлу? Якщо так, то як?

9. Як переглянути метадані зображення у Windows без сторонніх програм?

10. Що відбудеться, якщо спробувати записати нове значення у вже існуюче поле метаданих?

11. Як додати кілька метаданих одночасно у файл за допомогою FFmpeg?

Література: [14-17]

Практичне заняття № 8

«Створення мультимедійного каталогу»

Мета: набуття практичних навичок створення мультимедійного каталогу засобами Google Sites, ознайомлення з інтеграцією Google Таблиць як джерела даних, формування уявлення про побудову веб-інтерфейсу для представлення мультимедійного контенту, опрацювання принципів організації структури сайту, вбудовування медіаоб’єктів та налаштування доступу до інформації, розвиток умінь структурувати й оформлювати мультимедійні дані для публічного перегляду в зручному форматі.

Теоретичні відомості

8.1 Визначення

Мультимедійний каталог – це організована система зберігання, представлення та навігації по медіафайлах, що може включати зображення, відео, аудіо, а також супровідну текстову інформацію. Такий каталог зазвичай створюється з метою впорядкування великого обсягу мультимедійних матеріалів і забезпечення швидкого доступу до них для кінцевих користувачів. У цифровому середовищі мультимедійний каталог реалізується у вигляді вебінтерфейсу, бази даних або окремого вебресурсу, який дозволяє переглядати, шукати та фільтрувати об’єкти за певними критеріями.

Каталог, у широкому розумінні, – це структурований список або база, яка містить інформацію про об’єкти певного типу. В контексті мультимедіа, каталог є не просто зібранням файлів, а цілісною системою, де кожен медіаоб’єкт супроводжується метаданими: назвою, описом, категорією, датою створення, типом файлу тощо. Це дозволяє реалізовувати ефективний пошук, фільтрацію, групування матеріалів і полегшує орієнтацію в контенті.

Призначення мультимедійних каталогів у цифровому середовищі полягає у зручному управлінні та розповсюдженні цифрового контенту. У навчальних закладах це може бути каталог проєктів студентів або освітніх

відео. У творчому середовищі – портфоліо з фотографіями, музикою чи відеороботами. В адміністративних або архівних системах – електронні бази зображень, відео та аудіо для зберігання та обміну інформацією.

Прикладами використання мультимедійних каталогів є:

1) **Цифрові архіви** – публічні чи внутрішні ресурси для зберігання історичних, культурних або освітніх матеріалів у медіаформаті.

2) **Онлайн-портфоліо** – персональні сайти дизайнерів, фотографів, відеомейкерів, де представлено їхні роботи.

3) **Навчальні бази** – каталоги відеолекцій, презентацій, лабораторних занять або інтерактивних підручників, доступних через освітні платформи або Google Сайти.

4) **Каталоги продуктів** – сайти з колекціями товарів, кожен з яких має фото, відеоогляд і опис.

Таким чином, мультимедійний каталог – це інструмент, який поєднує зручність структурування інформації та можливість візуального чи аудіовізуального представлення даних, що робить його незамінним у багатьох сферах цифрової діяльності.

У сучасному цифровому середовищі існує велика кількість платформ, які дозволяють створювати мультимедійні каталоги. Ці інструменти різняться за рівнем складності, функціональністю та вимогами до технічної підготовки користувача. Найбільш поширені підходи – це використання CMS (систем управління контентом), онлайн-конструкторів сайтів і хмарних сервісів Google.

Системи управління контентом (CMS) – це спеціалізовані платформи, призначені для побудови та керування вебсайтами з великим обсягом інформації. Вони дозволяють створювати структуру сайту, додавати сторінки, керувати файлами, здійснювати фільтрацію та пошук контенту. Прикладами є WordPress, Joomla та Drupal. Такі системи зазвичай потребують базових знань веброзробки або щонайменше навичок роботи з адміністративними панелями. CMS-платформи пропонують велику кількість

тем оформлення, плагінів і розширень, що дозволяє адаптувати сайт під конкретні потреби користувача.

