Нова українська школа

Інна Тріщук, Олександр Лазарець

# ІНФОРМАТИКА

## 7 клас

Інна Тріщук
Олександр Лазарець

# ІНФОРМАТИКА

## ПІДРУЧНИК ДЛЯ 7 КЛАСУ
## закладів загальної середньої освіти

*Рекомендовано Міністерством освіти і науки України*

ТЕРНОПІЛЬ
НАВЧАЛЬНА КНИГА — БОГДАН
2024

УДК 004(075.2)
Т 67

*Рекомендовано Міністерством освіти і науки України*
(наказ Міністерства освіти і науки України
від 05.02.24 № 124)

**ВИДАНО ЗА РАХУНОК ДЕРЖАВНИХ КОШТІВ. ПРОДАЖ ЗАБОРОНЕНО**

Підручник створено за Модельною навчальною програмою
«Інформатика. 7–9 класи» для закладів загальної середньої освіти
(авт. Пасічник О.В., Козак Л.З., Ворожбит А.В.)

**Тріщук І. В.**

Т 67 Інформатика : підручник для 7 кл. закладів загальн. середн. освіти / І. В. Тріщук, О. Ю. Лазарець. — Тернопіль : Навчальна книга — Богдан, 2024. — 288 с.

ISBN 978-966-10-8768-1

Зміст підручника відповідає Державному стандарту загальної середньої освіти та Модельній навчальній програмі «Інформатика. 7–9 класи» (авт.: Пасічник О.В., Козак Л.З., Ворожбит А.В.).

Для учнів та учениць 7 класу.

**УДК 004(075.2)**

Завантажуйте безкоштовний інтерактивний додаток,
використовуючи детальну інструкцію,
за покликанням:
https://edodatok.com/8768-1/



ISBN 978-966-10-8768-1

## Шановні семикласниці та семикласники!

Ви тримаєте в руках підручник з інформатики, який допоможе Вам розширити й поглибити здобуті в 6-му класі знання. У цьому підручнику буде багато новинок про використання штучного інтелекту, генерування зображень, способи захистити свою інформацію та багато тем із програмування! Ми будемо навіть створювати свій YouTube-канал та публікувати власні відео!

Підручник складається із п'яти розділів:

- Персональний цифровий простір
- Цифрове середовище для навчання та співпраці
- Візуальний контент
- Тексти та публікації
- Графічне програмування. Медіадизайн

Розділи поділено на теми, кожна з яких містить теоретичний матеріал, практичну роботу за персональним комп'ютером, домашнє завдання та перевірку знань. Після виконання практичної роботи Ви зможете перевірити себе, відсканувавши QR-код та відповівши на кілька запитань. Також, скориставшись посиланням за QR-кодами, Ви матимете можливість переглянути інтерактивні презентації, що допоможуть Вам закріпити вивчене.

Тож старанно вивчайте теорію, вдосконалюйте практичні навички користування комп'ютером, створюйте цікаві інтерактивні проєкти, а головне — робіть усе із задоволенням!

*Автори*

### УМОВНІ ПОЗНАЧЕННЯ

Для вчителя/вчительки: додайте тест у свою бібліотеку ClassTime, щоб бачити статистику учнів

Для учня/учениці: проскануйте QR-код та пройдіть перевірку

Для учня/учениці: проскануйте QR-код та виконайте завдання для кмітливих

# ПЕРСОНАЛЬНИЙ ЦИФРОВИЙ ПРОСТІР

Тема 1

## Цифрове середовище та інформаційні технології для професійної діяльності та розв'язання проблемних життєвих ситуацій. Цифрові пристрої для побудови локальної (домашньої, персональної) мережі. Драйвери для підключення пристроїв до мережі

**Ключові поняття**

**Повторення**
**Цифрове середовище**
**Права дитини в цифровому середовищі**
**Захист в цифровому просторі**
**Комп'ютерна мережа**
**Способи під'єднання до мережі «Інтернет»**

### Інформаційні технології

**Комп'ютерна техніка та системи** зв'язку використовуються для збирання, передавання, отримання, пошуку, опрацювання, поширення інформації. А це дозволяє організувати будь-яку людську діяльність найбільш ефективно.

Інформаційні технології роблять наше життя зручнішим і допомагають нам здійснювати різні завдання швидко та ефективно.

Інформаційні технології використовуються практично в усіх професіях. Наприклад, лікарі застосовують ІТ для ведення медичних записів, учителі — для навчання, а інженери — для проєктування та моделювання.

Інформаційні технології можуть допомогти розв'язувати проблеми, пов'язані з організацією розкладу, плануванням подорожей, пошуком інформації про товари та послуги, а також для комунікації з друзями та рідними.

Припустимо, Ви зібрали багато фотографій з Вашої мандрівки і хочете створити альбом. Завдяки можливостям інформаційних технологій маєте змогу використати спеціалізовані програмні засоби для оброблення та редагування знімків, внесення текстових описів та створення оригінальних композицій.

Також можете легко обмінюватися повідомленнями, фотографіями та відео з друзями через соціальні мережі, електронну пошту або месенджери. Це дозволяє Вам залишатися на зв'язку з іншими людьми, навіть коли вони далеко від Вас.

## Цифрове середовище

**Цифрове середовище** — означає всі способи, завдяки яким використовують комп'ютери та інтернет: смартфони й планшети, комп'ютерні ігри й соціальні мережі для зберігання, обробки і передачі інформації.

Цифрове середовище розширює наші горизонти, пропонуючи безліч нових можливостей, однак водночас зобов'язує нас бути обачними та свідомими у використанні цих технологій, дбаючи про безпеку та відповідальність.

### Основні складові цифрового середовища

**1 Комп'ютери та пристрої**

Це наші ноутбуки, смартфони, планшети та інші гаджети, які ми використовуємо для роботи, навчання, розваг та спілкування.

**2 Інтернет**

Це велика мережа, яка дозволяє нам шукати інформацію, спілкуватися з іншими людьми, ділитися даними та користуватися онлайн-сервісами.

**3 Програми та додатки**

Це спеціальні програми, які ми встановлюємо на наші пристрої для різних цілей, наприклад, для соціальних мереж, гри, редагування фотографій чи написання текстів.

**4 Соціальні мережі**

Це платформи, які дозволяють нам спілкуватися з друзями, ділитися своїми думками та фотографіями в інтернеті.

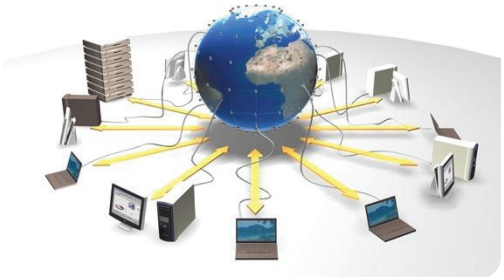
**5 Цифрові дані**

Це інформація у вигляді чисел і кодів, яку ми можемо зберігати, обробляти та передавати за допомогою комп'ютерів.

## Комп'ютер для навчання

## Комп'ютерна мережа

**Комп'ютерна мережа** — система зв'язку між двома чи більше комп'ютерами з метою обміну даними.



Найбільш поширені бездротові з'єднання:

- Wi-Fi
- Bluetooth
- Мобільний інтернет

Пристрої з'єднуються в мережу за допомогою спеціального мережевого обладнання. З'єднання може бути бездротовим або кабельним.

**Провайдер** (від англ. *provider* — постачальник) — це організація, що надає послуги, пов'язані з доступом до глобальної мережі.

**Комп'ютерні мережі бувають:**

- Персональні (Personal area networks — PAN)
- Локальні (Local area networks — LAN)
- Глобальні (Wide area networks — WAN)

Z-WAVE
ZigBee
Bluetooth SMART

**Персональна мережа** — це спеціалізована система комунікації, створена для взаємодії різних електронних пристроїв, які належать одній особі. Ця мережа дозволяє здійснювати обмін даними між пристроями, наприклад, між смартфоном, планшетом та іншими гаджетами.

**Локальні мережі** звичайно займають обсяг одного чи декількох поряд розташованих будинків. Кількість пристроїв, що складають мережу, типово не перевищує декількох тисяч.

**Глобальні мережі** розміщуються на великих географічних просторах.

Найкращим прикладом глобальної мережі є Internet.

**Мобільний інтернет** — це спосіб бездротового доступу до інтернету. В першу чергу такий доступ надає абоненту свободу дій і мобільність — можливість використовувати інтернет на смартфоні та роздавати інтернет на пристрої.


Статус підключення мобільного інтернету відображається в рядку стану.

Літера **E** — **2G** мережа,
**H** — **3G** мережа,
**LTE** — **4G** мережа.

Переваги використання мобільного інтернету сьогодні вже більш ніж очевидні:

- доступність практично по всій території України;
- доступ до мережі там, де інші варіанти підключення відсутні;
- можливість роздавати інтернет друзям.

## Драйвер

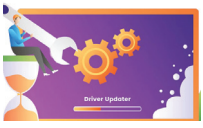
**Драйвер** (англ. *driver*) — це програмне забезпечення або компонент системи, який дозволяє операційній системі чи іншим програмам взаємодіяти з певним обладнанням або пристроєм.

Драйвери необхідні для правильної роботи різних пристроїв на комп'ютері, таких як принтери, графічні карти, мережеві адаптери, звукові карти, миші, клавіатури, сканери та інші.

Драйвери забезпечують операційну систему і програми доступом до функціональності пристрою і дозволяють користувачам ефективно використовувати обладнання.

Однак драйвери можуть бути специфічними для певної операційної системи (наприклад, Windows, Linux чи macOS) і конкретної моделі або виробника обладнання.

**Основна функція драйвера** — це забезпечити взаємодію між апаратним обладнанням пристрою та операційною системою або програмами, що використовують цей пристрій.




Встановлення драйверів може бути автоматичним, коли операційна система сама знаходить та встановлює відповідні драйвери, або вручну, коли Ви вибираєте драйвери зі списку та встановлюєте їх самостійно.

Після встановлення драйвера Ви зазвичай повинні перезавантажити комп'ютер, щоб зміни набули чинності.

Драйвери роблять Ваші пристрої сумісними з операційною системою. Вони перетворюють команди з Вашого комп'ютера в інструкції, які зрозумілі для певного пристрою.

## Способи під'єднання до мережі «Інтернет»

Підключення до інтернету є ключовим аспектом сучасного життя, дозволяючи нам залишатися на зв'язку, працювати, вчитися та розважатися. Існує кілька основних способів підключення до інтернету, кожен з яких має свої переваги та обмеження.

## Цифрові пристрої для побудови локальної мережі

**Маршрутизатор (Router)** — це цифровий пристрій, який дозволяє об'єднати кілька комп'ютерів в одну локальну мережу. Він розподіляє мережевий трафік між комп'ютерами та забезпечує доступ до інтернету. Маршрутизатор може бути підключений до мережі за допомогою Ethernet-кабелю або бездротового з'єднання Wi-Fi.

**Комутатор (Switch)** — це пристрій, який дозволяє підключити більше комп'ютерів до мережі та розподілити мережевий трафік між ними. Він працює на рівні даних і визначає, куди направити пакети даних в мережі.

**Мережева карта (Network Interface Card, NIC)** — це апаратний компонент комп'ютера, який дозволяє йому підключитися до мережі. Кожен комп'ютер повинен мати мережеву карту для участі в локальній мережі.

**Wi-Fi** — це технологія, яка дозволяє підключати комп'ютери, смартфони, планшети та інші пристрої до інтернету без потреби використовувати проводові з'єднання, такі як ethernet-кабелі.


Wi-Fi працює завдяки спеціальним пристроям, які називаються маршрутизаторами (роутер). Маршрутизатори розповсюджують бездротовий сигнал, який наші пристрої можуть сприймати і використовувати для підключення до інтернету. Таким чином, ми можемо користуватися інтернетом у будь-якому приміщенні, де є Wi-Fi сигнал, без потреби використовувати проводи.

Wi-Fi Direct дозволяє комп'ютерам і портативним гаджетам зв'язуватися один з одним без дротів, без використання маршрутизаторів і точок доступу. Тобто з'єднання встановлюється так само просто, як через Bluetooth.

## Цифрові пристрої для бездротового під'єднання до мережі

Wi-Fi роутер (Wireless Router)

Бездротова мережева карта (Wireless Network Adapter)

Смартфони, ноутбуки та планшети

**Wi-Fi роутер (Wireless Router)** — це пристрій, який створює бездротову мережу Wi-Fi у Вашому домі або офісі. Він дозволяє підключати різні цифрові пристрої, такі як комп'ютери, смартфони, планшети і навіть смарт-телевізори, до інтернету через бездротове з'єднання.

**Бездротова мережева карта (Wireless Network Adapter)** — це апаратний компонент, який дозволяє Вашому комп'ютеру підключатися до Wi-Fi мережі. Це може бути вбудованою частиною вашого комп'ютера або зовнішнім USB-адаптером.

Більшість сучасних смартфонів і планшетів мають вбудовану можливість підключення до Wi-Fi мережі. Вони можуть служити як бездротові точки доступу (Hotspot), що дозволяє іншим пристроям підключатися до інтернету через них.

## Бездротові точки доступу (Hotspot)

**Бездротові точки доступу (Hotspot)** — це як маленька «Wi-Fi станція» або «Wi-Fi точка», яка дозволяє іншим пристроям підключатися до інтернету через бездротове з'єднання.

Щоб створити Hotspot, виберіть відповідну опцію на своєму пристрої (наприклад, смартфоні) у налаштуваннях Wi-Fi або в спеціальному додатку. Потім Ви можете надати йому ім'я (SSID) та пароль, які інші пристрої будуть використовувати для підключення.

**Бездротова мережева карта** — це апаратний компонент, який дозволяє Вашому комп'ютеру підключатися до Wi-Fi мережі. Це може бути вбудованою частиною Вашого комп'ютера або зовнішнім USB-адаптером.

## *Додаткова інформація*

**NFC** — це технологія, яка дозволяє бездротово обмінюватися даними між пристроями на дуже короткій відстані, зазвичай менш ніж 10 сантиметрів.

Вона використовується для миттєвого обміну інформацією між смартфонами, картками оплати та іншими пристроями.

NFC працює на дуже близькій відстані, зазвичай не більш ніж 10 см. Це означає, що два пристрої, які використовують NFC, повинні бути дуже близько один до одного для передачі даних.

NFC вважається безпечним засобом передачі даних, оскільки він вимагає фізичного наближення пристроїв. Це ускладнює несанкціонований доступ до інформації.

NFC використовується для різних завдань, таких як безконтактна оплата (за допомогою NFC-карток чи смартфонів), обмін контактами або файлами між смартфонами, активація NFC-тегів для отримання інформації та інше.

Багато сучасних смартфонів мають вбудований чіп NFC, що дозволяє користуватися цією технологією для різних завдань.

## Практична робота №1

**Завдання.** Створіть інфографіку на тему за вибором. Покроковий туторіал за певною темою.

**Вимоги:**

- сервіс **Canva**;
- тип Інфографіка;
- додати кілька зображень;
- додати різну графіку (стрілки, номер, лінії і т.д.);
- пам'ятати про важливість не перевантажувати текстом графіку;
- дотримуватися одного стилю.

**Теми:**

1. Як створити мобільну точку доступу зі свого смартфону (також розказати, як змінити назву мережі та пароль).
2. Як під'єднатися до мережі Wi-Fi за допомогою смартфона та ноутбука (як переглянути властивості мережі та видалити дані про мережу).
3. Правила захисту при підключенні до безкоштовної Wi-Fi-мережі.
4. Чим небезпечні безкоштовні Wi-Fi-мережі і як в них безпечно працювати.
5. Технічні характеристики маршрутизатора та їх короткий опис.
6. Як обрати маршрутизатор.

Поширити за допомогою покликання Вашій вчительці/Вашому вчителеві на електронну пошту.

**Інтернет-трафік** (англ. *Traffic* — «рух») — обсяг інформації, переданої через комп'ютерну мережу за певний період часу. Кількість трафіку вимірюється як в пакетах, так і в бітах, байтах і їх похідних: кілобайт (КБ), мегабайт (МБ) і т. д.

Аналіз швидкості інтернет-трафіку за допомогою сервісу.

## Домашнє завдання

### Пошукова робота в мережі «Інтернет»

1. Що таке інтернет-трафік та які його різновиди.
2. Які порти є в роутері (маршрутизаторі) та яке їхнє призначення.
3. Використання мережі «Інтернет» для навчання та розвитку: онлайн-курси, корисні ресурси та інструменти для самостійного навчання.
4. Як підключити принтер до комп'ютерної мережі: налаштування та драйвери.
5. Wi-Fi та інтернет в мобільних пристроях: налаштування та оптимізація з'єднання.
6. Як працює бездротова технологія Wi-Fi: основи та принципи роботи.
7. NFC (Near Field Communication) — що це та де використовують цю технологію.
8. Zigbee — що це за технологія бездротового зв'язку та де вона використовується.

Зберегти в Google документі відповіді на ці запитання.

Надати доступ Вашій вчительці/Вашому вчителеві на електронну пошту.

***Вимоги:***

- 2-3 абзаци тексту — відповідь на 1 запитання;
- за потребою кілька зображень.

**Пам'ятайте про правила безпечного користування мережею «Інтернет» та критичне мислення.**

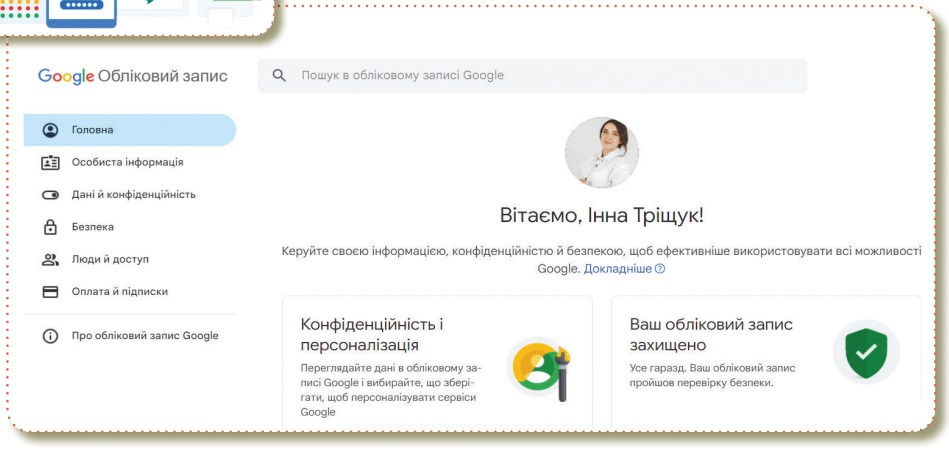
**Тема 2** **Розширення браузера для захищеної та продуктивної роботи й відпочинку онлайн. Технологія VPN. Безпека та конфіденційність облікового запису. Синхронізація даних між пристроями**

Ключові поняття

**Обліковий запис**
**Інструменти Google**
**Захист смартфонів та планшетів**
**Синхронізація даних**
**Розширення для браузера**
**Безпека в мережі**

## Обліковий запис (акаунт)

**Обліковий запис (акаунт)** — особистий профіль користувача, який дозволяє Вам входити та користуватися різними онлайн-сервісами та вебсайтами.




Це дуже зручно, оскільки Ви можете мати доступ до багатьох інтернет-ресурсів, не створюючи окремих профілів для кожного з них.

**Ім'я користувача** — це ім'я, яке Ви обираєте для ідентифікації себе на вебсайті або сервісі. Іноді це також може бути вашою електронною адресою.

**Пароль** — це секретний код, який Ви створюєте, щоб захистити свій обліковий запис від несанкціонованого доступу.

**Електронна адреса** — це унікальна адреса, яка дозволяє Вам отримувати повідомлення за допомогою електронного листування та відновлювати доступ до облікового запису в разі потреби.

## Яка будова електронної адреси


**1 Ім'я користувача**

Ця частина перед символом «@» вказує на ім'я користувача

**2 Символ «@»**

Символ «@» служить роздільником між ім'ям користувача і доменом

**3 Домен**

Ця частина адреси після символу «@» вказує на сервер, на якому розташована електронна пошта користувача

**Двофакторна автентифікація** — це додатковий рівень безпеки при вході в обліковий запис, який вимагає від користувача надати два різні методи ідентифікації для підтвердження своєї особи.

Це робить процес входу в обліковий запис більш безпечним, оскільки навіть, якщо хтось зламає або дізнається Ваш пароль, він не зможе отримати доступ до Вашого облікового запису без додаткового підтвердження.

## Надійний пароль — який він?

**Надійність пароля**

Важливо використовувати паролі, які складаються з різних символів, таких як букви, цифри та спеціальні символи. Уникайте використання очевидних паролів, таких як «123456» або «пароль».

**Довжина пароля**

Довжина пароля також важлива. Зазвичай рекомендується, щоб пароль мав не менш ніж 8 символів.

**Індивідуальність пароля**

Кожен обліковий запис повинен мати власний пароль. Не використовуйте однаковий пароль для всіх сервісів.

**Зміна пароля**

Регулярно змінюйте паролі для підвищення безпеки облікового запису.

**Зберігання пароля**

Нікому не повідомляйте свій пароль, не залишайте його відкритим на громадських комп'ютерах.

**Перевір себе за допомогою інтерактивних ігор**

## Різновиди облікових записів

**Соціальні мережі**

Обліковий запис в соціальних мережах, таких як **Facebook**, **Instagram**, **Twitter** та інші

**Електронна пошта**

Обліковий запис електронної пошти, наприклад, на **Gmail** або **Yahoo**, дозволяє Вам відправляти та отримувати електронні листи

**Хмарні сервіси**

Деякі сервіси, наприклад, **Google Drive** або **Dropbox**, дозволяють зберігати та ділитися файлами в хмарному сховищі, і Ви можете створити обліковий запис для доступу до них

**Інтернет-банкінг**

Банківські облікові записи дозволяють Вам керувати своїми фінансами в інтернеті, переказувати гроші та перевіряти баланс

**Обліковий запис Google** — це особистий профіль користувача, який надає доступ до різних сервісів та продуктів, розроблених компанією **Google**.


Це включає в себе пошту **Gmail**, **Google Drive** для збереження файлів у хмарі, **Google Docs** для роботи з текстовими документами, **Google Sheets** для створення таблиць, **Google Calendar** для планування подій та багато інших сервісів.

Під час створення облікового запису Ви обираєте електронну адресу **Google**, яка також називається Gmail-адресою. Це буде Вашою основною адресою для надсилання та отримання електронних листів.

**Пошукова система Google. Google Search або Google** — найбільша пошукова система, що належить корпорації Google Inc.

**Карти Google (Google Maps)** — безкоштовний картографічний сервіс. Допоможе визначити ваше місцезнаходження, дороги для велосипедистів/велосипедисток, ґрунтові/неасфальтовані доріжки, затори, дослідити нові місця та ін.

**Ютуб (YouTube)** — популярний відеохостинг, що надає послуги з розміщення відеоматеріалів.

**Google Фото (Google Photos)** — сервіс, призначений для зберігання, упорядкування та демонстрації ваших фото та відео.

**Контакти Google (Google Contacts)** — дають можливість зберегти імена, електронні адреси, номери телефонів та інші дані.

**Диск Google (Google Drive)** — хмарне сховище даних Google надає всім користувачам/користувачкам безкоштовні 15 ГБ онлайн-простору.

**Google Календар (Google Calendar)** — сервіс для планування зустрічей, подій, справ із прив'язкою до календаря. Можливе спільне використання календаря групою користувачів/користувачок.

## Для налаштування власного облікового запису Google потрібно:

1. Налаштування особистої інформації
2. Налаштування конфіденційності та безпеки
3. Налаштування даних та конфіденційності
4. Налаштування безпеки

Детальний алгоритм налаштування поданий на с. 28 (практична робота).

## Захист смартфонів та планшетів

### 1 Встановлення пароля або ПІН-коду

Встановіть пароль або ПІН-код для блокування Вашого пристрою. Це перший шар захисту в разі втрати чи крадіжки.

### 2 Графічний патерн (Pattern Lock)

Замість пароля чи ПІН-коду Ви можете використовувати графічний патерн, який потрібно намалювати на екрані для розблокування.

### 3 Розпізнавання відбитка пальця (Fingerprint Recognition)

Більшість сучасних смартфонів та планшетів підтримують розпізнавання відбитка пальця (обличчя), що забезпечує швидкий та безпечний доступ до пристрою.

### 4 Віддалене відслідковування (Remote Tracking)

Увімкніть службу віддаленого відслідковування (наприклад, «Find My iPhone» для Apple або «Find My Device» для Android), яка дозволяє Вам відстежувати місцезнаходження та навіть віддалено блокувати чи видаляти дані на пристрої у випадку втрати.

### 5 Віддалене видалення даних (Remote Wipe)

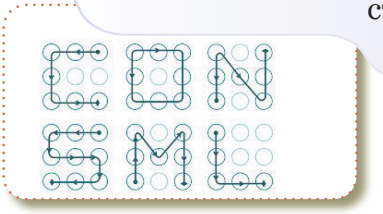
У разі втрати пристрою Ви можете видалити всі Ваші особисті дані віддалено, щоб уникнути несанкціонованого доступу.

### 6 Встановлення додатків з надійних джерел

Завжди завантажуйте додатки лише з офіційних джерел, таких як Google Play Store або Apple App Store, щоб уникнути встановлення шкідливого програмного забезпечення.

## Графічний патерн (Pattern Lock)

**Графічний патерн (Pattern Lock)** — це метод захисту Вашого смартфона або планшета, який дозволяє Вам блокувати доступ до пристрою за допомогою рисунку, створеного Вами на екрані сенсорного пристрою.



### Складний патерн

Створіть складний графічний патерн, який важко вгадати. Уникайте використання простих форм, таких як "L" або "Z", і робіть патерн непередбачуваним.

### Чистота екрану

Регулярно чистіть екран Вашого пристрою від відбитків пальців, щоб ніхто не міг вгадати Ваш патерн за слідами.

### Використовуйте інші методи блокування

Графічний патерн може бути одним з методів блокування, але ви також можете використовувати паролі, ПІН-коди, розпізнавання відбитка пальця або обличчя для додаткової безпеки.

## Віддалене відслідковування (Remote Tracking)

**Віддалене відслідковування (Remote Tracking)** — це функція, яка дозволяє Вам визначити місцезнаходження вашого смартфона або планшета в реальному часі, використовуючи GPS-технології та інтернет-з'єднання.

### GPS-технологія

Багато сучасних смартфонів та планшетів обладнані GPS-приймачами, які дозволяють точно визначити місцезнаходження пристрою.

**GPS (Global Positioning System)** — це система супутникової навігації, яка використовує сигнали з супутників для визначення координат пристрою.

### Інтернет-з'єднання

Для передавання інформації про місцезнаходження Вам потрібне активне підключення до інтернету. Ви можете використовувати мобільний інтернет або Wi-Fi для цього.

### Спеціальні програми і сервіси

Для віддаленого відслідковування Ви повинні встановити спеціальну програму або скористатися сервісом віддаленого відслідковування, які доступні для багатьох смартфонів.

## Віддалене видалення даних (Remote Wipe)

**Віддалене видалення даних (Remote Wipe)** — це функція, яка дозволяє Вам видалити всі дані й налаштування на своєму смартфоні або планшеті віддалено, навіть якщо пристрій був втрачений, викрадений або недоступний.

**1** **Активація функції**

Для використання віддаленого видалення даних Вам потрібно активувати цю функцію на Вашому смартфоні або планшеті. Зазвичай це робиться через обліковий запис Google (для Android-пристроїв) або Apple ID (для iPhone і iPad).

**2** **Видалення даних**

Після активації Ви можете віддалено видалити всі дані на пристрої. Наприклад, фотографії, контакти, повідомлення, додатки та іншу інформацію.

**3** **Заблокування пристрою**

Також Ви можете віддалено заблокувати пристрій за допомогою пароля або ПІН-коду. Це запобігає доступу до пристрою навіть після видалення даних.

## Синхронізація даних

**Синхронізація даних між пристроями** — це процес обміну інформацією між двома або більше пристроями (наприклад, комп'ютерами, смартфонами, планшетами чи іншими пристроями), щоб зробити дані на цих пристроях ідентичними (однаковими).

### Аспекти синхронізації даних

**1** **Хмарні послуги**

Синхронізація даних часто виконується за допомогою хмарних послуг, таких як Google Drive, Dropbox, iCloud тощо. Це онлайн-сервіси, які зберігають Ваші дані на серверах в інтернеті, а не на конкретних пристроях.

**2** **Один обліковий запис**

Для синхронізації даних між пристроями Ви можете використовувати той самий обліковий запис (наприклад, обліковий запис Google або Apple ID) на всіх своїх пристроях. Це дозволяє Вашим даним бути доступними на всіх пристроях, які використовуєте. Наприклад, Google контакти або Ваші місця перебування.

3

**Доступ з будь-якого місця**

Однією з переваг синхронізації є можливість отримати доступ до Ваших даних з будь-якого місця, де є інтернет. Наприклад, Ви можете переглядати свої файли або календар навіть під час подорожі.

4

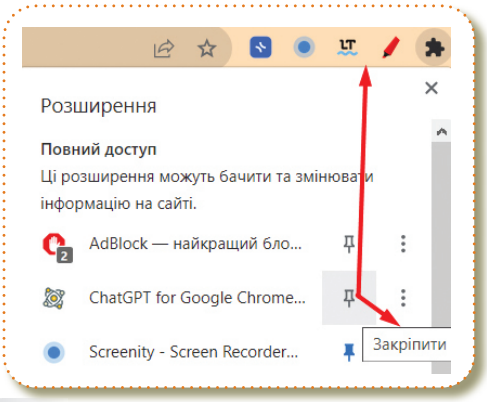
**Резервне копіювання**

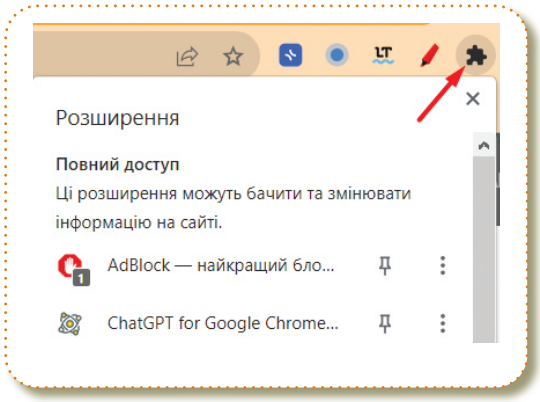
Синхронізація даних також забезпечує резервне копіювання Ваших даних. Це означає, що навіть якщо один з Ваших пристроїв пошкодиться або втратиться, Ваші дані залишаться в безпеці на інших пристроях або в хмарному сховищі.

## Розширення для браузера

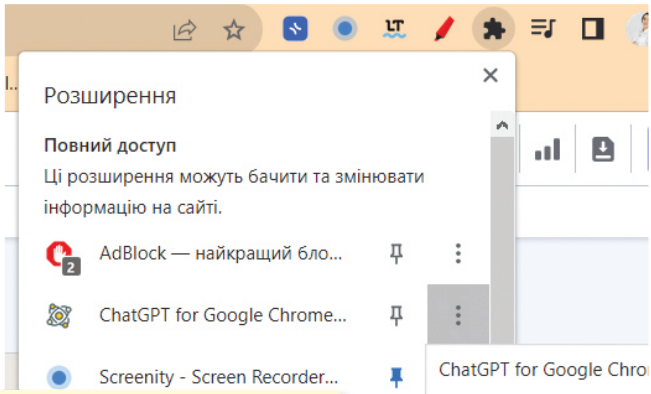
**Розширення для браузера** — це додаткові програми, які можна встановити в браузері для розширення його функціональності та забезпечення додаткових можливостей.

Після встановлення розширення Ви можете використовувати його, натискаючи на його значок у верхньому правому або лівому куті вашого браузера.



При виборі розширень для встановлення завжди важливо використовувати надійні джерела, такі як офіційні магазини додатків для браузерів.



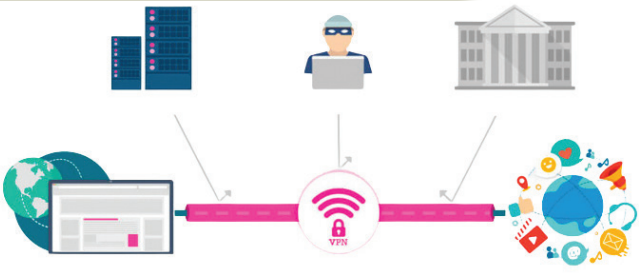

Якщо Ви більше не використовуєте якенебудь розширення або воно Вас обмежує, то можете видалити його з браузера.

## Безпека в мережі

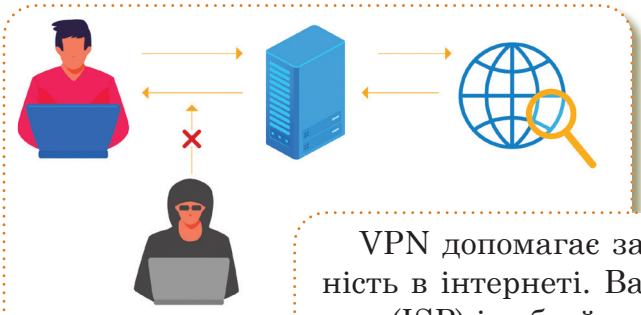
### Технологія VPN

**VPN (Virtual Private Network)** — технологія, яка дозволяє створювати безпечне та приватне з'єднання між Вашим пристроєм (комп'ютером, смартфоном, планшетом) та інтернетом.

Вона допомагає захищати Вашу інформацію від несанкціонованого доступу.



Коли Ви підключаєтеся до VPN, Ваш інтернет-трафік (наприклад, вебсторінки, файли, повідомлення) переходить через зашифрований тунель, який проходить через сервери VPN.



VPN допомагає захищати Вашу приватність в інтернеті. Ваш інтернет-постачальник (ISP) і вебсайти не можуть відстежувати, які сайти Ви відвідуєте або що Ви робите в інтернеті.

**Фішинг** — це форма атаки з використанням соціальної інженерії, в ході якої зловмисник/зловмисниця, маскуючись під надійний суб'єкт, виманює конфіденційну інформацію.

## Ознаки фішингу

1. **Загальні або неофіційні привітання** — листи без персоналізації (наприклад, «шановний клієнт/клієнтка») та формальностей мають викликати підозри. Те ж стосується псевдо-персоналізації з використанням випадкових, підроблених посилань.

2. **Запит на особисту інформацію** — часто використовують кіберзлочинці, але банки, фінансові установи та більшість онлайн-сервісів намагаються цього уникати.

3 **Терміновість** — фішингові повідомлення часто намагаються викликати відчуття терміновості дій, залишаючи жертвам менше часу на роздуми.

4 **Пропозиція, від якої важко відмовитися** — якщо лист занадто хороший, щоб бути правдою, він, напевно, є фішинговим.

5 **Некоректна граматика** — орфографічні та друкарські помилки, а також незвичні фрази часто можуть означати небезпеку (але відсутність помилок не є доказом легітимності).

6 **Підозрілий домен** — сумнівно, що дійсно американський чи німецький банк надсилатиме Вам електронний лист з китайського домену.

## Цифровий слід

**Цифровий слід.** Усе, що залишається в пам'яті мережі: відкриття акаунтів, створення паролів, відвідування певних сайтів, лайки, коментарі, покупки тощо.

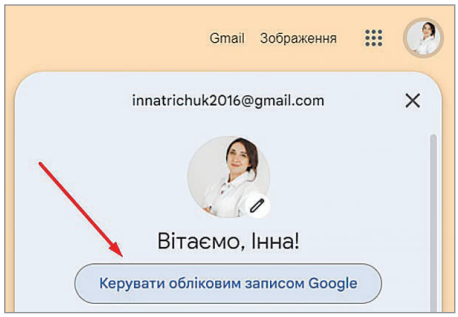
Коли хтось відправляє якусь фотографію через мобільний телефон чи інший гаджет, вона більше не належить конкретній людині — ця інформація стає власністю мережі «Інтернет», тому ніхто не знає, як її можуть використати і як це позначиться на репутації людини.

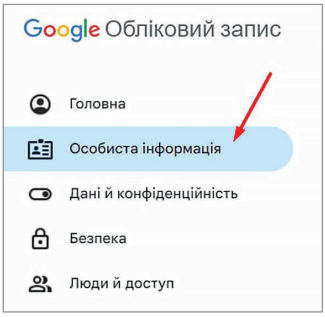
З повідомлення, яке надіслане, а потім видалене, можуть зробити скріншот (сфотографувати), а фото зберегти на іншому носії. Навіть якщо Ви впевнені у тому, кому надсилаєте щось особисте, інтимне чи компрометуюче — цифровий слід залишиться назавжди.

## Практична робота №2

### Частина 1. Налаштування власного облікового запису Google

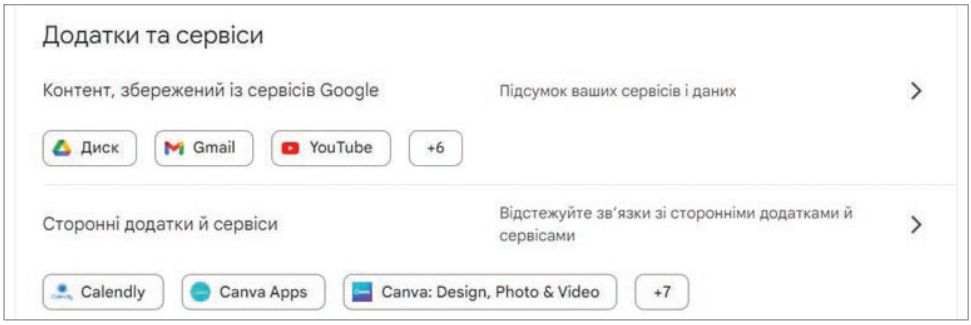
**1.** В обліковому записі **Google** оберіть **Керувати обліковим записом Google**.



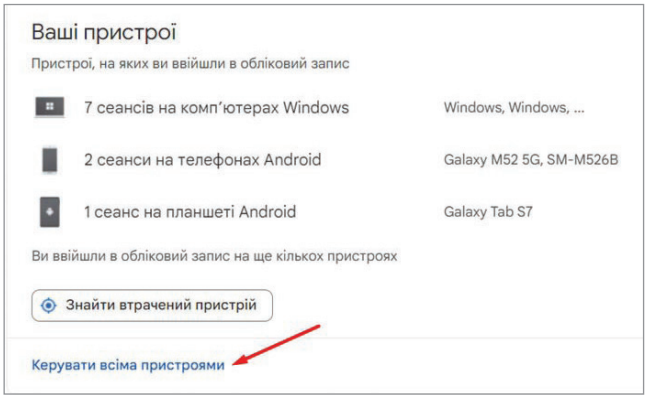

**2.** Оберіть пункт **Особиста інформація.**

- Перевірте Ваше ім'я та прізвище. За бажанням можете додати псевдонім.
- Перевірте Вашу дату народження та хто може бачити Вашу дату народження.
- Додайте резервну електронну адресу (потрібно ввести пароль безпеки, який прийде на електронну резервну адресу).
- Додайте номер телефону (якщо не доданий, аналогічно за допомогою SMS-повідомлення прийде автоматичне підтвердження).
- Перевірте свій пароль та дату його створення.
- Перевірте, чи мова Вашого облікового запису українська. Якщо ні — змініть на українську мову.



**3.** Оберіть пункт **Дані та конфіденційність** / **Додатки та сервіси**.

- Перевірте пункт **Контент**, збережений із сервісів та перегляньте дані, які зберігаються у Вашому обліковому записі **Google**.
- Перевірте, в які сторонні сервіси та додатки Ви авторизувалися за допомогою облікового запису **Google**.
- Створіть резервну копію даних облікового запису **Google**. Оберіть сервіси, які Вам необхідні, та завантажте матеріали.



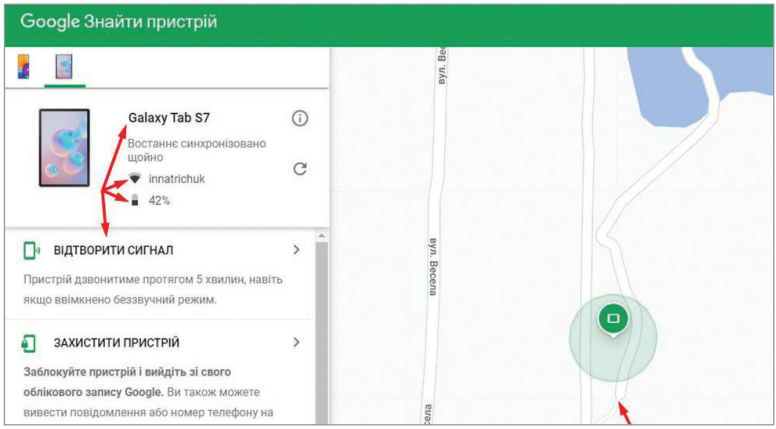

**4.** Оберіть пункт **Безпека**.

- Перегляньте, чи увімкнена у Вас двоетапна перевірка (налаштуйте за допомогою сповіщень на смартфоні при вході у Ваш обліковий запис).
- Перевірте, коли була зміна паролю останній раз (за потреби змініть пароль).
- Перевірте номер телефону для відновлення та резервну електронну адресу (може бути кілька варіантів).
- Оберіть Ваші пристрої / керувати всіма пристроями та перегляньте, на яких пристроях Ви ввійшли в обліковий запис. Якщо серед цих пристроїв є підозрілі або невідомі Вам — завершіть сеанс.

**5.** Знайдіть втрачений пристрій.

- Оберіть пункт **Знайти** втрачений пристрій.
- Серед переліку пристроїв, у яких Ви увійшли в свій обліковий запис, оберіть Ваш.
- У діалоговому вікні перегляньте місце розташування пристрою, рівень заряду та підключення до мережі «Інтернет».

- Оберіть **Відтворити сигнал** та очікуйте дзвінка.




**6.** Перевірте перелік сторонніх застосунків та сервісів, у які Ви входили, авторизуючись за допомогою облікового запису **Google**.

- Пункт **Зв'язки** із сторонніми додатками та сайтами.
- Перегляньте перелік.
- Якщо Ви бачите підозрілі сайти або сервіси — видаліть дані авторизації та всі дозволи до Вашого облікового запису.

**7.** В обліковому записі **Google** можуть зберігатися Ваші паролі. Менеджер паролів полегшує вхід на сайти та сервіси на будь-якому пристрої, де Ви увійшли в свій обліковий запис.

- Перевірте перелік паролів та сайтів.

### Частина 2. Синхронізація зображень

Працюйте в сервісі **Google фото**.

- З мережі «Інтернет» завантажте кілька зображень про символи України на **Робочий стіл** (5-6 зображень буде достатньо).
- Відкрийте сервіс **Google фото**.
- Створіть альбом **«Символи України»**.
- Завантажте в альбом дані зображення (обрати **Вибрати з комп'ютера**).
- Створіть покликання для альбому.
- Надішліть це покликання електронним листом Вашій вчительці/Вашому вчителю.
- Перевірте **Google фото** на смартфоні (чи наявний альбом та чи наявні зображення в альбомі).

## Частина 3. Синхронізація папки на Google диску

Працюйте в сервісі **Google диск**.

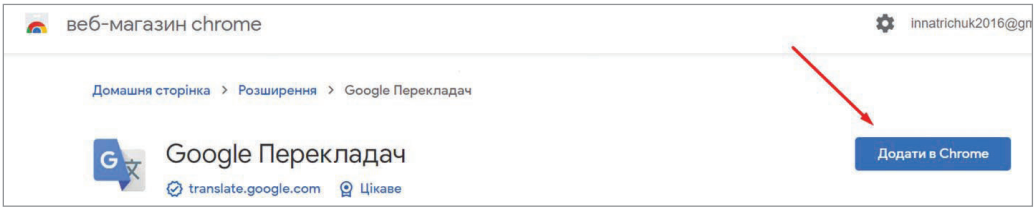
- Створіть папку на **Google диску “Практична робота №2”**.
- У папці створіть **Google документ** та **Google презентацію** (назвати на Ваш вибір).
- Завантажте зображення з **Робочого столу** (папка із зображеннями, яку ми використовували в попередній частині).
- Відкрийте **Google диск** на смартфоні.
- Перевірте наявність папки та файлів у папці.

## Частина 4. Інсталяція розширень Google Chrome

1) Перейдіть за покликанням https://chrome.google.com/webstore.

2) У рядку пошуку напишіть **Google Translate**.

3) Оберіть **Додати в Chrome**.




4) Інсталюйте розширення та закріпіть.

5) Перейдіть на будь-який англомовний сайт та за допомогою розширення перекладіть сайт.

## Розширення Page Market

- Інсталюйте це розширення аналогічно попередньому алгоритму дій.
- Закріпіть розширення на панелі.


- Створіть кілька позначок за допомогою розширень на будь-якому сайті.
- Збережіть у форматі **.png.**

## Домашнє завдання

Пошукова робота в мережі «Інтернет».

1. Порівняй роботу хмарних сховищ за певними критеріями.
2. Створи **Google документ** з таблицею за зразком.
3. Після виконання завдання — відправ на електронну пошту вчительці/вчителеві.

| Критерій | OneDrive | Dropbox | Google Drive |
|---|---|---|---|
| Обсяг безкоштовного місця | | | |
| Обмеження за розміром файлу, що завантажується | | | |
| Чи можна отримати додаткове місце | | | |
| Платне використання | | | |
| Підтримка операційних систем | | | |

1. Склади перелік «ТОП 5 розширень **Google Chrome** для продуктивної роботи».
2. Створи **Google документ** з таблицею за зразком.
3. Після виконання завдання — відправ на електронну пошту вчительці/вчителеві.

| № | Назва розширення | Призначення розширення | Покликання розширення |
|---|---|---|---|
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |

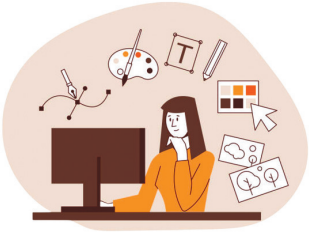
## Тема 3 Інформаційне наповнення персонального цифрового простору. Інформаційні потреби. Інформаційні джерела. Достовірна та недостовірна інформація. Пошук даних в мережі «Інтернет». Цифрові інструменти перевірки факту редагування фото, зображень, аудіо, відео тощо. Інформаційне сміття та способи його зменшення

Ключові поняття

**Персональний цифровий простір**
**Інформаційні потреби**
**Інформаційні джерела**
**Інформаційне сміття**

### Персональний цифровий простір

**Персональний цифровий простір** — віртуальний простір, де Ви зберігаєте та обробляєте інформацію за допомогою цифрових пристроїв, таких як комп'ютери, смартфони та планшети.



Він містить файли, документи, фотографії, відео та інше, що Ви створюєте або отримуєте в інтернеті.

Персональний цифровий простір утворюється завдяки цифровим пристроям, які ми використовуємо, таким як комп'ютери, смартфони та планшети.

Ви створюєте цей простір, коли зберігаєте на ньому Ваші файли та дані. Ви можете зберігати їх на внутрішньому сховищі пристрою або у хмарних сховищах в інтернеті.

Ваш персональний цифровий простір дозволяє Вам зберігати важливу інформацію та документи, доступ до яких можна мати з будь-якого пристрою, підключеного до інтернету. Ви можете використовувати його для зберігання шкільних робіт, фотографій, музики, відео та багатьох інших речей. Він також корисний для навчання, спілкування, розваг та інших завдань.

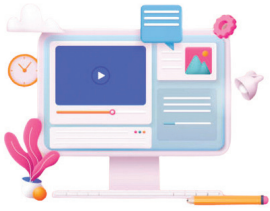
## Що можна робити в персональному цифровому просторі?

### 1 Зберігати інформацію

Ваш персональний цифровий простір — це місце, де Ви зберігаєте свої файли, такі як фотографії, документи та відео. Вони зберігаються на Вашому комп'ютері, смартфоні або в хмарному сховищі в інтернеті.

### 2 Робота з інформацією

Ваш персональний цифровий простір дозволяє Вам редагувати та обробляти інформацію. Наприклад, Ви можете редагувати фотографії, створювати текстові документи або робити презентації.



### 3 Навчання

Ви можете використовувати свій цифровий простір для навчання. Доступ до інтернету допомагає знаходити відповіді на запитання та вивчати нову інформацію.

### 4 Поширення інформації

Ви можете відправляти файли і листи електронною поштою, ділитися фотографіями у соціальних мережах або надсилати посилання на важливі матеріали.

## Інформаційні потреби

**Інформаційні потреби** — це те, що ми хочемо знати або дізнатися, коли шукаємо інформацію. Це може бути будь-яка інформація, яка цікава або корисна для нас.

Важливо бути критичним споживачем інформації, перевіряти джерела та критично ставитися до інформації.

**Відповіді на запитання.** Ми шукаємо інформацію, щоб знайти відповіді на наші запитання. Наприклад: «Які річки найбільші в світі» або «Як працює сонячна енергія?».

**Навчання.** Ми шукаємо інформацію, щоб дізнатися про нові речі, які цікавлять нас. Наприклад, як навчитися програмувати або вивчити іноземну мову та багато іншого.

## Інформаційні джерела

**Інформаційні джерела** — це місця, де ми можемо знайти необхідну інформацію.

- Інтернет
- Бібліотеки
- Вчителі та експерти
- Медіа

Інтернет — це велике джерело інформації, де ми можемо знаходити відповіді на наші запитання, читати новини, дивитися відео та багато іншого.

Бібліотеки містять книги, журнали та інші джерела, які можна використовувати для навчання і дослідження.

Ваші вчителі та інші експерти можуть бути джерелами інформації та знань.

Телевізор, радіо, газети і журнали — це інші джерела інформації, де можна знаходити новини та розважальний контент.

## Достовірна і недостовірна інформація

Інформація, яку ми знаходимо в інтернеті або інших джерелах, може бути **достовірною (правдивою)** або **недостовірною (неправдивою)**.

**Достовірна інформація** — це інформація, яка підтверджена авторитетними джерелами, фактами або експертами. Якщо Ви можете перевірити інформацію та знайти її в більш ніж одному незалежному джерелі, то це може бути достовірною інформацією.

**Недостовірна інформація** — це інформація, яка не має підтвердження або може бути неправдивою. Вона може бути розповсюджена з метою обману або просто виникати на основі домислів.

## Пошук даних в мережі «Інтернет»

**Пошук інформації в мережі «Інтернет»** — це важлива навичка, є кілька моментів, які варто врахувати для ефективного пошуку.

**Ключові слова.** Ви можете використовувати ключові слова або фрази для пошуку в інтернеті. Вони повинні бути чіткими та пов'язаними з Вашою інформаційною потребою.

**Перевіряйте джерело.** Переконайтеся, що інформація походить від авторитетного джерела. Намагайтеся використовувати сайти, які мають добру репутацію, такі як урядові агентства, навчальні заклади або відомі видання.

**Порівнюйте джерела.** Якщо Ви знайшли інформацію, перевірте її в кількох джерелах, щоб переконатися в її точності та достовірності. Інформація може втратити актуальність з часом. Перевіряйте дату публікації, особливо для новин та статей.

**Будьте критичними.** Пам'ятайте, що не всі інформаційні джерела надійні. Будьте критичними до інформації.

## Критерії оцінювання сайтів

**1** **Наявність дати створення сайту, дати розміщення матеріалів та оновлення сайту**

Сайт, що містить наукові чи науково-популярні дані, має періодично оновлюватись, щоб вміщена на ньому інформація була не застарілою, а правильною та точною.

**2** **Інформація про автора**

Наявність потрібної кваліфікації у автора можна визначити, ввівши його ім'я в пошукову систему і переглянувши сайти, які дають додаткові відомості про автора, його місце роботи, публікації.

**3** **Забезпечення на сайті «зворотного зв'язку» з автором**

Наявність електронної адреси чи організація форуму, чату або діалогових вікон зв'язку.

**4** **Формальні ознаки надійності URL-адреси вебсайту**

Потрібно знати комбінацію літер в кінці доменного імені, наприклад:

**.gov** — вказує на те, що це сайт державної установи;
**.edu** — ознака освітніх установ, університетів;
**.com** — використовується для комерційних організацій, які створені для отримання прибутку;
**.org** — переважно ознака неприбуткових організацій.

**5** **Наявність граматичних та орфографічних помилок на сайті**

**6** **Наявність у статті слів узагальнювального та оцінювального характеру**

## Перевірка інформації

1. TinEye — зворотний пошук зображень
2. WebMii — шукає посилання з ім'ям людини
3. Is It Hacked — в режимі реального часу виконує перевірку
4. IsItPhishing — оцінює вказаний URL в режимі реального часу
5. Virus Total — перевірити посилання
6. Web of Trust — міжнародний ресурс перевірки сайтів

## Інформаційне сміття

**Інформаційне сміття**, також відоме як **«спам»** або **«надмірна інформація»** — це надмірна, непотрібна або небажана інформація, яка завалює наші поштові скриньки, соціальні мережі, інтернет-сайти та інші джерела.


Воно може містити небажану рекламу, фейкові повідомлення, спам-листи та інше.

Не пересилайте інформацію, яку Ви не перевірили, та не сприяйте поширенню сміття.

## Способи зменшення інформаційного сміття

1

**Повідомлення про спам**

Якщо Ви отримуєте спамові повідомлення, не соромтеся повідомляти про них Вашого поштового провайдера або адміністратора вебсайту. Це допоможе удосконалити фільтри та зменшити кількість небажаного контенту для всіх користувачів.

2

**Видалення підписок**

Періодично переглядайте свою поштову скриньку та видаляйте підписки на розсилки, які Вас більше не цікавлять.


3

**Використовуйте одноразові адреси електронної пошти**

Для реєстрації на різних сайтах або отримання інформації від невідомих джерел використовуйте одноразові адреси електронної пошти. Вони дозволять уникнути надмірного спаму в основній поштовій скриньці.

4

**Не діліться особистою інформацією**

Уникайте розголошення особистої інформації, такої як адреса електронної пошти, на публічних форумах та сайтах. Це допоможе зменшити кількість небажаних повідомлень.

5

**Фільтри та спам-фільтри**

Використовуйте фільтри для електронної пошти, які видаляють або відсилають спам в окремий ящик. Налаштуйте їх для блокування небажаних повідомлень.

## Практична робота №3

### Робота в інтернеті

**Завдання.** Створити інфографіку з порадами в **Canva** на обрану тему:

- «Інформаційне сміття. Як зменшити його в нашому цифровому просторі»;
- «Мій персональний цифровий простір. Як ефективно використовувати та за допомогою яких сервісів»;
- «Інформаційні джерела та які основні правила їх використання. Правила безпеки»;
- «Як оцінити інформацію та не потрапити на фейк».

Сервіс — **Canva**
Шаблон — **Інфографіка**
Кількість учнів у групі — 2–3 учні
Тривалість — 20–25 хв
Демонстрація інфографіки в **Canva**
Інфографіку надіслати Вашій вчительці/Вашому вчителю на електронну пошту.

## Домашнє завдання

Створити інфографіку на одну із запропонованих у практичній роботі тем.
Надіслати Вашій вчительці/Вашому вчителю на електронну адресу.

Розділ 2

# ЦИФРОВЕ СЕРЕДОВИЩЕ ДЛЯ НАВЧАННЯ ТА СПІВПРАЦІ

## Тема 4 Огляд операційних систем для різних пристроїв. Види програмного забезпечення (десктопні, застосунки, онлайнові версії). Встановлення програмного забезпечення

Ключові поняття

**Операційна система**
**Інтерфейс**
**Програмне забезпечення**
**Десктопні програми**
**Застосунки (або додатки)**
**Онлайнові версії програм**
**Інсталяція**

### Операційна система (ОС)

**Операційна система (ОС)** — це програмне забезпечення, яке керує роботою комп'ютера або іншого пристрою.


Існує багато різних ОС, розроблених для різних пристроїв і завдань.

Кожна операційна система призначена для певних видів пристроїв і має свої властивості та переваги. Вибір операційної системи залежить від типу пристрою та потреб користувача.

ОС є своєрідним мозком комп'ютера, який відповідає за виконання команд, організацію роботи і забезпечення взаємодії користувача з комп'ютером.

## Як обрати операційну систему

| | |
|---|---|
| Призначення | Вибір ОС залежить від того, для якої мети Ви використовуватимете пристрій. Наприклад, для роботи на ПК можна вибрати Windows або Linux, для смартфона — Android або iOS. |
| Сумісність | Переконайтеся, що ОС сумісна з Вашим обладнанням та програмами, які Ви плануєте використовувати. |
| Користувацький інтерфейс | Виберіть ОС з інтерфейсом, який Вам подобається і з яким Ви зручно взаємодієте. |
| Доступність додатків та програм | Переконайтеся, що ОС має необхідні для Вас додатки та можливості. |

## Інтерфейс

### Характеристики операційних систем

Інтерфейс | Системні вимоги | Безпека | Оновлення і патчі

- **Інтерфейс** — це спосіб, яким користувач взаємодіє з операційною системою. Це може бути графічним інтерфейсом (як у **Windows** або **macOS**) або командним рядком (як у **Linux**).
- Кожна ОС має певні **системні вимоги**, що вказують, яке обладнання потрібно для її роботи. Це включає процесор, оперативну пам'ять, місце на диску тощо.
- ОС має механізми **безпеки** для захисту від вірусів і зловмисного програмного забезпечення.
- Важливо, щоб ОС регулярно оновлювалася та мала можливість **встановлення патчів** для виправлення виявлених помилок та вразливостей.

## Види операційних систем

1. **Операційні системи для ПК**
Можна вибрати операційну систему із сімейства Windows або дистрибутив Linux.

2. **Операційні системи для мобільних пристроїв**
Це ОС для смартфонів і планшетів, такі як Android та iOS. HarmonyOS, розроблена компанією Huawei, є частиною їхньої стратегії створення інтегрованої екосистеми пристроїв. Google Fuchsia є відносно новою операційною системою, яка відрізняється від інших продуктів Google, таких як Android та Chrome OS.

3. **Операційні системи для серверів**
Ці ОС використовуються на серверах для надання послуг і ресурсів мережі.

4. **Вбудовані операційні системи**
Вони використовуються у вбудованих системах, таких як мікроконтролери, смарт-пристрої, автомобільні системи управління і багато інших.

### Операційні системи для ПК (персональних комп'ютерів)

Windows

Linux

macOS

**Windows** — це одна з найпоширеніших операційних систем для ПК. Вона розроблена корпорацією Microsoft. Windows надає зручний інтерфейс, велику кількість додатків і підтримку для багатьох пристроїв. Версії Windows включають Windows 10 і Windows 11.

**Linux** — це безкоштовна і відкрита операційна система, яка має безліч дистрибутивів (версій). Вона широко використовується як для особистих, так і для професійних цілей. Декілька популярних дистрибутивів Linux включають Ubuntu, Fedora і Debian.

**macOS** — операційна система, розроблена Apple для комп'ютерів Mac. Вона відома своєю стабільністю та дизайном; macOS працює тільки на обладнанні Mac.

## Операційні системи для мобільних пристроїв

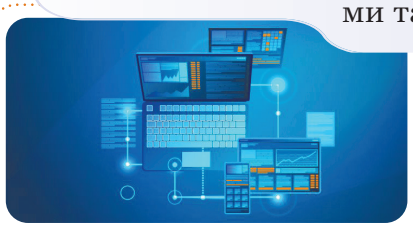
Android

iOS

**Android** — операційна система для смартфонів і планшетів, розроблена Google. Вона відома своєю великою кількістю додатків, включаючи Google Play Store.

**iOS** — операційна система, розроблена Apple для iPhone та iPad. Вона відома своєю безпекою, високоякісними додатками та екосистемою Apple.

**Програмне забезпечення (ПЗ)** — це сукупність програм, що використовуються для роботи з комп'ютерами та пов'язаними з ними пристроями.



- Десктопні програми
- Застосунки (Додатки)
- Онлайнові версії програм

### Десктопні програми

**Десктопні програми** — це програми, які встановлюються на локальному комп'ютері та працюють там без потреби постійного підключення до мережі «Інтернет».

### Переваги

- *Висока продуктивність.* Десктопні програми зазвичай працюють швидше, оскільки вони використовують ресурси комп'ютера напряму.
- *Розширені функції.* Вони надають більше можливостей і функцій порівняно з онлайновими версіями або застосунками.
- *Робота без інтернет-підключення.* Немає потреби у постійному доступі до інтернету для користування десктопними програмами.

### Недоліки

- *Встановлення та оновлення.* Вимагає встановлення та оновлення програм на кожному комп'ютері окремо.
- *Локальний доступ.* Доступ до даних обмежений локальним комп'ютером, що ускладнює спільну роботу та доступ до даних з інших пристроїв.

## Застосунки (або додатки)

**Застосунки (або додатки)** — це програми, які встановлюють на мобільних пристроях, таких як смартфони та планшети.

Вони призначені для виконання конкретних завдань або функцій. Прикладами застосунків є соціальні мережі, месенджери, ігри, фото- та відеоредактори, GPS-навігатори та багато інших.

### Переваги

- *Мобільність.* Застосунки розроблені для роботи на мобільних пристроях і надають зручний доступ до послуг у русі.
- *Інтуїтивний інтерфейс.* Зазвичай мають спеціально розроблений інтерфейс для сенсорних екранів.
- *Інтеграція.* Дозволяють використовувати функції пристрою, такі як камера, GPS та інші.

### Недоліки

- *Залежність від платформи.* Застосунки розробляються під певну платформу (iOS, Android, Windows Phone), що може обмежити їхню доступність.
- *Оновлення.* Необхідно встановлювати оновлення через магазини додатків.

## Онлайнові версії програм

**Онлайнові версії програм** — це програми, які працюють через веббраузер і не вимагають встановлення на локальний комп'ютер.

### Переваги

- *Доступ з будь-якого місця.* Користувачі можуть працювати з даними та програмами з будь-якого пристрою з інтернет-підключенням.
- *Автоматичні оновлення.* Перевагою є автоматичне оновлення програм без необхідності встановлення оновлень.
- *Колективна робота.* Дозволяє користувачам спільно працювати над документами та даними через інтернет.

### Недоліки

- *Залежність від інтернету.* Для користування потрібне постійне підключення до інтернету.
- *Обмежена функціональність.* Деякі онлайнові версії програм можуть мати обмежену функціональність порівняно з десктопними версіями.
- *Приватність та безпека.* Важливо бути обережним щодо обробки особистих даних та забезпечення їх конфіденційності в онлайнових програмах.

## Практична робота №4

**Індивідуальна пошукова робота на одну з тем:**

- ТОП 5 десктопних програм для навчання. Призначення, інсталяція, ліцензія та умови використання.
- ТОП 5 застосунків для навчання. Призначення, інсталяція, ліцензія та умови використання.
- ТОП 5 онлайнових програм для навчання. Призначення, інсталяція, ліцензія та умови використання.

Середовище виконання **Google Docs**

***Технічні вимоги:***

- 4-5 сторінок;
- єдиний стиль форматування документа на Ваш вибір;
- кілька зображень;
- покликання на програмне забезпечення;
- надіслати на електронну адресу Вашій вчительці/Вашому вчителеві.

## Домашнє завдання

Виконати індивідуальну пошукову роботу за аналогічними вимогами та надіслати Вашій вчительці/Вашому вчителеві на електронну адресу.

# Браузер. Порівняння браузерів. Налаштування. Розширення браузера. Облікові записи браузера

Тема 5

**Ключові поняття**

**Веббраузер**
**Автозаповнення**
**Анонімний режим**
**Управління паролями**
**Синхронізація**

## Веббраузер

**Веббраузер** — це програмне забезпечення, що дозволяє користувачеві переглядати та взаємодіяти з вебсторінками в інтернеті.

Браузер виконує ключову роль у відображенні вмісту в інтернеті та забезпечує зручний доступ до різноманітної інформації.

Призначення веббраузера:

*Перегляд вебсайтів.* Веббраузер дозволяє переглядати сторінки вебсайтів. Коли вводите адресу вебсайту (**www.google.com**, наприклад) в адресний рядок браузера та натискаєте **Enter**, браузер відвідує цей сайт і показує Вам його вміст.

*Пошук інформації.* Більшість веббраузерів мають вбудовану пошукову систему (як **Google** або **Bing**), яка дозволяє шукати інформацію в інтернеті, вводячи ключові слова.

*Мультимедіа.* Браузери дозволяють дивитися відео, слухати музику та грати в ігри безпосередньо в інтернеті без необхідності завантажувати додаткові програми.

## Основні характеристики веббраузера

1. **Користувацький інтерфейс**

   Адресний рядок: місце для введення вебадреси (URL).
   Кнопки **Назад** та **Вперед**: для навігації між сторінками.
   Кнопка **Оновити**: поновлення вмісту поточної сторінки та інші елементи.

2. **Вкладки**

   Можливість відкривати кілька сторінок в одному вікні.
   Зручне перемикання між вкладками для організації робочого простору.

3. **Закладки**

   Зберігання та організація улюблених вебсайтів.
   Швидкий доступ до часто відвідуваних ресурсів.

4. **Історія перегляду**

   Запис останніх відвіданих вебсторінок.
   Можливість повертатися до попередніх сторінок.

5. **Розширення та додатки**

   Розширення для розширення функціоналу браузера.
   Додатки для різних задач (рекламний блокер, менеджер паролів тощо).

6. **Приватність та безпека**

   Режим приватності для анонімного перегляду.
   Безпека від шкідливих програм та загроз.

7. **Автоматичне оновлення**

   Регулярні оновлення для забезпечення безпеки та нових функцій.

## Різновиди браузерів

### Настільні браузери

*Призначення.* Настільні браузери розроблені для використання на комп'ютерах та ноутбуках зі стаціонарними операційними системами, такими як Windows, macOS, Linux та інші.

Зазвичай мають багатий функціонал, розширені можливості керування вкладками та додатками, інтеграцію з операційною системою.

*Інтерфейс та функціонал.* Можливості синхронізації із мобільними версіями, розширеннями, менеджерами закладок та іншими розширеними налаштуваннями.

Велика бібліотека розширень для забезпечення різноманітних функцій.

*Повна версія сайтів.* Відображення вебсайтів у повному обсязі, з усіма графічними елементами та динамічним контентом.

### Мобільні браузери

*Призначення.* Мобільні браузери розроблені для використання на смартфонах та планшетах під управлінням мобільних операційних систем, таких як Android та iOS.

Оптимізовані для екранів невеликого розміру, зі спеціальною версією для мобільних платформ.

*Інтерфейс та функціонал.* Мобільні браузери призначені для оптимізації швидкодії та зменшення споживання ресурсів.

Мобільні браузери часто підтримують синхронізацію з версіями для настільних комп'ютерів.

*Мобільні версії сайтів.* Деякі вебсайти автоматично відображають мобільні версії для оптимального перегляду на мобільних пристроях.

## Налаштування браузера

Терміни, які потрібно знати:

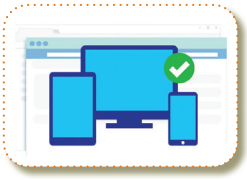
**Домашня сторінка**

Вебсторінка, яка відкривається при запуску браузера. Користувач може вибрати свою улюблену сторінку або пошуковик.

**Тема оформлення**

Зовнішній вигляд браузера, який містить кольорову палітру та стиль. Можна обрати свою тему та фон.

**Протокол https**



Захищений протокол, що забезпечує безпечний обмін даними між браузером та вебсайтом.

**Кеш та кукіси**

Тимчасове зберігання даних, що полегшує швидкий доступ до вебсайтів.

**Автозаповнення**

Функція браузера, яка автоматично заповнює форми на вебсайтах основною інформацією користувача.

Браузер запам'ятовує введені користувачем дані та пропонує їх при наступному введенні на подібних вебсайтах.

Користувач може переглядати, редагувати та видаляти збережені дані в налаштуваннях браузера.

Збереження приватної інформації для автозаповнення може бути ризиком без захищеного доступу до браузера.

Інформація, яку можна автоматично заповнити:

*Ім'я та прізвище:* Ваші особисті дані.
*Адреса:* Постійна адреса.
*Електронна пошта:* Адреса електронної пошти.
*Номер телефону:* Ваш номер телефону.

## Управління паролями

Браузер може зберігати паролі для різних вебсайтів та автоматично їх вводити при наступних відвідуваннях.

Під час введення пароля на сайті браузер може запропонувати зберегти його для подальшого використання.

При наступних входженнях на сайт браузер може автоматично вводити ваші збережені паролі.

Управління автозаповненням та паролями спрощує процес входження на вебсайти та гарантує безпеку Ваших особистих даних. Але варто подбати про безпеку своєї приватної інформації.

Браузер може запропонувати зберегти пароль при його введенні, а потім автоматично вводити його при наступних входженнях на сайт.

## Анонімний режим

У режимі інкогніто браузер не зберігає історію перегляду, файли cookie або дані автозаповнення.

Дані автозаповнення, введені в анонімному режимі, не зберігаються для майбутнього використання.

Використовується для перегляду вебсайтів, не залишаючи слідів у історії браузера.

В анонімному режимі браузер не записує в історію відвідування вебсайтів жоден збережений пароль чи дані автозаповнення форм.

Особливий режим роботи, призначений для приватного та анонімного перегляду вебсайтів.

### Переваги синхронізації даних

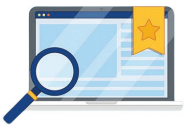
1. **Постійний доступ**
   Закладки та інші дані завжди доступні на будь-якому підключеному пристрої.

2. **Зручність та ефективність**
   Синхронізація забезпечує зручний та ефективний перехід між різними пристроями без втрати важливих даних.

3. **Безпека та захист**
   Більшість браузерів використовують шифрування для захисту синхронізованих даних.

## Закладки та диспетчер закладок

**Диспетчер закладок** — інструмент у браузері, який дозволяє користувачам організовувати, зберігати та керувати своїми закладками або вибраними вебсайтами.



Диспетчер закладок — це потужний інструмент для організації та збереження Ваших улюблених вебсайтів.

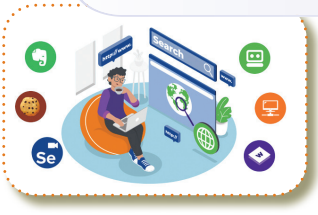
Організація закладок за допомогою створення папок для групування відповідно до тем чи категорій.

Можливість редагувати назви та адреси закладок, а також видаляти непотрібні закладки.

Більшість браузерів використовують шифрування для захисту синхронізованих даних.

## Розширення браузера

**Розширення браузера** — це додаткові програми, які розширюють функціональність вебпереглядача, додаючи нові можливості, інструменти та сервіси.




Зазвичай Ви можете встановити розширення з магазину розширень свого браузера, такого як Chrome Web Store для Google Chrome, Mozilla Add-ons.

Після встановлення розширення Ви знаходите його значок на панелі інструментів браузера. Ви можете використовувати цей значок для доступу до налаштувань та управління розширенням.

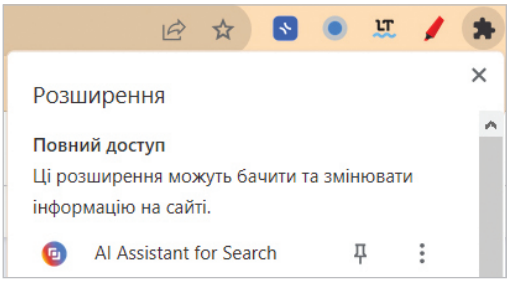
Більшість браузерів використовують шифрування для захисту синхронізованих даних.

Значки розширень розташовані праворуч від адресного рядка.

1. Відкрийте браузер Chrome на комп'ютері.
2. Перетягніть піктограму на потрібне місце.

Щоб зробити приховані розширення видимими, потрібно виконати такі дії:

1. Натисніть піктограму **Розширення** .
2. Знайдіть приховане розширення.
3. Натисніть на значок **Закріпити** .



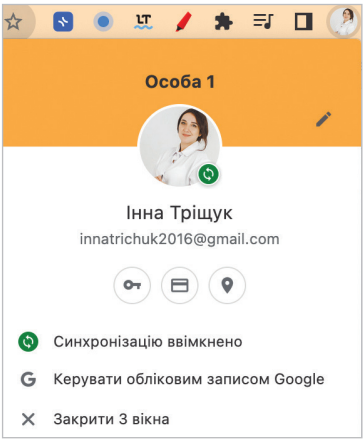

**Меню розширень**

Ви можете будь-яке розширення закріпити на панелі розширень, перемістити, переглянути додаткову інформацію про розширення.

У вебмагазині Chrome можна знайти багато корисних додатків, розширень і тем для вебпереглядача **Google Chrome**. Нижче наведено поради, як працювати з цим сервісом.

## Обліковий запис браузера

**Обліковий запис браузера** — це електронний обліковий запис, пов'язаний з конкретним браузером, який забезпечує доступ до різних функцій та сервісів браузера, таких як синхронізація даних, зберігання паролів, історії та інші персоналізовані налаштування.




Обліковий запис дозволяє синхронізувати закладки, історію перегляду, паролі та інші дані між різними пристроями користувача.

Зберігає налаштування браузера, теми та інші персоналізаційні елементи.

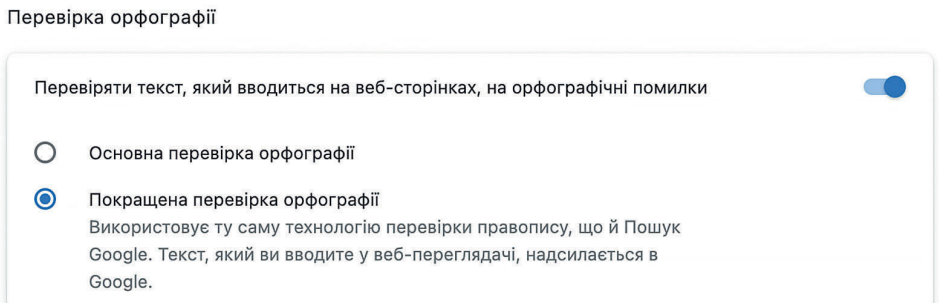
## Практична робота №5

### Частина 1. Налаштовуємо браузер

1. Відкрийте браузер **Google Chrome**.
2. Відкрийте **Налаштування**:

Вкладка **Мови**

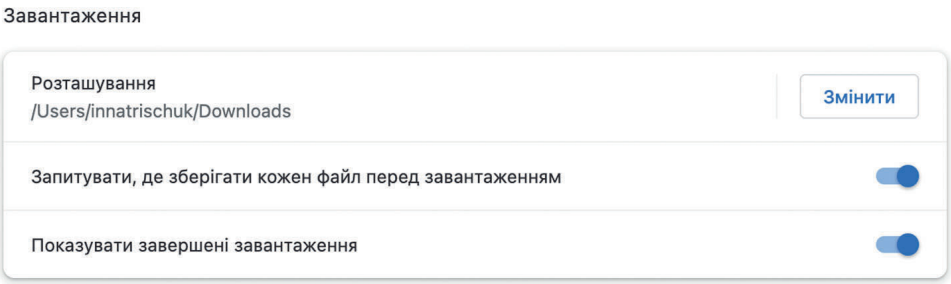
3. В налаштуваннях мови — перевірте вибрані мови. Повинна бути українська мова.
4. Увімкніть покращену перевірку орфографії.




5. В налаштуваннях **Google перекладача** — оберіть **Завжди перекладати на українську мову**.

Вкладка **Завантаження**

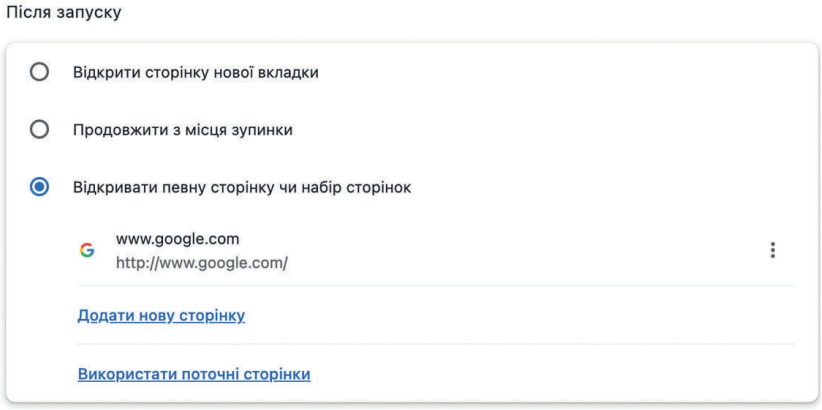
6. Оберіть папку, в яку завантажуватимете файли.
7. Оберіть, чи потрібно нам запитувати перед кожним завантаженням місце збереження файлів.
8. Оберіть, чи варто показувати повідомлення про вже завантажені файли.



**9.** Вкладка **Після запуску**.

**10.** Оберіть **Відкривати певну сторінку чи набір сторінок**.

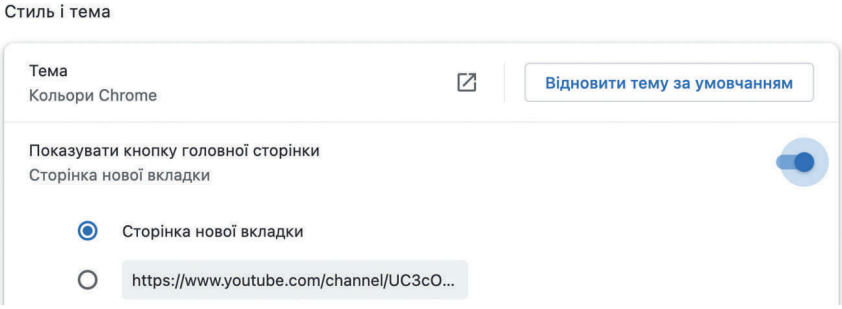
**11.** Укажіть 3 стартові сторінки на Ваш вибір.




**12.** Перевірте, чи пошукова система, яка використовується в адресному рядку — **Google**. Якщо інша — змінити на **Google**. Налаштуйте стиль **Google Chrome**.

Вкладка **Стиль та тема**

**13.** Додайте кнопку **Головної сторінки** та налаштуйте її вміст на Ваш вибір.




**14.** Оберіть тему з галереї тем **Google**.

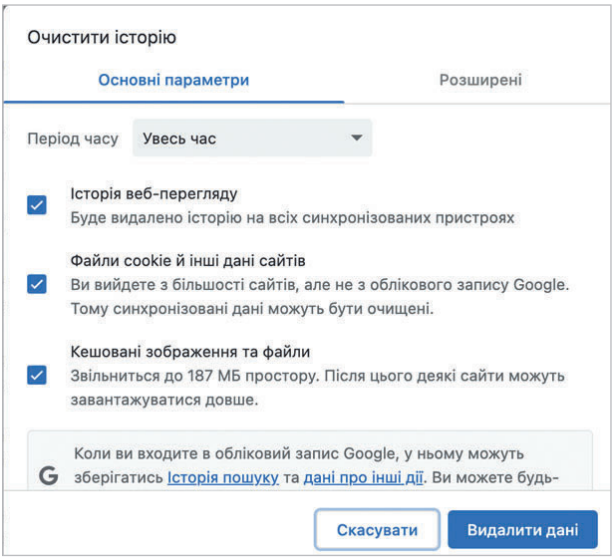
**15.** Обов'язково увімкніть панель закладок.

**16.** Всі інші параметри налаштуйте на Ваш вибір.

- Розмір шрифту
- Шрифти
- Масштабування сторінки

### Вкладка **Конфіденційність та безпека**

**1.** Очистіть історію.




**2.** У налаштуваннях **Безпеки** оберіть покращений захист.

### Вкладка **Автозаповнення та паролі**

**1.** У менеджері паролів перегляньте паролі (за потреби видаліть).

**2.** У налаштуваннях оберіть **Пропонувати зберегти паролі** та вимкніть.

### Вкладка **Ви** й **Google**

**1.** Увійдіть у свій шкільний обліковий запис (якщо шкільний — лишаємо шкільний).

**2.** Оберіть **Синхронізація** та налаштуйте **Дані для синхронізації**.

**3.** Налаштуйте профіль **Google.**

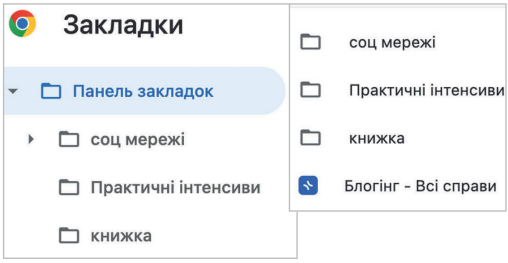
- Додайте назву свого профілю **Chrome**
- Оберіть тему
- Оберіть аватар

## Частина 2. Робота з диспетчером закладок

**1.** Відкрийте диспетчер закладок.

**2.** Створіть 3 нові папки (назви на Ваш вибір).

3. Додайте закладки в нові папки на Ваш вибір.
4. Розмістіть папки на панелі закладок.
5. Редагуйте назву папки.
6. Редагуйте закладку.




## Частина 3. Робота з розширеннями

1. Розширення **PageMarker** — дозволяє робити помітки одразу на вебсайтах та зберігати скріншот разом з помітками.

Page Marker - Draw on Web

Інсталюйте та використайте.

2. Розширення **Коректор** граматики та орфографії — **Language Tools**.
Інсталюйте та використайте.

## Домашнє завдання

Порівняльна характеристика браузерів.
В **Google документі** створити табличку за зразком.
За допомогою мережі «Інтернет» заповнити таблицю.
Надати доступ Вашій вчительці/Вашому вчителю за допомогою електронної адреси.

| Офіційний сайт | Google Chrome | Microsoft Edge |
|---|---|---|
| Логотип | | |
| Чи є мобільна версія | | |
| Особливості браузера | | |
| Чи є автозаповнення даних | | |

**Тема 6** **Хмарні сервіси. Моделі надання хмарних сервісів (IaaS, PaaS, SaaS). Рівні доступу до мережевих документів. Інтеграція сервісів. Резервна копія файлів. Синхронізація. Обмін файлами**

Ключові поняття

**Хмарні сервіси**
**Спільний доступ до документа**
**Інтеграція сервісів**
**Синхронізація в обліковому записі**
**Обмін файлами**

## Хмарні сервіси

**Хмарні сервіси** — це онлайн-платформи та сервіси, які дозволяють зберігати, обробляти та обмінюватися інформацією через мережу «Інтернет».

Вони отримали назву «хмарні» через символічне зображення хмари, що вказує на те, що дані не зберігаються на одному конкретному пристрої, а доступні з будь-якого пристрою, підключеного до мережі.

### Основні переваги хмарних сервісів

- **Доступність.** Ваші дані доступні в будь-якому місці та в будь-який час, де є мережа «Інтернет».
- **Зручність спільної роботи.** Можливість одночасно працювати над проєктами та документами з іншими користувачами.

- **Зберігання та резервне копіювання.** Дані зберігаються в безпечних центрах обробки даних, існує менше ризику втрати інформації.
- **Економія простору на пристрої.** Не потрібно завантажувати великі файли на свій пристрій, оскільки вони можуть бути збережені в хмарі.

## Основні хмарні сервіси

Google Drive

**Google Drive** є частиною екосистеми **Google** та пропонує безкоштовне зберігання файлів у хмарі.

*Зберігання та обмін документами:* можливість завантажувати та зберігати документи, презентації, архіви тощо.

*Фотографії та відео:* завантаження та спільний доступ до фотографій та відео.

*Робота в режимі реального часу:* колективна робота над файлами в режимі реального часу.

Microsoft OneDrive

**OneDrive** є хмарним сховищем, розробленим **Microsoft**, та інтегрованим із пакетом офісних програм **Office 365**.

*Офісні документи:* зберігання, редагування та обмін офісними документами, такими як **Word**, **Excel**, **PowerPoint**.

*Файли та фотографії:* завантаження та синхронізація файлів, фотографій із пристроїв.

Dropbox

**Dropbox** — це хмарний сервіс, що надає можливість зберігання файлів та спільної роботи з ними.

*Зберігання та синхронізація:* завантаження файлів та їх синхронізація між пристроями.

*Спільна робота:* легка спільна робота над файлами, можливість надсилання запрошень на спільний доступ.

## Моделі надання хмарних сервісів

**1** **Infrastructure as a service (IaaS) — інфраструктура як сервіс**

IaaS — це модель хмарних сервісів, яка надає віртуальні ресурси для створення, розгортання та управління інфраструктурою без необхідності володіти та управляти фізичним обладнанням.

**Amazon Web Services (AWS)**

AWS дозволяє користувачам орендовувати віртуальні сервери, зберігати дані та використовувати інші інфраструктурні ресурси безпосередньо через мережу «Інтернет».

**2** **Platform as a service (PaaS) — платформа як сервіс**

PaaS — це модель, яка надає платформу для розробки, тестування та розгортання програм, спрощуючи процес створення програмного забезпечення.

**Google App Engine**

Google App Engine — це платформа для розробки та розгортання вебдодатків, яка надає інфраструктуру та середовище для створення високомасштабованих додатків без необхідності вручну управляти інфраструктурою.

**3** **Software as a service (SaaS) — програмне забезпечення як сервіс**

SaaS — це модель, яка надає готове програмне забезпечення через мережу «Інтернет» без необхідності встановлення чи обслуговування на власних пристроях.

**Microsoft 365**

Microsoft 365 — це пакет офісних програм, таких як Word, Excel та PowerPoint, які надаються як послуга через хмару. Користувачі можуть працювати з документами в хмарі, не встановлюючи їх на власний комп'ютер.

## Рівні доступу до мережевих документів

Рівні доступу до мережевих документів (наприклад, в **Google** сервісах, таких як **Google docs**, **Google sheets**, **Google slides**) можуть бути різними і залежать від конфігурації доступу, яку Ви встановлюєте для кожного документа.

1 **Приватний (Private)**

Тільки Ви маєте доступ до документа.

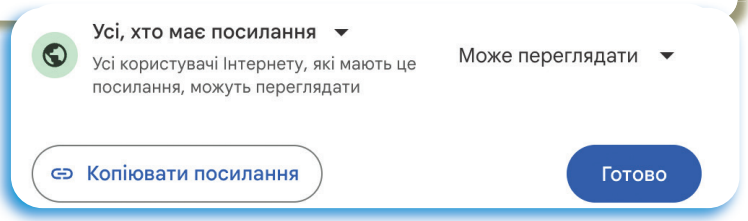
Інші користувачі не можуть переглядати, редагувати або коментувати документ, якщо Вони не отримають від Вас окремого доступу.


2 **Загальний доступ (Public)**

Кожен, хто має посилання на документ, може переглядати, редагувати або коментувати його. Це залежить від прав доступу, які Ви надасте.

3 **Доступ за посиланням (Link access)**

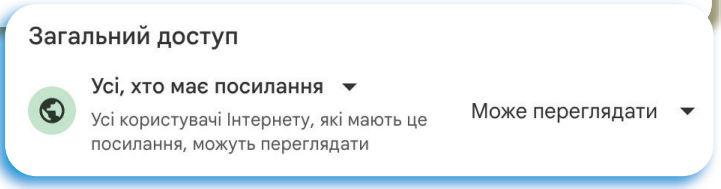
Доступ до документа можна отримати за посиланням, але без спеціальних прав доступу.




4 **Лише перегляд (View only)**

Інші користувачі можуть тільки переглядати вміст документа, але не можуть його редагувати.

Цей рівень доступу зручний для широкого розповсюдження інформації без можливості внесення змін.



**5** **Коментування (Comment only)**

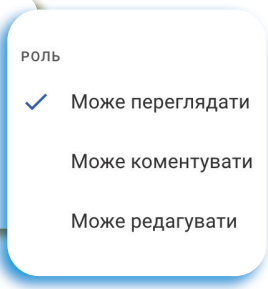
Інші користувачі можуть переглядати та залишати коментарі, але не можуть редагувати основний текст документа.

Це дозволяє іншим користувачам висловлювати зауваження та давати поради.

**6** **Редагування (Edit)**

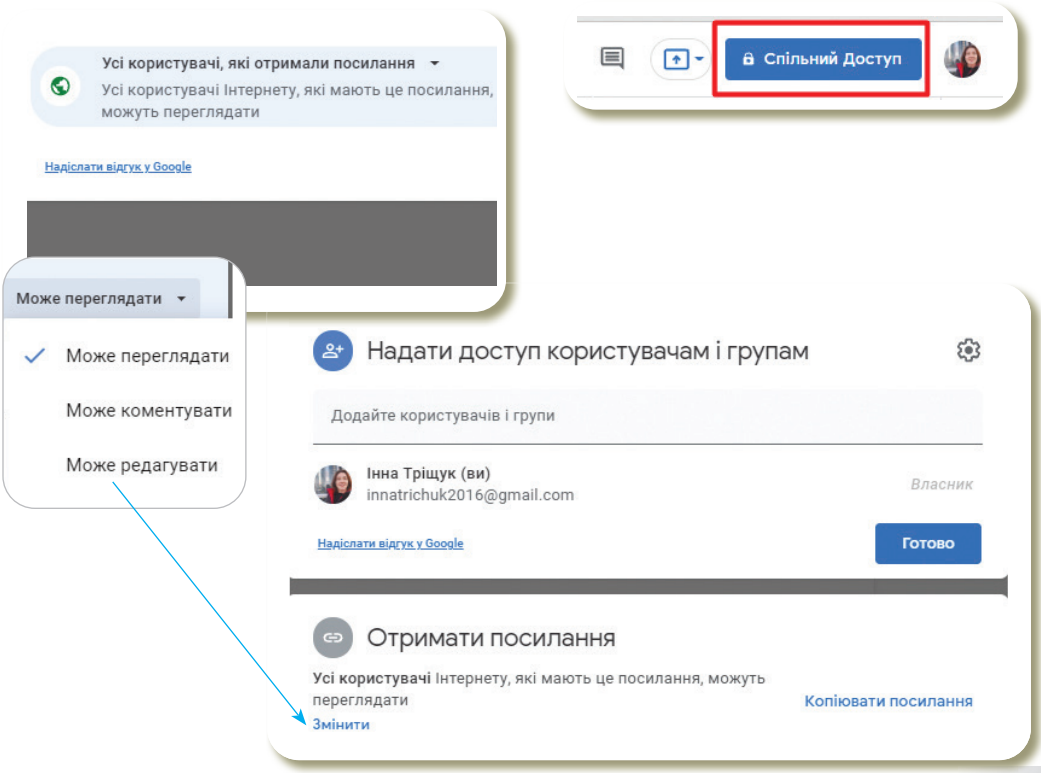
Інші користувачі можуть редагувати вміст документа.

Вони можуть вносити зміни в текст, таблиці, графіку і т.д.




## Спільний доступ до документа

Коли Ви надаєте доступ до файлів на **Google диску**, то можете вказати, які дії користувачі зможуть виконувати з ними: редагувати, коментувати або лише переглядати.



## Інтеграція сервісів

**Інтеграція сервісів** означає об'єднання різних програм або сервісів для полегшення обміну інформацією та взаємодії між ними.

Інтегруючи **Google Диск** з **Google Документами**, Ви можете створювати та редагувати документи прямо у своєму сховищі файлів.

У **Google Документах** виберіть **Вставити** — **Документ Google Диска** для вибору файлів із свого Диска.

**Переваги інтеграції сервісів:**

- *Ефективність:* зменшення потреби перемикатися між різними сервісами.
- *Зручність користувача:* створення єдиного інтерфейсу для взаємодії з різними сервісами.

**Резервне копіювання даних** — це процес створення копій Ваших важливих файлів і зберігання їх на іншому місці, яке відмінне від Вашого основного комп'ютера.

Це робиться для того, щоб в разі втрати даних (наприклад, через віруси або випадкового видалення) Ви могли відновити свої файли.

Використовуйте зовнішній жорсткий диск або хмарне сховище (наприклад, **Google Drive**, **Dropbox**) для збереження копій Ваших файлів.