Онлайн-конструктори сайтів (наприклад, Wix, Webflow, Tilda, Squarespace) надають зручний інтерфейс для створення сайтів за принципом "перетягни і встав". Вони орієнтовані на користувачів без досвіду програмування, тому основні дії виконуються за допомогою графічного редактора. Такі платформи дозволяють швидко зібрати презентаційний сайт або портфоліо, розмістити мультимедійний контент (зображення, відео, інтерактивні елементи), а також забезпечують базову SEO-оптимізацію. Конструктори також часто пропонують інтеграцію з хмарними сховищами та соціальними мережами, що робить їх зручними для створення каталогів навчальних або творчих проєктів.

Google-сервіси, зокрема Google Sites, Google Sheets і Google Drive, представляють безкоштовне рішення для створення простих вебсайтів із мультимедійним наповненням. Google Sites дозволяє створити сайт без знання HTML або CSS, використовуючи готові блоки для додавання тексту, зображень, відео, таблиць та інших елементів. Google Таблиці можуть слугувати джерелом структурованої інформації, яка інтегрується безпосередньо на сторінки сайту. Google Диск, у свою чергу, забезпечує зберігання та спільний доступ до медіафайлів. Такий підхід особливо зручний у навчальному процесі, для внутрішніх проєктів, студентських портфоліо чи демонстрацій контенту, коли потрібна швидка реалізація без складних налаштувань.

Таким чином, вибір платформи залежить від цілей каталогу, технічних вимог та досвіду розробника. Для складних і динамічних проєктів краще використовувати CMS або професійні конструктори, а для простих, швидких рішень – хмарні сервіси Google.

8.1.1. Переваги використання Google Sites як безкодового рішення

Google Sites – це потужний інструмент для створення вебсайтів, який не потребує знань програмування. Завдяки своєму простому інтерфейсу та

інтеграції з іншими продуктами Google, він є зручним рішенням для тих, хто хоче швидко створити мультимедійний каталог без складних технічних налаштувань.

1. Простота використання

Google Sites пропонує інтуїтивно зрозумілий інтерфейс, що дозволяє навіть користувачам без технічного досвіду створювати вебсайти. Інтерфейс базується на принципі "перетягни і встав", що дозволяє додавати текстові блоки, зображення, відео, таблиці, посилання та інші елементи всього за кілька кліків. Користувачам не потрібно володіти знаннями HTML, CSS чи JavaScript, що значно спрощує процес створення сайту.

2. Інтеграція з іншими продуктами Google

Однією з найбільших переваг Google Sites є його інтеграція з іншими інструментами Google, такими як Google Sheets, Google Drive, Google Docs, YouTube та іншими. Це дозволяє без зусиль вставляти на сайт документи, таблиці, відео та інші мультимедійні елементи, що дуже зручно для створення мультимедійних каталогів. Наприклад, дані з Google Таблиць можна вбудовувати у вигляді інтерактивних таблиць, що дозволяє користувачам легко сортувати і фільтрувати інформацію.

3. Безкоштовність і доступність

Google Sites є безкоштовним інструментом для створення сайтів, що робить його доступним для всіх користувачів з акаунтом Google. Для початку роботи не потрібно здійснювати додаткові витрати на хостинг чи доменне ім’я. Користувачі отримують значний обсяг вільного простору на Google Диску для зберігання медіафайлів, що також знижує витрати на хмарні сховища.

4. Спільна робота і доступність

Як і інші сервіси Google, Google Sites дозволяє налаштовувати доступ до сайту та редагувати його спільно з іншими користувачами. Це дозволяє кільком людям одночасно працювати над створенням та оновленням сайту. Крім того, користувачі можуть налаштувати рівні доступу для перегляду або редагування сайту, що важливо для командних проєктів або освітніх ініціатив.

5. Адаптивний дизайн

Google Sites автоматично забезпечує адаптивний дизайн, що дозволяє сайту коректно відображатися на різних пристроях: комп'ютерах, планшетах і мобільних телефонах. Це особливо важливо для мультимедійних каталогів, де користувачі можуть переглядати медіафайли на різних платформах, не переживаючи про проблему сумісності.

6. Безпека та підтримка

Google забезпечує високий рівень безпеки для своїх продуктів, тому Google Sites також гарантує захист даних. Користувачі можуть бути впевнені, що їхній сайт буде захищений від несанкціонованого доступу. Крім того, Google надає користувачам технічну підтримку, що додає додатковий рівень безпеки та зручності.

Таким чином, Google Sites є ідеальним вибором для тих, хто хоче створити мультимедійний каталог швидко та без складних технічних вимог. Це зручне, безкоштовне та гнучке рішення для людей з будь-яким рівнем підготовки.

8.1.2 Обмеження безпрограмного підходу

Попри численні переваги безкодового підходу до створення мультимедійних каталогів, існують і певні обмеження, які варто враховувати. Хоча інструменти, такі як Google Sites, є простими у використанні і дозволяють швидко створювати базові вебсайти, вони не завжди підходять для складних або спеціалізованих проєктів.

1. Обмежена функціональність і кастомізація

Одним із головних обмежень безкодового підходу є обмеження функціональності та кастомізації. Для створення мультимедійного каталогу за допомогою Google Sites чи подібних платформ, користувачі обмежені стандартними шаблонами та елементами дизайну. Це може бути проблемою, якщо потрібно реалізувати специфічні функції, які не підтримуються безкодовиими платформами. Наприклад, складні інтерактивні елементи,

спеціалізовані фільтри для пошуку чи складні системи сортування даних можуть вимагати більш гнучкого підходу, який передбачає програмування.

2. Обмежений контроль над дизайном

Інтерфейси безкодових платформ зазвичай мають обмежену можливість для повного налаштування дизайну. Хоча Google Sites дозволяє змінювати стилі і макети, можливості для детальної налаштування вигляду сайту можуть бути обмеженими порівняно з платформами, які дозволяють писати власний код. Це може стати проблемою для користувачів, які бажають створити унікальний або брендований інтерфейс для свого мультимедійного каталогу.

3. Проблеми з масштабуванням

Для малих проєктів Google Sites і подібні інструменти є чудовим вибором, але для великих проєктів з величезною кількістю даних або високими вимогами до масштабованості, безкодова платформа може не бути достатньо ефективною. Наприклад, при створенні мультимедійного каталогу з великою кількістю медіафайлів та необхідністю їх швидкої обробки або інтеграції з іншими базами даних, без програмування можуть виникнути проблеми з продуктивністю або складністю керування великими обсягами даних.

4. Обмежені можливості для інтеграцій

Безкодові платформи часто мають обмежену кількість вбудованих інтеграцій з іншими сторонніми сервісами або API. Хоча Google Sites інтегрується з іншими продуктами Google, для більш складних інтеграцій з іншими сторонніми інструментами (наприклад, спеціалізованими базами даних, медіасервісами або кастомізованими інтерфейсами) може знадобитися програмування або використання сторонніх API, чого неможливо досягти без використання коду.

5. Безпека та конфіденційність

Використання безкодових платформ для створення мультимедійних каталогів може також мати обмеження в аспекті безпеки та конфіденційності. Вебсайти, створені на таких платформах, можуть бути уразливими до атак,

якщо не налаштувати відповідний рівень доступу чи безпеки. Користувачі, які працюють із чутливими даними або приватними медіафайлами, можуть стикнутися з проблемами при спробі налаштувати належний захист інформації.

Таким чином, безкодові платформи є чудовим варіантом для створення базових мультимедійних каталогів, особливо для невеликих проєктів, де простота і швидкість є головними перевагами. Однак для складних, масштабованих і високонавантажених проєктів, можливо, доведеться звернутися до більш потужних інструментів, що потребують програмування та більш гнучкого підходу до дизайну і функціональності.

8.2 Приклад виконання

Крок 1. Створення Google Сайту

1. Відкрити сервіс Google Sites.
2. Натиснути "+" (Створити новий сайт/Порожній сайт).
3. У верхній частині сторінки задати заголовок сайту, наприклад **«Мультимедійний каталог»** або іншу назву відповідно до теми.
4. У верхньому банері можна змінити фон (наприклад, поставити кольорову текстуру або тематичне фото).
5. За бажанням – вставити логотип у верхній лівий кут (через **"Дизайн теми" → Логотип"**).

Порада: У розділі «Тема» (праворуч) обрати естетичне оформлення – наприклад, тему "Прості" або "Мінімалістичні".