**Резервне копіювання даних облікового запису Google** — це важливий процес, який допомагає зберегти Вашу інформацію та дані, пов'язані з **Google**, в безпечному місці, щоб уникнути їх втрати у разі втрати доступу до облікового запису, видалення даних або інших негараздів.

## Синхронізація в обліковому записі

**Синхронізація в обліковому записі** — це процес, за допомогою якого дані, пов'язані з Вашим обліковим записом, оновлюються автоматично на всіх пристроях і платформах, на яких Ви використовуєте цей обліковий запис.

Наприклад, якщо Ви додаєте новий контакт до свого телефону або змінюєте запис в календарі через вебінтерфейс **Google**, ці зміни автоматично синхронізуються з усіма пристроями, на яких Ви використовуєте свій обліковий запис **Google**.

Це дозволяє Вам отримувати доступ до своїх даних, таких як закладки, історія, паролі, контакти, календарі та налаштування на всіх Ваших пристроях.

Ви можете вибрати синхронізацію всіх даних або лише деяких. Після ввімкнення синхронізації Ваші дані будуть автоматично синхронізуватися на всіх Ваших пристроях, які використовують той самий обліковий запис **Google**.

**Обмін файлами** — це передача файлів між різними користувачами чи пристроями. Це може означати завантаження файлів на хмарне сховище або поширення файлів за допомогою мережі «Інтернет».

### Методи обміну файлами

| | |
|---|---|
| Електронна пошта | Користувачі можуть відправляти файли одне одному через електронні листи як вкладення. |
| Месенджери | Популярні месенджери, такі як **WhatsApp**, **Telegram** чи **Viber***, дозволяють Вам відправляти файли своїм контактам. |
| Хмарні сховища | Сервіси, такі як **Google Drive**, **Dropbox** чи **OneDrive**, дозволяють користувачам завантажувати файли в хмарне сховище та ділитися доступом до них. |

**Ян Кум (Jan Koum)** є одним з співзасновників WhatsApp, популярного месенджера для смартфонів. Його історія є втіленням «американської мрії»: він народився та виріс в Україні, а в 1990-х роках емігрував до Сполучених Штатів зі своєю матір'ю. Перед створенням WhatsApp Кум працював у компанії Yahoo, де він зустрів свого майбутнього бізнес-партнера Браяна Ектона.

*Слід мати на увазі, що розробники месенджерів Telegram та Viber належать до ворожих нам країн.

## Практична робота №6

### Частина 1. Колективна робота

**1.** Колективна робота з можливістю коментування.

*Примітка для вчительки/вчителя:* надати доступ учням/ученицям з можливістю коментування документа.

*Алгоритм дій для вчительки/вчителя*

- Копіювати документ на свій **Google диск**.
- Надати доступ з можливістю коментування для учнів/учениць.

**2.** Колективна робота з можливістю редагування.

*Примітка для вчительки/вчителя:* надати доступ учням/ученицям з можливістю редагування документа.

*Алгоритм дій для вчительки/вчителя*

- Копіювати документ на свій **Google диск**.
- Надати доступ з можливістю редагування для учнів/учениць.

### Частина 2. Обмін файлами

**1.** Відкрийте **Google диск**.

**2.** Створіть папку на **Google диску Ваше прізвище та ім'я. Зображення.**

**3.** Завантажте у відповідну папку кілька зображень з Робочого столу.

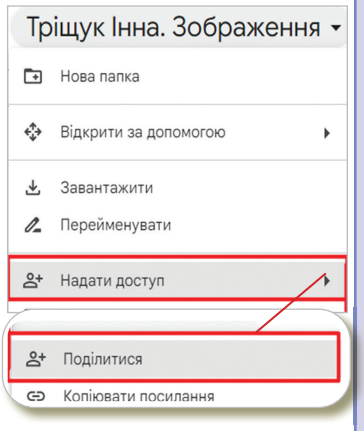
**4.** Надайте доступ до папки за допомогою електронної адреси своїм однокласникам/однокласницям та Вашій вчительці/Вашому вчителю.

**5.** Надайте можливість редагувати.

**6.** Створіть папку **Ваше прізвище та ім'я. Документи**.

**7.** Завантажте у відповідну папку кілька документів з **Робочого столу**.

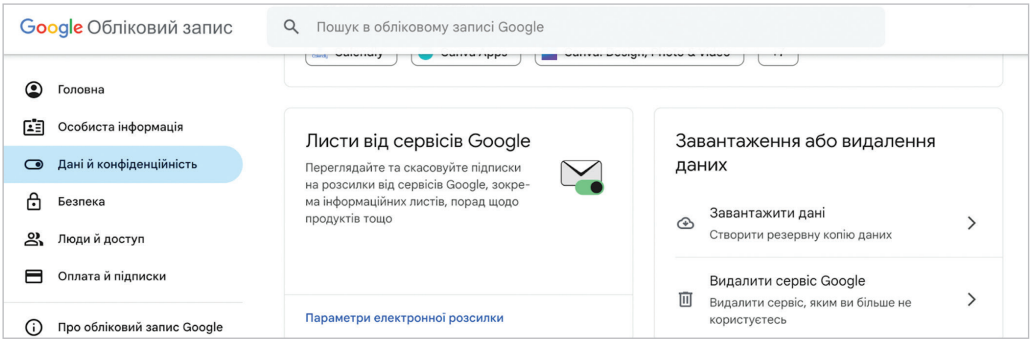
**8.** Створіть покликання з можливістю коментувати.




**9.** Покликання відправте електронною поштою своїм однокласникам/однокласницям та Вашій вчительці/Вашому вчителеві.

## Частина 3. Резервне копіювання

1. Увійдіть у свій обліковий запис **Google**.
2. Оберіть команду **Керувати обліковим записом** та **Дані та конфіденційність.**
3. Оберіть завантаження або видалення даних та **Завантажити дані**.




4. Оберіть сервіси, які Вам потрібні з переліку.
5. Оберіть спосіб доставки — на **Google диск** / частоту — **кожних 2 місяці**.
6. Очікуйте перебіг експортування.

## Домашнє завдання

1. Зробити порівняльну характеристику хмарних сервісів за зразком.

| Назва хмарного сховища | Google диск | Microsoft OneDrive | Dropbox |
|---|---|---|---|
| Обсяг зберігання | | | |
| Вартість | | | |
| Доступ та спільне використання | | | |

2. Надіслати на електронну пошту Вашій вчительці/Вашому вчителю.

Тема 7

# Види прикладних програм: текстові та графічні редактори, електронні таблиці тощо. Власний віртуальний образ. Цифрова взаємодія, вплив на інших осіб. Налаштування облікового запису

**Ключові поняття**

**Текстові редактори**
**Електронні таблиці**
**Віртуальний образ**
**Обліковий запис**
**Авторизація**
**Верифікація**

## Текстові редактори


Доступність та платформа

Колективне редагування

Інтеграція та синхронізація

Офлайн-режим

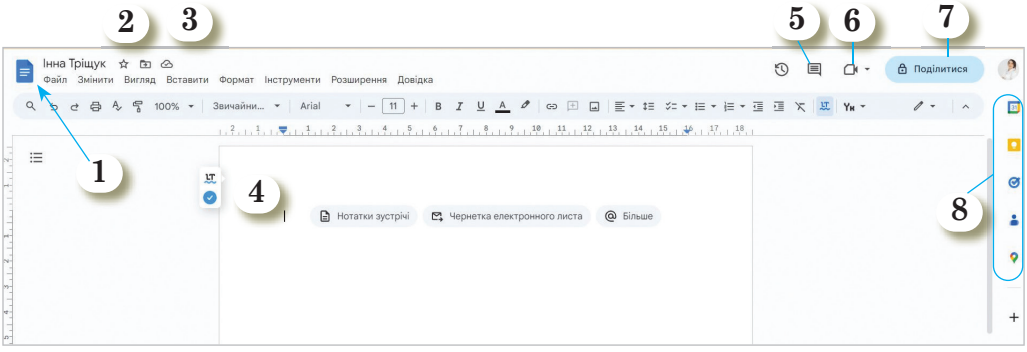
### Google Docs

- Це хмарний сервіс, який доступний через браузер. Робота відбувається онлайн, а документи автоматично зберігаються в хмарі **Google Drive**.
- Визначається можливістю одночасного колективного редагування. Користувачі можуть працювати над документом одночасно та спостерігати за змінами і коментарями інших у реальному часі.
- Інтегровано з іншими сервісами **Google**, такими як **Google Drive**, **Gmail** тощо. Забезпечує автоматичну синхронізацію документів.
- Має обмежений офлайн-режим через веброзширення та офіційні мобільні додатки.

### Microsoft Word (Microsoft 365)

- Частина офісного пакету **Microsoft Office**, доступного як онлайн-версія Office Online або в офлайн-режимі через програми для **Windows** та **MacOS**.
- У десктопній версії **Microsoft Office** — недоступне коментування та спільна робота, в онлайн-версії **Office** є колективна робота.
- Інтеграція з **Microsoft OneDrive**, **Outlook** та іншими сервісами. Також надає автоматичну синхронізацію файлів.
- Дозволяє працювати в офлайн-режимі за допомогою програм для настільних комп'ютерів.

**Google Docs** — це безкоштовний хмарний сервіс для створення та редагування текстових документів в інтернеті.




1. Пункти меню.
2. Розташування документа на **Google диску**. У **Google Docs** всі документи створюються і зберігаються в **Google Drive**, хмарному сховищі від **Google**. Це дозволяє легко організовувати та знаходити Ваші документи, а також легко ділитися ними з іншими користувачами.
3. Автозбереження документа на **Google диску**.
4. Робоча область.

5. Переглянути історію коментарів.
6. Створити відеодзвінок для учасників, які працюють над цим документом.
7. Спільний доступ. У **Google Docs**: кілька користувачів можуть одночасно працювати над документом у режимі реального часу, бачити зміни інших учасників та спілкуватися через вбудований чат. Це забезпечує ефективну співпрацю в групі.
8. Інтеграція з іншими сервісами **Google**.

## Особливості в Google Docs

1

### Вбудований перекладач

У **Google Docs**: має вбудований перекладач, який дозволяє перекладати вибрані фрагменти тексту відразу в документі. Це особливо корисно для міжнародної співпраці або навчання мов.


2

### Функція останні зміни

У **Google Docs**: ця функція дозволяє швидко переглядати останні зміни в документі, включаючи автора та час внесення змін. Вона особливо корисна при спільній роботі над проєктами.

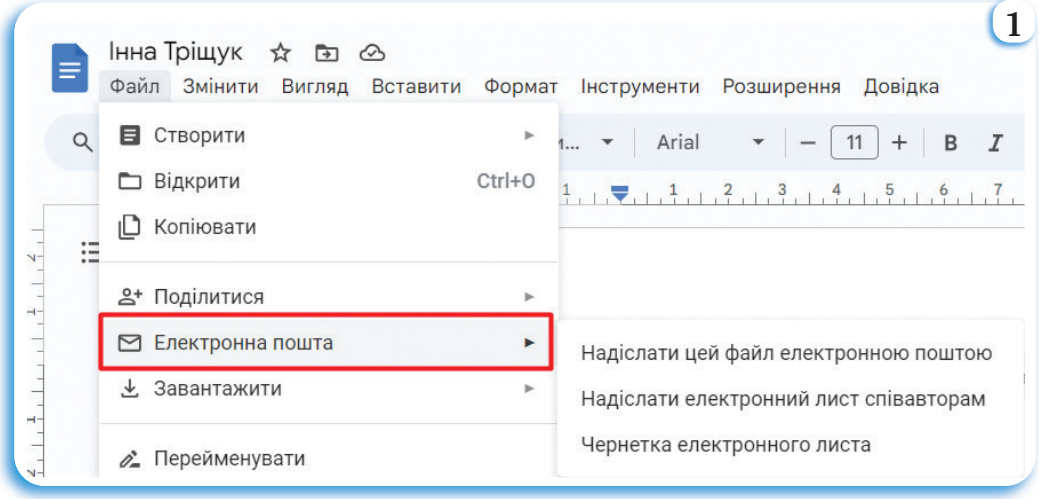

3

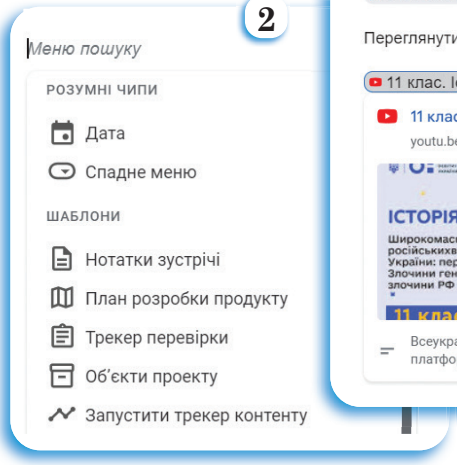
### Доступність на різних пристроях

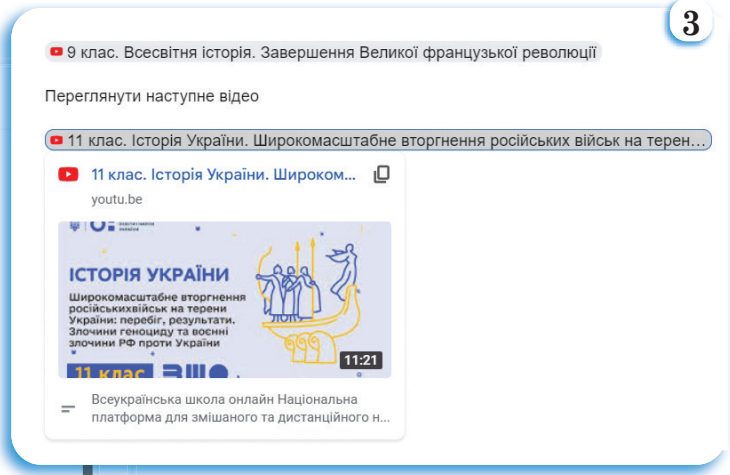
У **Google Docs**: можливість працювати з документами з будь-якого пристрою з підключенням до мережі «Інтернет» — це комп'ютер, планшет чи смартфон. Мобільні додатки для **iOS** та **Android** спрощують роботу в дорозі.

## Особливості в Google Docs

1. Надіслати посилання на документ листом просто з **Google Docs**.
2. Меню пошуку в **Google Docs** на сторінці.
3. Можливість вставляти відео безпосередньо в документ.

## Електронні таблиці

| Доступність та платформа | Колективне редагування | Інтеграція та синхронізація | Офлайн-режим |
|---|---|---|---|

### Google таблиці

- Це хмарний сервіс, доступний через браузер, інтегрований з **Google Drive**.
- Можливість спільного редагування таблиць кількома користувачами в режимі реального часу.
- Базовий, але достатній для більшості завдань функціонал. Підтримка формул, діаграм, фільтрів тощо. Інтегровано з **Google Drive** та іншими сервісами **Google**.
- Є можливість редагування таблиць в офлайн-режимі через офіційні мобільні додатки.

### Microsoft Excel (Microsoft 365)

- Є як частина офісного пакету **Microsoft Office**, так і онлайн-версії **Office Online**, доступної через браузер.
- Онлайн-версія також підтримує колективне редагування, в десктопній версії колективної роботи нема.
- Розширений набір функцій, діаграм, графіків, фільтрів, можливість автоматизації завдань за допомогою макросів. Інтеграція з **OneDrive**, **Outlook** та іншими сервісами **Microsoft**.
- Дозволяє працювати в офлайн-режимі через програми для настільних комп'ютерів.

## Віртуальний образ

**Віртуальний образ учня** — це цифровий представник реальної особи, який може взаємодіяти в інтернеті, виконувати завдання та навчатися.

Створюється для взаємодії з іншими користувачами та використання з освітньою чи розважальною метою.

## Цифрова взаємодія, вплив на інших осіб

Взаємодія між особами, яка відбувається через цифрові технології та платформи.

Це охоплює різні форми взаємодії, такі як електронна пошта, соціальні мережі, чати, віртуальні конференції та інші технології.

### Переваги

- Глобальний доступ до інформації та ресурсів.
- Зручність та швидкість взаємодії.
- Можливість підтримки та спілкування з різними культурами та групами.

### Способи цифрової взаємодії

*Чат-боти та месенджери.* Спілкування через письмові повідомлення, голосові повідомлення і т.д.

*Соціальні мережі.* Ділитися інформацією, коментувати, взаємодіяти з контентом та іншими користувачами.

*Відеоконференції.* Зустрічі та обговорення за допомогою відеозв'язку.

## Вплив цифрової взаємодії на інших осіб

**Сприяння комунікації**

Можливість легко спілкуватися та обмінюватися інформацією з іншими, навіть на великій відстані.

**Посилення взаєморозуміння**

Швидке інтернет-спілкування може сприяти швидкому розумінню точок зору та думок інших осіб.

**Збільшення соціальної активності**

Участь в онлайн-спільнотах та соціальних мережах може збільшити соціальну активність та розширити коло знайомств.

**Можливість поділитися думками та інформацією**

Кожен користувач може висловлювати свої думки, погляди та ідеї через різні цифрові канали.

## Обліковий запис

**Налаштування облікового запису** — це процес налаштування інформації про Вас у віртуальному світі, такому як ім'я, пароль, адреса електронної пошти та інші деталі.

***Важливо вказати:***

| | |
|---|---|
| Електронна пошта та Номер телефону | Додайте актуальну електронну адресу та номер телефону для відновлення доступу та отримання важливих сповіщень. |
| Пароль | Використовуйте надійний пароль, який складається з букв, цифр та спецсимволів. Змінюйте пароль регулярно. |
| Двоетапна перевірка | Увімкніть двоетапну перевірку для забезпечення додаткового рівня безпеки. Це може бути відправлення коду на телефон або використання автентифікатора. |
| Приватність | Налаштуйте рівень приватності Вашого облікового запису. Виберіть інформацію, якою ви хочете ділитися публічно та з ким. |

## Авторизація

**Авторизація** — процес входу в свій обліковий запис за допомогою ідентифікаційних даних, таких як ім'я користувача та пароль.

***Як це працює?***

*Введення ім'я користувача та пароля.* Після введення ім'я користувача та пароля на відповідних полях система перевіряє, чи правильно вони введені.

*Порівняння збережених даних.* Система порівнює введені дані з тими, які раніше були збережені під цим ім'ям користувача.

*Доступ або відмова в доступі.* Якщо введені дані збігаються зі збереженими, користувачу надається доступ до облікового запису.

## Верифікація

**Верифікація** — процес підтвердження вашої особистості або правильності інформації, яку Ви надаєте.

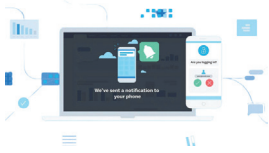

***Як це працює?***

*Введення додаткових даних.* Система може вимагати введення додаткових даних або виконання певних кроків для перевірки Вашої особистості.

*Відправлення коду або посилання.* Часто використовується відправлення спеціального коду або посилання на Вашу електронну адресу чи телефон, які Ви повинні підтвердити.

**Двофакторна автентифікація** — додатковий рівень безпеки, який вимагає введення двох видів ідентифікаційних даних для підтвердження особистості під час входу в обліковий запис.

Після введення пароля система вимагає додаткового коду або використання спеціального пристрою для підтвердження Вашої особистості.



Вам може бути відправлено SMS-код на Ваш телефон або Ви можете використовувати автентифікатор на телефоні для генерації унікального коду.

Введіть отриманий код або використайте пристрій для завершення входу в обліковий запис.

***Чому це важливо?***

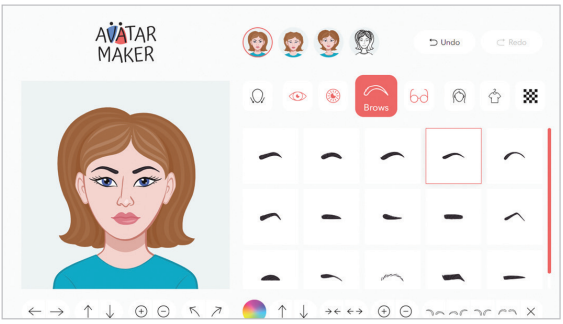
**Безпека.** Зловмисникам буде важко отримати доступ до Вашого облікового запису, навіть якщо вони дізналися Ваш пароль.

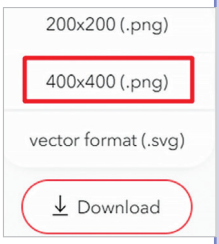
**Захист від несанкціонованого доступу.** Якщо хтось вирішить спробувати зламати Ваш обліковий запис, важко вдруге отримати додатковий код або доступ до Вашого пристрою.

## Практична робота №7

### Частина 1

**Завдання.** Створення віртуального образу.



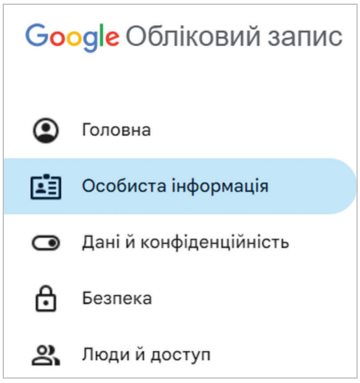




1. Перейдіть на онлайн-сервіс.
2. Оберіть налаштування свого аватару.
3. Збережіть у форматі 400x400(.png).
4. Завантажте Ваш аватар в папку **Аватар — прізвище та ім'я** на **Google диску**.
5. Надайте доступ для перегляду Вашій вчительці/Вашому вчителеві.

### Частина 2

**Завдання.** Налаштування свого облікового запису **Google**.

1. Авторизуйтеся в своєму обліковому записі **Google**.
2. Оберіть розділ **Особиста інформація**.




- Вказуємо основну інформацію:
  - ім'я,
  - дату народження,
  - стать.
- Вказуємо контактну інформацію:
  - номер телефону додатковий,
  - електронну адресу додаткову.
- Переходимо до розділу **Безпека**.
  1) Налаштовуємо двофакторну автентифікацію.
  2) Додаємо надійні пристрої.

3) Додаємо відновлення за допомогою текстового повідомлення.
4) Перевіряємо номери для відновлення та електронну пошту для відновлення.
5) Перевіряємо Ваші пристрої, якщо Ви бачите в переліку невідомий пристрій — завершіть сеанс.


## Частина 3

**Завдання.** Створити на **Google диску** документ та назвати своїм прізвищем та ім'ям. Поширити документ Вашій вчительці/Вашому вчителеві для перегляду.

1. Додайте 5 відео з **YouTube** з віртуальної школи-онлайн на Ваш вибір.
2. Напишіть кілька речень про себе та перекладіть одразу в документі.


3. Продиктуйте за допомогою голосового введення (якщо є можливість) кілька речень про свої хобі та інтереси.

## Домашнє завдання

Створити кілька аватарів своїх однокласників/однокласниць та надіслати їм на електронну пошту.

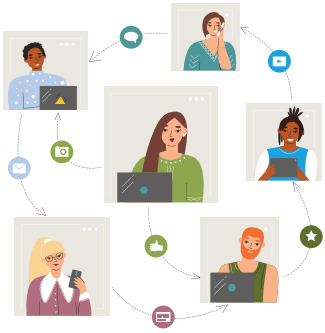
# Месенджери та відеоконференції. Правила нетикету. Соціальні мережі. Ігрові платформи. Кібербулінг

Тема 8

**Ключові поняття**

**Електронне спілкування**
**Месенджер**
**Відеоконференції**
**Соціальні мережі**
**Штучний інтелект**
**Нейромережі**

## Електронне спілкування



Завдяки інтернету, мобільному зв'язку дедалі більше людей використовують електронне спілкування.

**Computer-mediated communication, інтерактивне спілкування** — це особлива форма комунікації, в процесі якої відбувається спілкування людей одне з одним в мережі «Інтернет» і здійснюється шляхом обміну знаковими та/або мультимедійними повідомленнями.

Воно дозволяє передавати текстові повідомлення, зображення, відео та аудіо, а також дає можливість спілкуватися в режимі реального часу через онлайн-платформи.

## Переваги

### Зручність

Миттєвість електронного спілкування дозволяє швидко обмінюватися інформацією та відповідати на повідомлення без затримок. Від текстового спілкування до відеоконференцій — різноманітність способів спілкування дає можливість обрати найзручніший для конкретної ситуації.

### Доступність

Можливість спілкування з людьми з будь-якого куточка нашої планети, що робить світ меншим та більш доступним.

### Можливість віддаленого навчання та роботи

Ефективне використання для віддалених робочих чи навчальних процесів, особливо в умовах глобальних змін та пандемій.

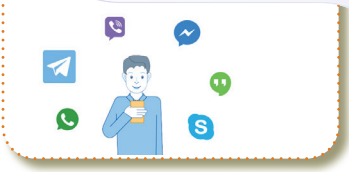
### Електронна документація

Можливість зберігання та відстеження історії спілкування, що полегшує організацію інформації.


У мережі «Інтернет» існує безліч спільнот. Їх називають інтернет-спільноти, вебспільноти або онлайн-спільноти.

## Месенджер

**Месенджер** — це програма чи застосунок для обміну повідомленнями та фотографіями через інтернет.



Використання месенджерів зазвичай безкоштовне, що дозволяє спілкуватися без додаткових витрат.

| | |
|---|---|
| Контакт | Особа, яка додана до списку друзів чи колег в месенджері та з якою можна спілкуватися. |
| Стікери | Мультимедійні зображення, які можна використовувати для вираження емоцій у чатах. |
| Емодзі | Маленькі зображення або смайлики, які використовуються для передавання емоцій у текстовому спілкуванні. |
| Груповий чат | Чат, у якому можуть брати участь багато користувачів одночасно, призначений для обговорення загальних тем. |
| Аудіовиклик | Можливість здійснювати голосові дзвінки між користувачами месенджера. |
| Відеовиклик | Можливість здійснювати відеодзвінки, під час яких можна бачити співрозмовника за допомогою вебкамери. |
| Таємний чат | Приватний чат з підвищеним рівнем шифрування, де повідомлення можуть самознищуватися. |
| Оновлення статусу | Можливість додавати короткі текстові оголошення чи статуси, які бачать контакти в списку друзів. |

Використання месенджерів вимагає дотримання певних правил та етикету, щоб забезпечити ефективне та ввічливе спілкування.

**Зберігайте культуру спілкування**
Спілкуйтеся з іншими користувачами з повагою та увагою. Уникайте використання нецензурної лексики та образливих висловлювань.

**Не розголошуйте особисту інформацію**
Уникайте надмірного розголошення особистої інформації, такої як адреса, номер телефону. Зберігайте конфіденційність своїх особистих даних.

**Будьте обережні зі стрікерами та емодзі**
Використовуйте стікери та емодзі розумно, уникаючи неприязних чи образливих висловлювань. Вони повинні допомагати у вираженні емоцій, але не ображати інших.

**Уникайте спаму та великої кількості повідомлень**
Не надсилайте багато повідомлень підряд, особливо якщо не отримали відповіді. Уникайте розсилання непотрібної інформації.

## Відеоконференції

**Відеоконференція** — це онлайн-зустріч між людьми, які перебувають в різних місцях, за допомогою відеокамер та аудіозв'язку.

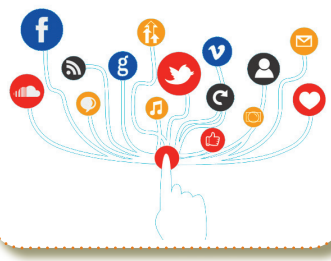
- Програмне забезпечення
- Вебкамера
- Мікрофон та динамік
- Інтернет

- Програмне забезпечення для відеоконференцій може містити спеціальні додатки, платформи або сервіси, які забезпечують функції відео- та аудіоз'єднання. Платформи для відеоконференції — **Zoom**, **Microsoft Teams**, **Google Meet** або інші.

- Це пристрій, який записує або передає відеозображення учасників конференції. Більшість сучасних пристроїв мають вбудовані вебкамери.
- Вбудований мікрофон і динамік у Вашому пристрої чи зовнішні аудіопристрої дозволять Вам чути та говорити під час конференції.
- Стабільне та достатньо швидке інтернет-з'єднання важливе для безперебійного обміну відео- та аудіосигналами.

## Соціальні мережі

**Соціальні мережі** — спеціальний вебсайт або додаток, який дозволяє людям об'єднуватися та спілкуватися в інтернеті.



Головна ідея соціальних мереж — це створення віртуальних майданчиків, де користувачі можуть обмінюватися інформацією, знаходити друзів, висловлювати свої думки та взаємодіяти.

| | |
|---|---|
| Профіль | Створення особистого простору для користувача, де він може додавати інформацію про себе, фотографії, інтереси та інше. |
| Друзі та підписники | Можливість підключатися до інших користувачів, додавати їх до списку друзів чи підписників, щоб бути в курсі їхніх публікацій та спілкуватися. |
| Повідомлення та чати | Можливість особистого обміну повідомленнями з іншими користувачами або участь у групових чатах. |
| Хештеги | Використання тегів для групування та пошуку контенту за певною темою чи ключовим словом. |

**Вподобайка**

Натискання на спеціальну кнопку під певним постом чи фотографією. Якщо Вам подобається фотографія Вашого друга чи подруги, то можете біля світлини натиснути «вподобайку», щоб йому/їй було приємно.

**Коментар**

Залишення текстового повідомлення під постом чи фотографією. Якщо у Вас є щось цікаве чи приємне для відгуку на пост свого друга, то можете написати коментар, щоб поділитися своєю думкою.

**Репост**

Поширення чужого посту на своїй сторінці чи в своїй спільноті. Якщо Ви бачите цікавий пост про подію чи новину і хочете, щоб і Ваші друзі/подруги могли його побачити, Ви можете зробити репост цього посту на свою сторінку.

- **Будь ввічливим та поважай інших**
  Спілкуйся ввічливо та поважай думки інших користувачів. Уникай використання грубих чи образливих слів.
- **Зберігай приватність**
  Не ділися особистою інформацією, такою як адреса чи номер телефону, з незнайомими особами. Використовуй налаштування приватності.
- **Будь обережним із фотографіями**
  Перевір, хто бачить твої фотографії. Не публікуй занадто особисті зображення або фото без дозволу інших осіб.
- **Будь відповідальним за свої дії**
  Відповідай за свої слова та дії в інтернеті так само, як і в реальному житті. Не роби нічого, що ти не зробив би офлайн.
- **Обмеж час, витрачений в інтернеті**
  Визнач, наприклад, 30-60 хвилин на день для спілкування в соціальних мережах і відведи час на інші активності — наприклад, читання чи заняття спортом.

## Штучний інтелект

**Штучний інтелект** — це галузь комп'ютерних наук, яка ставить за мету створення програм і систем, здатних виконувати завдання, які зазвичай вимагають людського інтелекту.

Ці завдання можуть містити розпізнавання образів, розуміння мови, прийняття рішень, самонавчання та інші.

**Bard** — це експериментальний сервіс, у якому можна взаємодіяти з генеративним штучним інтелектом. Як розмовна модель штучного інтелекту, Bard може допомогти вам відшукати нові ідеї, підвищити продуктивність, а також розкрити творчий потенціал.

**Чат GPT** — це чат-бот від компанії OpenAI, який дозволяє користувачам спілкуватися зі штучним інтелектом. Бот може вести діалог з людиною різними мовами, створювати та перекладати текст і виконувати інші текстові завдання.

Нова **пошукова система ШІ** використовує технології обробки природної мови і машинного навчання, щоб зрозуміти, що шукають користувачі, а потім пропонує їм найкращі результати. Також представляє джерела пошуку.

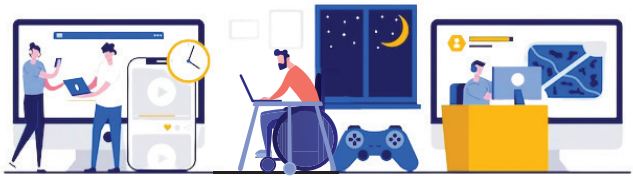
## Нейромережі

**Нейромережі** — це комп'ютерні системи, які моделюють роботу людського мозку, складаючись з великої кількості штучних нейронів, що співпрацюють для виконання завдань.

## Ігрові платформи

**Ігрові платформи** — це апаратне та програмне забезпечення, яке використовується для гри у відеоігри.

Іноді в ігри можна грати безпосередньо в мережі «Інтернет», не встановлюючи їх на свій комп'ютер чи пристрій.



Комп'ютер

Гральна консоль

Мобільний телефон або планшет

**Комп'ютер** — це пристрій, на якому можна грати в ігри, використовуючи клавіатуру та мишу або інші пристрої введення.

**Гральна консоль** — це спеціальний пристрій, створений для гри. Зазвичай він під'єднується до телевізора та має власний контролер для управління грою.

**Мобільний телефон** або **планшет** — це пристрої, які можна використовувати для гри в ігри, використовуючи сенсорний екран або додаткові контролери.

## Інформаційна безпека

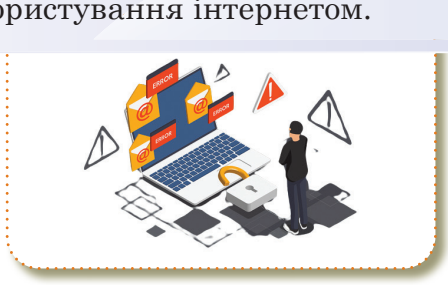
**Інформаційна безпека** — стан захищеності інформаційного середовища суспільства, який забезпечує конфіденційність, доступність і цілісність даних.

Проблема захисту даних від втрати, викрадення, спотворення або пошкодження потребує посиленої уваги, оскільки зростає роль **інформаційно-комунікаційних технологій** у сучасному суспільстві.

**Розкриття конфіденційної інформації у мережі** — повідомлення повного власного імені чи імен членів родини, адреси проживання і навчального закладу, номерів телефонів та іншої приватної інформації в мережі «Інтернет».

## Загрози безпеці в мережі «Інтернет»

**Загрози безпеці в мережі «Інтернет»** — це потенційні ризики та небезпеки, які можуть виникнути під час користування інтернетом.


**Віруси та інші шкідливі програми**

Віруси, черви, троянські коні та інші види шкідливого програмного забезпечення можуть інфікувати Ваш комп'ютер або пристрій. Вони можуть пошкодити Вашу операційну систему, вкрасти особисті дані або перешкоджати нормальній роботі комп'ютера.

**Доступ до облікових записів**

Зловмисники можуть намагатися взяти під контроль Ваші облікові записи, такі як пошта, соціальні мережі чи онлайн-банкінг, і використовувати їх для шкідливої діяльності.

**Крадіжка ідентичності**

Зловмисники можуть намагатися вкрасти вашу ідентичність, використовуючи Ваші особисті дані для злочинних цілей.

**Фішинг**

Атаки фішингу — це спроби обману користувачів, зазвичай через електронну пошту або вебсайти, з метою витягнути особисті або фінансові дані, такі як паролі чи номери кредитних карток.

**Спам**

Спам — це небажані повідомлення, які надсилаються користувачам без їхньої згоди. Спам може містити небезпечні посилання або вкладення.

## Практична робота №8

### Частина 1

**1.** Зробіть порівняльну характеристику месенджерів за зразком.

**2.** Надішліть доступ Вашій вчительці/Вашому вчителеві для перегляду.

| № | Питання | Viber | Telegram | WhatsApp | Signal | Skype |
|---|---|---|---|---|---|---|
| 1 | Яка особливість месенджера? | | | | | |
| 2 | Чи можна додавати в текстові повідомлення емодзі, смайли? | | | | | |
| 3 | Чи можна відправити фото чи відео в месенджері? | | | | | |
| 4 | Можливість створювати та керувати груповими чатами, проводити групові відео- та аудіоконференції. | | | | | |
| 5 | Інтерфейс та зручність використання | | | | | |
| 6 | Мультимедійні можливості | | | | | |

### Частина 2

Колективна робота (3-4 учні в групі)

**1.** Увійдіть у свій обліковий запис **Google**.

**2.** У своєму **Google диску** створіть **Google** презентацію (керівник/керівниця групи).

**3.** Надайте доступ учасникам/учасницям групи та Вашій вчительці/Вашому вчителеві.

**4.** Тема презентації «Соціальні мережі та їх особливості».
Теми:
- Pinterest
- Instagram

- LinkedIn
- Facebook

Вимоги до Google презентації:

- Титульний слайд (назва презентації, інформація про авторів).
- 1 слайд — Основна інформація про соціальні мережі.
- 2 слайд — Соціальна мережа та її особливості.
- 3 слайд — З якого віку можна створити акаунт.
- 4 слайд — Який контент можна поширювати.
- 5 слайд — Чим корисна соціальна мережа.
- 6 слайд — Історія створення.
- 7 слайд — Можливості спілкування.
- 8 слайд — Висновок.

### Частина 3. Робота з нейромережами

**Завдання.** Відкрити три нейромережі та порівняти результати пошуку.

| № | Запитання | ChatGPT | Bard | perplexity.ai |
|---|---|---|---|---|
| 1 | Дати визначення, що таке інформатика. | | | |
| 2 | Що вивчає предмет інформатика? | | | |
| 3 | Дати визначення терміна «інформація». | | | |
| 4 | Яка класифікація видів інформації? | | | |

## Домашнє завдання

Створити презентацію на тему «Історія виникнення соціальної мережі». Надати доступ для перегляду Вашій вчительці/Вашому вчителю.

# ВІЗУАЛЬНИЙ КОНТЕНТ

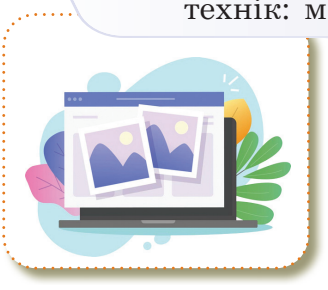
**Тема 9** **Пошук та генерування зображень в інтернеті. Кодування графічних даних. Типи файлів з графічними даними. Програмні засоби створення візуального контенту. Генерування зображень в мережі. Етичність і відповідальність при використанні генерованого контенту**

Ключові поняття

**Графічне зображення**
**Пошук зображень**
**Google об'єктив**
**Генерування зображень**
**Кодування графічних даних**

## Графічне зображення

**Графічне зображення** — це візуальне представлення об'єкта, створене за допомогою графічних елементів, таких як кольори, форми та текстура, за допомогою різних технік: малювання, фотографія, комп'ютерна графіка, векторна графіка тощо.



Це може бути будь-яке зображення чи малюнок, що Ви бачите на екрані комп'ютера, аркуші паперу, на екрані смартфона чи будь-якому іншому носії. Графічні зображення можуть бути створені або зафіксовані з використанням різних технік, таких як малювання, фотографія, комп'ютерна графіка, векторна графіка тощо.

Графічні зображення використовуються у різних галузях, таких як графічний дизайн, веброзробка, реклама, друковані матеріали, ігри та інші сфери, де важлива візуальна представленість.

### Растрові зображення

- Представлені масивом пікселів.
- Зображення може втрачати якість при збільшенні.
- Формати: JPEG, PNG, GIF.

### Векторні зображення

- Представлені математичними об'єктами (шляхами, лініями, кривими).
- Зберігають чітку якість навіть при збільшенні.
- Формати: SVG, EPS, AI.

### Текстові графічні зображення

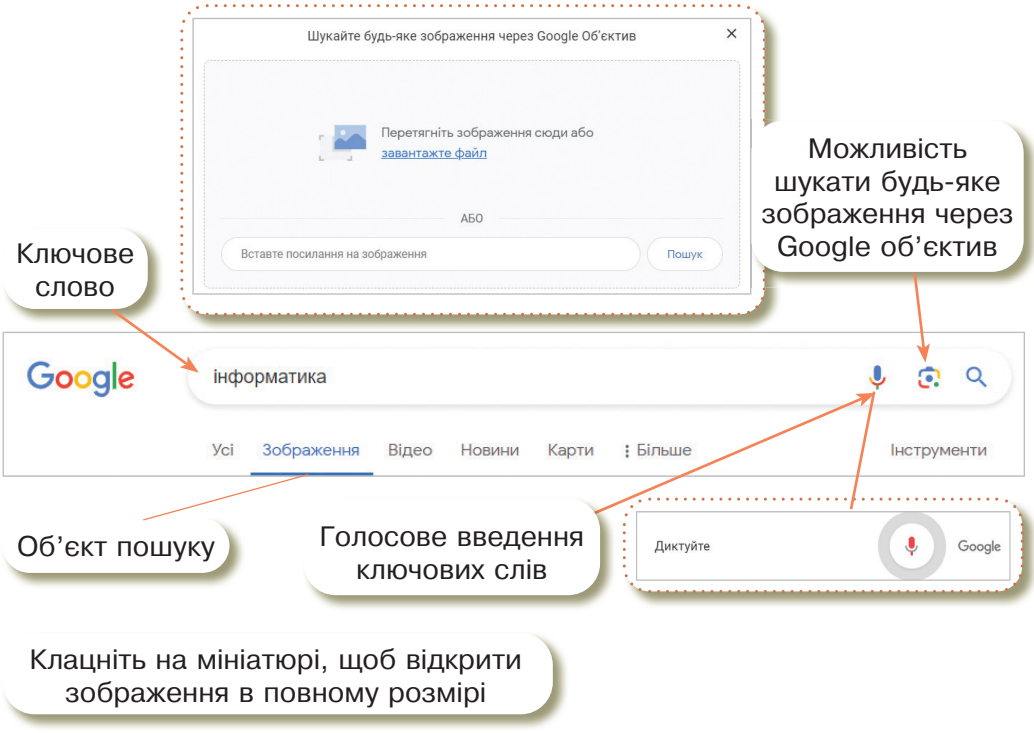
- Створені з використанням текстових символів та шрифтів.
- Зазвичай використовуються для створення артграфіки або логотипів.

### 3D-зображення

- Використовуються для представлення тривимірних об'єктів та сцен.
- Використовуються в графічному дизайні та іграх.

## Пошук зображень

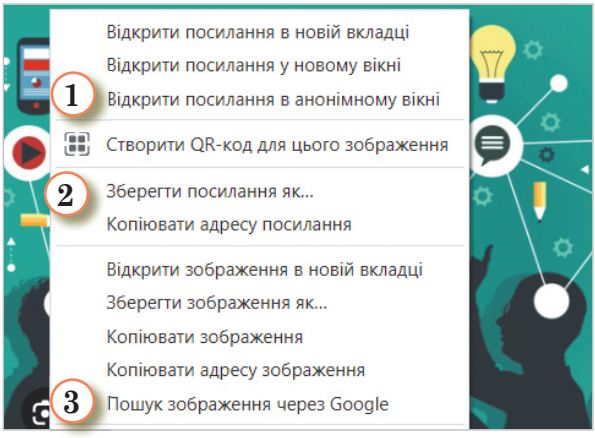
### Використання пошукової системи Google



## Google об’єктив

**Google об’єктив** (англ. *Google lens*) — технологія розпізнавання зображень, розроблена **Google**, покликана збирати відповідну інформацію, що стосується об’єктів, які вона ідентифікує за допомогою візуального аналізу на основі нейронної мережі.


Ви можете навести камеру телефону на текст, а потім виділити цей текст у Google Lens і скопіювати його для використання на телефоні.




1. Дуже швидко створити QR-код з доступом до цього зображення.
2. Копіювати зображення в буфер обміну.
3. Об’єкт пошуку — зображення.
4. Можна знайти джерела зображення.
5. Можна копіювати та перекладати текст за допомогою **Google Lens**.

Онлайн-платформи або ресурси, які надають користувачам доступ до різноманітних графічних ресурсів, таких як фотографії, вектори, ілюстрації, відео та інші візуальні елементи, можуть пропонувати як безкоштовний, так і платний контент, а також різні типи ліцензій для використання зображень та іншого матеріалу.

**Unsplash** є платформою для обміну високоякісними фотографіями, які доступні для використання безкоштовно. Фотографії на Unsplash мають високий стандарт та часто використовуються у вебдизайні, маркетингу та інших проєктах. Користувачі можуть завантажувати свої фотографії та використовувати зображення з сайту без обмежень.

**Pixabay** є іншою платформою для безкоштовних високоякісних зображень та відеороликів. Як і Unsplash, Pixabay також має ліцензію від Creative Commons Zero, що дозволяє використовувати зображення без обмежень. Крім фотографій, на Pixabay можна знайти ілюстрації, вектори, аудіофайли та відео. Сайт має широкий спектр категорій та зручний пошук, що полегшує знаходження потрібного контенту.

**Adobe Stock** — це платформа для придбання високоякісних фотографій, ілюстрацій, векторів та відео. Вона інтегрована з продуктами Adobe, такими як Photoshop та Illustrator, що спрощує редагування та використання контенту безпосередньо в програмах Adobe. Adobe Stock пропонує як безкоштовні, так і платні зображення високої якості. Вартість контенту може варіюватися залежно від типу ліцензії та використання.

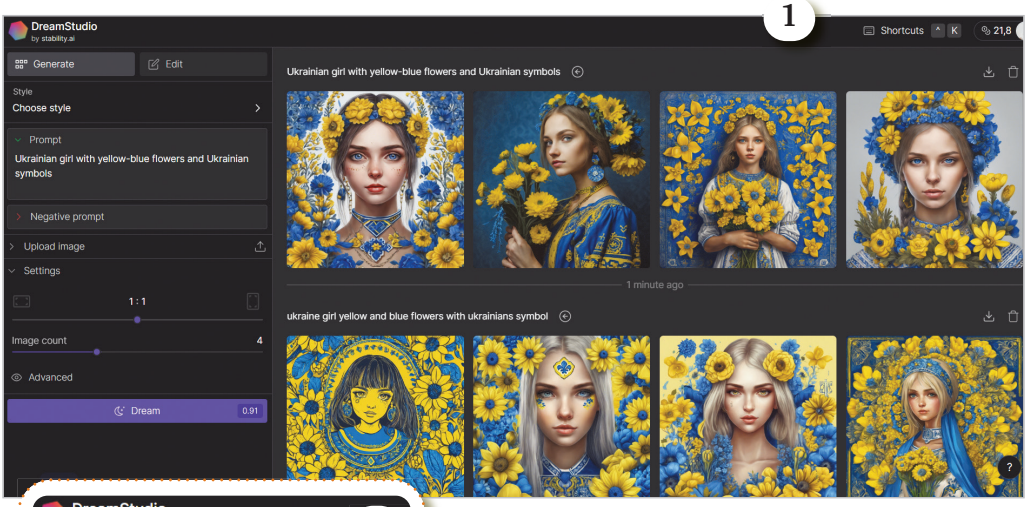
## Генерування зображень

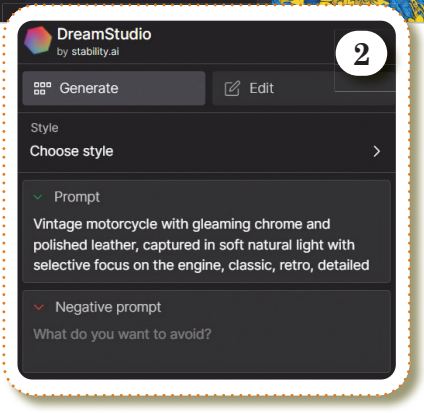
**Генерування зображень** — це створення нових зображень за допомогою комп'ютерних програм або штучного інтелекту.