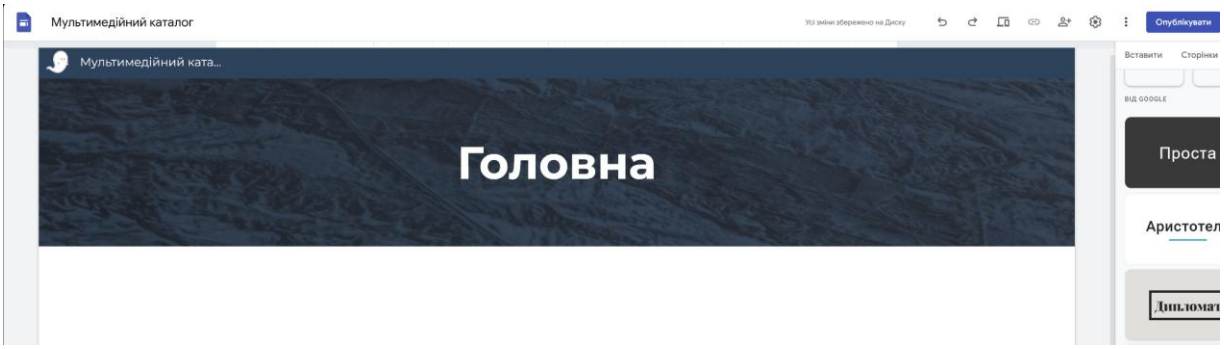


Рисунок 8.1 – Головна сторінка створеного сайту

Крок 2. Створення структури сайту

1. Перейти до вкладки **"Сторінки"** у правій панелі.
2. Натиснути кнопку "+" внизу і створити нові сторінки:

- **Головна** – залишити як стартову сторінку. Сюди додати коротке вітання, назву проєкту та зображення або логотип.
- **Каталог** – сторінка, де буде вбудована таблиця з мультимедіа.
- **Пошук** – створити сторінку з інструкцією, як шукати потрібні файли за допомогою фільтрів у таблиці.
- **Про нас** – сторінка з описом автора проєкту, його мети або команди.

Порада: Можна перетягувати сторінки в меню для зміни порядку їх появи на панелі навігації.

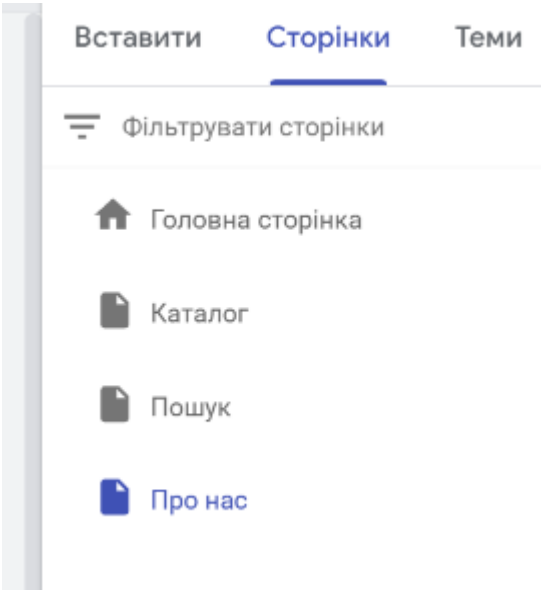


Рисунок 8.2 – Структура сайту з кількома сторінками

Крок 3. Підготовка Google Таблиці та її вставка

1. Створити Google Таблицю, у якій буде база даних мультимедійних об'єктів.

2. У таблиці рекомендується використовувати такі стовпці (можна взяти таблицю з 6 практичної роботи):

1) ID
2) Назва
3) Тип медіа (зображення / відео / аудіо)
4) Категорія
5) Короткий опис
6) Посилання на файл (Google Drive або інше джерело)

3. Надати таблиці доступ для перегляду:

Файл → Налаштування доступу → Усі користувачі з посиланням можуть переглядати

4. У Google Сайті перейти на сторінку **Каталог** → натиснути **"Вставити → Таблиця"** → обрати потрібну таблицю зі списку.

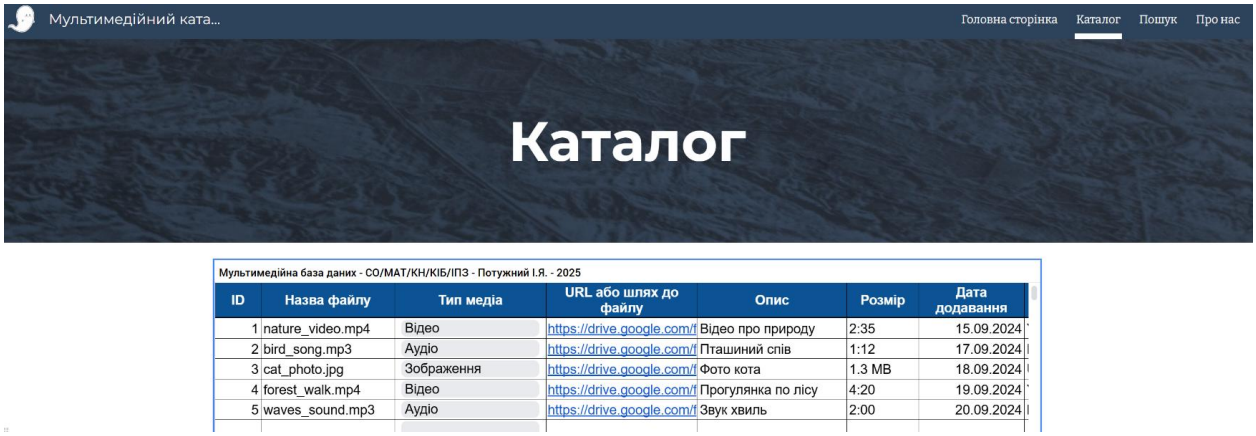


Рисунок 8.3 – Вбудована таблиця з мультимедійними файлами

Крок 4. Додавання мультимедійних блоків

На сторінці **Головна** або **Каталог** додати приклади медіа вручну:

Для **зображень**:

- Натиснути **"Вставити → Зображення"**
- Обрати з Диску або завантажити файл

Для **відео**:

- Натиснути **"Вставити → YouTube"** або **"З Диску"**, вказати посилання на відео

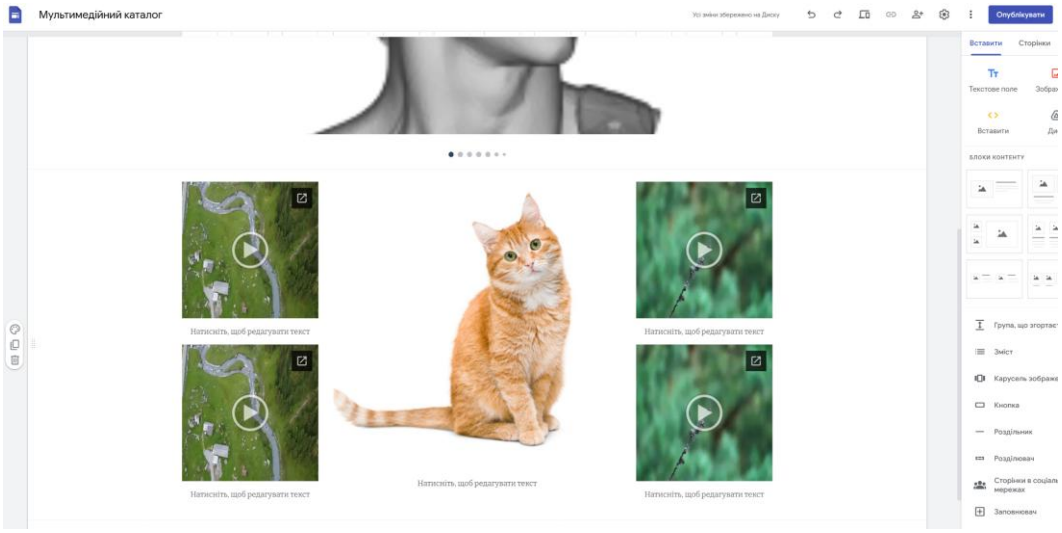


Рисунок 8.4 – Медіа-блоки на сайті

Крок 5. Публікація сайту

1. Натиснути кнопку **"Опублікувати"** у верхньому меню.
2. У полі "Веб-адреса" вказати бажану URL-адресу (латиницею).
3. У розділі **"Кому надати доступ"** обрати **"Усі користувачі з посиланням"**.
4. Натиснути **"Опублікувати"**.

Порада: після публікації натиснути на стрілочку поруч із кнопкою – вибрати "Переглянути сайт" і переконатися, що все працює.