Промт Ukrainian girl with yellow-blue flowers and ukrainian symbols

## Dreamstudio.Ai

**Dream studio** — це вебсайт для створення зображень за допомогою stable diffusion, моделі для перетворення тексту в зображення.



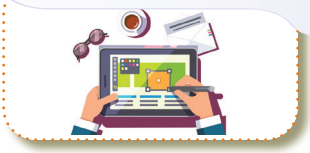





1. **Prompt** — текстовий опис зображення, яке Ви хочете згенерувати.
2. **Negative prompt** — опис того, чого потрібно уникати при генерації зображення.
3. **Styles** — вибір стилю для згенерованого зображення, наприклад, аніме, цифрове мистецтво, фотографія та інші.

## Кодування графічних даних

**Кодування графічних даних** — процес перетворення зображень у цифровий формат, який може бути записаний на комп'ютерний носій.



- Растрове кодування
- Векторне кодування

Растрове кодування використовує набір пікселів (крапок) для представлення зображення. Кожен піксель має свої координати (x, y) та колір. Розмір зображення визначається кількістю пікселів по горизонталі та вертикалі. Що більше пікселів, то вища роздільна здатність зображення, а отже, й краща якість.

Векторне кодування використовує геометричні об'єкти для представлення зображення. Об'єкти можуть бути лініями, кривими, фігурами та іншими елементами. Кожний об'єкт має свої координати, розміри, колір та інші властивості.

### Типи файлів з графічними даними

Є різні способи зберігання даних зображення на комп'ютері. Кожен тип файлу має свої особливості та призначення.

**Jpeg (joint photographic experts group)**

Формат, оптимізований для стиснення фотографій та зображень з реальними сценами. Зазвичай використовується для фотографій в інтернеті, де прийнятна втрата якості для зменшення розміру файлів.

**Png (portable network graphics)**

Формат з підтримкою прозорості та високої якості стиснення без втрат. Часто використовується для зображень з прозорістю та для графічних робіт.

**Gif (graphics interchange format)**

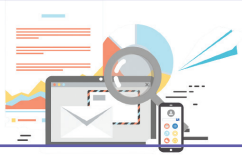
Формат, який підтримує анімацію та обмежений палітрою кольорів. Використовується для невеликих анімованих зображень та простих анімацій.

**Svg (scalable vector graphics)**

Векторний формат, який представляє графіку як математичні об'єкти. Використовується для векторної графіки та зображень у вебдизайні.

**Pdf (portable document format)**

Хоча не є лише графічним форматом, PDF може містити графічні дані та використовуватися для зберігання електронних документів з графікою.



**Raw (camera raw image)**

Формат, який зберігає непереробленні дані з камери, необроблені алгоритмами компресії чи корекції кольорів. Зазвичай використовується фотографами для зберігання максимальної кількості інформації з фотоапарата.

## Canva

**Canva** — вебплатформа для графічного дизайну, яка дозволяє користувачам створювати різноманітні графічні матеріали, такі як постери, презентації, логотипи, соціальні зображення, візитки та багато іншого.


*Простота у використанні.* Canva має простий у використанні інтерфейс, який дозволяє користувачам створювати професійні дизайни без необхідності мати навички дизайну.

*Широкий вибір шаблонів.* Canva пропонує широкий вибір шаблонів для різних типів дизайнів, що робить його ідеальним вибором для користувачів, які не хочуть починати з нуля.

*Доступність.* Canva має доступний онлайн, що означає, що Ви можете використовувати його з будь-якого пристрою з доступом до інтернету.

## Практична робота №9

### Частина 1. Пошук зображень

1. Знайдіть та завантажте по 5 зображень з кожного сервісу на Ваш вибір.
2. Створіть папку на своєму **Google диску** «Ваше прізвище та ім'я».
3. Завантажте Ваші зображення у свою папку.
4. Надайте доступ Вашій вчительці/Вашому вчителеві.
   Сервіси:



Потрібно авторизуватися за допомогою свого облікового запису **Google**.

### Частина 2. Генеруємо зображення за допомогою нейромережі dreamstudio.ai

1. Авторизуйтеся за допомогою облікового запису **Google**.
2. Сформулюйте промт (якщо є потреба — використайте **Google перекладач**).
3. Оберіть різні стилі згенерованих зображень.
4. Завантажте 10 згенерованих зображень за допомогою цієї нейромережі.
5. Завантажте в свою папку на **Google диску** (в окрему папку з допомогою нейромережі).

### Частина 3. Графічний редактор Canva

1. Перейдіть за покликанням **Canva**.
2. Авторизуйтеся за допомогою облікового запису **Google**.
3. Оберіть шаблон **Інфографіка**.
   Оберіть шаблон, який Вам подобається!

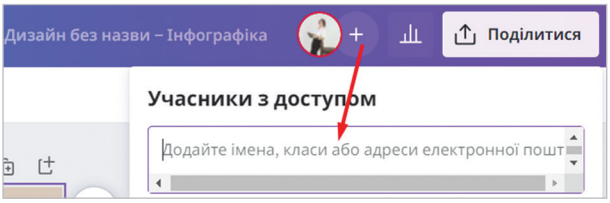
Тема інфографіки «Як правильно написати промт для чату GPT».

При пошуковій роботі пам'ятай про правила безпеки в мережі «Інтернет» та про критичне мислення під час пошукової діяльності.


Вимоги:

- Мінімум 5 пунктів.
- Додавати скріншоти.
- Додавати об'єкти з колекції **Canva**.

Надайте доступ до своєї інфографіки Вашій вчительці/Вашому вчителеві за допомогою електронної пошти.




## Домашнє завдання

Створити дизайн за допомогою сервісу **Canva**, шаблон **Інфографіка**, тема «ТОП 5 нейромереж для навчання». Надати доступ Вашій вчительці/Вашому вчителеві за допомогою електронної пошти.

Тема 10

# Види інформаційних продуктів. Ліцензії на використання інформаційних продуктів. Поняття про статичну і динамічну графіку. Поняття доповненої реальності, інструменти її створення та використання

**Ключові поняття**

**Інформаційні продукти**
**Статична графіка**
**Динамічна графіка**
**Доповнена реальність**
**Вибір ліцензії**

## Інформаційні продукти

Різноманітні форми інформації можуть бути створені та оброблені за допомогою комп'ютерної технології, а саме: текст, зображення, відео, аудіо, програмний код, таблиці даних та інші форми інформації.

**Зображення.** Якщо Ви створюєте малюнок у графічному редакторі — це буде інформаційний продукт у формі зображення. Ви можете зберегти його, роздрукувати чи використовувати для створення презентацій.

**Текст.** Набираючи звіт про свою улюблену книгу або есе на певну тему в текстовому редакторі, Ви створюєте інформаційний продукт — текстовий документ. Цей продукт можна зберігати, редагувати та ділитися з іншими.

**Відео.** Якщо Ви записуєте коротке відео, розповідаючи про свій проєкт, це теж інформаційний продукт. Ви можете використовувати камеру або смартфон, зберігати відео та обмінюватися ним з іншими.

**Аудіо.** Якщо записуєте своє озвучення для презентації — це інформаційний продукт у формі аудіо. Такі файли можна слухати на комп'ютері, телефоні чи іншому пристрої.

## Статична і динамічна графіка

**Статична графіка** — це зображення, яке залишається незмінним без будь-якої рухомої або анімаційної дії. Іншими словами, це статичне, непорушне зображення. Наприклад, фотографії, малюнки, картини — це статична графіка. Коли Ви переглядаєте знімок, то бачите лише фіксований момент часу без будь-якого руху або змін.

**Динамічна графіка**, навпаки, містить елементи руху чи анімації. Це означає, що об'єкти або зображення змінюють свій стан чи положення з часом. Наприклад, відеоролики, анімовані мультфільми та ігри — це форми динамічної графіки. У динамічній графіці об'єкти можуть рухатися, змінювати свою форму або реагувати на введення користувача.

## Доповнена реальність

**Доповнена реальність** — технологія, яка дозволяє додавати в існуючий реальний світ додаткові візуальні, звукові або інші сенсорні враження за допомогою комп'ютерних сил, часто через мобільні пристрої, смартокуляри чи інші пристрої.

Основна ідея полягає в тому, щоб об'єднати в реальному часі віртуальні об'єкти або інформацію з реальним навколишнім світом, створюючи таким чином враження розширення чи «доповнення» реальності.

### Arloopa

Додаток доповненої реальності (augmented reality), який, здається, розмиває межі між реальним і віртуальним світом. Містить кейси з готовими AR-об'єктами, що об'єднані за різними категоріями: освіта, тварини, мистецтво, наука і технології, історія і культура, архітектура тощо.

### Assemblr

Додаток, що дозволяє переглядати готові AR-об'єкти та створювати власні моделі для доповненої реальності. Для зручності всі елементи згруповані за категоріями: тварини, архітектура, мистецтво, мультиплікаційна анімація, герої, культура, освіта, природа, наука, технології та багато інших.

## Вибір ліцензії

Ліцензії для використання інфопродуктів визначають правила та обмеження, які стосуються того, як можна використовувати та поширювати зображення.

Різні типи ліцензій передбачають різні рівні прав та зобов'язань для користувачів.

**Ліцензії для використання зображень** — це правила, які визначають, як ми можемо використовувати фотографії, малюнки та інші графічні зображення. Коли ми кажемо «використання», маємо на увазі розміщення цих зображень в інтернеті, у проєктах чи в інших місцях.

- **Відкрита ліцензія (open source)**. Продукти з відкритою ліцензією зазвичай надають Вам більше можливостей. Вони можуть включати можливість перегляду, зміни та розповсюдження вихідного коду або вмісту.
- **Комерційна ліцензія (commercial license)**. Зазвичай використовується для продуктів, призначених для комерційного використання. Користувачі зобов'язані купити ліцензію для використання продукту.
- **Користувацька ліцензія (custom license)**. Може включати специфічні умови, які стосуються конкретного випадку використання. Можливо, це передбачає обмеження на використання у певних індустріях або для конкретних цілей.
- **Creative commons.** Найпопулярніша ліцензія, що охоплює будь-які види контенту. Різні версії ліцензій Creative Commons (наприклад, CC BY, CC BY-SA) мають різні обмеження. Деякі можуть дозволяти використання зображення з обов'язковим зазначенням автора, інші — з обов'язковим розповсюдженням похідних творів на тих самих умовах.
- **CC BY (Attribution)** — «із зазначенням авторства». Найбільш вільна ліцензія. Зображення з цією ліцензією можна поширювати, змінювати і використовувати навіть із комерційною метою за умови, що відсутні відомості про автора.
- **Cc BY-SA (Sharealike)** — «поширення на тих же умовах — копілефт», дозволяє поширювати, змінювати і використовувати зображення із комерційною метою, але за умови вказівки авторства і ліцензування всіх змінених / створених зображень на аналогічних умовах. Цю ліцензію часто порівнюють із безкоштовними ліцензіями програмного забезпечення з відкритим вихідним кодом.
- **CC BY-ND (Noderivs)** — «із зазначенням авторства — без похідних», дозволяє поширювати, змінювати і використовувати зображення із комерційною метою, але за умови вказівки авторства і ліцензування всіх змінених / створених зображень на аналогічних умовах. Цю ліцензію часто порівнюють із безкоштовними ліцензіями програмного забезпечення з відкритим вихідним кодом.

## Практична робота №10

### Частина 1

Колективна робота (3-4 учні/учениці в команді).

Теми:

- Сфери використання доповненої реальності.
- AR-інструменти для створення доповненої реальності.
- Історія та розвиток доповненої реальності.
- Доповнена реальність в освіті, для навчання.
- Етика використання доповненої реальності.

*Середовище виконання:* **Canva** — безкоштовний онлайн-інструмент графічного дизайну.

*Шаблон:* Презентація.

Вимоги до презентації:

- Мінімум 10 слайдів.
- На кожному слайді — текстові поля, зображення, графічні елементи, покликання.
- Додати до колективної роботи Вашу вчительку/Вашого вчителя.
- Єдиний стиль презентації.

Пам'ятаємо про безпеку в мережі «Інтернет» та критичне ставлення до інформації.

### Частина 2. Плакат на тему: «Етикет спілкування в інтернеті»

*Середовище виконання:* **Canva** — безкоштовний онлайн-інструмент графічного дизайну.

*Шаблон:* Плакат.

Надати доступ Вашій вчительці/Вашому вчителеві.

Пам'ятаємо про безпеку в мережі «Інтернет» та критичне ставлення до інформації.

## Домашнє завдання

Створити інфографіку в **Canva** на тему «ТОП 10 небезпек в інтернеті та як захистити себе».

Сервіс — **Canva**. Шаблон — **Інфографіка**.

Надати доступ Вашій вчительці/Вашому вчителю.

Пам'ятаємо про безпеку в мережі «Інтернет» та критичне ставлення до інформації.

## Тема 11 Сайт-портфоліо

**Ключові поняття**

**Сайт**
**URL (Uniform Resource Locator)**
**Контент**
**Хостинг**
**Ергономіка сайту**

Сайт

**Сайт** — сукупність вебсторінок та вмісту, доступних у мережі «Інтернет», які об'єднані за змістом та за навігацією під єдиним доменним ім'ям.


Сайти служать для представлення інформації, розважання, спілкування, комерційної діяльності та інших цілей, вони доступні для перегляду за допомогою веббраузерів.

**URL (Uniform Resource Locator)** — адреса в мережі «Інтернет», яка вказує на конкретний ресурс, такий як вебсторінка чи зображення.

**Контент** — весь матеріал на вебсайті, такий як текст, зображення, відео. Кожен розділ має власний унікальний контент.

**Хостинг** — це послуга, яка дозволяє розміщувати Ваш вебсайт на сервері, щоб він був доступний в інтернеті. Коли Ви купуєте хостинг, то отримуєте простір на сервері, де зберігаються файли Вашого вебсайту та звідки їх відправляють користувачам, які відвідують Ваш сайт.

**Розділ сайту** — це частина сайту, яка має свою тему чи завдання. Наприклад, розділ про книги може мати URL «www.example.com/books».

## Сайт-портфоліо

**Сайт-портфоліо** — вебсайт, який використовується для представлення особистої або професійної інформації в інтернеті.

Зазвичай ці сайти створюються з метою представлення особистості, просування особистого бренду або надання спрощеної контактної інформації.

| | |
|---|---|
| Головна сторінка | Тут Ви можете вітати відвідувачів та коротко розповісти про себе. Додати свою фотографію та основні інформаційні блоки. |
| Про мене | Розказати трошки про себе: свої інтереси, хобі, чому Ви створили цей сайт. Зробити це як історію про себе. |
| Мої проєкти | У цьому розділі можна показати свої творчі роботи або проєкти. Додати фотографії, описи та, можливо, як Ви працювали над ними. |
| Мої навички | Виділити свої основні навички. Це може бути все: від програмування та малювання до інших речей, які Ви вмієте робити. |
| Контакти | Вказати контакти для того, щоб люди могли зв'язатися з Вами. Додати свою електронну адресу чи посилання на соціальні мережі. |

## Вимоги до сайту-портфоліо

Дизайн та контент повинні відповідати темі портфоліо та відображати стиль особистості власника.

Зазвичай ці сайти створюються з метою представлення особистості, просування особистого бренду або надання спрощеної контактної інформації.

**Мінімалізм** — Сайт повинен бути зрозумілим та не перевантаженим інформацією.

**Навігація** — Меню повинно бути легким для використання та дозволяти швидко переміщатися між розділами.

**Фотографії та зображення** — Рекомендуємо, щоб робота чи проєкт мали фотографії або зображення, щоб люди могли бачити Вашу творчість.

**Стиль та особистість** — Ваш сайт повинен відображати Вашу індивідуальність. Вибирайте кольори та дизайн, які відповідають Вашому стилю.

Створення вебсайтів за допомогою конструкторів спрощує процес розробки, оскільки вони надають інтерфейс інтуїтивно зрозумілий для користувачів, які не мають глибоких знань в галузі веб-програмування або дизайну.

## 1 ЕТАП РЕЄСТРАЦІЯ ТА ВИБІР ШАБЛОНУ

*Реєстрація:*

- Перейдіть на сайт конструктора (canva).
- Створіть обліковий запис або увійдіть, якщо Ви вже маєте обліковий запис.

*Вибір шаблону:*

- Оберіть категорію (наприклад, «особистий сайт», «онлайн-магазин», «блог»).
- Оберіть шаблон, який Вам подобається.

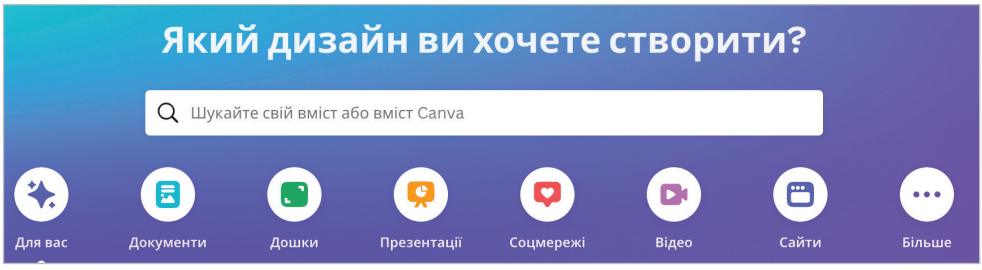

## 2 ЕТАП НАЛАШТУВАННЯ СТРУКТУРИ ТА ВИГЛЯДУ

*Додавання та редагування сторінок:*

- Додайте нові сторінки (наприклад, **Головна**, **Про нас**, **Контакти**).
- Редагуйте тексти та зображення на кожній сторінці.

*Вибір кольорової палітри та шрифтів:*

Змінюйте кольори та шрифти за допомогою панелі налаштувань.



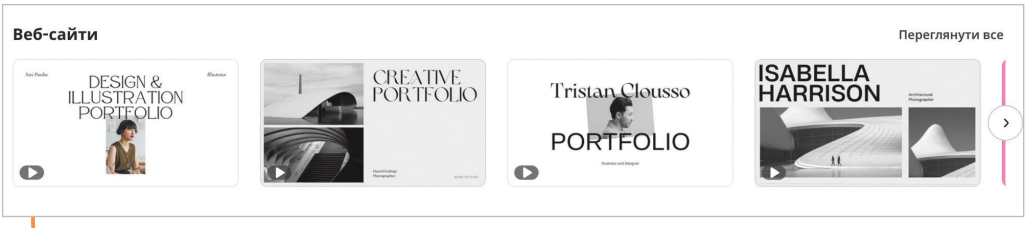

## 3 ЕТАП ДОДАВАННЯ ВМІСТУ

*Текст та зображення:*

- Додайте блоки тексту для описів та інструкцій.
- Вставляйте зображення, наприклад, фотографії Вашого класу чи проєктів.

*Відео та аудіо:*

- Додайте відео, якщо у Вас є відеоматеріали (наприклад, відеозвіт із заходу).
- Уставте аудіо, якщо це відповідає тематиці Вашого сайту.

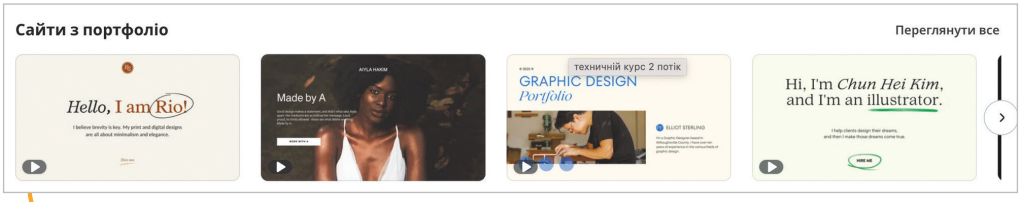


## 4 ЕТАП ПУБЛІКАЦІЯ ТА ДОМЕН

*Вибір домену:*

- Придбайте власний домен.
- Якщо є можливість, прикріпіть власний домен.

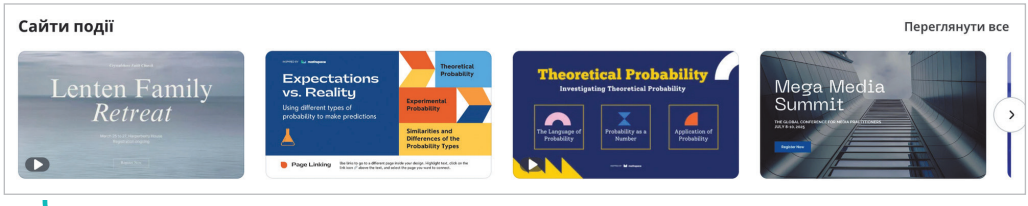
*Публікація:* Натисніть **Опублікувати**, щоб Ваш сайт став доступним в інтернеті.




## 5 ЕТАП ПІДТРИМКА ТА ОНОВЛЕННЯ

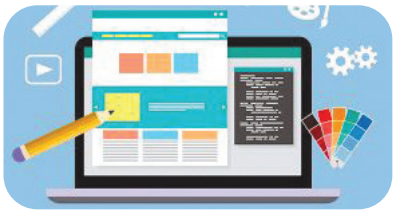
*Оновлення вмісту:*

- Регулярно додавайте нові інформації, фотографії чи оголошення.
- Зберігайте вміст свіжим та актуальним.



## Ергономіка сайту

**Ергономіка сайту** — це наука, що досліджує, як створити вебсайт зручним для користувачів, легким у застосуванні.



### Основні аспекти ергономіки сайту

1. **Зручність використання**
Вебсайт повинен бути простим та зрозумілим для користувачів. Навігація має бути легкою, зрозумілою та послідовною.

2. **Читабельність та вигляд**
Текст на сайті повинен бути читабельним та зрозумілим, шрифт — зручним, обсяг — оптимальним.

3. **Зображення та мультимедіа**
Використання якісних зображень та мультимедійних елементів. Забезпечення швидкості завантаження сторінок, щоб користувачі не чекали надто довго.

4. **Адаптивний дизайн**
Забезпечення зручності перегляду вебсайту на різних пристроях (комп'ютерах, планшетах, смартфонах). Адаптування інтерфейсу під різні розміри екрана.

## Практична робота №11

**Завдання.** Створення сайту-портфоліо.

**1.** Авторизуватися на сайті **Canva**.

**2.** Обрати шаблон — **Сайт**.

**3.** Обрати шаблон, який Вам сподобається.

У презентації запропоновано приблизні розділи сайту-портфоліо, але Ви можете створити свої розділи.

Наповнюємо контентом про себе та свою творчість.

Надаємо доступ для перегляду Вашій вчительці/Вашому вчителю.

## Домашнє завдання

Створити сайт-портфоліо про своє хобі.

Сервіс — **Canva**.

Шаблон — **Сайт**.

Сайт-портфоліо про вид хобі.

Розкажіть детально про своє захоплення, людей, які Вас надихають, покажіть власні роботи.

Створіть свої розділи.

Наповніть контентом про свою творчість.

Надайте доступ для перегляду Вашій вчительці/Вашому вчителю.

Тема 12

# Текстові документи. Різновиди публікацій. Друковані та електронні публікації. Лінійний та нелінійний текст. Формування, форматування, збереження текстових документів та публікацій у різних форматах

Ключові поняття

**Текстовий документ**
**Форматування текстового фрагмента**
**Недруковані знаки**
**Публікації**
**Лінійний та нелінійний текст**

## Текстовий документ

**Текстовий документ** — документ, який складається з текстових, графічних, мультимедійних та інших об'єктів.

Текстові документи дуже розповсюджені та використовуються з різною метою, наприклад, для написання листів, складання звітів, створення шкільних робіт тощо.

Текстовий документ — це інструмент для створення і зберігання текстової інформації, який допомагає оформити Ваші ідеї, думки та документи.

Текстові документи можна зберігати на комп'ютері або навіть у хмарних сервісах для зручного доступу до них з будь-якого пристрою з інтернетом.

### Microsoft Office

**Microsoft Office** — офісний пакет, створений корпорацією **Microsoft** для операційних систем **Windows**, **macOS**, **iOS** та **Android**.

Можливість спільної роботи над документами через хмарний сервіс **Microsoft 365**.

Текстовий редактор (**Microsoft Word**):

- Створення та редагування текстових документів.
- Форматування тексту, додавання зображень, таблиць та інших об'єктів.
- Перевірка правопису та граматики.

Пакет інтегровано працює з хмарними службами, такими як **Microsoft OneDrive**, що дозволяє зберігати та обмінюватися документами в інтернеті.

Можливість використання **Microsoft Office** на різних платформах, включаючи **Windows**, **macOS**, **iOS** та **Android**, що робить його доступним для користувачів на різних пристроях.

## Google docs

**Google Docs** — безкоштовний хмарний офісний пакет, що містить текстовий редактор, табличний редактор і службу для створення презентацій, розроблений **Google**.


Користувачі можуть одночасно працювати над документом, редагуючи його та коментуючи зміни в режимі реального часу. Це робить спільну роботу ефективною та зручною.

Він є частиною офісного пакета **Google Workspace**, який також має й інші інструменти, такі як **Google Sheets** (таблиці) та **Google Slides** (презентації).

**Google Docs** дозволяє створювати та редагувати текстові документи просто у веббраузері, без необхідності встановлення спеціального програмного забезпечення.

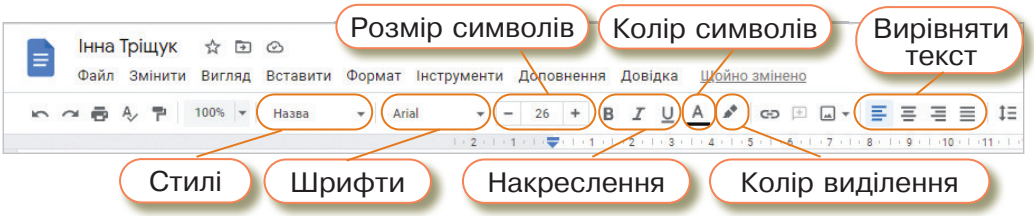
Документи автоматично зберігаються в хмарному сховищі **Google Drive**, що дозволяє легко отримувати доступ до них з будь-якого пристрою з підключенням до інтернету.

## Дії над об'єктами текстового документа

Уведення | Редагування | Форматування | Вставлення об'єктів | Макетування | Друкування | Робота з файлами

**Уведення** — введення з клавіатури символів у документ.
**Редагування** — виправлення помилок, видалення, переміщення, копіювання, вставляння символів, слів, абзаців, рядків тощо.
**Форматування** — зміна вигляду документа: встановлення кольорів тексту, вирівнювання абзаців, зміна орієнтації сторінки тощо.
**Вставлення об'єктів** — додавання до документа рисунків, таблиць, формул, діаграм, схем тощо.
**Макетування** — підготовка документа до друку: оформлення заголовків, розбивка на сторінки, нумерація їх, опрацювання рисунків, створення змісту тощо.
**Друкування** — виведення на папір усіх або вибраних сторінок створеного документа.
**Робота з файлами** — збереження текстового документа у файлі на диску, відкриття текстового файлу в текстовому процесорі.

## Форматування текстового фрагмента




### Властивості символів та їх форматування

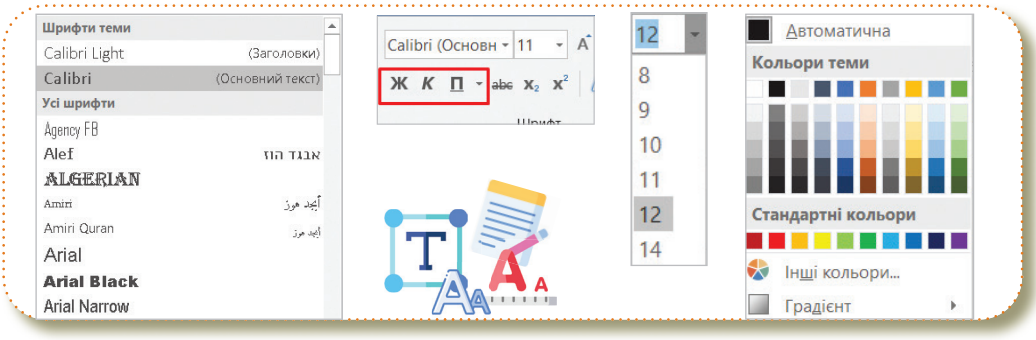
Символ як об'єкт текстового документа має такі властивості: шрифт, розмір, колір, накреслення та інші.

**Шрифт** (нім. *Schrift* — письмо) визначає графічну форму символів, як почерк у людей. Наприклад, Algerian, SimSun, Impact, Times New Roman, Monotype Corsiva.

**Накреслення** визначає особливості зовнішнього вигляду символів. Може набувати значень: звичайний, жирний, курсив, жирний курсив, підкреслений, закреслений.

**Розмір** символів вказується в спеціальних одиницях — пунктах. Наприклад, 8,5 пт, 14 пт, 28,5 пт, 72 пт.

**Колір** символів може набувати різних значень: жовтий, синій, зелений тощо.




## Властивості абзаців та їх форматування

Абзац як об'єкт текстового документа має такі властивості: вирівнювання, відступи, міжрядковий інтервал та інші.

**Вирівнювання абзацу** — спосіб розташування рядків абзацу відносно його меж. Значення: зліва, справа, по центру, за шириною.

**Відступи** — відстань усіх рядків абзацу від межі лівого та правого поля сторінки, відступ першого рядка абзацу відносно його лівої межі. Наприклад, 0 см, 1 см, 2,25 см.

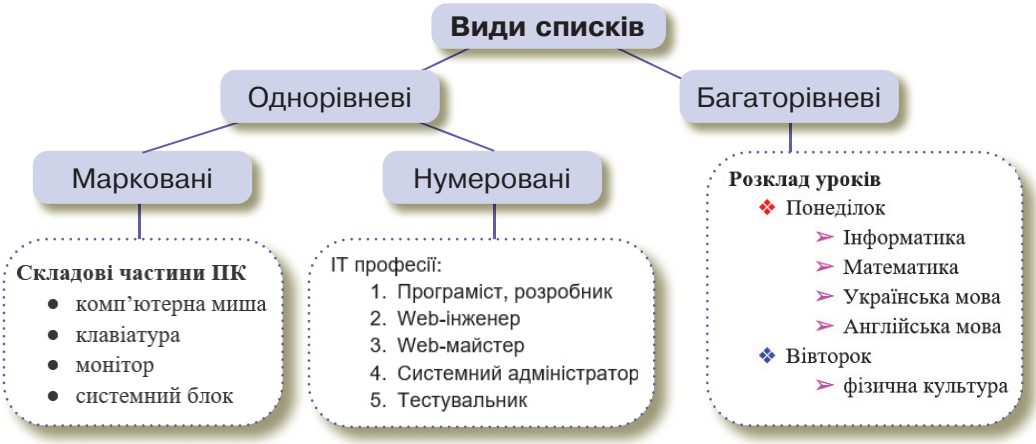
**Міжрядковий інтервал** — відстань між рядками тексту в абзаці, вимірюється в пунктах. Можна встановити такі значення міжрядкового інтервалу: одинарний, подвійний, множник.

Також для абзацу можна встановити **межі** та **колір заливки**.

1. Кнопки для встановлення значення вирівнювання абзацу.
2. Кнопки для збільшення та зменшення відступу абзаців зліва.
3. Кнопка зі списком для встановлення значення міжрядкового інтервалу.
4. Кнопки для встановлення кольору заливки абзацу.
5. Кнопка для встановлення межі абзацу.
6. Кнопка для відкриття діалогового вікна **Абзац**.

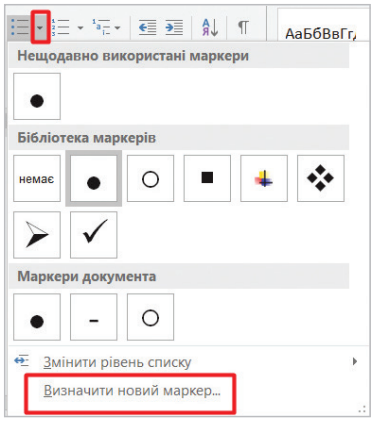
Установити значення властивостей абзаців можна елементами керування групи **Абзац** вкладки **Основне** або **мініпанелі форматування**.

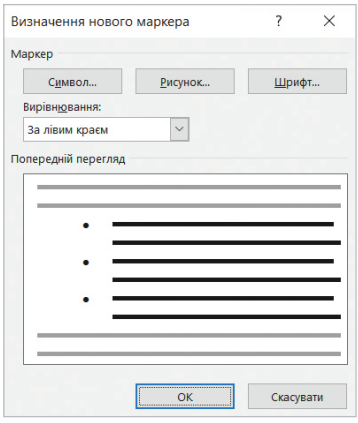
## Створення списків у текстовому документі

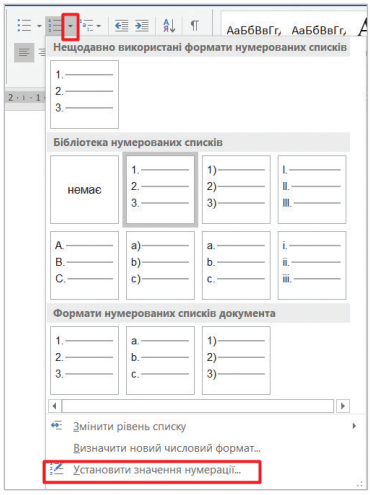

### Змінити стиль маркерів або нумерації

1. Натиснути на стрілку.
2. Обрати команду **Визначити новий маркер**.
3. Обрати інший маркер або вставити спеціальний символ.

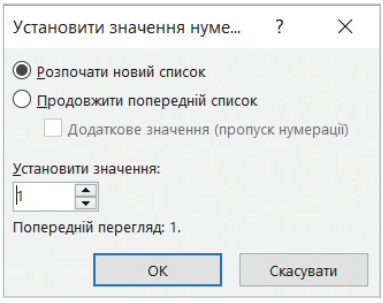


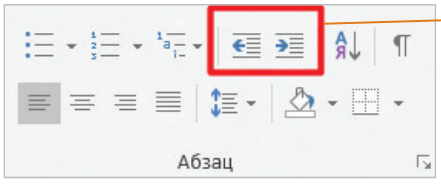
## Параметри списку




У параметрах списку можна почати нумерацію заново або продовжити попередню нумерацію.



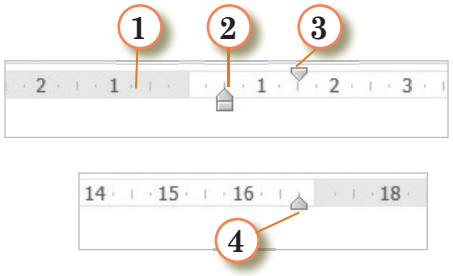

## Встановлення відступів елементів списку




Збільшити/зменшити відступ

Також можна користуватися клавішами **Tab** та **Enter**.

Для встановлення відступів абзацу зручно користуватися маркерами горизонтальної лінійки, перетягуючи їх вздовж лінійки.



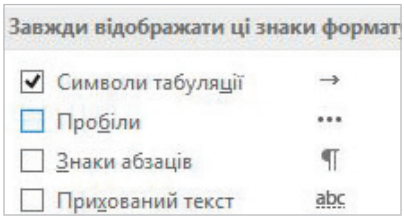

1. Маркер нависаючого відступу
2. Маркер відступу зліва
3. Маркер відступу першого рядка
4. Маркер відступу справа

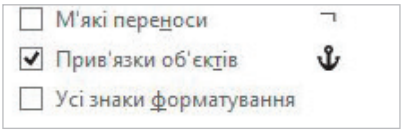
### Недруковані знаки

**Недруковані знаки** (іноді їх називають невидимими символами) у **Microsoft Word** — це спеціальні символи, які показують різні форматувальні елементи та пробіли в текстовому документі.

Вони допомагають Вам контролювати форматування тексту і вирівнювання.

- **Пробіл (Spacebar).** Простий пробіл між словами або символами. Він позначається як просто порожня позиція між словами.
- **Знак абзацу (¶).** Показує розрив абзацу, коли Ви натискаєте клавішу **Enter** на клавіатурі для переходу на новий рядок або абзац.
- **Знак розриву сторінки (Page Break).** Показує, де розривається сторінка, коли Ви додаєте перехід до нової сторінки.
- **Знаки табуляції (Tab Characters).** Вказують на вставку табуляції або відступу в тексті.
- **Знак розділу (Section Break).** Вказує, де починається новий розділ документа, який може мати власні параметри форматування.
- **Знаки переносу слова (Hyphenation).** Показують, як слово було розділене на дві частини для перенесення на новий рядок.




## Публікації

**Публікації** — текстові матеріали або інші роботи, які призначені для публічної інформації та розповсюдження серед читачів чи аудиторії.

**Друковані публікації**

- Книги
- Журнали
- Каталоги
- Газети

**Електронні публікації**

- Електронні книги (е-книги)
- Вебсайти
- Онлайн-журнали та вебжурнали
- Блоги та відеоблоги

**Друковані публікації** — видання, які виробляються у паперовій формі та призначені для розповсюдження серед читачів.

**Книга** є особливим видом паперової публікації, яка представляє собою довгий та розгорнутий текст, зазвичай зібраний в книжковій формі.

**Журнал** — це видання, яке виходить періодично і має статті на різні теми, такі як мода, наука, спорт чи мистецтво.

**Каталог** — це книгоподібний документ, який містить перелік товарів чи послуг, іноді з фотографіями та цінами.

**Газети** — регулярні видання, що містять актуальні новини, коментарі, статті та інші матеріали. Зазвичай призначені для широкого кола читачів, які цікавляться подіями та інформацією загального характеру.

**Електронні публікації** — публікації, які можна читати на електронних пристроях, таких як електронні книги, планшети чи комп'ютери.

**Електронні книги (е-книги)** — книги, які можна читати на електронних пристроях, таких як електронні книги, планшети чи комп'ютери.

**Вебсайти** — різноманітні ресурси в інтернеті, які містять інформацію на різні теми.

**Онлайн-журнали та вебжурнали** — електронні версії традиційних журналів чи спеціалізовані вебжурнали, які публікуються тільки в мережі.

**Блоги та відеоблоги** — онлайн-платформи, де автор (блогер) регулярно публікує текстові записи, які можуть містити різні теми, думки, інформацію чи особисті враження. Описи до відеороликів на **YouTube**.

## Лінійний та нелінійний текст

Лінійний та нелінійний текст — це два різні способи організації і подання інформації.

**Лінійний текст** має чіткий лінійний порядок, який йде від початку до кінця. Це означає, що читач повинен прочитати текст послідовно, дотримуючись зазначеного порядку речень.

**Нелінійний текст** означає, що інформація не подається в порядку, що йде одна за одною. Читач може обирати шлях читання, переходити від одного елемента до іншого за своїм вибором.

Цей тип тексту може містити гіпертекстові документи, де слова або фрази є посиланнями, які спрямовують до інших частин тексту.

## Збереження текстових документів

Збереження текстових документів передбачає вибір формату файлу та місця для збереження.

Вибір формату залежить від того, як Ви плануєте використовувати документ. Наприклад, для редагування в **Microsoft Word** оберіть **docx**, а для обміну документами — **pdf**.


Після вибору формату надайте документу зрозумілу та логічну назву, що відображає його вміст. Виберіть папку для збереження в локальному сховищі. Це допоможе вам швидко знаходити документ.

Після цього виберіть папку на хмарному сховищі, де Ви хочете зберегти документ. Завантажте документ в хмарне сховище.

За потреби Ви можете поділитися документом з друзями.

## Формати текстових документів

| | |
|---|---|
| docx | текст з форматуванням і вставленими об'єктами, основний формат **Microsoft Word** |
| pdf | формат для перегляду та друкування з незмінним вихідним форматуванням для різних платформ |
| txt | текст з розбиттям на абзаци без форматування |
| rtf | текст з форматуванням і вставленими об'єктами |
| html | стандартна мова розмітки вебсторінок в інтернеті і відповідне розширення файлу |

Microsoft Word (.docx)
Формат OpenDocument (.odt)
Формат RTF (.rtf)
Документ PDF (.pdf)
Простий текст (.txt)
Вебсторінка (.html, у стиснутому вигляді)
Публікація EPUB (.epub)

## Практична робота №12

**Завдання.** Створити текстовий документ на тему «Порівняльна характеристика **Microsoft Word** та **Google документ**».

Середовище виконання: **Microsoft Word** (можна **Google документи**).

Вимоги до текстового документа:

- Наявність кількох списків.
- Встановлення правильних відступів.
- На кожній сторінці додати кілька ілюстрацій, зображень, скріншотів.
- Додати вступ та висновок.
- Додати номер сторінки.
- Єдиний стиль форматування тексту.

Зберегти цей документ у свою папку на **Робочому столі**, назвавши своїм прізвищем та ім'ям.

Зберегти у форматах:

- .doc
- .pdf
- .txt

Завантажити текстовий документ у всіх форматах на свій **Google диск** (створити окрему папку).

Надати доступ Вашій вчительці/Вашому вчителеві за допомогою електронної пошти.

## Домашнє завдання

Створити текстовий документ на тему «Новинки Google документів, або Що тут унікальне?».

Середовище виконання: **Google документи**.

Вимоги до текстового документа:

- Наявність кількох списків.
- Встановлення правильних відступів.
- На кожній сторінці додати кілька ілюстрацій, зображень, скріншотів.
- Додати вступ та висновок.
- Додати номер сторінки.
- Єдиний стиль форматування тексту.

Надати доступ Вашій вчительці/Вашому вчителю за допомогою електронної пошти.

# Форматування текстових документів за допомогою стилів. Автоматизоване формування змісту документа

Тема 13

**Ключові поняття**

**Стилі**
**Зміст документа**
**Гіперпокликання**
**Схема документа**

## Стилі

Оформлення текстових документів за допомогою стилів — це спосіб надати тексту конкретний вигляд та структуру, не використовуючи ручного форматування кожного окремого елемента.

Це робиться для того, щоб зробити текст більш читабельним, організованим та професійним.

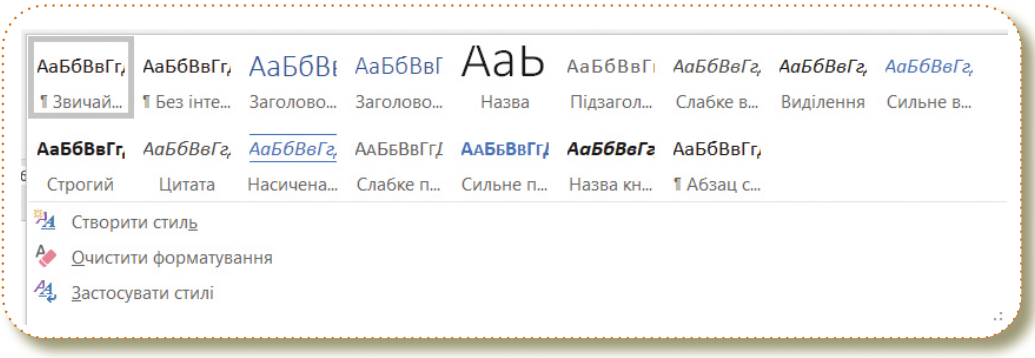
Панель стилів містить список доступних стилів. Вона розташована на вкладці **Основне** в групі **Стилі**.

**Стилі текстового редактора**

Готові стилі, які надаються текстовим редактором. Вони зазвичай використовуються для форматування звичайного тексту, заголовків, списків тощо.

**Власні стилі**

Створюються користувачем. Вони можуть бути налаштовані відповідно до індивідуальних потреб і вимог.



## Переваги використання стилів

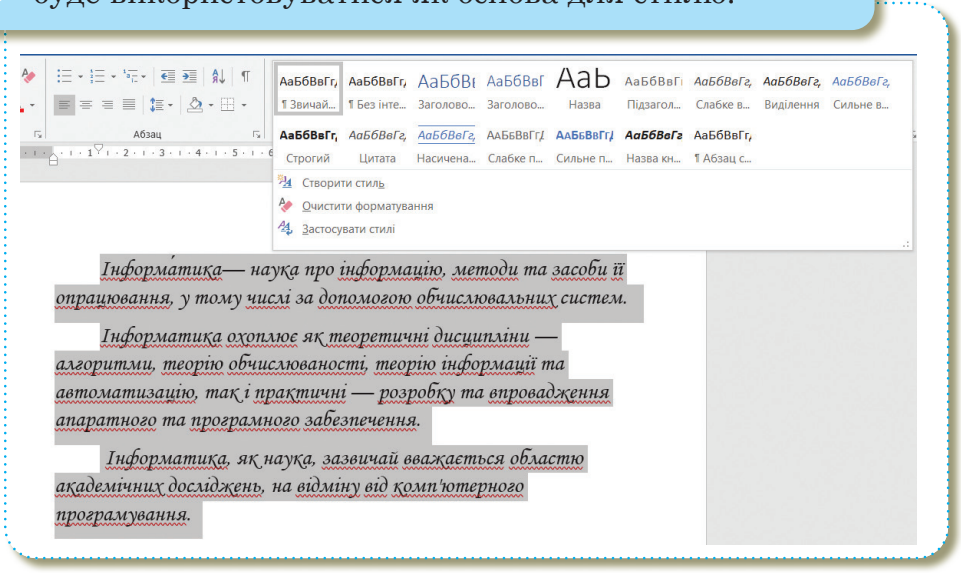
Для застосування стилю до тексту або іншого об'єкта документа необхідно виділити цей об'єкт і вибрати потрібний стиль на панелі стилів.

Стилі дають змогу швидко та легко змінювати зовнішній вигляд документа, а також зберігати його уніфікованість.

- *Швидкість і зручність форматування.* За допомогою стилів можна швидко та легко змінювати зовнішній вигляд документа, не заходячи в діалогові вікна форматування.
- *Уніфікованість документа.* Стилі дозволяють уніфікувати зовнішній вигляд документа, що покращує його читабельність і естетику.
- *Зручність редагування документа.* Якщо Ви змінюєте параметри стилю, то ці зміни автоматично відбуватимуться в усіх об'єктах, до яких цей стиль застосовано.

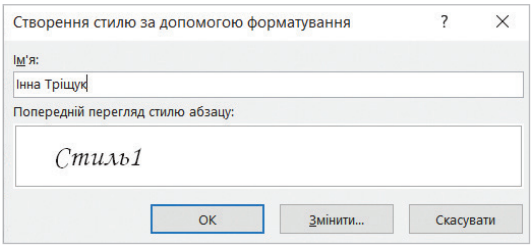
## Створення власних стилів

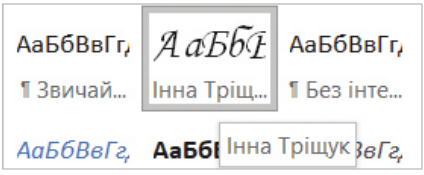
**1** Виділити текст або інший об'єкт документа, який буде використовуватися як основа для стилю.




**2** На панелі стилів натиснути кнопку **Створити стиль**.

**3** У діалоговому вікні **Створення стилю** за допомогою форматування ввести ім'я стилю та налаштувати його параметри форматування.




Параметри форматування стилів залежать від типу об'єкта, до якого застосовується стиль.

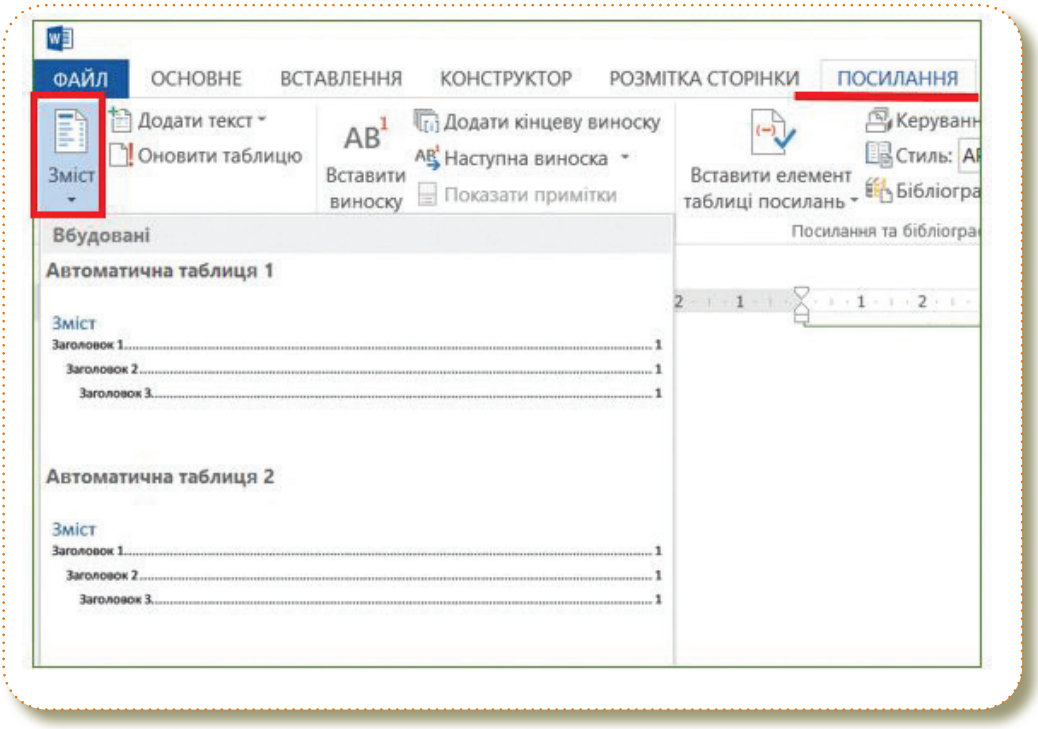
- **Стилі символів** — застосовуються до одного чи більшої кількості виділених символів.
- **Стилі абзаців** — задають оформлення поточному абзацу.
- **Зв'язані стилі** — залежно від вибору застосовуються або до абзацу, або до виділених символів.
- **Стилі списків** — застосовуються як до маркерів або номерів списку, так і до тексту списку.

## Зміст документа

**Зміст документа** — це перелік назв структурних частин документа з відповідними номерами сторінок, де цей фрагмент тексту розпочинається.

Для швидкого переміщення по документу, пошуку потрібних структурних частин — розділів, пунктів тощо слід у змісті документа вибрати вказівником потрібний заголовок. Після наведення вказівника на гіперпосилання курсор набуває вигляду вказівного пальця, для переходу за гіперпосиланням потрібно втримувати натиснутою клавішу **Ctrl**.

Щоб автоматично створити зміст документа, слід для кожної із його структурних частин використати форматування із застосуванням стилів заголовків — **Заголовок 1**, **Заголовок 2** тощо.



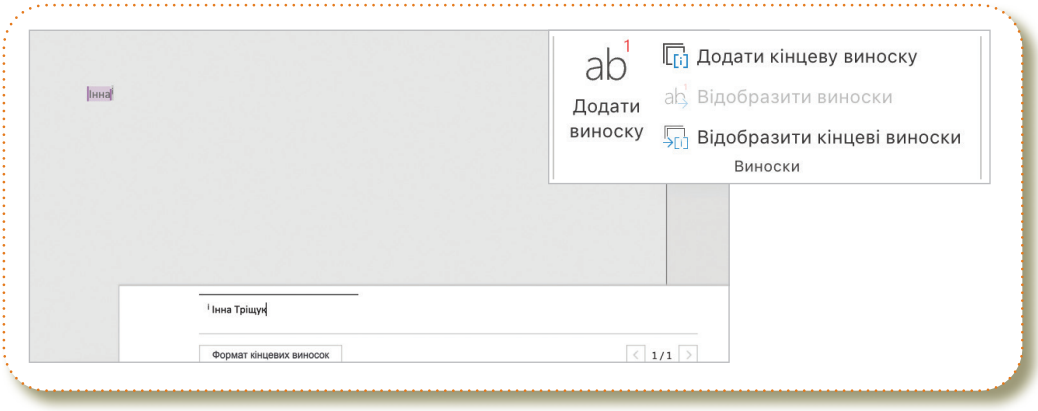

## Автоматизоване форматування змісту документа

### Для автоматизованого створення змісту потрібно:

1. Відформатувати назви розділів, параграфів тощо відповідними стилями.
2. Вибрати місце вставлення змісту (на початку або в кінці документа).
3. На вкладці **Посилання** в групі **Зміст** розгорнути список **Зміст**.
4. Вибрати стиль змісту та натиснути клавішу **Enter**.

**Виноски (алфавітний покажчик)** — список слів із зазначенням номерів сторінок, на яких вони згадуються.

Щоб видалити елемент виноски, потрібно відобразити недруковані символи тексту (**Основне** →) і видалити біля елемента виноски позначку у фігурних дужках.




## Гіперпокликання

**Покликання (гіперпокликання)** — частина документа (малюнок, текст тощо), клацанням по якій здійснюється перехід у певне місце документа або до іншого файлу, розміщеного на комп'ютері користувача або в інтернеті.

**Внутрішні гіперпосилання** можуть бути створені для переходу в межах одного документа (на фрагмент тексту, зображення, таблицю, схему, бібліографічні джерела або інші об'єкти).

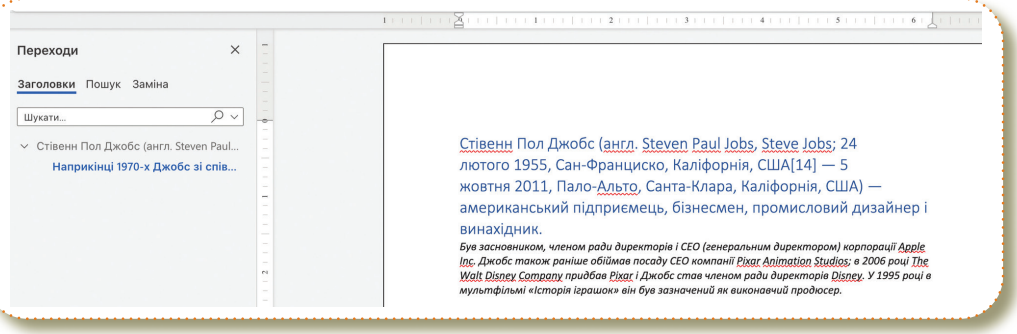
**Зовнішні гіперпосилання** створюються для переходу з текстового документа на зовнішні ресурси (вебсайти, матеріали на **Google Диску**, сторінку в соціальних мережах, папку в локальній мережі, на відправлення листа електронною поштою тощо).

## Схема документа

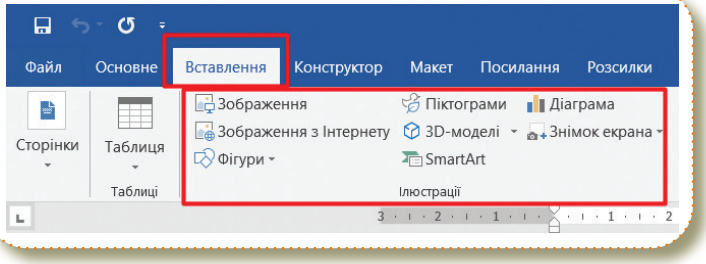
Відобразити структурну схему документа можна декількома способами: або перейти до вкладки **Подання**, або обрати **Навігація**.

При роботі з достатньо об'ємними документами користувач неминуче стикається з потребою швидко здійснювати переходи між розділами цього документа або мати постійний доступ до його структури. Однак в багатьох випадках такі документи не мають структури.

Швидко встановити стилі заголовків **Heading 1 (Заголовок 1)**, **Heading 2 (Заголовок 2)** … можна за допомогою клавіатурних скорочень **Ctrl + Alt + 1, 2, 3**.

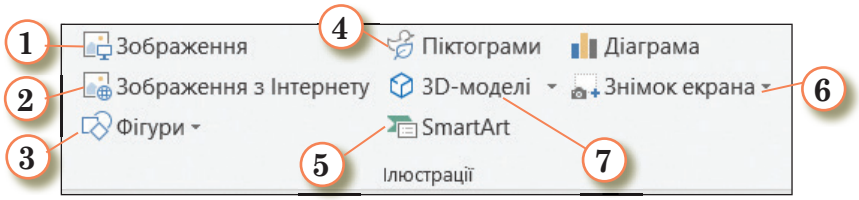



## Вставка графічних об’єктів

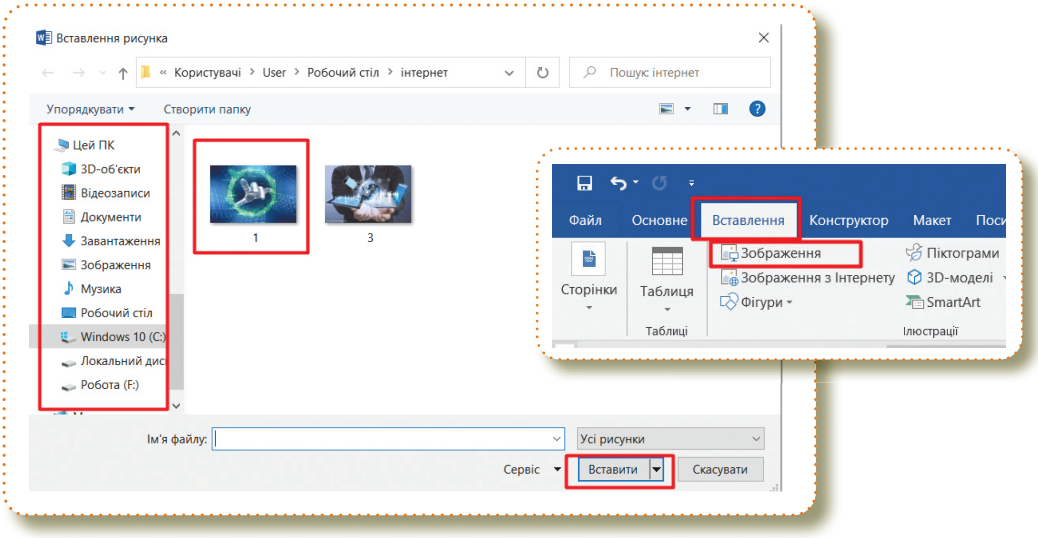



Вставка графічних об’єктів здійснюється з вкладки **Вставлення.**

1. Завантаження зображень, збережених на Вашому комп’ютері.
2. Вставка зображень за допомогою **bing** — пошукової системи, що належить компанії **Microsoft**.
3. Вставка готових фігур.
4. Піктограми в колекції **Microsoft** — векторні файли (**svg**).
5. Зображення **SmartArt** для візуального подання інформації.
6. Вставка зображення екрана (скріншот).
7. Вставка **3D-моделей** із веббібліотеки **Microsoft**.



## Вставити зображення з файла



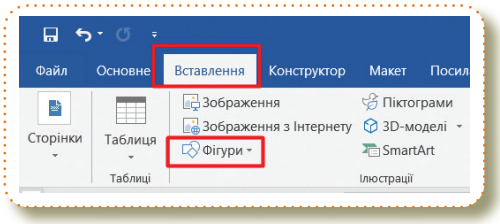

1. Обрати вкладку **Вставлення/ Зображення**.
2. У діалоговому вікні за допомогою файлового провідника знайти папку з потрібним файлом.
3. Виділити один файл або декілька файлів зображень.
4. Натиснути кнопку **Вставити**.

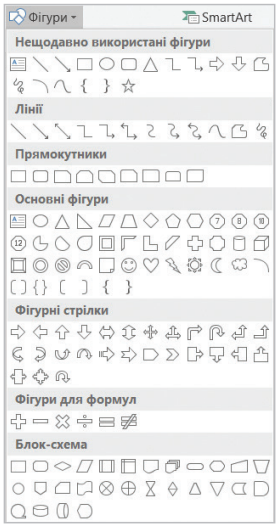
## Вставити зображення з інтернету


1. Обрати вкладку **Вставлення/ Зображення** з інтернету.
2. У діалоговому вікні ввести ключове слово.
3. Обрати одне або кілька зображень.
4. Натиснути кнопку **Вставити**.

## Вставити фігуру

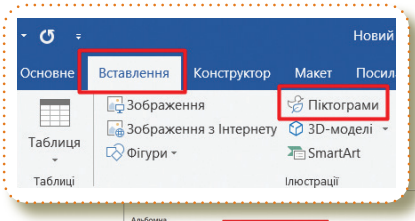
1. Обрати вкладку **Вставлення/ Фігури**.
2. Вибрати потрібну фігуру.
3. Клацнути у місці її вставлення та перетягнути вказівник миші, щоб накреслити фігуру.
4. Редагувати та форматувати фігуру.

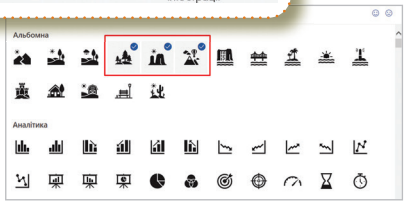



## Вставити піктограму

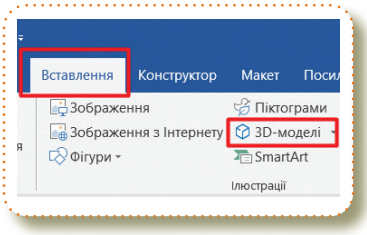
1. Обрати вкладку **Вставлення/ Піктограми**.
2. Вибрати одну або кілька піктограм.
3. Натиснути кнопку **Вставити**.
4. Редагувати та форматувати піктограму.

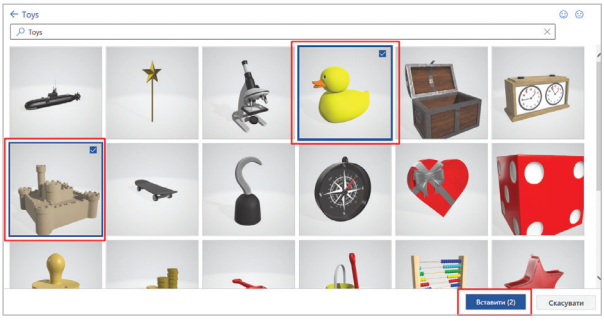


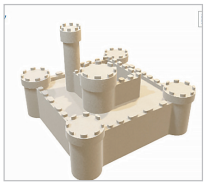



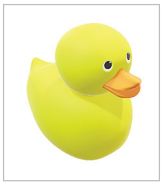

## Вставити 3D-модель

1. Обрати вкладку **Вставлення/3D-моделі**.
2. Вибрати одну або кілька 3D-моделей.
3. Натиснути кнопку **Вставити**.



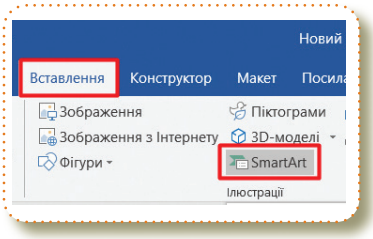



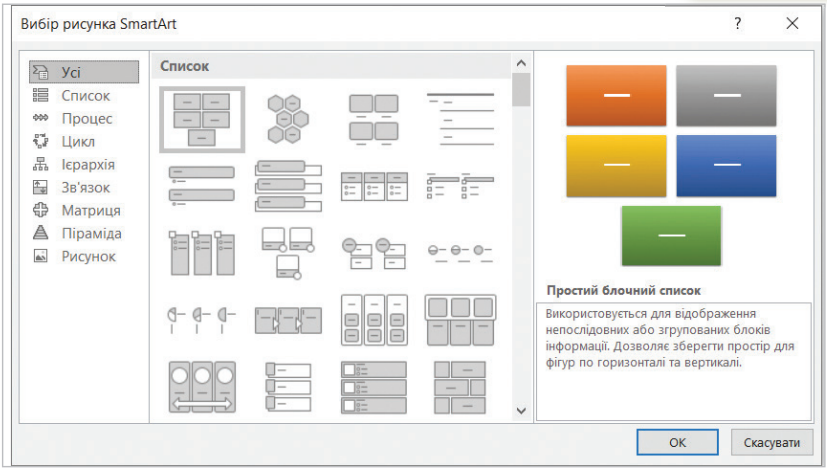





## Вставити SmartArt

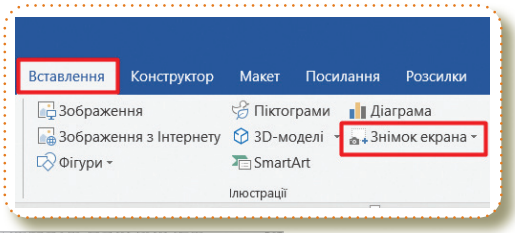
1. Обрати вкладку **Вставлення/ SmartArt**.
2. Вибрати тип **SmartArt**.
3. Натиснути кнопку **Ok**.
4. Редагувати та форматувати **SmartArt**.

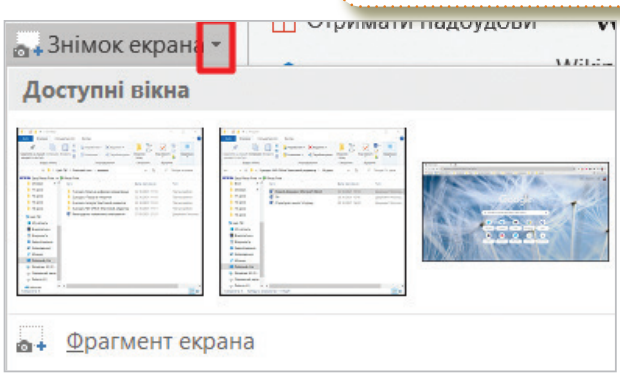


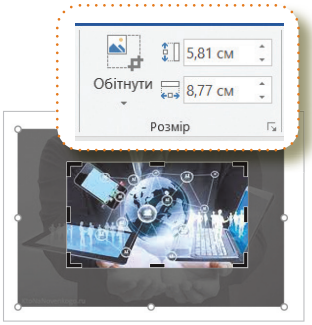

## Вставити знімок екрана

1. Обрати вкладку **Вставлення/ Знімок екрана**.
2. Обрати потрібне вікно.
3. Розмістити його в текстовому документі.
4. Редагувати та форматувати зображення.



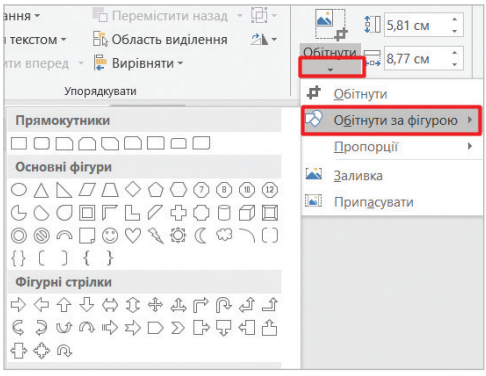
## Обітнути зображення




1. Виділити зображення.
2. Обрати вкладку **Знаряддя для зображення** / **Формат**.
3. Обрати інструмент **Обітнути**.
4. Вибрати потрібні пропорції та натиснути **Обітнути** (або **Enter**).

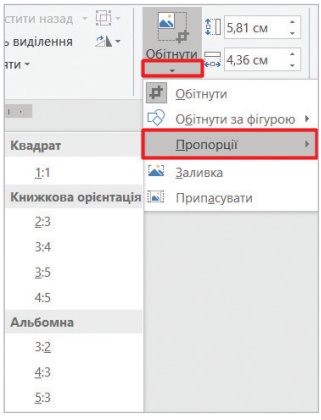
## Обітнути зображення за фігурою

1. Виділити зображення.
2. Обрати вкладку **Знаряддя для зображення** / **Формат**.
3. Обрати інструмент **Обітнути**/ **Обітнути за фігурою**.
4. Переглянути результат.



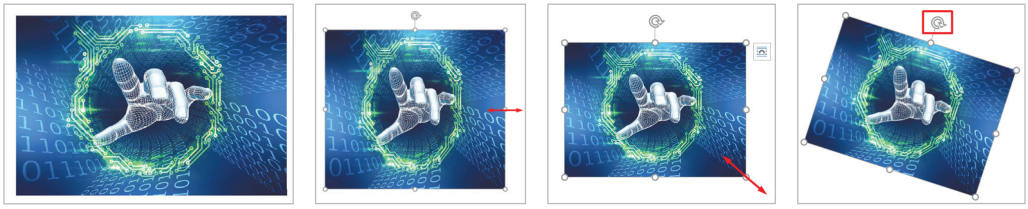

## Обітнути зображення до пропорцій

1. Виділити зображення.
2. Обрати вкладку **Знаряддя для зображення** / **Формат**.
3. Обрати інструмент **Обітнути**/ **Пропорції**.
4. Переглянути результат.



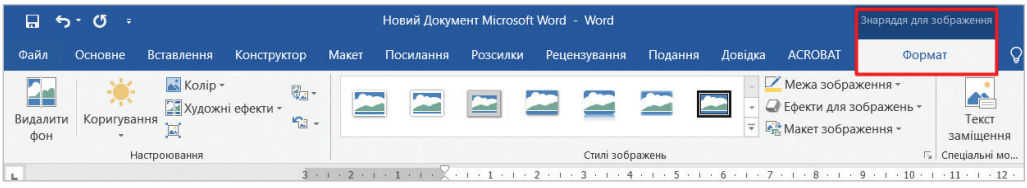
## Зміна розмірів та кута нахилу зображення

Щоб повернути зображення, потрібно затиснути лівою кнопкою мишки й утримувати кругову ручку, прикріплену до верхньої межі зображення.



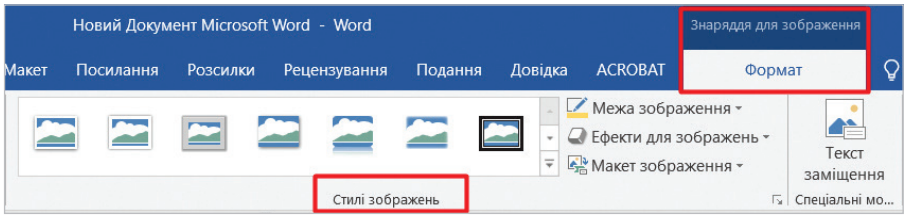
## Параметри зображення

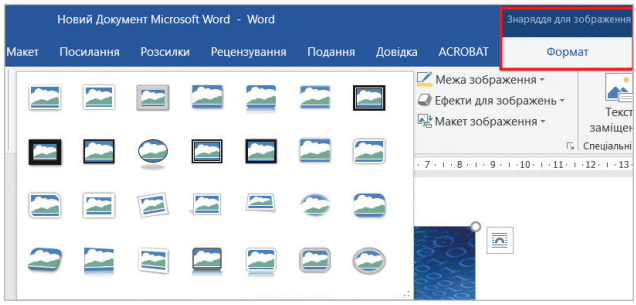
Щоб відкрити параметри зображення, потрібно натиснути на зображення та обрати вкладку **Знаряддя для зображення/ Формат**.




## Зміна стилю зображення

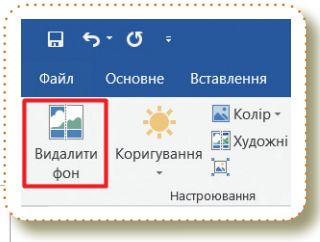
До зображень, що містяться в текстовому документі, можна застосовувати деякі стилі та ефекти оформлення.




## Видалення окремих частин із зображення

1. Виділити зображення.
2. Перейти до вкладки **Формат зображення**.
3. Обрати інструмент **Видалити фон**.
4. Для застосування результату натиснути на кнопку **Зберегти зміни**.




## Властивості зображення в текстовому документі

**Розташування на сторінці** — визначає місце розміщення зображення на сторінці відносно країв (полів) сторінки. Властивість може набувати значення: вгорі ліворуч, посередині по центру, внизу праворуч, посередині ліворуч та інше.

**Спосіб обтікання зображення текстом** — визначає взаємне розміщення тексту і зображення на сторінці документа.

Може набувати значення: у тексті, навколо рамки, за контуром, за текстом, перед текстом тощо.

| *У тексті* | *Навколо рамки* | *Перед текстом* | *За текстом* |
|---|---|---|---|
| Тризуб є давнім українським символом. Його найдревніше зображення, віднайдене археологами, датується X століттям. | Тризуб є давнім українським символом. Його найдревніше зображення, віднайдене археологами, датується X століттям. | Тризуб є давнім українським символом. Його найдревніше зображення, віднайдене археологами, датується X століттям. | Тризуб є давнім українським символом. Його найдревніше зображення, віднайдене археологами, датується X століттям. |

## Практична робота №13

### Частина 1. Вставка графічних елементів

***Примітка для вчительки/вчителя***

Завантажте документ на Робочий стіл **ПР №13 (частина 1).docx**.

Учні/учениці працюють у цьому документі.

**Завдання.**

- Додати до кожного розділу 2 зображення.
- Додати скріншот додаткової інформації про IT-стартапи.
- Додати кілька піктограм та фігур.
- Зберегти документ у форматі **pdf**.
- Завантажити на свій **Google диск** та надати покликання з можливістю перегляду для Вашої вчительки/Вашого вчителя.

### Частина 2. Створення власних стилів

**Завдання.**

- Створіть текстовий документ.
- Здійсніть пошукову роботу в мережі «Інтернет» на тему: «Копірайт та рерайт — основні характеристики».
- Зробіть 2 абзаци з різним форматуванням тексту та зображень.
- Створіть 2 власні стилі та назвіть своїм прізвищем та ім'ям.
- Збережіть документ у форматі **pdf**.
- Завантажте на свій **Google диск** та надайте покликання з можливістю перегляду Вашій вчительці/Вашому вчителеві.

## Домашнє завдання

Проекспериментувати з власними стилями.

Створити кілька власних стилів на Ваш вибір.

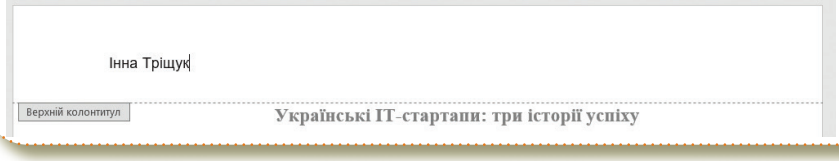
## Тема 14 Колонтитули та нумерація сторінок. Шаблони документів (резюме, портфоліо, буклет, афіша тощо)

Ключові поняття

**Колонтитул**
**Нумерація сторінок**
**Орієнтація сторінки**
**Шаблон документа**

### Колонтитул

**Колонтитули** — це інформація, яка розміщується у верхній, нижній чи бічній частині сторінки, наприклад, назва розділу, номер сторінки, дата.




*Як зробити:*

- Перейдіть на вкладку **Вставка**.
- Оберіть **Колонтитули** і виберіть потрібний стиль чи створіть власний.
- Додайте текст чи елементи до колонтитула.

Ви можете мати різні колонтитули для різних частин документа (наприклад, різні для титульної сторінки та основного тексту).

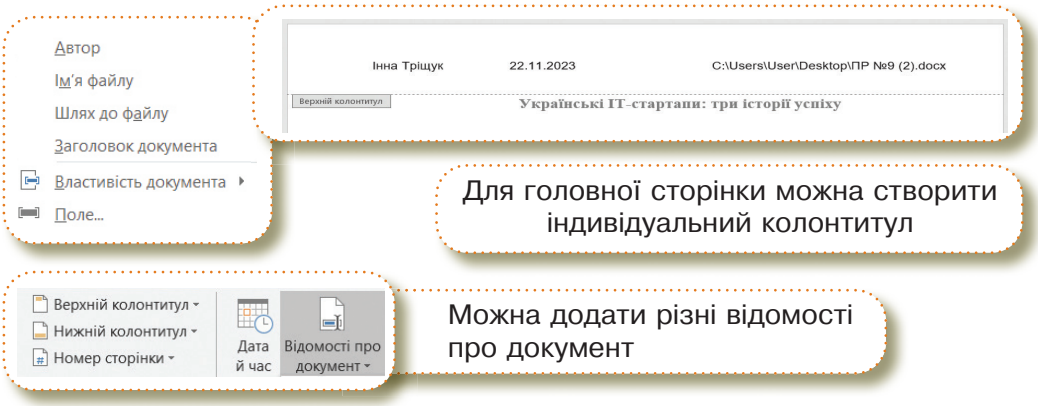
*Як зробити:*

- Виберіть **Розділ** або розділіть документ на розділи.
- Налаштуйте колонтитули для кожного розділу окремо.

Колонтитули дозволяють стандартизувати та уніфікувати вигляд документа, особливо коли це документ з багатьма сторінками. Їх можна форматувати, змінювати шрифти та розміщення тексту, додавати лінії розділу.

## Редагування колонтитулів

Також можна клікнути у верхній або нижній частині екрана, після чого Ви побачите позначення зони колонтитула і зможете внести туди текст або графіку.

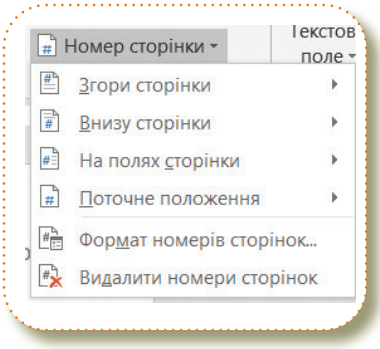



### Нумерація сторінок

**Нумерація сторінок** — це процес присвоєння унікальних номерів кожній сторінці вашого документа.

*Як зробити:*

- Перейдіть на вкладку **Вставка**.
- Оберіть **Номер сторінки** і виберіть розташування та формат нумерації.




Ви можете вибрати, з якої сторінки починати нумерацію та додати інші налаштування.

Такий знак (номер) може розташовуватися у верхньому, нижньому, лівому чи правому куті сторінки і вказувати на те, на якій саме сторінці перебуваєте.

Якщо потрібно, Ви можете налаштувати формат та вигляд номера сторінки. Тут можете вказати, з якої сторінки починати нумерацію.

Якщо Ви вставляєте нові сторінки перед існуючими або після них, програма автоматично перераховує та оновлює номери сторінок відповідно до нової структури документа.

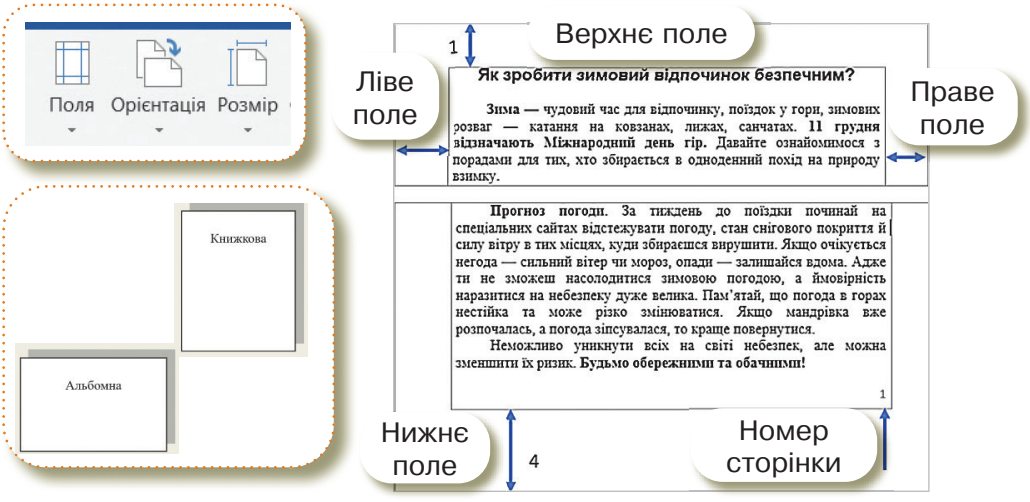
## Властивості сторінки текстового документа

**Поля** — це вільні від основного тексту ділянки сторінки вздовж країв аркуша, які залишають для різних приміток і кращого сприйняття тексту. Наприклад, у Вашому зошиті чи підручнику теж є поля. На сторінці є верхнє, нижнє, ліве і праве поля.

**Розмір полів** — це відстань від відповідного (верхнього, нижнього, лівого, правого) краю аркуша до тексту. Розміри полів за замовчуванням у текстовому процесорі задаються в сантиметрах.

**Орієнтація сторінки** — це спосіб розміщення сторінки на площині. Ця властивість може набувати такі значення: книжкова (вертикальна) та альбомна (горизонтальна).

**Розміри сторінки** — це висота і ширина аркуша, на якому планується друкувати документ. Наприклад, більшість текстових документів друкують на аркуші паперу формату А4, який має такі розміри: ширина 21 см та висота 29 см 7 мм. А аркуш паперу формату А5 має такі розміри: ширина — 14 см 8 мм, висота — 21 см.




На сторінці документа, за потребою, вказують її номер. Він може розміщуватися угорі, внизу чи на лівому або правому полях сторінки з вирівнюванням по центру, лівому чи правому краю тощо. Завдяки нумерації в документах можна швидко знайти потрібну сторінку.

## Шаблон документа

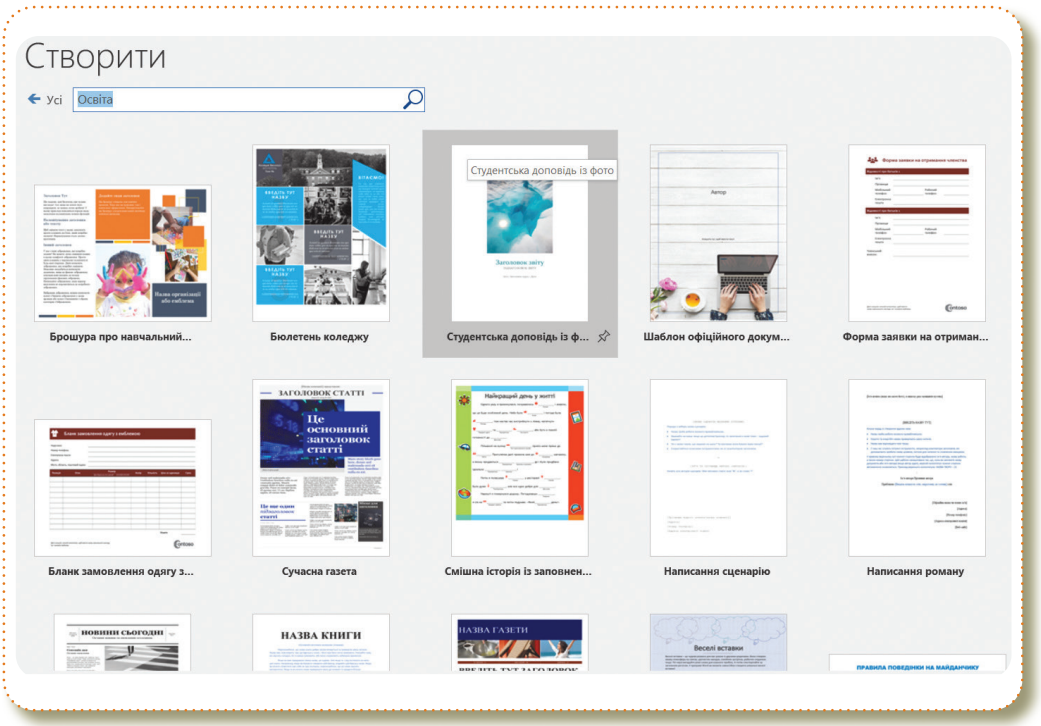
**Шаблон документа** — зразок або заготовка документа з певною структурою, форматуванням та вже встановленими елементами (наприклад, заголовками, номерами сторінок).

Під час створення нового документа виберіть **Новий** та оберіть шаблон, який Вам підходить. Тут можуть бути шаблони для листів, резюме, наукових статей тощо.

Шаблони можуть містити основні елементи, такі як заголовки, підписи, вставки зображень, вже налаштоване форматування тощо.

Ви також можете створювати власні шаблони, визначаючи вигляд та структуру, а потім зберігаючи їх для майбутнього використання.

В **MS Word** є вбудована колекція шаблонів, які Ви можете вибрати при створенні нового документа.



## Практична робота №14

**Завдання.** Створюємо шаблони документів.

### Частина 1

- В шаблонах **MS Word** обираємо категорію **Освіта** та шаблон **Розклад щотижневих завдань**.
- Наповнюємо вмістом.
- Обов'язково додаємо колонтитули з різним вмістом.
- Зберігаємо в форматі **pdf** та **doc** в локальному сховищі (рекомендовано на **Робочий стіл**).
- Створюємо папку на своєму **Google диску**.
- Завантажуємо в папку на **Google диску** та надаємо доступ Вашій вчительці/Вашому вчителеві.

Розклад щотижневих завд...

### Частина 2

- У шаблонах **MS Word** обираємо категорію **Освіта** та шаблон **Подяка**.
- Наповнюємо вмістом.
- Обов'язково додаємо колонтитули з різним вмістом.
- Зберігаємо в форматі **pdf** та **doc** в локальному сховищі (рекомендовано на **Робочий стіл**).
- Створюємо папку на своєму **Google диску**.
- Завантажуємо в папку на **Google диску** та надаємо доступ Вашій вчительці/Вашому вчителеві.


Подяка із зображенням к...

### Частина 3

- В шаблонах **MS Word** обираємо категорію **Освіта** та шаблон **Навчальний календар**.
- Наповнюємо вмістом.

- Обов'язково додаємо колонтитули з різним вмістом.
- Зберігаємо в форматі **PDF** та **DOC** в локальному сховищі (рекомендовано на **Робочий стіл**).
- Створюємо папку на своєму **Google диску**.
- Завантажуємо в папку на **Google диску** та надаємо доступ Вашій вчительці/Вашому вчителеві.


**Навчальний календар (од...**

## Домашнє завдання

В шаблонах **MS Word** обираємо категорію **Освіта** та шаблони.

- **Розклад домашніх завдань на тиждень**.
- **Розклад щотижневих завдань**.
- **Навчальний календар**.

Наповнюємо вмістом.

Обов'язково додаємо колонтитули з різним вмістом.

Зберігаємо в форматі **pdf** та **doc** в локальному сховищі (рекомендовано на **Робочий стіл**).

Створюємо папку на своєму **Google диску**.

Завантажуємо в папку на **Google диску** та надаємо доступ Вашій вчительці/Вашому вчителеві.

## Тема 15 Публікації для соціальних мереж. Роль тексту для формування цифрового образу

Ключові поняття

**Публікації для соціальних мереж**
**Хештег**
**Репост**
**Коментар**
**Цифровий образ**

### Публікації для соціальних мереж

**Публікації для соціальних мереж** — текстовий та візуальний контент, змістом якого Ви ділитеся на своїй сторінці або профілі в соціальних мережах.

Такі публікації містять фотографії, відео, тексти, посилання та інші елементи, що дозволяють Вам спілкуватися та ділитися враженнями з Вашими друзями/подругами.

**Вибір теми**

Оберіть тему або ідею, якою Ви хочете поділитися з іншими. Це може бути Ваша подорож, захоплення, особисті думки або події.

**Формат публікації**

Розгляньте, який формат найкраще підходить Вашій темі. Це може бути фотографія, кілька фотографій в слайд-шоу, відео, тексти або комбінація цих елементів.

**Опис та хештеги**

Додайте короткий опис або пояснення до Вашої публікації. Також використовуйте хештеги для підвищення видимості вашого контенту серед інших користувачів/користувачок.

**Графіка та дизайн**

Зверніть увагу на дизайн Вашої публікації. Вибирайте гармонійні кольори та шрифти, які підходять до Вашого стилю.

**Перевірка перед публікацією**

Перегляньте Ваш контент перед публікацією, переконайтеся, що всі елементи виглядають так, як Ви хочете.

## Підготовка текстів для дописів

1. **Персональний стиль**
Розвивайте власний стиль написання. Якщо Ви ведете особистий блог чи акаунт, намагайтеся зберігати узгодженість у виразі та тоні.

2. **Цільова аудиторія**
Перед тим як почати писати, подумайте про те, до кого Ви звертаєтеся. Спробуйте визначити інтереси та вподобання своєї аудиторії.

3. **Коротко та зрозуміло**
Спробуйте виражати свої думки коротко і зрозуміло. Короткі речення та прості слова роблять Ваші повідомлення легкими для читання.

4. **Інтерактивність**
Заохочуйте своїх читачів/читачок до спілкування, вживаючи питання чи заклики до дії. Наприклад, «Що Ви думаєте про це?» або «Поділіться власним досвідом».

5. **Приклади та описи**
Додавайте приклади або конкретні деталі, що зроблять Ваш текст більш живим та цікавим.

## Основні терміни при роботі з соціальними мережами та написанні текстів для публікацій

**Хештег (#Hashtag)** — ключове слово або фраза, яким передує символ #. Хештеги потрібні для легкого пошуку публікацій.

**Репост (Retweet/Share)** — дія, яка дозволяє поділитися чужою публікацією зі своїми підписниками. В різних соціальних мережах це може мати різні назви.

**Коментар (Comment)** — відгук чи думка, яку користувач/користувачка залишає під публікацією чи фото.

**Локація (Location Tagging)** — додавання географічної мітки до публікації для показу місця, де вона була створена.

**Особисті повідомлення (Direct Message)** — приватне повідомлення, яке користувач/користувачка може відправити іншому користувачеві, якщо вони обидва підписані один на одного.

### Цифровий образ

**Цифровий образ** — спосіб, яким користувач/користувачка представляє себе в онлайн-середовищі, і він/вона може суттєво впливати на сприйняття користувача/користувачки іншими в інтернеті.

Текст відіграє важливу роль у формуванні цифрового образу особистості в інтернеті.

**Самовираження.** Текст дозволяє користувачам/користувачкам висловлювати свої думки, ідеї та почуття. Він є засобом для вираження особистості, уподобань та поглядів.

**Брендування.** Текст допомагає формувати бренд особистості в інтернеті. Відповідно до того, як людина висловлює свої думки та взаємодіє з іншими, формується образ, який стає її цифровим брендом.

**Спілкування та взаємодія.** Текст є основним засобом комунікації в інтернеті. Користувачі взаємодіють між собою через написані повідомлення, коментарі, блоги тощо. Текст створює можливість для обміну інформацією та вираження підтримки чи відмови.

## Практична робота №15

**Завдання 1.** Створити допис для соціальної мережі «Інстаграм».

Тема допису **«Мої лайфхаки ефективного навчання»**.

**1.** Готуємо текст.

- Середовище виконання — **Google Docs**.
- Кількість символів — 500–700.
- Додати хештеги.
- Використовуємо смайлики та відступи.

**2.** Готуємо графіку за допомогою графічного онлайн-редактора **Canva**.

- Авторизуємося в графічному онлайн-редакторі **Canva**.
- Обираємо шаблон — **Допис**.
- Пишемо заголовок нашого допису.
- Підбираємо елементи графіки.
- Завантажуємо зображення та вставляємо в текстовий документ.

Надаємо доступ Вашій вчительці/Вашому вчителеві за допомогою електронної пошти.

**Завдання 2.** За аналогічним алгоритмом створити допис для Instagram.

Тема на Ваш вибір.

Створіть графіку та текст допису.

Тема **«Фотодайджест — красиві місця мого рідного міста»**.

Надайте доступ Вашій вчительці/Вашому вчителеві за допомогою електронної пошти.

**Завдання 3.** За аналогічним алгоритмом створити допис для **Instagram**.

Створіть серію розповідей в **Instagram**.

Тема **«Онлайн-сервіси для навчання»**.

На графіці додайте текст публікації та кілька скріншотів сервісу.

Надайте доступ Вашій вчительці/Вашому вчителеві за допомогою електронної пошти.