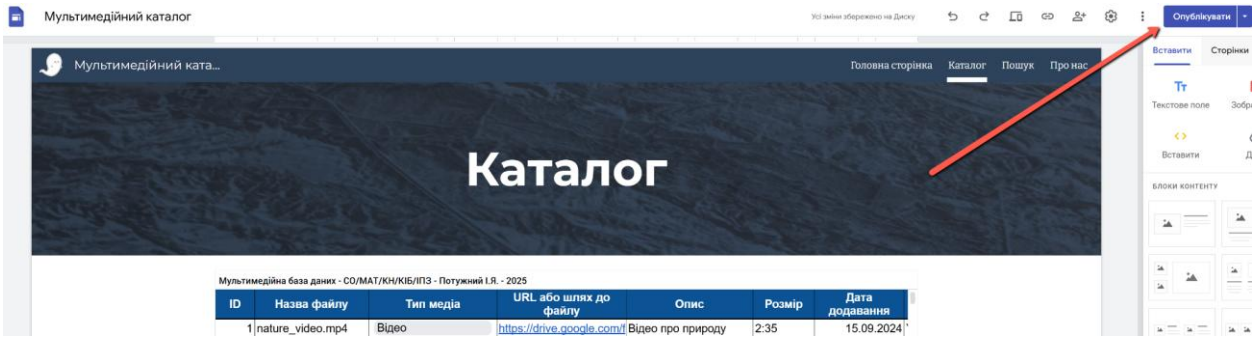


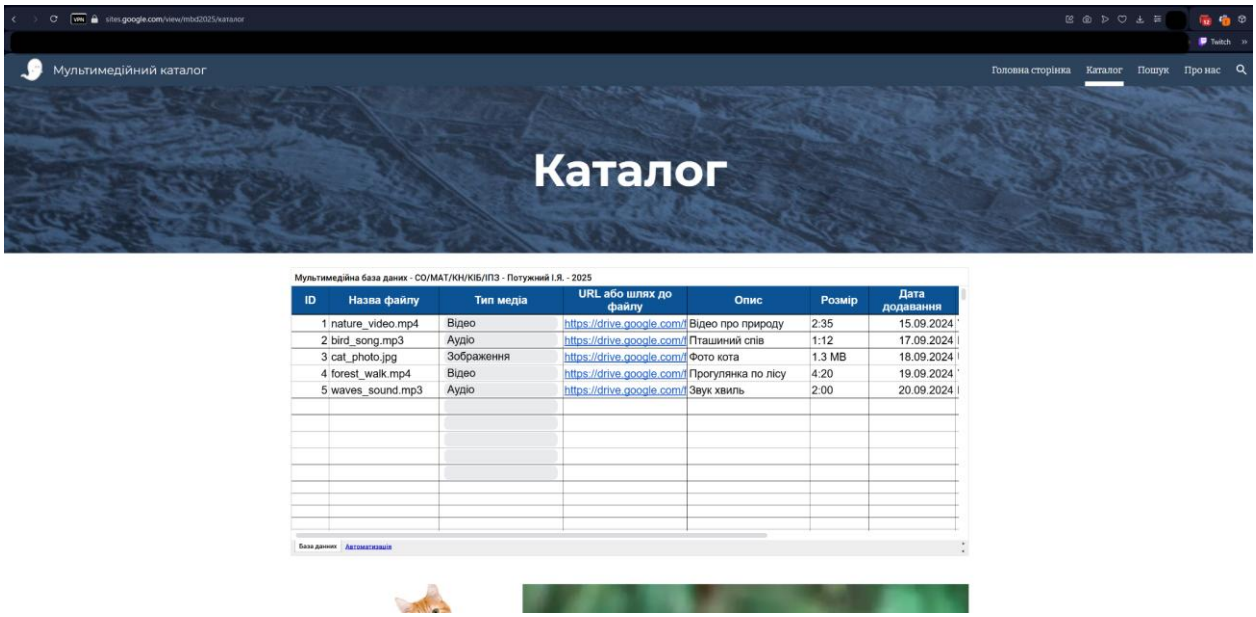
Рисунок 8.5 – Публікація сайту

Крок 6. Перевірка сайту

1. Перейти за посиланням на опублікований сайт.
2. Протестувати:
   - Роботу навігаційного меню.

- Перехід між сторінками.
- Роботу фільтрів у таблиці.
- Перегляд фото, відео, аудіо.