**Завдання 4.** За аналогічним алгоритмом створити допис для **Instagram**.

Анімований допис в **Instagram**.

Тема на Ваш вибір.

Надайте доступ Вашій вчительці/Вашому вчителеві за допомогою електронної пошти.

## Домашнє завдання

Створення допису для соціальної мережі «Інстаграм».

Тема допису — на Ваш вибір.

Готуємо текст.

- Середовище виконання — **Google Docs**.
- Кількість символів — 500-700.
- Додати хештеги.
- Використовуємо смайлики та відступи.

Готуємо графіку за допомогою графічного онлайн-редактора **Canva**.

- Авторизуємося в графічному онлайн-редакторі **Canva**.
- Обираємо шаблон — **Допис**.
- Пишемо заголовок нашого допису.
- Підбираємо елементи графіки.
- Завантажити зображення та вставити в текстовий документ.

Надати доступ Вашій вчительці/Вашому вчителеві за допомогою електронної пошти.

# Засоби для перекладу текстів

Тема 16

**Ключові поняття**

**Онлайн-перекладач**
**Google Translation**
**Bing Translator**
**DeepL**

## Онлайн-перекладач

**Онлайн-перекладач** — програма або сервіс, що дозволяє виконувати переклад текстів, слів, документів і вебсайтів, не завантажуючи її на свій комп'ютер.

*Переваги перекладачів:* зручно, швидко, допомагає зрозуміти суть тексту, безкоштовно.

*Недоліки перекладачів:* неможливість користування без доступу до інтернету, неточність перекладу, іноді навіть некоректність, помилки.

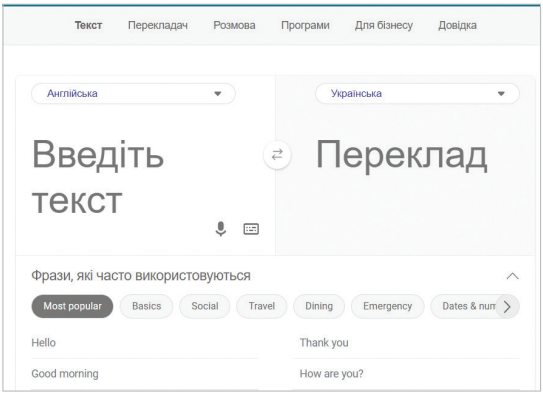
Варто звернути увагу на:

- **Контекст.** Перекладачі не завжди можуть правильно розуміти контекст речення чи тексту. Важливо враховувати смислові нюанси та можливість множинного тлумачення.
- **Перевірка та корекція.** Після отримання перекладу важливо його перевірити та внести корекції, якщо це необхідно. Автоматизовані перекладачі не завжди відображають усі відтінки мовлення.
- **Мовні особливості.** Кожна мова має свої унікальні особливості, такі як вживання ввічливих форм, поняття однини чи множини та інші. Важливо враховувати ці особливості для найточнішого перекладу.

## Bing Translator

Хороший переклад онлайн тексту на 50 мов. Працює зі звичайним текстом і вебсторінками. Є можливість озвучування слів.

Дизайн сучасний, але не такий інтуїтивно зрозумілий, як у **Google**, доведеться трохи розібратися з навігацією по сайту до початку використання.



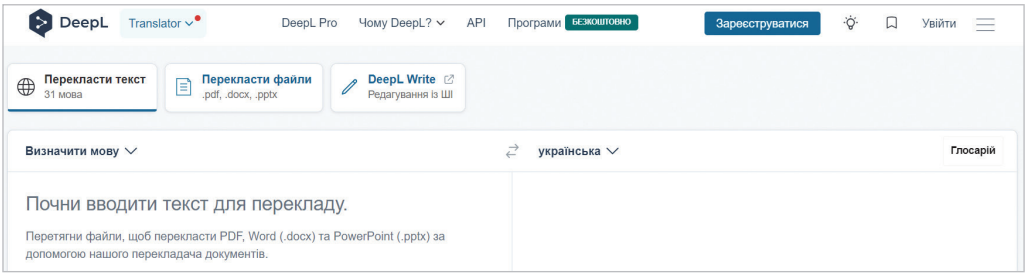

## DeepL

**DeepL** використовує машинне навчання для створення близьких за змістом перекладів. Ним можна користуватися для перекладу коротких текстів або навіть великих документів.

**DeepL** відомий своєю високою точністю перекладів завдяки використанню технологій глибокого навчання. Він спроможний розпізнавати контекст та виражати смислові відтінки.

**DeepL** може перекладати короткі фрази, а також великі тексти, що робить його корисним для різноманітних завдань — від перекладу листівок до роботи з об'ємними документами.

**DeepL** надає візуальний редактор, який дозволяє користувачам/користувачкам переглядати та вдосконалювати переклади за необхідності, щоб забезпечити ще більшу точність.



## Практична робота №16

*Примітка для вчителя/вчительки:*

Відправте учням/ученицям за допомогою електронної пошти документ для роботи **ПР №16 (частина 1).docx**.

### Завдання 1.

Відкрийте текстовий документ, надісланий за допомогою електронної пошти.

Збережіть на **Робочий диск**, відкрийте та заповніть документ.

*Використати:*

### Завдання 2.

- Створіть новий документ.
- Перекладіть запропонований текст на англійську мову за допомогою онлайн-перекладача на Ваш вибір.
- Текст знайдіть за покликанням **ПР №16 (частина 2).docx**.
- Перекладіть текст онлайн-перекладачем на Ваш вибір.
- Додайте кілька світлин, згенерованих штучним інтелектом.

## Домашнє завдання

Створити **Google документ** та назвати «Ефективність».

Здійснити пошукову діяльність в мережі «Інтернет» на тему «Поради щодо ефективного навчання для учнів».

В документі зробити ТОП 10 правил ефективного навчання.

Перекласти текст за допомогою онлайн-перекладача на Ваш вибір.

Надати доступ Вашій вчительці/Вашому вчителеві за допомогою електронної адреси.

Розділ 5

# ГРАФІЧНЕ ПРОГРАМУВАННЯ

## Тема 17 Середовище програмування: функції та можливості

Ключові поняття

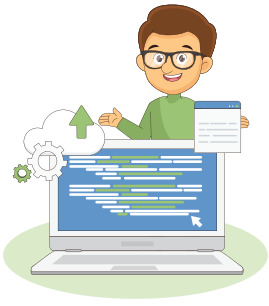
**Що таке програмування**
**Програми, в яких пишуть код Python**
**Величини**
**Типи даних**
**Значення величини**
**Команда виводу Print ()**

### Що таке програмування

**Програмування** — це процес створення програмного коду, який керує поведінкою комп'ютера чи іншого обчислювального пристрою.


Простими словами, програмування — дія, яку повинен виконати комп'ютер.

Програмування передбачає створення алгоритмів (чіткі послідовні інструкції) для виконання завдань.



**Алгоритм** — це набір чітких та послідовних інструкцій, які вказують комп'ютеру, як виконувати конкретне завдання.

**Мови програмування** — це спеціальні мови, якими користувачі комунікують з комп'ютером, формуючи програмний код.

Сьогодні у світі існує понад 300 мов програмування. Кожна з них має свою сферу застосування та відповідає за виконання конкретних завдань.

## Мова програмування PYTHON

**Мова програмування PYTHON** — мова високого рівня, досить «молода», проте дуже популярна.

**Python** розробив Гвідо ван Россум. Першу версію Python (0.9.0) випустили в лютому 1991 року.

## Переваги мови Python

1. синтаксис зрозумілий і легко читається, подібний до англійської мови
2. вдалий вибір як для першої мови в навчанні програмуванню
3. стандартна бібліотека містить багато корисних функцій

Синтаксис **Python** дозволяє розробникам писати програми з меншою кількістю рядків, ніж деякі інші мови програмування.

Важливий і той факт, що необхідне програмне забезпечення, включаючи середовища розробки, переважно безкоштовне.

## Призначення Python

На сьогодні **Python** використовується при реалізації найрізноманітніших проєктів, серед яких:

- створення ігор та комп'ютерної графіки;
- створення програм з графічним інтерфейсом;
- для роботи штучного інтелекту;
- для веброзробки та багато іншого.

## Поняття, пов'язані з процесом виконання програм у мовах програмування

**Програмний код** — це набір інструкцій або команд, написаних мовою програмування, які визначають, як повинна виконуватися програма.

Якщо ми хочемо написати програму на мові **Python**, то для цього потрібно набрати відповідний програмний код.

**Інтерпретатор** — це програма або середовище, яке виконує програмний код.

Інтерпретатор читає кожну команду з програмного коду, перетворює її на машинний код і виконує її. Тому, в принципі, якщо програма складається з декількох команд, її можна організувати у вигляді файлу з програмним кодом, а потім «відправити» цей файл на виконання.

**Компілятор** — це програма, яка перетворює програмний код у код, який зрозумілий комп'ютеру, так званий бінарний код.

Процес компіляції відбувається перед виконанням програми, і після компіляції можна виконувати програму стільки разів, скільки потрібно, без повторної компіляції.

Програми, в яких пишуть код Python

## Де можна працювати і писати код мовою Python

**Для ПК**

IDLE PYTHON

**Для смартфонів**

PYDROID3

**Онлайн-версія**

ONLINEGDB

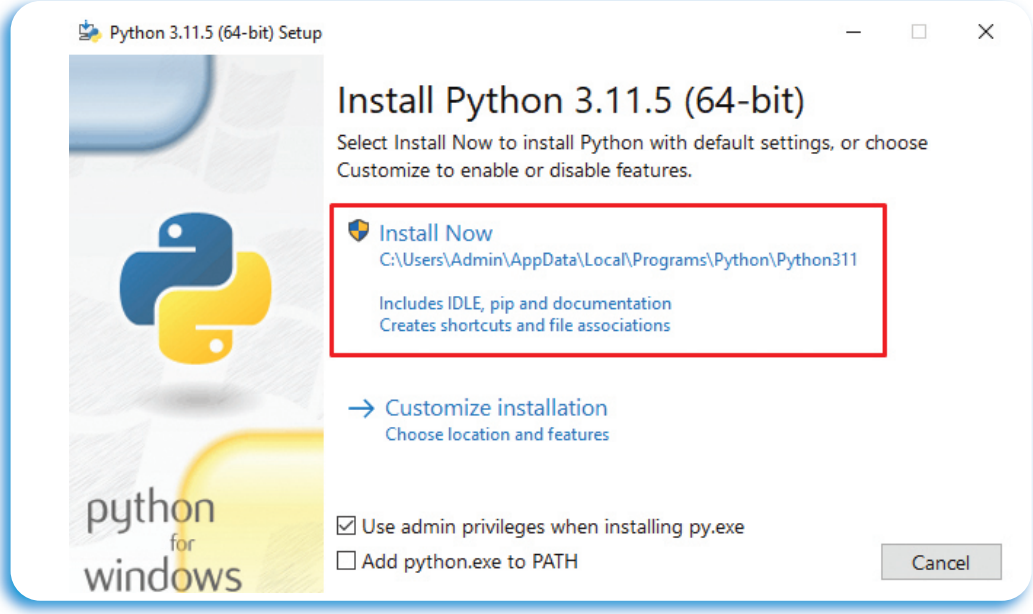
## Інсталяція ПЗ

1. Натисніть на вказану кнопку **Download Python 3/11/5**, очікуйте завантаження інсталяційного файлу.

1

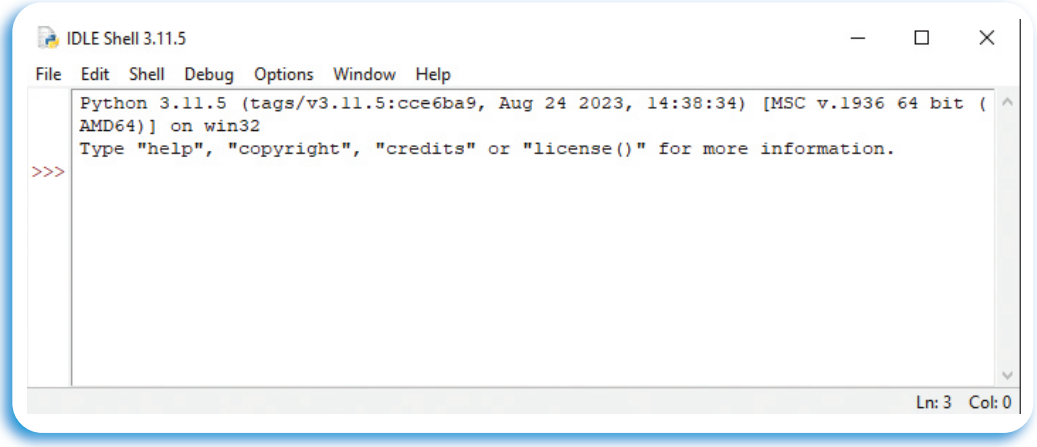

2. Після завантаження інсталяційного файлу запустіть його на виконання. В діалоговому вікні — натисніть на **Install Now** та чекайте, поки програма встановиться на ваш комп'ютер.

2



## Середовище IDLE

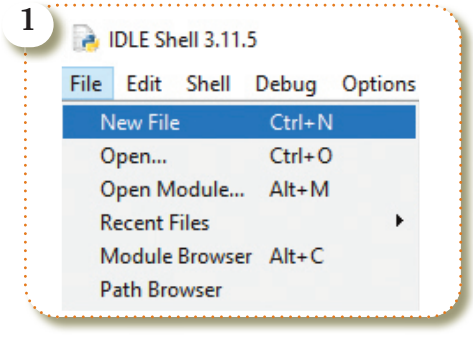
Щоб запустити програму IDLE, потрібно виконати команди: **Пуск — Всі програми — Python — IDLE (Python).**

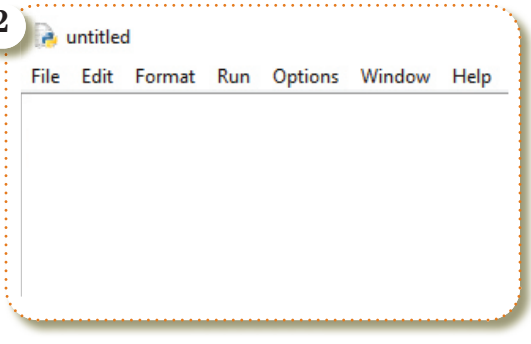



Перед нами командна оболонка інтерпретатора. Це вікно з декількома меню і великою робочою областю, в якій після символу потрійної стрілки **>>>** блимає курсор — це **командний рядок**. У цьому місці вводиться команда, яка виконується після натискання клавіші.

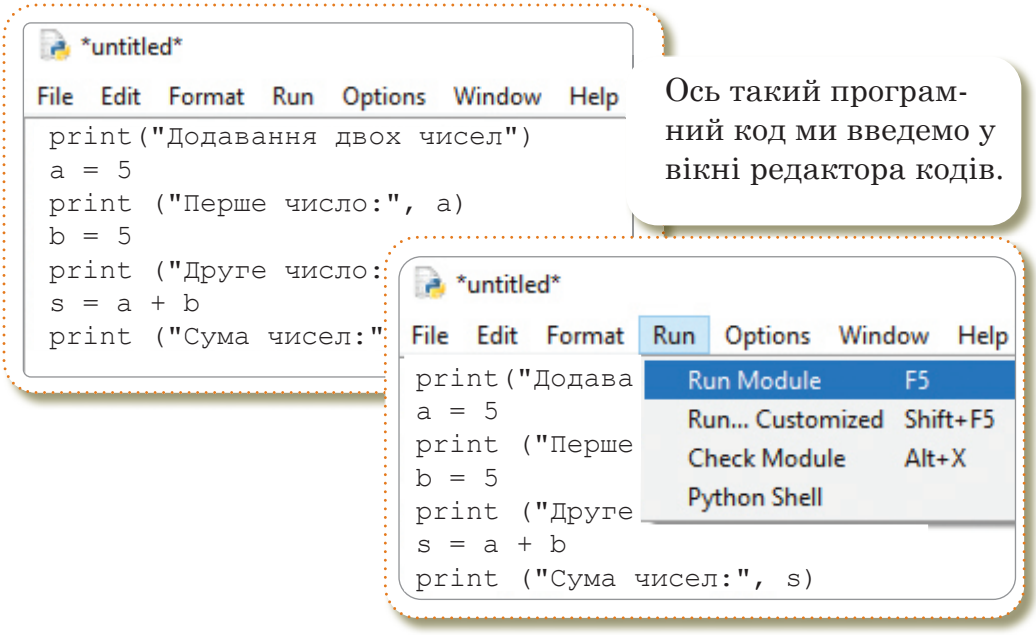
## Створення файлу

1. Для того, щоб створити файл програми, потрібно клацнути меню **File**, відкриється список команд і підменю, серед яких є й команда **New File**.
2. Одразу після вибору цієї команди відкривається редактор кодів. У вікні редактора вводимо програмний код.



## Запуск програми на виконання

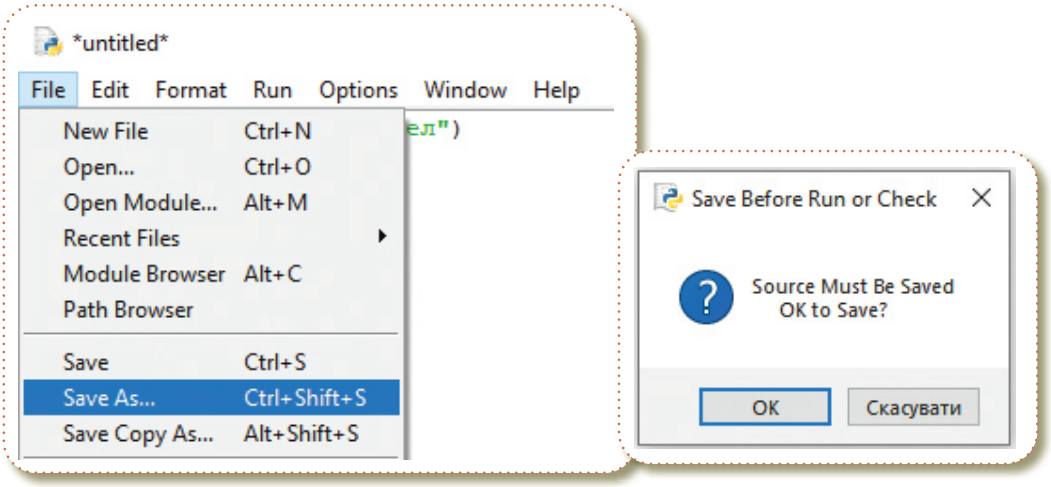


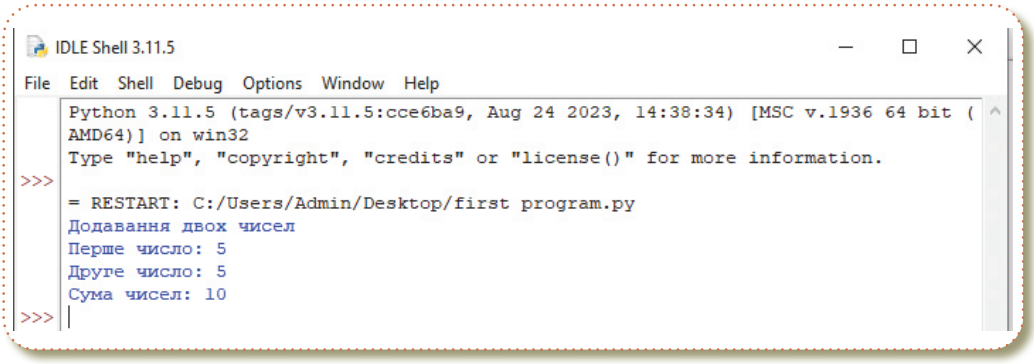

Після того, як програмний код набрано, його можна одразу виконати. Для цього в меню **Run** вибираємо команду **Run Module**.

## Збереження програми та результат

Щоб зберегти проєкт, потрібно натиснути команду **Save** із меню **File**.

Якщо перед запуском програми на виконання файл не зберегти, з'явиться діалогове вікно з пропозицією зберегти файл.






## Алфавіт мови Python містить:

1. великі (A ... Z) та малі (a ... z) букви латинського алфавіту
2. цифри: 1 2 3 4 5 6 7 8 9 0
3. операції: + − * ** / // % @ << >> & | ^ ~ < > <= >= == !=
4. роздільники: ( ) [ ] { } , : . ; @ = -> += -= *=

**Алфавіт мови** — це набір основних символів, з яких повинен складатися будь-який текст програмного коду цієї мови. Жодні інші символи не використовують!

Розмір літер має значення, тобто великі та малі літери вважають різними. Не використовують символи: $ ?

## Синтаксис мови Python

**Синтаксис мови програмування** — це набір правил, які визначають, як правильно структурувати програмний код у цій мові. Синтаксис визначає, які символи і конструкції можна використовувати в мові, як їх правильно розташовувати та комбінувати, щоб вони були прийнятні для інтерпретатора чи компілятора.

Коректний синтаксис дозволяє інтерпретатору чи компілятору розуміти і виконувати програмний код. Неправильно написаний код, який не відповідає синтаксису, може призвести до помилок під час виконання або компіляції.

Для зручності читання та розуміння програмного коду його складові мають свої кольори:

- фіолетовий — команди **Python (print, input, ...)**;
- помаранчевий — службові слова **(if, for, while, ...)**;
- зелений — рядки в лапках;
- чорний — інший текст;
- червоний — коментарі та помилки у вікні **IDLE**.

```
print("Синтаксис")
if (7 > 5):
    print("Перше число більше")
```

## Коментарі

У мові програмування **Python** коментарі використовують для додавання пояснень до коду. Коментарі не впливають на виконання програми та ігноруються інтерпретатором **Python**.

Однорядковий коментар починається з символа «решітка» **#**, тому все, що стоїть після **#** в рядку, не буде виконуватися.

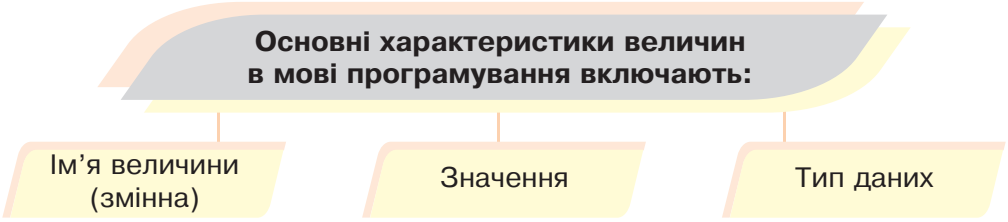
```
#Це однорядковий коментар
print("Додавання двох чисел")
a = 5
b = 8 #це також однорядковий коментар
s = a + b
print("Сума чисел:",s)
```

Для багаторядкового коментаря можна використовувати потрійні лапки (одинарні або подвійні) у поєднанні з апострофами. Такий коментар може розтягуватися на кілька рядків і часто використовується як рядок документації для опису функцій і модулів.

```
'''
Це багаторядковий коментар
або рядок документації,
який може бути використаний для опису функції або модуля
'''
print("Hello world!")
```

**Величина в Python** — це іменований контейнер для зберігання даних певного типу. Величина має унікальне ім'я, за яким можна отримати доступ до її значення, і може змінюватися (якщо це змінна) або бути незмінною (якщо це константа). Величини в **Python** використовують для зберігання, обробки та передачі даних в програмах.

Величиною в **Python** може бути:
число, літера, символ, рядок, список, словник і т.д.



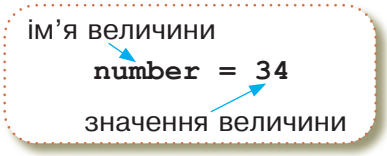

**Ім'я величини або змінна** — це об'єкт, який має ім'я і в якому зберігається певне значення або дані. Ім'я повинно відповідати правилам синтаксису мови програмування і часто повинно бути описовим, щоб полегшити зрозуміння призначення величини.

**Значення величини** — це конкретні дані, які вона містить. Значення може бути числом, рядком, списком, об'єктом, булевим значенням тощо, залежно від типу величини.

**Тип даних** визначає, які види даних можуть бути збережені в даній величині та як ці дані будуть оброблятися.

## Ім'я величини

У **Python** **символ присвоєння** — це оператор **=**
Він використовується для присвоєння значення змінній. Коли Ви пишете змінна **=** значення, Ви надаєте змінній ім'я і призначаєте їй значення.




Змінна величина може мати коротке ім'я (наприклад, **x** та **y**) або більш описове ім'я (наприклад — ціна, вартість, об'єм, площа і т. д.).

### Правила для надання імен змінних в Python:

- ім'я змінної має починатися з літери або символа підкреслення;
- ім'я змінної не може починатися з числа;
- ім'я змінної може містити лише літерно-цифрові символи та підкреслення (A-z, 0-9 та _);
- назви змінних чутливі до регістру (age, Age and AGE — це три різні змінні).

## Типи даних

**Python** підтримує різні типи даних, які використовуються для представлення різних видів інформації.

### Основні типи даних в Python включають:

```
ціле_число = 42
```

**Цілі числа (int)**
Використовуються для представлення цілих чисел, які можуть бути додатними, від'ємними або нулем.

```
правда = True
неправда = False
```

**Булеві(логічні) значення (bool)**
Використовуються для вираження істинності або хибності. Може бути два можливі значення: **True** або **False**.

```
дійсне_число = 3.1415926
```

**Дійсні числа (float)**
Використовуються для представлення чисел з рухомою комою.

```
рядок = 'Привіт, світ!'
```

**Рядки (str)**
Використовуються для представлення текстової інформації. Рядки можуть бути в одинарних або подвійних лапках.

## Надання та зміна типів даних

```
int(5)=1          int(3.5)=3          int('10')=10
float(5)=5.0      float(3.5)=3.5      float('10')=10.0
str(5)='5'        str(3.5)='3.5'      str('10')='10'
```

Таким чином можна змінювати типи даних: ціле число подавати у вигляді дійсного або рядка; дійсне число подати у вигляді цілого або перетворити в символи; рядок (який складається з цифр) подати у вигляді числа.

### Значення величини

**Значення величини (або змінної)** — це конкретні дані, які зберігаються в цій величині. Значення визначають тип даних величини і представляють інформацію, яка пов'язана з цією величиною. Значення може бути числовим, рядковим, списком, об'єктом, булевим значенням або іншим типом даних, відповідно до визначення величини.

Наприклад, у мові **Python** величина **x** зі значенням **5** буде мати тип **int** і містити ціле число **5**.

```
x = 5 #Змінна x має значення 5, тип int
```

У величини також може бути значення рядка.

```
ім_я = 'Іван' #Змінна ім_я має значення 'Іван', тип str
```

## Запис змінних

Надавати значення змінним можна по-різному:

- окремо по одній змінній (**number**, **name**);
- в одному рядку задати значення кількох змінних (**a**, **b**, **car**);
- якщо кілька змінних мають однакові значення (**x**, **y**).

```
number = 6
name = 'Inna'
a, b, car = 5, 4.7, 'Mercedes'
x = y = 5
```

## Команда виводу Print ()

Розглянемо і вивчимо першу команду, яка дозволяє виводити інформацію в консоль, тобто на екран для користувача.

Такою командою є функція **print()**.

```
print()
```

Ця мова програмування була розроблена для зручності читання і має схожість з англійською мовою, тому команди мають свій логічний переклад.

*Print* (з англ.) — друкувати, тобто «друкувати» (виводити) інформацію на екран.

Команда **print** реалізується так: спочатку пишемо саму команду print, а в дужках те, що хочемо вивести: число, текст чи іншу конструкцію даних.

```
print(7)
print("Я вивчаю Python")
print(13-5)
```

У нашому випадку в консоль буде виведено таку інформацію:

```
7
Я вивчаю Python
8
```

Розглянемо різні варіанти виводу через команду **print**.

- Щоб вивести текстову інформацію, використовують одинарні або подвійні лапки.

```
print('Я вивчаю Python')
print("Я вивчаю Python")
```

- Якщо потрібно вивести однією командою кілька рядків тексту — використовують потрійні лапки.

```
Любіть Україну у сні й наяву,
    вишневу свою Україну.
Красу її, вічно живу і нову,
    і мову її солов'їну.''')
```

Також можна виводити і числа та операції над ними:

```
print(11)
print(12 + 5)
print(8.5 * 2)
```

```
11
17
17.0
```

Можна виводити однією функцією кілька різних значень, розділяючи їх комою.

```
print('текст', 45, '5')
print('5 + 7 =',5 + 7)
```

```
текст 45 5
5 + 7 = 12
```

До речі, якщо число записати в лапках — комп'ютер його буде розуміти як текст, а не число.

Можна виводити значення змінних.

```
a = 4
b = 7
print(a,'+',b,'=',a+b)
```

```
4 + 7 = 11
```

Як могли помітити, роздільником між об'єктами, за замовчуванням, є пробіл.

```
print(1, 2, 3)
print(1, 2, 3, sep=':')
```

```
1 2 3
1:2:3
```

**sep=''** — роздільник, який потрібно ставити між об'єктами, що виводяться. В лапках встановлюємо символ, який буде роздільником.

**end='\n'** — символ, що ставиться в кінці рядка (за замовчуванням – символ кінця рядка).

Наприклад:

```
a = 5
b = 7
c = 10
print(a, b, c,
sep=',', end='.')
Результат: 5,7,10.
```

```
print('комп'ютер)
print('комп\'ютер')
```

Якщо потрібно вивести текст, у якому використовуються спеціальні символи (і щоб спеціальні символи були звичайними) — використовують **Escape-послідовності**. Ось деякі з них:

| | |
|---|---|
| **\'** — одинарна лапка | **** — зворотний слеш |
| **\''** — подвійна лапка | **\n** — новий рядок |
| **\?** — знак питання | **\t** — горизонтальна табуляція |

При виведенні кількох значень, розділених комою, у вікні виведення між значеннями стоїть пробіл. Щоб позбутися їх, можна використати **sep=''** або використати знак **+**.

```
name = 'Olga'
print('Привіт, ',name,'!')
print('Привіт, ' + name + '!')
```

```
Привіт, Olga!
Привіт, Olga!
```

## Практична робота №17

**1.** Напишіть програму, яка виводить на екран Ваше ім'я, захоплення та назву улюбленої книги.

Порядок виконання задачі:

- запустити **IDLE Python**;
- натиснути **File — New File**;
- у відкритому редакторі ввести код програми, наприклад:

```
print('Мене звати Христина')
print('Мої захоплення: малювання, гра на скрипці')
print('Моя найкраща книга "Чарівна крамниця"')
```

- зберегти код програми у своїй папці, натиснувши **File — Save as...** та надати ім'я файлу **about me**;
- запустити код на виконання, натиснувши **Run — Run Module**;
- закрити вікно з кодом;
- відкрити свою програму знову, використовуючи команду **File — Open**;
- запустити повторно код на виконання;
- показати вчителеві/вчительці результат.

**2.** Напишіть програму, яка виведе на екран визначення таких понять: алгоритм, програмування.

**3.** Напишіть програму, яка виведе на екран вірш про Україну.

*Рекомендації:*

- у коді використати лише один раз команду **`print`**.
- додати коментар, у якому буде написано автора та назву вірша.

**4.** Знайдіть периметр трикутника зі сторонами **`a = 5`**, **`b = 7`**, **`c = 6`**.

Порядок виконання задачі:

- у відкритому редакторі ввести код програми:

```
a = 5
b = 7
c = 6
P = a + b + c
print('P =', P)
```

- запустити команду на виконання, перед цим зберігши її у своїй папці;
- замінити значення сторін трикутника іншими числами (використати десятковий дріб);
- показати вчительці/вчителеві результат.

**5.** Напишіть програму, яка визначає сумарну вартість покупки, знаючи назви та ціну товарів.

*Рекомендації:*

- у програмі має бути не менше трьох товарів;
- імена змінних повинні відповідати назвам товарів;
- числові значення змінних повинні бути як цілими числами (лише гривні), так і дробовими (гривні та копійки).

**6.** Дано значення трьох змінних: ім'я, вік та улюблений предмет. Виведіть дані цих значень у такому форматі, використовуючи команди **sep** та **end**.

```
Христина;12;інформатика.
```

## Домашнє завдання

**1.** Написати програму, яка виводить на екран перелік ключових слів, з якими Ви ознайомилися під час цієї теми. Додати коментар «Список ключових слів мови **Python**».

**2.** Використовуючи змінні **name** та **age**, вивести на екран інформацію такого типу:

*Привіт! Мене звати (name). Мені (age) років. Я вивчаю мову* ***Python***.



Тема 18

# Основні команди мови програмування. Робота зі змінними

Ключові поняття

**Команда Input**
**Арифметичні оператори**
**Правила обчислення виразів**
**Модулі в Python**
**Модуль Math**
**Модуль Random**

## Команда Input

### Команда введення інформації

Ви вже ознайомилися з командою **print**, яка дозволяє виводити інформацію на екран для користувача.

Розглянемо функцію, яка дозволяє, навпаки, вводити інформацію до пам'яті комп'ютера, тобто з клавіатури.

Це можна зробити за допомогою команди **input()**.

```
a = input()
```

Ця функція зчитує рядок, введений користувачем з клавіатури, і перетворює його як рядковий об'єкт (тип даних **str**).

У дужках функції, в лапках можна вказати підказку для користувача, що саме потрібно вводити:

```
name = input('введіть своє ім\'я')
```

```
name = input('Введіть своє ім\'я: ')
print('Привіт, ' + name + '!')
```

У цьому прикладі програма запитує користувача про його ім'я, зчитує введений текст і виводить повідомлення «`Привіт, [ім'я]!`», де `[ім'я]` — це введене користувачем ім'я.

Після виклику **input()** програма буде очікувати введення від користувача. Користувач повинен ввести текст та натиснути клавішу **Enter**. Введений текст буде збережений у змінній **name**, і його можна використовувати для подальшої обробки.

Зверніть увагу, що **input()** повертає дані у вигляді рядка. Якщо Вам потрібно використовувати введені дані як число, то слід сконвертувати їх у відповідний тип даних за допомогою функцій, таких як **int()** або **float()**.

```
name = input('Введіть своє ім\'я: ')
age = int(input('Введіть свій вік'))
number = float(input('Введіть десятковий дріб'))
print('Текст:', name)
print('Ціле число:', age)
print('Дійсне число:', number)
```

```
Введіть своє ім'я: Inna
Введіть свій вік: 25
Введіть десятковий дріб: 4.5
Текст: Inna
Ціле число: 25
Дійсне число: 4.5
```

Зверніть увагу, що десятковий дріб розділяється не комою, як в математиці, а крапкою.

## Способи запиту та введення даних

1

Наприклад, щоб запитати на введення двох цілих чисел, можна окремо записати команди в два рядки.

```
a = int(input())
b = int(input())
```

```
5
7
>>>
```

У цьому випадку вводимо перше число, натискаємо **Enter**, вводимо друге число, натискаємо **Enter**.

2

Також для введення кількох значень одного типу можна використати функцію **map**.

```
a, b = map(int, input('Введіть два числа через пробіл').split())
```

```
Введіть два числа через пробіл 5 7
```

У даному випадку потрібно ввести дані через пробіл і натиснути **Enter**. В обох випадках комп'ютер отримає значення даних та зможе виконувати операції над ними.

У цьому прикладі:

- функція **`input`** використовується для отримання рядка, який вводить користувач;
- метод **`split()`** розділяє введений рядок на окремі слова (за пробілами);
- функція **`map(int, ...)`** застосовує функцію **`int`** до кожного елемента списку рядків, перетворюючи їх на цілі числа.

(Більш детально з методом **`split`** Ви ознайомитеся в старших класах).

## Арифметичні оператори

У програмуванні дуже часто потрібно виконувати різні математичні дії над змінними.

| Оператор | Приклад | Опис |
|---|---|---|
| `+` | 7 + 4 = 11 | **Додавання(+):** використовується для додавання двох чисел або об'єднання рядків (конкатенація). |
| `-` | 7 − 4 = 3 | **Віднімання(-):** використовується для віднімання одного числа від іншого. |
| `*` | 7 * 4 = 28 | **Множення(*):** використовується для множення двох чисел або для повторення рядка задану кількість разів. |
| `/` | 7 / 4 = 1.75 | **Ділення(/):** використовується для ділення одного числа на інше. |
| `//` | 7 // 4 = 1 | **Цілочисельне ділення(//):** повертає цілу частину від ділення одного числа на інше, відкидаючи дробову частину. |
| `%` | 7 % 4 = 3 | **Ділення залишку(%):** повертає залишок від ділення одного числа на інше. |
| `**` | 2 ** 3 = 8 | **Піднесення до степеня(**):** використовується для піднесення числа до певного степеня. |

## Математичні функції

Для обчислення в **Python** модуля числа використовується функція **abs**

```
a = -10
b = abs(a) #10
```

Для округлення числа в **Python** використовується функція **round**

```
x = 5.4
res = round(x) #5
```

```
x = 5.6789
res = round(x, 2) #5.68
```

### Правила обчислення виразів

1. Спершу виконуються операції в дужках.
2. Потім піднесення до степеня (за допомогою **).
3. Потім множення та ділення (зліва направо).
4. Нарешті, додавання та віднімання (зліва направо).

## Запис скорочених операцій

Розробники мови **Python** прагнули зробити програмний код більш лаконічним і компактним. Ще одним способом скоротити запис виразів з арифметичними операціями є скорочена форма. Використовується вона тоді, коли значення даної змінної змінюється значенням іншої змінної або константою (переприсвоєння).

| Стандартний запис | Скорочений запис |
|---|---|
| `x = x + 3` | `x += 3` |
| `x = x - 3` | `x -= 3` |
| `x = x * 3` | `x *= 3` |
| `x = x / 3` | `x /= 3` |
| `x = x // 3` | `x //= 3` |
| `x = x % 3` | `x %= 3` |
| `x = x ** 3` | `x **= 3` |

Пропуск між операцією і оператором присвоєння не ставиться

## Модулі в Python

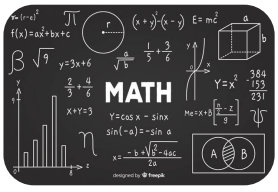
Окрім стандартної бібліотеки з досить широким набором функцій, мова **Python** містить велику кількість додаткових бібліотек, які можуть бути використані при написанні програм.

У мові програмування **Python** **модулі** — це файли, які містять **Python-код** (змінні, функції тощо).
Модулі використовуються для організації коду в більшу програму і для забезпечення впорядкованості та легкого управління функціями, класами та змінними.

Для роботи з модулем та для використання його функцій в нашому коді — його необхідно імпортувати (підключити). Підключення модуля виконується на самому початку програми командою **import**, далі через пропуск вказується ім'я модуля.

```
import name_module
```

### Модулі, з якими Ви ознайомитеся в курсі 7 класу




**import math**


**import random**

**import turtle**

## Модуль Math

Бібліотека **(модуль) Math** — одна з бібліотек, яка містить математичні функції і призначена для роботи з числовими даними.

Для того, щоб використовувати функцію з модуля, який імпортували, то у коді слід вказати ім'я модуля і після крапки обрати функцію, яку потрібно використати. На прикладі ми застосовуємо функцію **pow**, яка підносить число до степеня.

```
import math
x = math.pow(2,3)
```

```
from math import pow
x = pow(2,3)
```

```
from math import *
x = pow(2,3)
```

Записом **import math** — ми імпортуємо всю бібліотеку та всі функції, які в ній містяться. Якщо потрібно використати в коді лише одну функцію чи невелику кількість — можна імпортувати так: **from math import pow**. І тоді в самому коді перед викликанням функції не потрібно писати ім'я модуля.

Якщо після **import** поставити ***** — це означає, що ми імпортуємо всі функції з модуля, і тоді також у коді не потрібно ставити ім'я модуля перед викликанням потрібної функції.

## Основні функції модуля Math

| Функція | Опис |
|---|---|
| **pi** | Число π |
| **pow(a, b)** | Піднесення числа **a** до степеня **b** |
| **fabs(a)** | Модуль числа **a** |
| **sqrt(a)** | Корінь з числа **a** |
| **factorial(a)** | Факторіал числа **a** |
| **ceil(a)** | Заокруглення числа **a** |
| **round(a, k)** | Заокруглення числа **a** з **k** знаками після коми |

```
import math
print(math.pi)
print(math.pow(2,3))
print(abs(-5))
print(math.sqrt(16))
print(math.factorial(4))
print(math.ceil(3.7))
```

```
3.141592653589793
8.0
5
4.0
24
4
```

Щоб знайти модуль числа **a**, можна використовувати функцію **abs(a)**, для якої не потрібно підключати бібліотеку **Math**.

З функціями **sqrt** та **factorial** Ви ознайомитеся детальніше на уроках математики в старших класах.

**Корінь з числа 16** — знайти таке невід'ємне число, яке при піднесенні до другого степеня дасть результат 16 (таким числом буде 4).

Факторіал з числа **4** — це добуток всіх послідовних натуральних чисел від **1** до **4**. **4! = 1*2*3*4**.

## Модуль Random

**Модуль Random** — у мові програмування **Python** використовується для генерації випадкових чисел і випадкового вибору об'єктів.

Цей модуль дозволяє створювати випадкові дані, що може бути корисним в багатьох програмах, таких як ігри, симуляції, генерація випробувальних даних і багато інших. Ось деякі основні функції та методи, доступні у модулі **random**.

### Основні функції модуля Random

| Функція | Опис |
|---|---|
| **random()** | генерує випадкове число від 0 до 1 |
| **randint(a,b)** | генерує випадкове число від **a** до **b** |
| **randrange(x,b,x)** | генерує випадкове число від **a** (включаючи) до **b** (не включаючи) з кроком **x** |

```
import random

a = random.random()
print('Випадкове число від 0 до 1: ', a)
b = random.randint(10, 99)
print('Випадкове двохцифрове число: ', b)
c = random.randrange(-50, -30, 5)
print('Випадкове число з кроком: ', c)
```

```
Випадкове число від 0 до 1:  0.5500748436559528
Випадкове двохцифрове число:  91
Випадкове число з кроком:  -45
```

## Практична робота №18

**1.** Напишіть програму, яка виведе результат числових виразів:

а) $4,2\cdot(-7)-9,3:(5,8-8,9)$

б) $-55+12\cdot3+(-4)\cdot5:22$

в) $(-3,15)\cdot5,15+3,12:(-18)$

г) 25% від числа $\left(18^2-15^2\right)\cdot|2,5-3,1|$

Порядок виконання задачі:

- у відкритому редакторі ввести код програми:

```
a = 4.2*(-7)-9.3/(5,8-8,9)
print('Значення виразу: ', a)
```

- запустити програму на виконання і перевірити результат:

```
Значення виразу: -26.400000000000002
```

- повернутися до коду програми та відредагувати код, округливши результат до десятих (один знак після коми):

```
a = 4.2*(-7)-9.3/(5,8-8,9)
print('Значення виразу: ', round(a, 1))
```

- запустити програму на виконання;
- показати вчительці/вчителеві результат.

**2.** Напишіть програму, яка шукатиме значення виразу при введенні дійсного значення x із клавіатури:

а) $x^2-24x-56$

б) $\frac{3x-2}{4}$

в) $\frac{x+1}{x-2}+\frac{2x-1}{3}$

г) $\frac{x^2-5x+6}{4|x-3|}$

д) $(x+1)^3+\frac{x-7}{5-|x|}$

Порядок виконання задачі:

- у відкритому редакторі ввести код програми:

```
x = float(input('x = '))
result = (x + 1)**3 + (x - 7)/(5 - abs(x))
print('Значення виразу: ', round(result, 2))
```

- запустити програму на виконання та звіритися з результатом, якщо **x =-1,5**:

```
x = -1.5
Значення виразу: -2.55
```

- знайти значення виразу при іншому довільному значенні **x**;
- показати вчительці/вчителеві результат.

**3.** Знайдіть суму цифр трицифрового числа, введеного з клавіатури.

Порядок виконання задачі:

- у відкритому редакторі ввести код програми:

```
n = int (input('Введіть трицифрове число: '))
a = т // 100
b = (n // 10) % 10
c = n % 10
print(a, b, c, sep=',')
sum = a + b + c
print('Сума цифр: ', sum)
```

- запустити тричі програму на виконання та звіритися, чи програма правильно шукає суму цифр;
- показати вчительці/вчителеві результат.

**4.** Школа має в розпорядженні **p** грн. Скільки комп'ютерних мишок вартістю **n** грн можна закупити для школи? Виведіть залишок грошей.

*Примітка:* кількість комп'ютерних мишок — ціле число.

**5.** Напишіть програму, яка шукатиме площу круга радіусом *r*.

*Примітка:* функція **pi** імпортується з модуля **math**.

**6.** У класі **m** хлопців і **n** дівчат (**m** і **n** — рандомні числа від 5 до 15). Напишіть програму, яка знайде кількість учнів в класі та виведе, в якому відсотковому відношенні поділяються хлопці та дівчата в цьому класі.

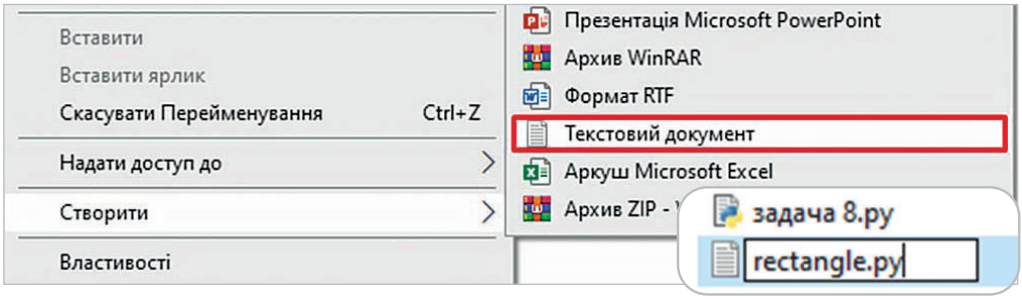
***Примітка:***

- використати функцію **randint** з модуля **random**;
- для знаходження відсоткового відношення використати математичну формулу **ab100%**.