3. За потреби – повернутись до режиму редагування і внести правки.



Мультимедійна база даних - СО/МАТ/КН/КІБ/ІПЗ - Потужний І.Я. - 2025

ID	Назва файлу	Тип медіа	URL або шлях до файлу	Опис	Розмір	Дата додавання
1	nature_video.mp4	Відео	https://drive.google.com/f	Відео про природу	2:35	15.09.2024
2	bird_song.mp3	Аудіо	https://drive.google.com/f	Пташиний спів	1:12	17.09.2024
3	cat_photo.jpg	Зображення	https://drive.google.com/f	Фото кота	1.3 MB	18.09.2024
4	forest_walk.mp4	Відео	https://drive.google.com/f	Прогулянка по лісу	4:20	19.09.2024
5	waves_sound.mp3	Аудіо	https://drive.google.com/f	Звук хвиль	2:00	20.09.2024

Рисунок 8.6 – Попередній перегляд опублікованого сайту

Завдання

Завдання 1. Створення Google Сайту

Створити Google Сайт, який слугуватиме платформою для мультимедійного каталогу. Назва сайту має відображати його тематику (наприклад, «Каталог зображень природи», «Мультимедійна бібліотека студентських проєктів» тощо). Забезпечити загальнодоступний перегляд – через публікацію сайту або надання доступу за посиланням.

Завдання 2. Структурування сайту

Організувати логічну структуру сайту шляхом додавання кількох основних розділів:

– *Головна* – розмістити вступну інформацію, вітальний текст, логотип або ілюстративне зображення;

– *Каталог* – створити розділ для відображення мультимедійного вмісту;

– *Пошук* – додати інструкцію з використання фільтрів і навігації в каталозі;

– *Про нас* – вказати автора проєкту, джерела контенту.

Завдання 3. Інтеграція Google Таблиці

Додати до розділу «Каталог» Google Таблицю, яка містить дані про медіафайли (наприклад: назва, тип, категорія, посилання, опис). Для цього використати функцію "Вставити" → "Таблиці". Переконатися, що таблиця доступна для перегляду (публічний режим або доступ за посиланням). Таблиця має виступати основним елементом навігації каталогом.

Завдання 4. Додавання галерей та мультимедійних блоків

Оформити сайт за допомогою мультимедійних елементів:

– створити галерею зображень (через блоки “Зображення” або вставку з Google Диску);

– додати відео з YouTube або зі збережень на Google Диску;

Усі елементи мають бути органічно вбудовані в структуру сторінок, відповідати тематиці та сприяти зручності використання каталогу.

Загальні рекомендації до виконання звіту

1. **Чіткість і структурованість**.

Опис кожного кроку має бути чітким і послідовним, з логічними поясненнями дій. Використовувати заголовки для кожного етапу роботи, щоб полегшити орієнтацію в звіті.

2. **Скріншоти**.

Для кожного етапу роботи додавати чіткі скріншоти з відповідними підписами. Підписи повинні пояснювати дії, що виконуються, та налаштування, що застосовуються.

3. **Аналіз і пояснення**.

У звіті повинно бути пояснення вибору кожного інструменту та налаштувань. Порівняти результати аналізу різних форматів зображень, зокрема, роздільну здатність, розмір файлу і прозорість.

4. **Оформлення**.

Використовувати чітку нумерацію та форматування для кожного розділу. Підписати всі скріншоти, що додаються до звіту.

5. **Висновки**.

Висновки мають бути логічними та чіткими, з підсумком виконаної роботи і порівнянням результатів.

Контрольні питання:

1. Що таке мультимедійний каталог і яку роль він відіграє в організації мультимедійних файлів?
2. Які типи мультимедійних файлів можуть бути включені до мультимедійного каталогу?
3. Чим відрізняється структура мультимедійного каталогу від традиційних баз даних?
4. Як метадані допомагають у структурованому представленні мультимедійних файлів?
5. Як здійснюється класифікація мультимедійних об'єктів у каталозі?

6. Що таке фільтрація та сортування даних у каталозі, і чому це важливо для зручності користувачів?

7. Які переваги має використання безкодових платформ (таких як Google Sites) для створення мультимедійних каталогів?

8. Які основні функції можуть бути реалізовані на сайті мультимедійного каталогу?