**7.** Використовуючи дані із зовнішнього модуля, знайдіть площу та периметр прямокутника.

Порядок виконання задачі:

- відкрити редактор коду та зберегти даний файл у своїй папці;
- у папці з кодом створити текстовий документ з назвою **rectangle** та з розширенням **py**;




- відкрити файл **rectangle.py** в IDLE або в блокноті та ввести в нього значення довжини та ширини прямокутника, зберегти зміни;
- написати код програми, який буде імпортувати дані з Вашого модуля та знайде площу та периметр прямокутника:

```
import rectangle

P = (rectangle.a + rectangle.b)*2
S = rectangle.a * rectangle.b
print('P = ', P)
print('S = ', S)
```

```
P =  26
S =  40
```

- запустити код на виконання та перевірити результат;
- у модулі змінити значення сторін прямокутника та заново запустити програму на виконання;
- показати вчительці/вчителеві результат.

## Домашнє завдання

1. Знайти суму покупки комп'ютерної миші, клавіатури та монітора, ввівши їхню вартість з клавіатури.
2. Використовуючи дані із зовнішнього модуля, розв'язати задачу: *Даринка вирішила зібрати молодшому братикові подарунок, купивши машинку, фломастери та дві шоколадки. Визначіть вартість покупки, якщо відомо ціну кожного товару.*

# Лінійний алгоритм. Черепашача графіка

Тема 19

**Ключові поняття**

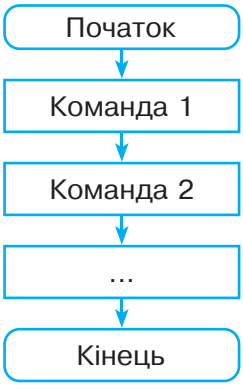
**Лінійний алгоритм**
**Графіка в Python**
**Модуль Turtle**
**Властивості вікна**
**Властивості Черепашки**
**Лінійний алгоритм в Turtle**

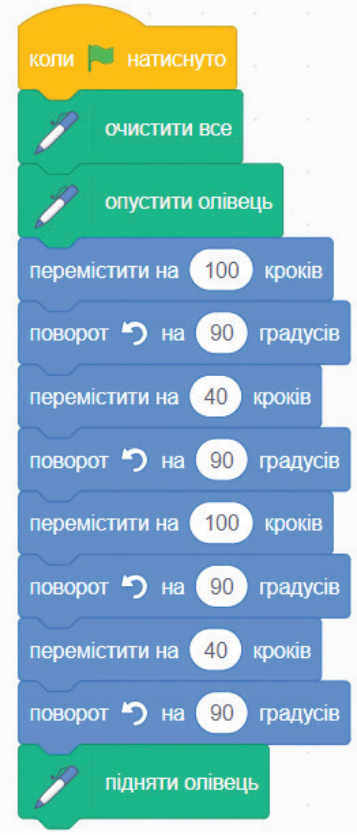
## Лінійний алгоритм

**Лінійний алгоритм** — це алгоритм, у якому всі дії виконуються послідовно одна за одною.

Вивчаючи мову програмування **Scratch** в молодших класах, Ви вже реалізовували лінійний алгоритм.

Використовуючи блок-схеми, лінійний алгоритм можна представити так:




Прикладом може бути, коли рудий кіт малює прямокутник зі сторонами 100 та 40.

Та сама задача, яка знаходить суму двох чисел, є прикладом лінійного алгоритму.

Реалізована мовою **Scratch:**


Реалізована мовою **Python**:

```
a = int(input('Введи перше число: '))
b = int(input('Введи друге число: '))
S = a + b
print('Сума двох чисел:', S)
```

Розглянемо детальніше лінійний алгоритм мовою **Python**:

```
#Крок 1: Отримання вхідних даних від користувача
name = input("Введіть своє ім'я: ")

#Крок 2: Виведення привітання
print("Привіт, " + name + "!")

#Крок 3: Обчислення суми чисел
num1 = 5
num2 = 10
summ = num1 + num2

#Крок 4: Виведення результату на екран
print("Сума чисел 5 і 10 дорівнює:", summ)

#Крок 5: Завершення програми
```

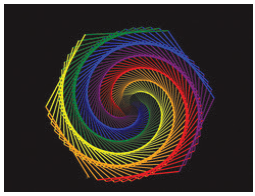
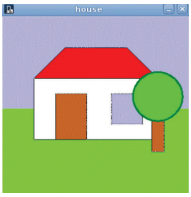
У цьому прикладі операції виконуються послідовно. Кожен крок виконується відразу після попереднього, і програма завершується.

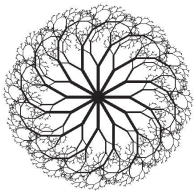
Лінійні алгоритми є простими для розуміння, і вони є важливою частиною будь-якої програми.

У разі необхідності виконання складних завдань лінійні алгоритми можуть поєднуватися з умовними виразами та циклами для створення більш складних програм.

## Графіка в Python

Графіка в **Python** може бути реалізована за допомогою різних бібліотек та інструментів. Основні бібліотеки для роботи з графікою в **Python** див. на форзаці.



## Модуль в Turtle

У 7 класі Ви ознайомитеся з однією з таких бібліотек — а саме з бібліотекою **Turtle**.

Модуль **Turtle** є чудовим інструментом для опанування світу програмування та малювання, і він може бути використаний для створення різноманітних цікавих малюнків та анімацій.

### Основні функції і можливості модуля Turtle

1. **Створення вікна для малювання**
   Модуль **Turtle** дозволяє створювати вікна для малювання, в яких буде відображатися рух Черепашки.
2. **Керування рухом Черепашки**
   Ви можете керувати рухом Черепашки, вказуючи їй рух уперед, назад, повороти і так далі.
3. **Малювання ліній та фігур**
   Черепашка може малювати лінії, криві, круги, трикутники та інші геометричні фігури.
4. **Зміна кольорів та стилів малювання**
   Ви можете налаштовувати колір ліній, заливки фігур, розмір пензлів і багато іншого.

5 **Використання циклів та умовних виразів**

Ви можете використовувати цикли та умовні вирази для створення складних малюнків та анімацій.

6 **Збереження малюнків**

Модуль **Turtle** дозволяє зберігати створені малюнки у файлі для подальшого використання.

## Початок роботи з Черепашкою

Для того, щоб використовувати функції модуля **Turtle**, спочатку потрібно його імпортувати.

```
import turtle

window = turtle.Screen()
```

```
from turtle import *

window = Screen()
```

У першому випадку використовуємо імпорт самої бібліотеки. При цьому в коді програми, щоб викликати функцію з даної бібліотеки, потрібно перед нею прописувати назву модуля **Turtle**.

У другому випадку імпортуємо одразу всі функції бібліотеки **Turtle**. При цьому в коді програми, щоб викликати функцію з даної бібліотеки, не потрібно перед нею прописувати назву модуля **Turtle**.

Властивості вікна

## Створення вікна

Щоб створити вікно для малювання, використовують функцію **Screen()**.

```
window = Screen()
```

Спочатку надаємо ім'я об'єкту «вікно» та після знака «дорівнює» викликаємо функцію **`Screen()`**.

***Увага!*** Ім'я цієї функції починається з великої літери.

Після запуску програми з'явиться вікно, в якому надалі будемо створювати різні зображення та анімації.


## Основні функції вікна в Turtle

| Функція | Опис |
|---|---|
| `setup(width,height)` | Налаштовує розмір вікна у ширину і висоту. |
| `bgcolor("color")` | Змінює колір фону вікна. |
| `title("Заголовок")` | Встановлює заголовок вікна. |
| `clear()` | Очищає вікно від малюнків. |
| `reset()` | Очищає вікно та повертає Черепашку в центр. |
| `home()` | Повертає Черепашку в початкове положення. |
| `bye()` | Закриває вікно. |

Щоб надати об'єкту значення властивості, потрібно написати таку комбінацію:

**назва_об'єкта.властивість(значення)**

Наприклад: **`window.title('name')`**

```
from turtle import *

window = Screen()
window.setup(600, 400)
window.bgcolor('yellow')
window.title('Моє перше графічне вікно')
```

Написавши даний код програми та запустивши його на виконання, отримаємо вікно жовтого кольору з розмірами 600 пікселів по довжині та 400 пікселів по висоті й назвою вікна «Моє перше графічне вікно».

Щоб створити Черепашку (графічний об'єкт), яка буде рухатися та малювати зображення, використовують функцію **`Turtle()`**. `t = Turtle()`

Спочатку надаємо ім'я нашій Черепашці й після знака **=** пишемо функцію **`Turtle()`**.

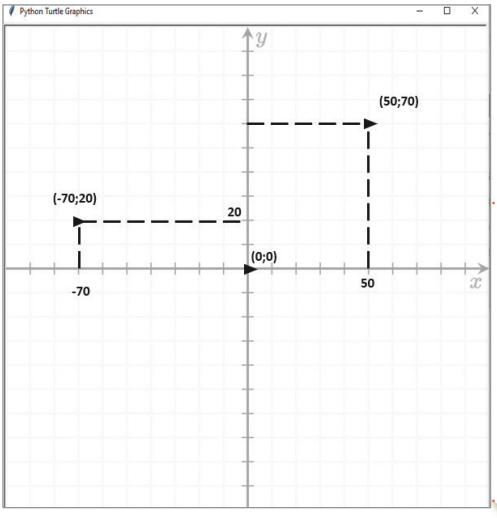
***Увага!*** Ім'я цієї функції пишеться з великої літери.

**Графічний об'єкт Черепашка** в **Python** — це абстракція, яка представляє собою віртуальну Черепашку, якою можна керувати для малювання на вікні. Черепашка може рухатися вгору, вниз, вліво та вправо, і вона залишає за собою слід, коли переміщується.

За замовчуванням, після запуску програми Черепашка має вигляд чорної стрілочки, яка показує напрям і розміщена в центрі вікна.

Положення Черепашки у вікні визначає координати на координатній площині, яка уявно розміщена у вікні.

Центр вікна та початкове положення Черепашки — це точка (**`0;0`**).



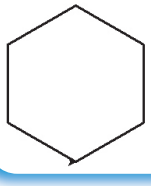

Якщо Черепашку перемістити вправо на 50 одиниць та на 70 одиниць вгору — вона зупиниться в точці **`(50;70)`**; аналогічно, якщо перемістити на 70 одиниць вліво та на 20 одиниць вгору — зупиниться в точці **`(-70;20)`**.

## Властивості Черепашки

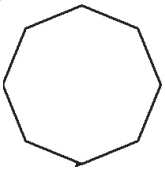
### Основні функції руху Черепашки

| Функція | Опис |
|---|---|
| `forward(n)` | Переміщує Черепашку вперед на **n** кроків. |
| `left(g)` | Повертає Черепашку вліво (проти годинникової стрілки) на **g** градусів. |
| `right(g)` | Повертає Черепашку вправо (за годинниковою стрілкою) на **g** градусів. |
| `seth(grad)` | Надає напрямок Черепашці на градус **grad** (від 0 до 360) (0 — вправо, 90 — вгору, 180 — вліво, 270 — вниз). |
| `goto(x, y)` | Переміщає Черепашку в точку (**x**, **y**). |
| `circle(r)` | Будує коло з радіусом **r**. |
| `circle(r, g)` | Будує дугу кола з радіусом **r** та градусною мірою **g**. |
| `dot(d)` | Будує точку з діаметром **d**. |

Функція **circle** може мати і третій параметр — кількість сегментів, на які потрібно поділити коло.



Функція `circle (100, 360, 6)` малює коло з радіусом 100 одиниць, на кут 360 градусів (повне коло) і розділяє його на 6 сегментів, створюючи таким чином вигляд «шестикутника».

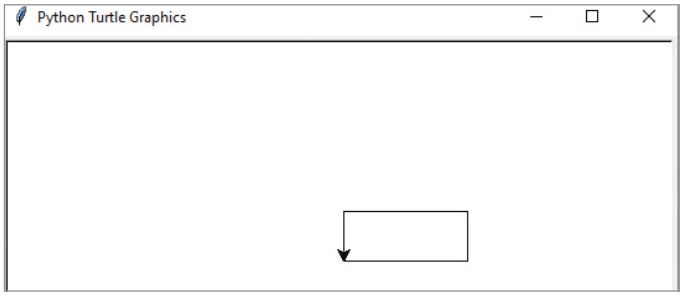


Функція `circle (100, 360, 8)` малює коло з радіусом 100 одиниць, на кут 360 градусів (повне коло) і розділяє його на 8 сегментів, створюючи таким чином вигляд «восьмикутника».

Функція `circle (100, 180, 5)` малює половину кола з радіусом 100 одиниць і розділяє цю половину кола на 5 сегментів, створюючи таким чином вигляд «п'ятикутника», який є половиною кола.

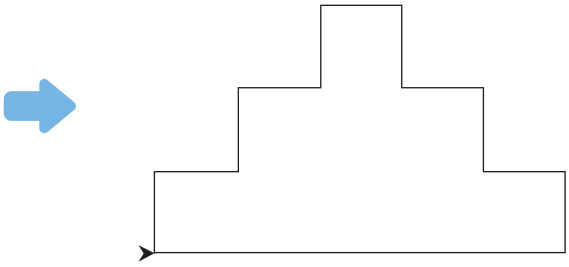
```
from turtle import *

# Створення нової черепашки і надання їй імені "t"
t = Turtle()

# Рух черепашки
t.forward(100) # Рух вперед на 100 пікселів
t.left(90)     # Поворот на 90 градусів
t.forward(40)  # Рух вперед на 40 пікселів
t.left(90)     # Поворот на 90 градусів
t.forward(100) # Рух вперед на 100 пікселів
t.left(90)     # Поворот на 90 градусів
t.forward(40)  # Рух вперед на 40 пікселів
```

У цьому прикладі **Turtle()** створює нову Черепашку і призначає їй ім'я **"t"**. Потім Черепашка виконує послідовні операції руху: вона рухається вперед на 100 пікселів, повертає на 90 градусів ліворуч, рухається вперед на 40 пікселів, знову повертає на 90 градусів і так далі. В кінці утворився прямокутник зі сторонами 100 і 40 пікселів.




```
from turtle import *

t1 = Turtle()

t1.goto(200, 0)
t1.goto(200, 40)
t1.goto(160, 40)
t1.goto(160, 80)
t1.goto(120, 80)
t1.goto(120, 120)
t1.goto(80, 120)
t1.goto(80, 80)
t1.goto(40, 80)
t1.goto(40, 40)
t1.goto(0, 40)
t1.goto(0, 0)
```

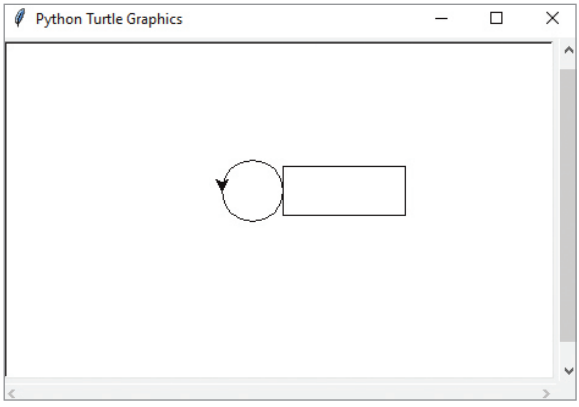
Також можна будувати лінії за допомогою переміщення в довільне місце площини, за даними координатами.



Для побудови комбінацій різних фігур часом потрібно, щоб Черепашка при русі не залишала за собою сліду. Тому є відповідні функції.

| Функція | Опис |
|---|---|
| **down()** | Опускає перо, тобто включає функцію малювання, щоб Черепашка залишала слід при русі. |
| **up()** | Піднімає перо, тобто відключає функцію малювання, щоб Черепашка не залишала сліду при русі. |
| **speed(n)** | Змінює швидкість руху Черепашки (від 1 до 10), якщо **n=0** — миттєве створення малюнка. |

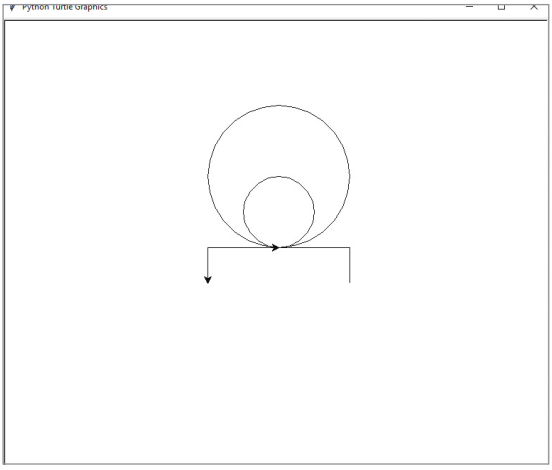
```
from turtle import *
t = Turtle()
t.forward(100)
t.left(90)
t.forward(40)
t.left(90)
t.forward(100)
t.left(90)
t.forward(4)

t.up()
t.goto(-50, 20)
t.down()

t.circle(25)
```




У цьому прикладі **Turtle()** створює нову Черепашку і призначає їй ім'я **"t"**. Потім Черепашка виконує послідовні операції руху: утворює прямокутник; потім піднімає перо, переміщається в інше місце, опускає перо і будує коло.

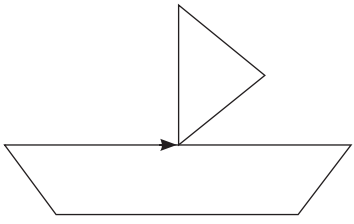
Функція **Turtle()** може бути використана для створення багатьох Черепашок з різними іменами та параметрами, які можуть малювати на вікні незалежно одна від одної або разом, щоб створити складні малюнки та анімації.

```
from turtle import *

t1 = Turtle()
t2 = Turtle()

t1.circle(100)

t2.forward(100)
t2.right(90)
t2.forward(50)

t2.up()
t2.home()
t2.down()
t2.seth(180)

t2.forward(100)
t2.left(90)
t2.forward(50)

t1.circle(50)
```



## Домашнє завдання

1. На одну шальку терезів всадовили Вінні-Пуха, маса якого **m** кг, і П'ятачка, який на 20 кг легший, а на іншу шальку поставили горнятко з медом, маса якого **n** кг (що набагато більше за масу обох друзів). Напишіть програму, яка визначить, скільки кілограмів меду потрібно з'їсти героям, щоб шальки терезів зрівноважилися.
2. Напишіть програму, яка створить зображення:



## Практична робота №19

**1.** Напишіть програму, яка буде отримувати від користувача два цілі числа: кількість учнів **n** та кількість яблук **k**. Учні ділять між собою яблука порівну, залишок залишається в кошику. Виведіть на екран, скільки яблук дістанеться кожному учню/учениці, а скільки залишиться в кошику.

***Примітка:***

- для отримання даних від користувача використовуйте функцію `input()`;
- для знаходження цілочисельного ділення та залишку використовуйте оператори `//` та `%`.

**2.** Напишіть програму для обчислення тривалості поїздки, якщо вона складається із трьох відрізків по **s1**, **s2**, **s3** кілометрів і на кожному з них автомобіль рухався зі швидкістю **v1**, **v2**, **v3** відповідно. Між відрізками дороги автомобіль зупинявся один раз на **t** хвилин.

***Примітка:***

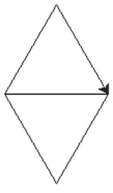
- вся тривалість — це сума часу на кожному відрізку, додати час зупинки між кожним відрізком;
- щоб знайти час на одному відрізку, використовують математичну формулу $t = \frac{s}{v}$.

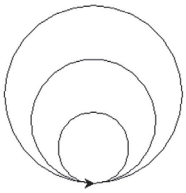
**3.** Напишіть програму, яка створює вікно синього кольору з розмірами 600×350 пікселів та замініть ім'я заголовка на Ваше прізвище та ім'я.

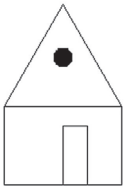
**4.** Використовуючи функцію **goto(x, y)**, напишіть програму, яка намалює малюнок за координатами:
(20, 40), (20, –40), (40, –40), (40, 80), (60, 80), (60, 60), (140, 60), (160, 0), (160, –60), (140, –60), (140, –80), (100, –80), (100, –60), (–20, –60), (–20, –80), (–60, –80), (–60, –60), (–80, –60), (–80, –40), (–100, –40), (–120, –20), (–140, 20), (–160, 40), (20, 40).

**5.** Напишіть програму, яка створить зображення:

а) 

б) 

в) 

г) 

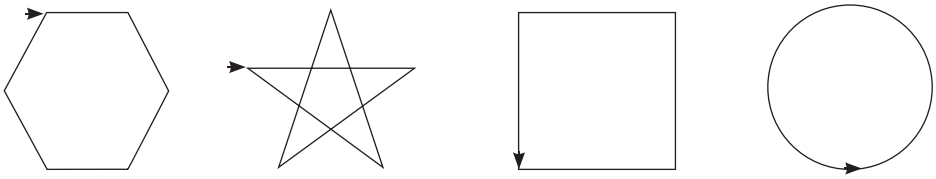
**Тема 20** **Алгоритм створення зображень за допомогою графічного модуля Tutrle**

Ключові поняття

**Властивості ліній**
**Властивості замкненої фігури**
**Кольори в Turtle**
**Зміна властивостей Черепашки**
**Вставка тексту**
**Атрибути тексту**

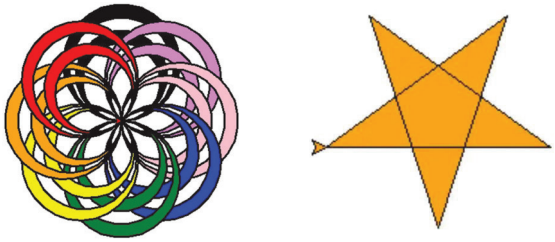
## Властивості ліній

Ви уже ознайомилися із основними функціями модуля **Turtle**, навчилися визначати координати Черепашки у вікні та створювати графічні примітиви.



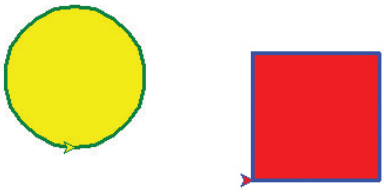
Проте всі зображення до цього в нас були чорного кольору.

Модуль **Turtle** надає можливість змінювати властивості об'єктів графічного вікна та працювати з кольоровими зображеннями (як Ви уже могли помітити при зміні кольору фону графічного вікна).



## Властивості лінії (сліду Черепашки)


### Властивості замкненої фігури



- Товщина контуру фігури
- Колір контуру фігури
- Колір заповнення фігури

Для зміни товщини (ширини) лінії при малюванні у модулі **Turtle** в **Python** Ви можете використовувати метод **`pensize(n)`** або **`width(n)`** у графічного об'єкта Черепашка.

```
from turtle import *
t1 = Turtle()
t1.width(3)
```

```
from turtle import *
t1 = Turtle()
t1.pensize(7)
```

У властивостях **`pensize(n)`** та **`width(n)`** — число **`n`** — довільне натуральне число (**`1`** — найтонша лінія, яка йде за замовчуванням).

Для зміни кольорів фігур і ліній у модулі **`Turtle`** в **Python** Ви можете використовувати такі властивості:

| Функція | Опис |
|---|---|
| **`pencolor(c)`** | Встановлює колір лінії, якою малює Черепашка. |
| **`color(c)`**<br>**`color(c1, c2)`** | Ця властивість дозволяє встановити одночасно кольори лінії та заповнення фігури. Ви можете передавати або один аргумент для кольору лінії, або два аргументи для кольору лінії та заповнення. |
| **`fillcolor(c)`** | Встановлює колір заповнення для замкнених фігур, таких як кола або прямокутники. |

Проте потрібно зауважити, що властивість заповнення сама по собі не працюватиме. Для заповнення закритої фігури в модулі **Turtle** в **Python** Ви можете використовувати методи **`begin_fill()`** та **`end_fill()`**.

| Функція | Опис |
|---|---|
| **`begin_fill()`** | Позначає початок заповнення фігури. |
| **`end_fill()`** | Завершує заповнення фігури. |

Функція **`begin_fill()`** позначає початок заповнення, і після цього малюємо закриту фігуру. Коли її намальовано, **`end_fill()`** завершує заповнення, і фігура буде заповнена кольором за допомогою властивостей **`fillcolor()`** або **`color()`** Черепашки.

```
from turtle import *

t1 = Turtle()
t1.width(3)
t1.color('red', 'yellow')
t1.circle(60)
```

Неправильний код для заливки

```
from turtle import *

t1 = Turtle()
t1.width(3)
t1.color('red', 'yellow')
t1.begin_fill()
t1.circle(60)
t1.end_fill()
```

Правильний код для заливки

Коли ми задаємо колір лінії та заливки, наша Черепашка набуває вигляду цих властивостей.

## Кольори в Turtle

Модуль **`Turtle`** в **Python** має підтримку базових кольорів, які можна використовувати для малювання. Ось деякі з цих кольорів:

| | | | |
|---|---|---|---|
| black | yellow | red | green |
| blue | white | orange | purple |
| pink | brown | gray | cyan |

Для того, щоб дізнатися назви інших кольорів — можна перейти за посиланням:

Кольори в **Turtle** також можна надавати у форматі **hex**, що дозволяє використовувати всі можливі відтінки кольорів:

```
from turtle import *

t1 = Turtle()
t1.width(3)
t1.color('red', '#CD26BC')
t1.begin_fill()
t1.circle(60)
t1.end_fill()
```


У цьому прикладі колір подано у вигляді 16-го коду #CD26BC, що визначає один із відтінків фіолетового кольору.

Щоб використовувати кольори у форматі **hex** — можна перейти на сайт за посиланням: →

У вікні-палітрі обираємо потрібний нам відтінок, копіюємо код кольору і використовуємо його в програмному коді.

## Зміна властивості Черепашки

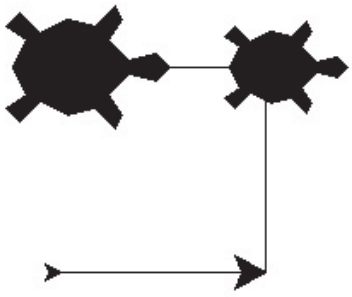
Зовнішній вигляд Черепашки також можна змінювати, а саме форму та розмір.

```
t1.shape('turtle')
t1.shape('circle')
t1.shape('square')
t1.shape('arrow')
t1.shape('arrow')
t1.shape('classic')
```

- Для того, щоб змінити форму Черепашки, використовують функцію **shape('name')**.
- Для того, щоб змінити розмір Черепашки, використовують функцію **shapesize(n)**.

**Метод `stamp()`** в модулі **`Turtle`** в **Python** використовується для того, щоб отримати відбиток поточного положення та вигляду Черепашки на малюнку. Це потрібно, якщо Ви хочете залишити сліди Черепашки на малюнку або створити певні візуальні ефекти.

```
t1 = Turtle()

t1.stamp()
t1.goto(120, 0)
t1.shapesize(2)
t1.stamp()
t1.goto(120, 120)
t1.shape('turtle')
t1.shapesize(3)
t1.stamp()
t1.goto(0, 120)
t1.shapesize(4)
t1.stamp()
```

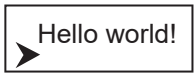


У цьому прикладі ми використовуємо **`t1.stamp()`** для залишення відбитка Черепашки на малюнку. Ви можете помітити, як Черепашка залишає відбиток в точці, де вона перебуває. Ви можете переміщати Черепашку та залишати відбитки на різних частинах малюнка для створення цікавих ефектів.

## Атрибути тексту

Для виведення тексту у модулі **`Turtle`** використовують метод **`write()`**.

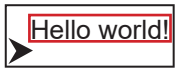
```
from turtle import *
t1 = Turtle()
t1.write('Hello world!')
```




Ось як він працює:

- Вказуєте Черепашці, де розташувати текст на малюнку. Це визначає позицію, в яку буде виведено текст.
- Викликаєте метод **`write()`** для Черепашки, передаючи йому текст, який Ви хочете вивести.

При цьому Черепашка створює прямокутну область, у якій розміщує сам текст (лівий нижній кут прямокутника — це початкове положення Черепашки).




Ви можете налаштовувати розмір та стиль шрифту, вказавши аргумент **font** в методі **`write()`**.

```
from turtle import *
t1 = Turtle()
t1.color('blue', 'red')
t1.write('Hello world!', font = ('Cambria', 36, 'bold'))
```


У цьому прикладі шрифт встановлено на «cambria» розміром 36 зі стилем «bold», але Ви можете змінити ці параметри на власний вибір.

Обережно встановлюйте позицію тексту, щоб впевнитися, що текст виводиться в потрібному місці на малюнку.

Окрім **`font`**, метод **`write()`** має ряд інших атрибутів (параметрів), які дозволяють налаштовувати відображення тексту на малюнку. Ось список основних атрибутів методу **`write()`**:

| Атрибут | Опис |
|---|---|
| **`text`** | Це обов'язковий атрибут, який визначає текст, який буде виведено на малюнку. |
| **`font=(шрифт, розмір, стиль)`** | Цей атрибут встановлює шрифт для тексту. Ви можете вказати рядок з іменем шрифту, розміром шрифту та стилем. Наприклад ("Arial",12, "normal"). |
| **`align`** | Визначає вирівнювання тексту. Можливі значення цього атрибута: "left"(ліворуч), "center" (по центру), "right" (праворуч). |
| **`move`** | Це логічний атрибут, який вказує, чи слід переміщати Черепашку після виведення тексту. Якщо True, то черепашка після виведення тексту переміщується вперед. |

```
t1.write('Hello, World!',
font = ('Arial', 36, 'normal'),
align='center', move=True)
```

## Практична робота №20

**1.** Напишіть програму, виконання якої відкриватиме вікно з розмірами 600x400. У цьому вікні намалюйте 4 фігури довільного розміру:

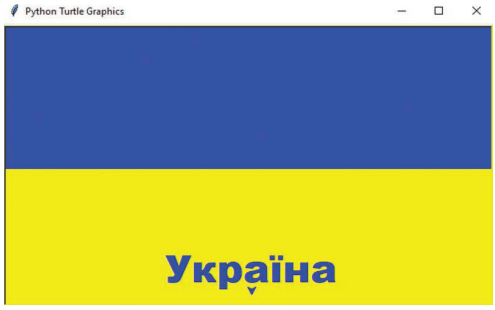
- помаранчевий квадрат з синьою рамкою, яка має товщину 3;
- жовтий трикутник з червоною рамкою, яка має товщину 1;
- прямокутник, який має заливку кольором “pink” та рамку сірого кольору;
- коло, яке має заливку за кодом #34C1AF та рамку чорного кольору.

**2.** Напишіть програму, яка створить прапор України з підписом «Україна».

Порядок виконання задачі:

- створити вікно з розміром 600x360 жовтого кольору;
- перемістити Черепашку у центр лівої межі;
- побудувати прямокутник блакитного кольору, довжина 600 та ширина 180 (або за допомогою команди goto);
- перемістити в нижній центр вікна, наприклад, в точку (0;-160);
- у цьому місці вивести текст «Україна», вирівнювання *center*. Шрифти, розмір та колір підібрати самостійно.

*Зразок:*




**3.** Намалюйте зображення засобами модуля **Turtle**:

Порядок виконання задачі:

- створити вікно чорного кольору;
- створити червону Черепашку з рожевою заливкою;
- повернути Черепашку вліво на 140 градусів;

- рухатися прямо на 220 кроків (це лінія від нижньої точки серця до початку частини кола);
- намалювати частину кола, використавши команду `circle(-110,200)` (`"-110"` означає коло радіусом 100, `"200"` — означає частину кола градусною мірою в 200 градусів, а 180 — то була б половина кола);
- надати напрямок Черепашки 60 градусів, використавши команду `seth(60)`;
- намалювати частину кола, використавши команду `circle(-110,200)`;
- рухатися прямо на 220 кроків (це лінія від частини кола до нижньої точки серця).

**4.** Намалюйте зображення засобами модуля `Turtle`:



## Домашнє завдання

**1.** Створіть графічну програму, в якій буде намальовано прапор будь-якої країни на вибір (пошук прапорів в інтернеті). Знизу по центру додати текст з назвою країни (Бельгія) або напис (Прапор Бельгії).

*Зразки:*



**2.** Намалюйте зображення засобами модуля **Turtle**:



**Тема 21** **Реалізація алгоритмів з розгалуженням. Умовний оператор if**

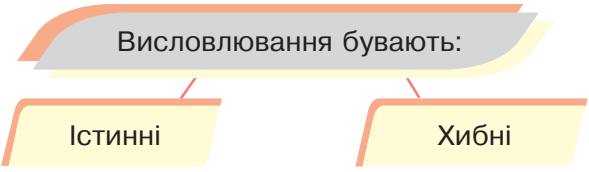
Ключові поняття

- **Висловлювання**
- **Логічний тип даних**
- **Оператори порівняння**
- **Логічні оператори**
- **Алгоритм розгалуження**
- **Оператори розгалуження**

## Висловлювання

### Пригадаймо

**Висловлювання** — це розповідне речення, яке містить ствердження або заперечення.




Речення, які не є висловлюваннями, можуть бути інформативними, але вони не містять умови, яку можна оцінити як істинну або хибну. Ось приклади:

— Поїдь гуляти в парк.
— Привіт!
— Скільки тобі років?

Приклади хибних висловлювань:

— Сума чисел 2 і 5 буде 8.
— Усі види птахів можуть літати.
— Київ — столиця Великобританії.

## Висловлювання з умовою (логічні)

**Висловлювання з умовою** містять умову, яка повинна бути оцінена як істинна або хибна, і залежно від цього виконується певна дія.

Якщо надворі падає дощ, то взяти парасольку.

Якщо кути вертикальні — то вони рівні.

Якщо користувач ввів правильний пароль, то відкрити доступ до системи.

Якщо на складі є товар, то відправити замовлення, інакше вказати термін очікування.

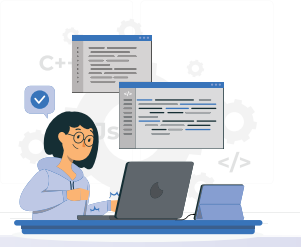
### Логічний тип даних

У програмуванні також часто потрібно знати, чи є якесь висловлювання істинним чи хибним, або ж слід перевірити умову, щоб виконувати певні дії.

Для цього використовують логічний тип даних **bool**, який може набувати двох значень: **True** та **False.**

Для роботи зі значеннями логічного типу призначені спеціальні оператори:

- оператори порівняння;
- логічні оператори.



### Оператори порівняння

**Оператори порівняння** — це оператори, які використовуються для порівняння двох значень або виразів і повертають логічне значення **True** або **False** залежно від результату порівняння.

| Оператор | Опис | Приклад |
|---|---|---|
| > | Більше (>): порівнює, чи перший операнд більший за другий. Якщо так, то вираз повертає **True**, в іншому випадку — **False**. | 10 > 5 # Поверне **True**<br>5 > 10 # Поверне **False** |
| < | Менше (<): порівнює, чи перший операнд менший за другий. Якщо так, то вираз повертає **True**, в іншому випадку — **False**. | 5 < 10 # Поверне **True**<br>10 < 5 # Поверне **False** |
| >= | Більше або рівне (>=): порівнює, чи перший операнд більший або рівний другому. Якщо так, то вираз повертає **True**, в іншому випадку — **False**. | 10 >= 5 # Поверне **True**<br>5 >= 5 # Поверне **True**<br>5 >= 10 # Поверне **False** |
| <= | Менше або рівне (<=): порівнює, чи перший операнд менший або рівний другому. Якщо так, то вираз повертає **True**, в іншому випадку — **False**. | 5 <= 10 # Поверне **True**<br>5 <= 5 # Поверне **True**<br>10 <= 5 # Поверне **False** |
| == | Рівність (==): порівнює, чи два операнди мають однакове значення. Якщо так, то вираз повертає **True**, в іншому випадку — **False**. | 5 == 5 # Поверне **True**<br>10 == 5 # Поверне **False** |
| != | Нерівність (!=): порівнює, чи два операнди мають різне значення. Якщо так, то вираз повертає **True**, в іншому випадку — **False**. | 5 != 10 # Поверне **True**<br>5 != 5 # Поверне **False** |

Оператори порівняння дозволяють Вам порівнювати значення у Вашому коді та приймати рішення на основі цих порівнянь. Вони часто використовуються в умовних виразах та циклах для керування логічним потоком програми.

## Логічні оператори

**Логічні оператори** — це символи чи ключові слова, які використовуються для об'єднання, порівняння чи змішування логічних висловлювань. Вони дозволяють Вам виражати складніші логічні умови в програмах.

| Оператор | Опис | Приклад |
|---|---|---|
| `and` | **Логічне "І" (and)**: повертає **True**, якщо обидва операнди є **True**. Якщо хоча б один з операндів є **False**, то вираз поверне **False**. | (5 < 10) and (7 > 3) #Поверне **True**<br>(5 < 10) and (7 < 3) #Поверне **False** |
| `or` | **Логічне "АБО" (or)**: повертає **True**, якщо хоча б один із операндів є **True**. Вираз повертає **False**, якщо обидва операнди є **False**. | (5 < 10) or (7 > 3) #Поверне **True**<br>(5 < 10) or (7 < 3) #Поверне **True**<br>(5 > 10) or (7 < 3) #Поверне **False** |
| `not` | **Логічне "НЕ" (not)**: змінює логічне значення на протилежне. Якщо операнд був **True**, то `not` зробить його **False**, і навпаки. | nnot (10 < 5) #Поверне **True**<br>not (5 < 10) #Поверне **False** |

## Алгоритм розгалуження

### Алгоритм розгалуження (алгоритм з умовою)

У реальному житті рішення приймаються на основі певних умов.

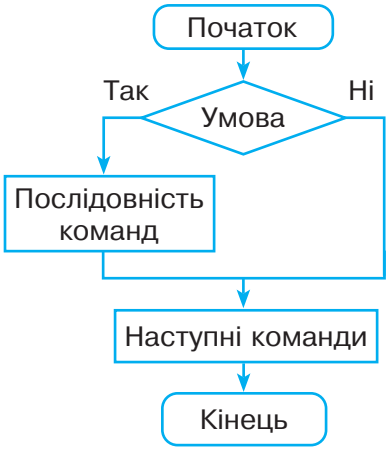
- Наприклад, ми можемо вирішити взяти парасолю, якщо йде дощ, або залишити її вдома, якщо немає дощу.
- Сенсори сучасних автомобілів реагують на автомобіль, який різко зупинився попереду, і натискають на гальма набагато швидше, ніж це змогли б зробити ми.

**Алгоритм розгалуження** — це послідовність інструкцій або операцій, які виконуються залежно від певної умови.

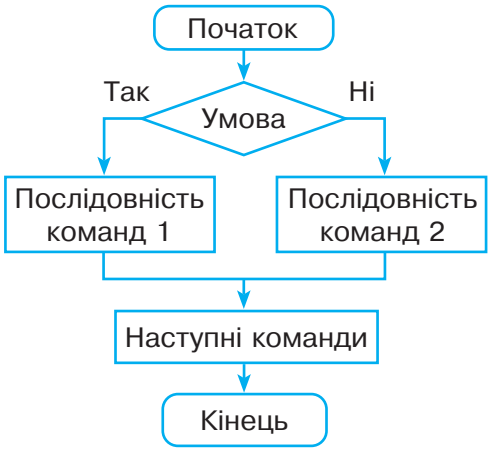
Ця умова може бути істинною або хибною, і відповідно до неї програма приймає рішення щодо виконання певних дій або операцій.

Розрізняють повне і неповне розгалуження.

**Виконання неповного розгалуження відбувається так:** виконавець виконує команду перевірки умови: якщо результат виконання цієї команди **Так**, то виконавець виконує послідовність **команд 1**, після чого переходить до виконання наступних команд програми; якщо результат виконання цієї команди **Ні**, то виконавець одразу переходить до виконання наступних команд програми.




*Неповне розгалуження*




*Повне розгалуження*

**Виконання повного розгалуження відбувається так:** виконавець виконує команду перевірки умови: якщо результат виконання цієї команди **Так**, то виконавець виконує послідовність **команд 1**, після чого переходить до виконання наступних команд програми; якщо результат виконання цієї команди **Ні**, то виконавець виконує послідовність **команд 2**, після чого переходить до виконання наступних команд програми.

Вивчаючи мову програмування **Scratch** в молодших класах, Ви також реалізовували алгоритм розгалуження.


*Неповне розгалуження*


*Повне розгалуження*


У цьому прикладі програма в нас запитувала суму двох чисел і перевіряла її: якщо ми відповіли правильно — то говорила **Молодець**, якщо ж відповіли неправильно — то говорила **Спробуй ще**.

## Оператори розгалуження

У **Python** розгалуження виконується за допомогою умовних операторів, таких як **`if`**, **`else`** і **`elif`**.
**Оператори розгалуження** дозволяють виконувати різні дії залежно від виконання певних умов.

### Оператор if (неповне розгалуження)

**Розглянемо реалізацію неповного розгалуження.**

Записується умова та команда або кілька команд, які виконуватимуться при істинності цієї умови. Умова записується після ключового слова **`if`** (якщо) за допомогою операторів порівняння та логічних операторів.

Обов'язково має бути відступ, так комп'ютер розуміє, що ці команди відносяться саме до цієї умови.

Якщо після умови є лише одна команда, то можна не переходити на наступний рядок:

**`if умова1:команда1`**
**`if умова2:команда2`**.

```
if умова1:
   команда

if умова2:
   команда1
   команда2
    ...
```

1. У цьому прикладі комп'ютер отримав дані про погоду. І перевіряє спочатку першу умову, яка хибна — і команду при цій умові не виконує. При перевірці другої умови, яка істинна — тому виконує команду при цій умові й виводить інформацію «Візьми парасольку».

2. Цей приклад порівнює два числа і виводить інформацію, яке з них більше, або якщо однакові — то виводить інформацію на екран, що вони рівні.

***Завдання:*** Введіть даний код, замініть числа **`a`** і **`b`** на свої та перевірте результат.

**1**

```
pogoda = 'дощ'

if pogoda == 'сонце':
   print('Гарної прогулянки!')
if pogoda == 'дощ':
   print('Візьми парасольку')
```

```
Візьми парасольку
```

**2**

```
a = 7
b = 13

if a > b:
  print('Перше число більше')
if a < b:
  print('Друге число більше')
if a == b:
   print('Числа рівні'
```

```
Друге число більше
```

**3.** Цей приклад отримує інформацію, в якому класі Ви навчаєтеся, і за допомогою складених умов, де використовуються логічні оператори **or** та **and**, перевіряє та виводить інформацію, учнем якої школи Ви є.

```
print('Учнем якої школи ви є')
клас = int(input('Введіть свій клас: '))
if клас >= 1 and клас <= 4:
    print('Ви є учнем початкової школи')
if клас >= 5 and клас <= 9:
    print('Ви є учнем середньої школи')
if клас == 10 or клас == 11:
    print('Ви є учнем старшої школи')
```

**3**

```
Учнем якої школи ви є
Введіть свій клас: 7
Ви є учнем середньої
школи
```

Проте ця задача не є повністю реалізована. Що ж буде, якщо ми введемо число, більше за 11, чи якісь інші дані, які не прописані в умові? Тоді програма нам нічого не виведе, і це логічно, бо для такого не описані спеціальні умови. І за допомогою неповного розгалуження це зробити важко, а часом і неможливо.

## Оператор else. Повне розгалуження

Розглянемо реалізацію повного розгалуження.

Повне розгалуження доповнюється ключовим оператором **else** (інакше). Комп'ютер перевіряє умову після ключового слова **if**, і якщо вона не є істинною, то лише тоді виконує команди з блоку **else**.

```
if умова:
    команда1
else:
    команда2
```

```
pogoda = 'дощ'
if pogoda == 'сонце':
    print('Гарної прогулянки!')
else:
    print('Візьми парасольку')
```

```
age = int(input('Введіть свій вік: '))
if age >= 18:
    print('Вам можна водити автомобіль')
else:
   print('''
   Вам заборонено водити автомобіль.
   Почекайте, коли виповниться 18 років.''')
```

## Алгоритм перевірки кратності числа

Дуже часто потрібно буде перевіряти, чи число є парним або чи ділиться на якесь конкретне число. Тому розглянемо, як можна реалізувати таку задачу.

```
num = int(input('Введіть число: '))
if num % 3 == 0:
    print('Дане число кратне трьом')
else:
    print('Дане число не ділиться на 3')
```

```
Введіть число: 12
Дане число кратне трьом
```

```
Введіть число: 5
Дане число не ділиться на 3
```

Щоб розв'язати цю задачу, спочатку потрібно зрозуміти: що означає «число, кратне 3». Тобто остача при діленні на три повинна бути нулем. Тому в умові йде перевірка: якщо остача при діленні на 3 дорівнює нулю — то вивести повідомлення «Число, кратне трьом», інакше «Число не ділиться на 3».

### Висновок

**Розгалуження** — це важливий елемент програмування, оскільки воно дозволяє створювати більш складні програми.

Воно застосовується в різних сферах програмування, включаючи розробку вебсайтів, ігор, десктопних додатків та багато інших. Розгалуження допомагає програмі приймати рішення на основі умов і виконувати різні завдання відповідно до контексту та потреб користувачів.

### Домашнє завдання

**1.** Ввести число. Перевірити, яке воно — парне чи непарне.

**2.** Катруся має **p** грн. Купивши покупку за **m** грн, їй на касі потрібно розрахуватися. Визначити, скільки решти потрібно віддати касиру. Якщо коштів бракуватиме, то повідомити, скільки не вистачає.

## Практична робота №21

**1.** Користувач вводить температуру води. Виведіть її фізичний стан.

***Примітка:*** є три стани води: *твердоподібний(лід), газоподібний(пара), рідкий(вода).*

**2.** Напишіть програму, яка знаходитиме корінь лінійного рівняння **ax = b**, ввівши параметри **a** і **b** з клавіатури.