9. Як забезпечити збереження метаданих при редагуванні мультимедійних файлів?

10. Які існують обмеження та недоліки безкодового підходу при створенні мультимедійних каталогів?

Література: [14-19]

СПИСОК РЕКОМЕНДОВАНОЇ ЛІТЕРАТУРИ

1. Основи цифрової обробки сигналів: навч. посіб. / І.І. Самборський, С.М. Шолохов, О.В. Юрченко, Б.А. Ніколаєнко, - Київ: ІСЗЗІ КПІ ім. Ігоря Сікорського, 2021. - 171 с.
2. Основи та методи цифрової обробки сигналів: від теорії до практики: навч. посіб. / уклад. Ю.О. Ушенко, М.С. Гавриляк, М.В. Талах, В.В. Дворжак. – Чернівці : Чернівецький нац. ун-т ім. Ю. Федьковича, 2021. - 308 с.
3. Рибальченко М. О. Цифрова обробка сигналів : навч. посіб. / М.О. Рибальченко, Єгоров О.П., Зворикін В.Б. – Дніпро: НМетАУ, 2018. – 79 с.
4. Gonzalez R. C. Digital Image Processing /R. C. Gonzalez, R. E. Woods. – Pearson, 2018. – 1024 p.
5. Ковальов Ю. М. Прикладна геометрія: нарисна геометрія, інженерна та комп’ютерна графіка, сучасні розділи: підручник / Ю. М. Ковальов, Ю. Н. Ковалев, В. М. Верещага. - Київ : ППП ОМЄГА-Л,2012. - 472 с.
6. Пічугін М. Ф. Комп'ютерна графіка: навч. посіб. / М. Ф. Пічугін, І. О. Канкін, В. В. Воротніков. - Київ : Центр учбової літератури, 2013. - 346 с.
7. Encarnaçao, J., Computer Graphics: Gerätetechnik, Programmierung und Anwendung graphischer Systeme / J. Encarnaçao, W. Straßer. - Walter de Gruyter GmbH & Co KG,2020.
8. Цифрова обробка аудіо- та відеоінформації у мультимедійних системах: навч. посіб. / О.В. Дробик, В.В. Кідалов, В.В. Коваль та ін. – Київ : Наукова думка, 2008. – 144 с.
9. Офіційний сайт Audacity [Електронний ресурс]. – Режим доступу: https://www.audacityteam.org. - Назва з екрана.
10. Онлайн-конвертація аудіоформатів [Електронний ресурс]. – Режим доступу: https://audio.online-convert.com. - Назва з екрана.

11. Документація до FFmpeg [Електронний ресурс]. – Режим доступу : https://ffmpeg.org. - Назва з екрана.

12. Metadata: The differences between XMP and IPTC – FotoWare [Електронний ресурс]. – Режим доступу: https://learn.fotoware.com/On-Premises/Getting_started/Metadata_in_the_FotoWare_system/02_Types_of_metadata_the_Fotoware_DAM_system_can_use/Metadata%3A_The_differences_between_XMP_and_IPTC. - Назва з екрана.

13. 5 ways to view and edit metadata – Daminion Blog [Електронний ресурс]. – Режим доступу: https://daminion.net/articles/tips/top-5-ways-to-view-and-edit-metadata. - Назва з екрана.

14. Gerkes, J. *Database management systems* (1st ed.) /J. Gerkes, R. Ramakrishnan. - McGraw-Hill,2002.

15. Google Cloud Documentation: Cloud Storage [Електронний ресурс]. – Режим доступу: https://cloud.google.com/storage. - Назва з екрана.

16. Google Developers: Google Sheets API [Електронний ресурс]. – Режим доступу: https://developers.google.com/workspace/sheets

17. Automating My Life with Google Apps Script: A Beginner's Journey [Електронний ресурс]. – Режим доступу: https://shamsfiroz.medium.com/automating-my-life-with-google-apps-script-a-beginners-journey-ac7239dc71df. - Назва з екрана.

18. Google Developers: Google Sheets Integration with Google Sites. [Електронний ресурс]. – Режим доступу: https://developers.google.com/sites. - Назва з екрана.

19. Tessema T. Beginners guide to web designing with Google Sites / T. Tessema. - CreateSpace Independent Publishing Platform, 2013. - 96 p.