***Примітка:*** розглянути три випадки, коли лінійне рівняння має безліч коренів, не має коренів і має 1 корінь.

**3.** Введіть число. Перевірте, чи воно ділиться на 4 і чи ділиться на 6.

**4.** Задача на підрахунок кількості. Дано три числа. Визначіть, скільки з них буде додатніх.

Порядок виконання задачі:

- ввести три числа з клавіатури **a**, **b**, **c** (або окремо, або використати **map**);
- ініціалізувати якусь змінну, яка відповідатиме за кількість (наприклад, **count** або **k**) і присвоїти цій змінній початкове значення 0;
- здійснити перевірку кожного числа на те, чи воно додатне. І якщо число буде додатним, то змінну, яка відповідає за кількість, збільшити на одиницю.

```
a = int(input('a = '))
b = int(input('b = '))
c = int(input('c = '))

count = 0 #Спочатку ми маємо 0 додатних чисел

if a > 0:
    count = count + 1 # 0 + 1 = 1 — переприсвоєння змінної count
if b > 0:
    count += 1 # бо такий запис переприсвоєння
if c > 0:
    count += 1

print('Додатних чисел: ', count)
```

- запустити задачу кілька разів і протестувати її, ввівши різні варіанти чисел;
- показати результат вчительці/вчителеві.

**5.** Дано радіус кола і сторону квадрата. Визначіть, у якої фігури більша площа.

***Примітка:*** `pi` знаходиться в модулі `math` під функцією `pi (math .pi)`.

**6.** Введіть трицифрове число з клавіатури. Перевірте, чи воно є паліндромом.

***Примітка:*** паліндром — це число, яке читається зліва направо так само, як і справа наліво `(373, 565, 808, 777)`.

**7.** Складіть програму, яка автоматично вестиме діалог та по заданій інформації шукатиме довжину лінії у формі трикутника, прямокутника або кола.

*Роз'яснення:*

Програма запитує в користувача форму лінії фігури, наприклад: «Введи форму лінії: `1` — трикутник, `2` — прямокутник, `3` — коло». Залежно від того, яку цифру введе користувач, програма повинна шукати довжину лінії. Для фігури «трикутник» подано зразок:

```
if form == 1:
    a = float(input('a = '))
    b = float(input('b = '))
    c = float(input('c = '))
    P = a + b + c
    print('Довжина лінії у формі трикутника буде:', P)
```

**8.** Розробіть програму, в якій користувач вводить пароль з чотирьох цифр, і якщо він збігається із наперед визначеним паролем, то вивести «Пароль правильний (істинний)», якщо ні — «Пароль неправильний (хибний)».

***Примітка:***

- пароль — це чотирицифрове число;
- створити змінну, яка матиме значення правильного пароля — за бажанням помістити дану змінну в зовнішній модуль;
- записати умови, в яких буде порівняння з введеним користувачем паролем та змінною, яка має значення правильного пароля;
- (* додатково) додати умову, якщо користувач ввів не чотирицифрове число, то вивести повідомлення «Потрібно ввести 4 цифри».

# Тема 22 Реалізація алгоритмів з повторенням. Оператор циклу while

Ключові поняття

**Цикл з умовою**
**Оператор While**
**Безкінечна ітерація**
**Оператор Break**
**Оператор Continue**
**Оператор Else**

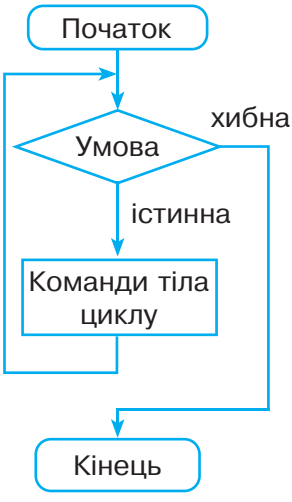
## Цикл з умовою

### Пригадаймо

**Алгоритм з повторенням** — це алгоритм, який передбачає виконання певної послідовності дій (інструкцій) багато разів.

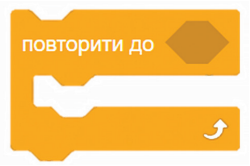
У мові програмування такі алгоритми реалізуються за допомогою циклів. Цикли використовуються для виконання однакового набору інструкцій декілька разів, і кількість повторень може залежати від умови або інших параметрів.

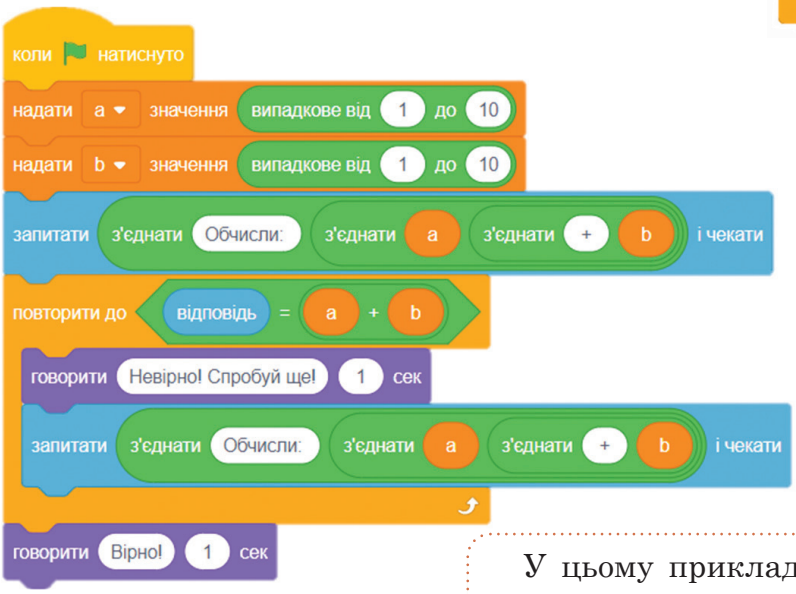
Послідовність команд, які повторюються під час виконання операторів циклу, називається **тілом циклу**.




**Цикл з умовою (або цикл з передумовою)** — цей тип циклу перевіряє умову перед кожним виконанням тіла циклу. Якщо умова істинна, тіло циклу виконується. Якщо умова стає хибною, цикл завершується.

Вивчаючи мову програмування **Scratch** в молодших класах, Ви також реалізовували алгоритм повторення з умовою.




У цьому прикладі програма в нас запитувала суму двох чисел і перевіряла її до тих пір, доки ми не відповіли правильно.

## Оператор While

В мові **Python** цикл з умовою реалізується за допомогою оператора **`while`**.

```
while умова:
    команди
```

Обов'язково має бути відступ, так комп'ютер розуміє, що ці команди відносяться саме до цього циклу.

**Принцип роботи оператора циклу дуже простий**: коли черга доходить до виконання цього оператора, перевіряється умова, яку вказано після ключового слова **`while`** (допоки). Якщо умова істинна, виконуються команди в тілі оператора циклу. Після виконання команд знову перевіряється умова. Якщо умова істинна, виконуються команди, а потім перевіряється умова, і так далі — доти, поки під час перевірки умови не виявиться, що вона хибна. На цьому виконання оператора циклу закінчується, і виконується команда, наступна після умовного оператора.

Той же приклад для перевірки суми чисел, тільки реалізований мовою **Python**.

```
import random

a = random.randint(1, 10)
b = random.randint(1, 10)
print('Обчисли:', a, '+', b)
s = int(input())
while s != a + b:
  print('Не вірно! Спробуй ще')
  print('Обчисли:', a, '+', b)
  s = int(input())
print('Правильно!')
```

Підключили бібліотеку **random** та згенерували два числа **a** і **b** від **1** до **10**. Програма запитує, чи користувач ввів суму даних чисел, і чекає відповіді. Після того, як користувач ввів число з клавіатури, програма перевіряє умову, чи дійсно введене число дорівнює сумі. Якщо ж не дорівнює (логічний оператор **!=**) — то виводить повідомлення, що неправильно, і повторює запит. Якщо ж користувач ввів число правильно — то команди тіла циклу не виконуватимуться — і лише виведе інформацію «Правильно!».

```
Обчисли: 5 +9
4
Неправильно! Спробуй ще
Обчисли: 5 + 9
14
Правильно!
```

**Ітерація** — це одне виконання тіла циклу.

Дуже часто в алгоритмі з циклом потрібно буде використовувати якусь змінну, яка виконуватиме роль лічильника (кількості ітерацій) циклу.

```
i = 1
while i <= 5:
   print('Привіт')
     i = i + 1
```

```
i = 1
while i <= 5:
      print(i)
      i += 1
```

1
2
3
4
5

Виведення чисел від 1 до 5 з кроком 1 (підряд)

```
Привіт
Привіт
Привіт
Привіт
Привіт
```

Цей приклад виводить 5 разів слово **Привіт**. Розберемо, як працює такий алгоритм: на початку в нас є значення змінної **i=1**. Ця змінна нам потрібна для підрахунку кількості ітерацій.

Поки **i = 1**, і за умовою, якщо **i**, тобто **1 <= 5**, то програма виконає команди з тіла циклу. Тобто виведе повідомлення **Привіт** перший раз. І якщо ми не змінимо значення **i**, то цикл не зможе завершитися, буде нескінченним, бо завжди умова **1 <= 5** буде істинною. Отже, додамо рядок **i = i + 1**, який переприсвоїть значення **i** на **2**. І так до тих пір, доки **i** не стане **5**.

Лічильник **i** можна збільшувати не лише на **1**, а й на довільне число, а також зменшувати.

```
i = 0
while i <= 10:
    print(i)
    i += 2
```

```
0
2
4
6
8
10
```

Виведення чисел від **0** до **10** з кроком **2**. Спочатку виводить значення змінної **i**, а потім збільшує на **2**.

```
i = 0
while i <= 10:
    i += 2
    print(i)
```

```
2
4
6
8
10
```

У цьому прикладі, навпаки, спочатку збільшує значення змінної **i** на **2**, а потім виводить на екран.

```
i = 10
while i >= 0:
    print(i)
    i -= 2
```

```
10
8
6
4
2
0
```

Виведення чисел від **10** до **0**, тобто в зворотному порядку. Значення змінної в цьому прикладі зменшується на **2** при кожній ітерації.

## Багаторазова та нескінченна ітерація

Розглянемо такий приклад:

```
a = int(input('Введіть число a або 0 для виходу: '))
while a != 0:
    print(a**2)
    a = int(input('Введіть число a або 0 для виходу: '))
```

Програма працює таким чином, що завжди запитує в користувача число і підносить його до квадрата. Проте цей цикл буде виконуватися до тих пір, доки користувач не введе **0**, як прописано в умові.

```
Введіть число a або 0 для виходу: 4
16
Введіть число a або 0 для виходу: 5
25
Введіть число a або 0 для виходу: 2
4
Введіть число a або 0 для виходу: 0
```

Таким чином, ми можемо вводити багато разів число і зробити ітерацію багаторазовою. Якщо ж нам у вводі потрібне саме число **0**, то при умові можна ввести якийсь інший символ чи набір символів для завершення циклу.

```
while True:
  a = int(input('a = '))
  print(a**2)
```

Для подібних задач в мові програмування **Python** є нескінченний цикл **while true**.

У разі використання такого циклу тіло циклу виконуватиметься завжди, оскільки умова **True** завжди виконується.

Нескінченні цикли часом потрібні, але слід бути обережними, щоб не створити цикл, який ніколи не завершується, як у пропонованій конструкції. Щоб «аварійно» вийти з такого зациклення — потрібно натиснути **Ctrl+C**.

```
a = 4
16
a = 6
36
a = 3
9
a =
Traceback (most recent call last) :
  File "C:\Users\Admin\AppData\Local\Programs\Python\
Python312\1234.py", line 2, in <module>
       a = int(input('a = '))
KeyboardInterrupt
```

натиснути **Ctrl+C**

## Оператор break

**Break** — це оператор у мові програмування **Python**, який використовується для виходу з циклу, доки він виконується.

```
while True:
  a = int(input('a = '))
  if a == 0:
     break
  print(a**2)
```

Коли виконується оператор **break**, виконання циклу припиняється, і виконання програми переходить до наступної інструкції поза циклом.

```
a = 4
16
a = 2
4
a = 6
36
a = 0
```

Потрібно уважно записувати порядок виконання команд в тілі циклу. Спочатку йде перевірка змінної **a**, якщо вона не дорівнює **0**, то виводить квадрат числа **a**, якщо користувач ввів **0**, то використовується оператор **break** і йде вихід з циклу.

У цьому прикладі цикл **while** виконується доти, доки **i** менше-рівне **5**. Але коли **i** досягає значення **3**, виконується оператор **break**, і цикл завершується.

```
i = 1
while i <= 5:
    print(i)
    if i == 3:
        break
    i += 1
```

Результат виводу буде:

```
1
2
3
```

Оператор **break** потрібен у ситуаціях, коли Вам необхідно вийти з циклу після досягнення певної умови або після виконання певної дії.

## Оператор continue

Також є оператор **continue**, який дозволяє пропустити певну частину коду в поточній ітерації циклу і перейти до наступної ітерації циклу.

Переважно оператор **continue** використовують в умовних конструкціях всередині циклу для того, щоб визначити, коли певні операції повинні бути пропущені в поточній ітерації циклу.

```
i = 0
while i < 5:
    i += 1
    if i == 3:
        continue
    print(i)
```

```
1
2
4
5
```

Зауважимо, що спочатку потрібно збільшувати ітерацію, командою **i=i+1**.

## Оператор else

В оператора циклу є й «розширена» версія з ключовим словом **else**.

```
while умова:
    команди 1
else:
    команди 2
```

Крім **while**-блоку в цьому випадку є ще й **else**-блок: після ключового слова **else** (і двокрапки :) розміщується блок команд, які виконуються тільки тоді, коли умова після інструкції **while** є хибною.

Складова **else** в цьому контексті не має нічого спільного зі структурою **if-else**, але вказує на дії, які виконуються після завершення циклу **while**.

Команди **else**-блоку виконуються один і тільки один раз, причому на завершальній стадії виконання оператора циклу. Такого ж ефекту можна досягти, якщо ці команди розташувати поза оператором циклу одразу після нього. Тому з **else**-блоку було б мало користі, якби не два ключові слова (дві інструкції): **break** і **continue**. Інструкція **break** завершує роботу оператора циклу без виконання **else**-блоку. Інструкція **continue** завершує виконання поточного циклу й дозволяє перейти одразу до перевірки умови в **while**-блоці.

```
i = 1
while i <= 5:
    print(i)
    i += 1
else:
   print('Цикл завершено.')
```

```
1
2
3
4
5
Цикл завершено.
```

```
i = 1
while i <= 5:
    print(i)
    i += 1
print('Цикл завершено.')
```

```
1
2
3
4
5
Цикл завершено.
```

```
i = 1
while i <= 5:
    print(i)
    i += 1
    if i == 3:
        break
print('Цикл завершено.')
```

```
1
2
Цикл завершено.
```

```
i = 1
while i <= 5:
    print(i)
    i += 1
    if i == 3:
        break
else:
    print('Цикл завершено.')
```

```
1
2
```

Ця конструкція необхідна, коли Вам потрібно виконати певні дії після завершення циклу, якщо умову було не виконано.

**Приклад 1.** Дано число **n**. Розділити його на цифри та вивести кількість цифр.

Щоб розділити дво- чи трицифрові числа на цифри, можна використовувати математичні оператори **%** та **//**. Щоб розділити довільне натуральне число на цифри, доцільно використати оператор циклу.

```
n = int(input('n = '))#Користувач вводить число
k = 0   #Ініціалізуємо змінну для лічильника цифр

#Розділяємо число на цифри і підраховуємо їх кількість
while n > 0:
    d = n % 10  # Отримуємо останню цифру числа
    print(d)    # Виводимо цифру
    k = k + 1   # Змінюємо значення кількості цифр на 1
    n = n // 10 # Утворюємо число n без останньї цифри

print('Кількість цифр числа: ', k)
```

У цьому коді користувач вводить число. Змінна **k** — потрібна для того, щоб рахувати кількість цифр числа. Цикл працює таким чином, що при кожній ітерації програма відділяє останню цифру числа, виводить її на екран та змінює значення числа **n** на інше — без останньої цифри, при цьому лічильник **k** збільшується на 1.

Наприклад, **43762**.

1 ітерація, **n = 43762 > 0**, тому виконується тіло циклу. Виділяємо цифру **2** і утворюємо нове число **4376**;

2 ітерація, **n = 4376 > 0**, тому виконується тіло циклу. Виділяємо цифру **6** і утворюємо нове число **437**;

3 ітерація, **n = 437 > 0**;

4 ітерація, **n = 43 > 0**;

5 ітерація, **n = 4 > 0**; тому виконується тіло циклу. Виділяємо цифру **4** і утворюємо нове число **n = 0**.

Коли число стає менше або дорівнює **0**, цикл завершується, і на екран виводиться кількість цифр у числі.

```
n = 43762
2
6
7
3
4
Кількість цифр числа: 5
```

**Приклад 2.** Написати програму, яка знаходитиме суму всіх чисел, які вводить користувач з клавіатури. Як тільки користувач введе 0 — програма зупиняється.

```
s = 0 # ініціалізуємо змінну, яка відповідатиме за суму
while True:
   n = int(input('n = ')) # вводимо число з клавіатури
   if n == 0:             # перевіряємо число на умову для
                            завершення програми
       break
   s = s + n              # переприсвоюємо значення змінної,
                            додаючи до неї число
   print('s = ', s)       # вивід суми
```

Якщо «винести» вивід суми за межі циклу, отримаємо такий результат, тобто лише кінцеву суму:

```
n = 5
n = 4
n = 7
n = 6
n = 3
n = 0
s = 25
```

```
n = 4
s = 4
n = 8
s = 12
n = 13
s = 25
n = 7
s = 32
n = 0
```

Цю задачу можна описати іншим способом. У цій програмі користувач спочатку вводить число, а потім уже перевіряє його для умови циклу: якщо число буде відмінне від нуля — додаватиметься з кожною ітерацією до змінної **s**.

```
n = int(input('n = '))
s = 0

while n != 0:
 s = s + n
 n = int(input('n = '))
 print('s = ', s)
```

```
n = 5
n = 6
n = 4
n = 7
n = 0
s = 22
```

```
n = int(input('n = '))
s = 0

while n != 0:
  s = s + n
  print('s = ', s)
  n = int(input('n = '))
```

```
n = 5
s = 5
n = 4
s = 9
n = 6
s = 15
n = 0
```

## Домашнє завдання

1. Написати програму, в якій користувач вводить число **n** і програма виводить суму всіх чисел від **1** до **n**.
2. Знайти найбільшу цифру числа **n**.
3. Вивести квадрати всіх чисел від **1** до **n**.

## Практична робота №22

**1.** Напишіть програму, в якій користувач вводить число **n** і програма виводить всі числа від **1** до **n**.

Порядок виконання задачі:

- написати команду вводу числа **n** з клавіатури;
- ініціалізувати змінну **s = 0**, яка буде відповідати за суму цифр числа;
- записати цикл з умовою, доки **n > 0**;
- в тілі циклу написати такі команди:
  – отримати останню цифру числа (**a = n % 10**);
  – додати до суми отриману цифру (**s = s + a**);
  – перепривоєння числа **n** без останньої цифри (**n = n // 10**).

**2.** Дано число **n**. Виведіть суму його цифр.

**3.** Виведіть всі числа в рядок, кратні **5**, від **1** до **100**. (Виведіть всі числа від **1** до **100** в рядок, кратні числу, введеному з клавіатури).

**4.** Дано число **n**. Виведіть його найменшу цифру.

Порядок виконання задачі:

- написати команду вводу числа **n** з клавіатури;
- ввести змінну найменшої цифри (наприклад, **min**) і присвоїти їй значення **9**;
- записати цикл з умовою, доки число **n** буде більше за **0**;
- в тілі циклу написати такі команди:

а) отримати останню цифру числа;

б) умова: якщо дана цифра менша за значення **min**, то перепривоїти значення **min** даною цифрою:

```
if a < min:
    min = a
```

в) утворити число без останньої цифри;

- вивести значення змінної **min**.

**5.** Використовуючи оператор **continue**, виведіть всі парні числа від **0** до **n**.

**6.** Гра «Вгадай число». Симулюйте гру, де програма обирає випадкове число, і користувач повинен вгадати його, отримуючи підказки: воно більше чи менше за введене ним число.

Порядок виконання задачі:

- підключити модуль **random**;
- створити змінну **secret_number** — яка дорівнюватиме рандомному числу, наприклад, від **1** до **100**;
- запитати користувача ввести число;
- записати цикл з умовою **True**;
- в тілі циклу написати такі команди:

а) *перша умова:* якщо користувач ввів менше число, ніж секретне, то повідомити про це та знову запропонувати користувачу ввести число:

```
if a < secret_number:
    print('Потрібно більше')
    a = int(input('Вгадай число: '))
```

або

```
if a < secret_number:
    a = int(input('Невірно! Потрібно ввести більше число: '))
```

б) *друга умова:* якщо користувач ввів більше число, ніж секретне, то повідомити про це та знову запропонувати користувачу ввести число;

в) *третя умова:* якщо користувач ввів таке ж число, як і секретне, то вивести повідомлення, що число вгадано та вийти з циклу через оператор **break**;

- продемонструвати роботу вчителю/вчительці.

*Додатково:* вивести кількість спроб **k**.

**7.** Удосконалення задачі «Пароль». Доступ до об'єкта закодовано паролем з 4-х цифр. Користувач повинен потрапити на об'єкт, увівши правильний пароль. Проте, якщо він помиляється, на табло висвічується повідомлення типу «Пароль неправильний» (Спробуй ще, Неправильно. Спроба(2), …) і дозволяє ще вводити значення. Якщо вводить правильно, то видає повідомлення типу «Правильно. Доступ надано».

*Додатково:* якщо користувач вводить 3 неправильні спроби — вивести повідомлення типу «Доступ заблоковано» і вийти з програми.

# Реалізація алгоритмів з повторенням. Оператор циклу for

Тема 23

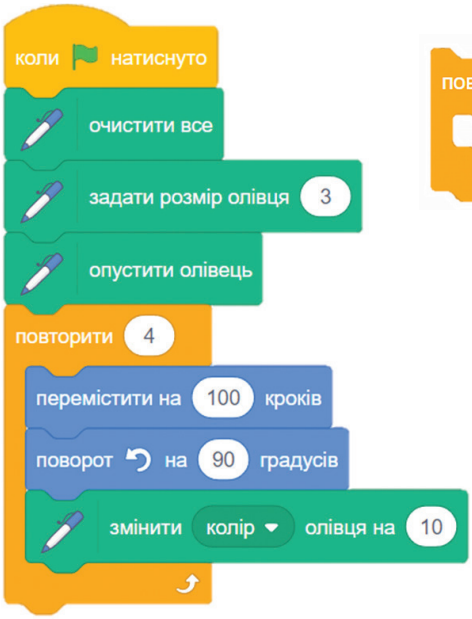
Ключові поняття

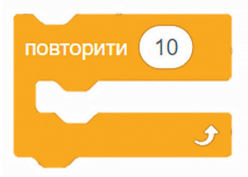
**Цикл з параметром**
**Оператор циклу for**
**Range(n)**
**Range(a,b)**
**Range(a,b,k)**
**Сума чисел**

## Цикл з параметром

**Цикл з параметром** — це один із видів циклів у програмуванні, який виконується певну кількість разів, використовуючи параметр або лічильник, який змінюється під час кожної ітерації.

Вивчаючи мову програмування **Scratch** в молодших класах, Ви також реалізовували алгоритм повторення з параметром.



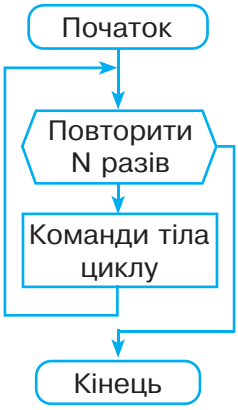




У цьому прикладі за допомогою алгоритму повторення виконавець малює квадрат.

## Оператор циклу for

У **Python** для реалізації циклу з параметром зазвичай використовують оператор **for** та функцію **range()**.

```
for змінна in range(початок, кінець, крок):
    команди
```




Такий тип циклу повторює команди тіла циклу стільки разів, скільки користувач вкаже після ключового слова **for**.

**Змінна** — змінна, яка використовується для збереження порядкового номера поточної ітерації (лічильник).

**Початок** — початкове значення параметра.

**Кінець** — кінцеве значення параметра (не включно в ітерацію).

**Крок** (необов'язковий) — значення, на яке параметр змінюється на кожній ітерації. За замовчуванням, крок дорівнює 1.

## Range(n)

### Способи подання діапазону значень лічильника циклу

**for i in range(n)** — це конструкція мови **Python**, яка створює цикл, який ітерується від **0** до **n-1**. Використовуючи цей шаблон, Ви можете виконати різні операції **n** разів.

У цьому коді **range(5)** створює послідовність чисел від **0** до **4**, і цикл **for** використовує цю послідовність для ітерації по значеннях **i**. Кожне значення **i** виводиться за допомогою функції **print()**. Зауважимо, що лічильник починає рахувати з нуля і не враховує межу "**5**".

```
for i in range(5):
    print(i)
```

```
0
1
2
3
4
```

*Приклади:*

```
S = 0
for i in range(5):
    s = s + i
print(s)
```

```
1 ітерація: i = 0, s = 0 + 0 = 0
2 ітерація: i = 1, s = 0 + 1 = 1
3 ітерація: i = 2, s = 1 + 2 = 3
4 ітерація: i = 3, s = 3 + 3 = 6
5 ітерація: i = 4, s = 6 + 4 = 10
```

Цей код знаходить суму чисел від **0** до **4**, при кожній ітерації додаючи значення **i** до змінної **s**. Потім виводиться сума, тобто **10**.

```
n = int(input('Введіть ціле число n: '))
for i in range(n):
    square = i**2
    print('Квадрат числа', i,'=', square)
```

У цьому прикладі програма отримує від користувача ціле число **n**, а потім використовує цикл **for** для виведення квадратів чисел від **0** до **n-1**. Введіть, наприклад, **5**, і вивід буде:

```
Введіть ціле число n: 5
Квадрат числа 0 = 0
Квадрат числа 1 = 1
Квадрат числа 2 = 4
Квадрат числа 3 = 9
Квадрат числа 4 = 16
```

## Range(a,b)

**for i in range(a, b)** — це конструкція мови **Python**, яка створює цикл, який ітерується від **a** до **b** (не включаючи).

У цьому коді **range(2, 7)** створює послідовність чисел від **2** до **6**, і цикл **for** використовує цю послідовність для 5-ти ітерацій по значеннях **i**. Кожне значення **i** виводиться за допомогою функції **print()**.

```
for i in range(2, 7):
    print(i)
```

```
2
3
4
5
6
```

*Приклади:*

```
a = 5
b = 50

for i in range(a, b):
    if i % 3 == 0:
        print(i, end=' ')
```

```
6 9 12 15 18 21 24 27 30 33 36 39 42 45 48
```

Цей код виводить усі числа від **5** до **49**, які є кратними **3**.

У такому коді використовується цикл **for**, щоб ітерувати змінну **i** від значення **a** до значення **b-1**. Всередині циклу використовується умовний оператор **if**, щоб перевірити, чи число **i** є кратним **3**. Якщо **i** кратне **3** (тобто **i % 3 == 0**), то виводиться значення **i**.

Ця програма на **Python** створює таблицю множення для заданого користувачем числа **n**. Використовуючи цикл **for**, програма обчислює добуток **n** на кожне число від **1** до **9** та виводить результат. Це простий спосіб вивести таблицю множення. Наприклад, якщо користувач введе **6**, програма виведе таблицю множення на **6**:

```
n = int(input('n = '))
for i in range(1, 10):
    res = n * i
    print(n,'*',i,'=',res)
```

```
n = 6
6 * 1 = 6
6 * 2 = 12
6 * 3 = 18
6 * 4 = 24
6 * 5 = 30
6 * 6 = 36
6 * 7 = 42
6 * 8 = 48
6 * 9 = 54
```

## Range(a,b,k)

**for i in range(a,b,k)** — це конструкція мови **Python**, яка створює цикл, який ітерується від **a** до **b** (не включаючи) з кроком **k**.

```
for i in range(3, 20, 4):
    print(i)
```

```
3
7
11
15
19
```

На кожній ітерації **i** збільшується на **4**, і це виводить числа вказаного діапазону, де крок між числами дорівнює **4**.

```
for i in range(20, 3, -4):
    print(i)
```
```
20
16
12
8
4
```

Ця конструкція також дозволяє використовувати числа в зворотному порядку, задаючи крок від'ємним числом.

*Приклад:*

```
a, b, k = 2, 20, 2

for i in range(a, b, k):
    print(i)
```
```
2 4 6 8 10 12 14 16 18
```

Вивести парні числа (або кратні числа якомусь конкретному **k**) можна і за допомогою використання циклу **for** з параметром **range(a, b, k)**, не прописуючи умову **if** на перевірку кратності.

## Сума чисел

Суму можна обчислити за допомогою вбудованої функції **sum()**.

Так, суму натуральних чисел від **1** до **n** (якщо значення цієї змінної задано) можемо обчислити як за допомогою циклу **for**, так і командою **sum(range(1,n+1))**.

- За допомогою циклу **for**

```
s = 0
for i in range(1, 11):
    s = s + i
print(s)
```

- За допомогою функції **sum**

```
s = sum(range(1, 11))
print(s)
```

У блоці команд оператора циклу **for** можна використовувати інструкції **break** і **continue**.

Ефект такий самий, як і під час використання цих інструкцій в операторі циклу **while**: виконання інструкції **break** приводить до завершення оператора циклу, а виконання інструкції **continue** приводить до завершення поточного циклу й переходу до наступного.

```
for i in range(5):
    if i == 3:
        break
    print(i)
```
```
0
1
2
```

У цьому випадку цикл **for** буде виконуватися для значень **i** від **0** до **4**. Якщо **i** стане дорівнювати **3**, то оператор **break** припинить виконання циклу.

```
for i in range(5):
    if i == 2:
        continue
    print(i)
```
```
0
1
3
4
```

У цьому випадку, коли **i** дорівнюватиме **2**, оператор **continue** припустить ітерацію і перейде до наступної ітерації циклу, минувши залишок коду всередині циклу.

## Практична робота №23

**1.** Напишіть програму, яка виведе всі числа від **a** до **b** в рядок.

***Примітка:*** щоб вивести в один рядок, можна використати команду **`sep = ' '`**.

**2.** Напишіть програму, яка виведе всі двоцифрові числа, які кратні числу **n**, введеному з клавіатури.

**3.** Напишіть програму, яка знаходить кількість всіх чисел від **a** до **b**, які закінчуються на **4**, та виведіть їх на екран.

**4.** Напишіть програму, яка всі парні числа від **0** до **n** збільшує вдвічі, а непарні підносить до квадрата.

***Примітка:***

- дані **a** і **b** вводяться з клавіатури;
- щоб знайти останню цифру, потрібно число **`i % 10`**;
- всередині циклу прописати умову, яка перевіряє, чи отримана остання цифра буде 4 (якщо так — то вивести це число в консоль за допомогою команди **print**).

**5.** Напишіть програму, яка знаходить факторіал натурального числа **n**.

***Примітка:*** факторіал числа **n** — це добуток всіх натуральних чисел від **1** до **n** включно.

Наприклад: **`4! = 1234 = 24`**.

**6.** Користувач вводить число з клавіатури. Напишіть програму, яка виводитиме всі дільники даного числа.

***Примітка:***

Для того, щоб вивести дільники числа **n**, потрібно записати цикл від **1** до **n+1**, і всередині циклу перевірити умову: якщо **`n % i = 0`**, то дане і буде дільником.

**7.** Користувач вводить з клавіатури **n** чисел. Виведіть найбільше з них.

**8.** Виведіть всі трицифрові числа, сума цифр яких дорівнює заданому **n**.

**9.** Програма запитує, скільки днів відслідковувати денну середню температуру. Виведіть кількість днів з додатною температурою.

Порядок виконання задачі:

- написати команду вводу кількості днів з клавіатури;
- ініціалізувати змінну **k** або **count**, яка відповідатиме за кількість шуканих днів;
- записати цикл **for** для введення температури кожного дня;
- в тілі циклу написати такі команди:

а) написати команду вводу температури даного дня;

б) записати умову, яка перевірятиме, чи ця температура додатна; якщо умова істинна, то збільшити кількість на одиницю: **k = k + 1**;

- вивести в консоль відповідь: кількість днів з додатною температурою.

**10.** Камера спостережень фіксує швидкість автомобілів, що проїжджають повз неї. За сьогодні було зафіксовано **n** знімків. Визначіть кількість автомобілів, які перевищили швидкість, якщо обмеження швидкості на цій ділянці **m** км/год.

## Домашнє завдання

**1.** Написати програму, яка знаходить суму всіх чисел від **a** до **b**, які закінчуються на **7**.

**2.** Написати програму, яка шукатиме добуток всіх чисел від **a** до **b**. Додатково: всіх чисел від **a** до **b**, які кратні **3**.

**3.** Користувач вводить з клавіатури **n** чисел. Вивести найбільше з них.

## Тема 24 Вкладені умови. Множинне розгалуження


**Повторення**
**Вкладені розгалуження**
**Множинне розгалуження**
**Оператор elif**

### Повторення

Пригадаємо, як реалізується умовний оператор **if**.

На попередніх уроках ми використовували повне та неповне розгалуження при перевірці різних умов.

Проте, якщо після перевірки деякої умови потрібно знову зробити вибір, то використовують вкладені розгалуження.

Таким чином, оператор **if** також можна розміщати всередині іншого оператора **if**. Це означає, що внутрішня умова **if** буде перевірятися лише тоді, якщо зовнішня умова **if** є істинною.

У загальному вигляді вкладений умовний оператор **if** має такий синтаксис:

```
if умова1:
    команда

if умова2:
    команда1
    команда2
        ...
```

```
if умова:
    команда1
else:
    команда2
```

```
if умова1:
  # Блок коду, який виконується,
  # якщо умова1 є істинною
  if умова2:
    # Блок коду, який виконується,
    # якщо обидві умови істинні
```

```
x = int(input('x = '))
if x > 0:
   print('Число додатне')
   if x % 2 == 0:
      print('Число парне')
else:
   print('Число не додатне')
```

```
x = 18
Число додатне
Число парне
```

```
x = 15
Число додатне
```

```
x = -5
Число не додатне
```

Цей код перевіряє введене користувачем число **x**. Якщо **x** є додатним числом, програма виводить «Число додатне» та перевіряє, чи є воно парним. Якщо **x** від'ємне або дорівнює нулю, виводиться «Число не додатне».

*Приклад:*

```
вік = int(input('Введи свій вік: '))
print('Чи закінчив ти навчання?')
навчання = int(input('якщо так - натисни 1, якщо ні - натисни 2 '))
print('Чи маєш ти водійське посвідчення?')
водійськеПосвідчення = int(input('якщо так - натисни 1, якщо
ні - натисни 2 '))

# Перевіряємо, чи особа має 18 років
if вік >= 18:
    print('Вам уже 18. Ласкаво просимо в доросле життя!')
    # Виконується, якщо особа закінчила навчання
    if навчання == 1:
        print('Вітаємо з отриманням диплому!')
    # Виконується, якщо особа має водійське посвідчення
    if водійськеПосвідчення == 1:
        print('Приємних поїздок!')
```

Цей код перевіряє вік особи. І лише тоді, коли людина повнолітня, запитує і виводить відповідну інформацію про навчання та наявність водійського посвідчення. Якщо особа буде мати вік менш ніж 18, то для цієї умови жодних команд немає. Тобто виводу на екран не буде.

```
if умова1:
  # Блок коду, який виконується,
  # якщо умова1 є істинною
else:
  # Блок коду, який виконується,
  # якщо умова1 є хибною
  if умова2:
    # Блок коду, який виконується, якщо
    # перша умова хибна, а друга істинна
```

Вкладену умову можна розміщати і в частині блоку, коли перша умова не виконується:

Цей код перевіряє, яким є введене число — додатним чи від'ємним, а якщо воно від'ємне, то чи є непарним.

```
x = int(input('Введіть число: '))

if x > 0:
  print('Ви ввели додатне число.')
else:
  print('Ви ввели від\'ємне число.')
  if x % 2 != 0:
    print('Ви ввели непарне число')
```

Вкладеною може бути також конструкція **if-else**.

```
if умова1:
# Блок коду, який виконується,
# якщо умова1 є істинною
if умова2:
# Блок коду, який виконується,
# якщо обидві умови істинні
else:
# Блок коду, який виконується,
# якщо умова1 істинна, а умова 2 - ні
else:
# Блок коду, який виконується,
# якщо умова 1 не є істинною
```

```
x = float(input('Введіть
число: '))

if x > 0:
   print('Ви ввели додатне
   число.')
else:
   if x < 0:
      print('Ви ввели від\'ємне
      число.')
   else:
      print('Ви ввели 0')
```

У цій задачі використовується вкладена умова для визначення, яким є введене число — від'ємним чи нульовим, якщо воно не є додатним.

*Приклад:*

За заданою кількістю очей і ніг потрібно відрізнити кішку, павука, морського гребінця і жучка.




Щоб написати код цієї задачі, спочатку потрібно з'ясувати, яка у кожного з них кількість очей і ніжок. У морського гребінця, наприклад, більше сотні очей, а у павуків — вісім. Також у павуків вісім ніг, а у морського гребінця їх немає зовсім. У кішки чотири ноги, а у жучка — шість ніг, але очей у обох по два. Тоді даний код можна записати так:

```
очі = int(input('Введіть кількість очей: '))
ніжки = int(input('Введіть кількість ніжок: '))

if очі >= 8:
    if ніжки == 8:
        print('павук')
    else:
print('гребінець')
else:
    if ніжки == 6:
        print('жучок')
else:
        print('кіт')
```

Ви можете вкладати умовні оператори як завгодно глибоко, але будьте обережні, щоб не створити надто складний код, який важко прочитати та зрозуміти.

Наприклад, пропонований код дуже важко читати та опрацьовувати:

```
if a > 1:
  if b < 20:
    if c == 'hello':
      if d == 30:
        print('Hello')
      else:
        if k == '80':
          print('k is 80')
        if a > 20:
          print('World')
        else:
          print('!!!')
```

## Множинне розгалуження. Оператор elif

Розглянемо ще один приклад з вкладеними розгалуженнями:

```
name = input('Введіть своє ім\'я: ')
if name == 'Микита':
    print('Привіт, Микита!')
else:
    if name == 'Аліса':
        print('Як справи, Аліса?')
    else:
        if name == 'Анна':
            print('Вітаю, Анна!!!')
        else:
            if name == 'Андрій':
                print('Привіт, Андрію)')
            else:
              print('Зачекайте! Я не знаю вашого імені')
```

Цей код можна переписати більш елегантним способом за допомогою оператора **`elif`**. Отже, нова версія того самого коду може бути:

```
name = input('Введіть своє ім\'я: ')
if name == 'Микита':
   print('Привіт, Микита!')
elif name == 'Аліса':
   print('Як справи, Аліса?')
elif name == 'Анна':
   print('Вітаю, Анна!!!')
elif name == 'Андрій':
   print('Привіт, Андрію)')
else:
   print('Зачекайте! Я не знаю вашого імені')
```

Погодьтеся, що цей код коротший, менш вкладений (бажано уникати вкладеності) і легший для читання.

Оператор **elif** використовується у конструкції умовного оператора **if** для перевірки декількох умов підряд. Він є скороченою формою виразу «**else if**», тобто виконується, якщо попередні умови були хибними, а сама умова **elif** виконується.

```
if умова1:
   # виконується, якщо умова 1 істинна
elif умова2:
   # виконується, якщо умова1 хибна, але умова2 істинна
elif умова3:
   # виконується, якщо умова1 і умова2 хибні, а умова 3 істинна
else:
   # виконується, якщо жодна з умов не виконується
```

Код у блоках **elif** виконується тільки для першої істинної умови, інші умови не перевіряються.

Таким чином, ми можемо удосконалити задачу, яку розглядали раніше:

Було:

```
print('Учнем якої школи ви є')
клас = int(input('Введіть свій клас: '))
if клас >= 1 and клас <= 4:
    print('Ви є учнем початкової школи')
if клас >= 5 and клас <= 9:
    print('Ви є учнем середньої школи')
if клас == 10 or клас == 11:
    print('Ви є учнем старшої школи')
```

Стало:

```
print('Учнем якої школи ви є')
клас = int(input('Введіть свій клас: '))
if клас < 1 or клас > 11:
    print('Некоректно введено клас')
elif клас < 5
    print('Ви є учнем початкової школи')
elif клас < 10:
    print('Ви є учнем середньої школи')
else:
    print('Ви є учнем старшої школи')
```

***Висновок.*** Вкладені умови та множинне розгалуження є важливими концепціями в програмуванні, які дозволяють створювати більш складні та гнучкі алгоритми. Вони дозволяють програмі виконувати різні дії залежно від різних умов, що може бути корисним у широкому спектрі завдань.

## Практична робота №24

**1.** Користувач вводить число з клавіатури. Напишіть програму, яка перевірятиме умову, якщо число парне — то чи воно ділиться на 6, а якщо число непарне, то чи воно кратне трьом.

**2.** Напишіть програму, яка допоможе банку автоматично визначити здатність видачі кредиту клієнту. Користувач вводить свій вік. Якщо вік більший за 18 років, то запитує в користувача його місячний прибуток. Якщо прибуток менший за 10 000 грн — то виводить інформацію «Ви можете взяти кредит на техніку», якщо прибуток від 10 000 до 30 000 грн — виводить інформацію «Ви можете взяти кредит на автомобіль», якщо прибуток більший, ніж 30 000 грн — виводить інформацію «Ви можете взяти житло в іпотеку». Якщо вік клієнта менший, ніж 18 років, то виводить інформацію «Ви не можете взяти кредиту».

**3.** Напишіть програму, яка визначає індекс маси тіла BMI людини. BMI (Body Mass Index) обчислюється за формулою: $BMI = \frac{\text{вага (кг)}}{(\text{зріст (м)})^2}$.

Користувач вводить свої зріст та вагу, а програма повідомляє йому його категорію BMI:

- якщо BMI менший, ніж 18,5 — «Недостатня маса тіла»;
- якщо BMI від 18,5 до 25 — «Нормальна маса тіла»;
- якщо BMI від 25 до 30 — «Надмірна маса тіла»;
- якщо BMI більший, ніж 30 — «Ожиріння».

**4.** Запитайте користувача про його оцінку за екзамен та виведіть відповідне повідомлення:

- якщо оцінка в діапазоні 90–100, виведіть «Відмінно!»;
- якщо оцінка в діапазоні 75–89, виведіть «Добре»;
- якщо оцінка в діапазоні 60–74, виведіть «Задовільно»;
- якщо оцінка менш ніж 60, виведіть «Потрібно покращити».

**5.** Дано координати точки **x** та **y**. Визначіть, в якій чверті декартової системи координат вона розміщена.

***Примітка:*** (5; 7) — 1 чверть, (–3; 5) — 2 чверть, (–5; –3) — 3 чверть, (4; –2) — 4 чверть, додатково: (0; 0) — початок координат, (5; 0) — на осі OX, (0; –5) — на осі OУ.

**6.** Користувач вводить число — кількість гривень. Напишіть програму, яка виведе число із відмінюванням слова «гривня». Наприклад: ввід: 25; вивід: 25 гривень.

***Примітка*** (один з випадків перебору):

- якщо число закінчується на 11, 12, 13, …, 19, то буде слово «гривень» (17 гривень, 312 гривень);

Якщо попередня умова хибна, то:

- якщо число закінчується на 1, то буде слово «гривня» (1 гривня, 131 гривня);
- якщо число закінчується на 2, 3 або 4, то буде слово «гривні» (32 гривні, 544 гривні);
- якщо число закінчується на 5, 6, 7, 8, 9, то буде також слово «гривень» (67 гривень, 325 гривень).

## Домашнє завдання

**1.** Написати програму, яка дозволить центру зайнятості розподілити клієнтів по місцях роботи в IT-компанії. Користувач вводить свій вік. Якщо вік менший, ніж 21 рік, то виводить інформацію «Вам потрібно отримати вищу освіту», якщо користувачу більше, ніж 21 рік, то запитує, в якій сфері він бажає працювати та очікує один з трьох варіантів відповіді: 1 — графічний дизайнер, 2 — програміст, 3 — тестувальник. Якщо користувач вводить 1, то виводить інформацію «Вам підійде IT-компанія *APT Студія 360*», якщо користувач вводить 2 — «Вам підійде IT-компанія *Кібер Віртуоз*», якщо користувач вводить число 3 — «Вам підійде IT-компанія *Тестова Вежа*».

**2.** Написати програму, яка допоможе визначити поведінку космічного апарата, що стартує на екваторі, залежно від його початкової швидкості **v**.

Ви вже знаєте з уроків фізики, що тут можливі чотири випадки:

- якщо $\mathbf{v} < 7{,}8$ км/с — апарат впаде на поверхню Землі;
- якщо $7{,}8 \le \mathbf{v} < 11{,}2$ км/с — апарат стане супутником Землі;
- якщо $11{,}2 \le \mathbf{v} < 16{,}4$ км/с — апарат стане супутником Сонця;
- якщо $\mathbf{v} \ge 16{,}4$ км/с — апарат покине Сонячну систему.

# Тестування та налагодження програм. Синтаксичні та логічні помилки. Покрокове виконання програми

Тема 25

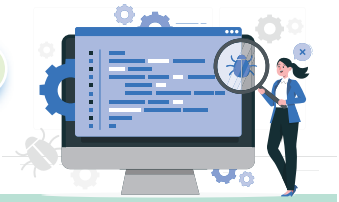
**Ключові поняття**

**Тестування програми**
**Види тестування**
**Налагодження програм**
**Синтаксичні помилки**
**Текстове перевантаження**
**Покрокове виконання програми**

Програмування (створення програм) — завдання досить складне, і цілком природно, що програмісти можуть припускатися помилок під час цього процесу.

Програмні помилки називають **«багами»** (від англ. *bug* — жучок).

## Тестування програми. Види тестування



**Тестування (testing)** — це процес пошуку помилок у програмі. Основна ідея: перевірити, чи працює програма так, як очікується.

### Види тестування

**Модульне тестування**
Перевірка окремих модулів або функцій

**Інтеграційне тестування**
Перевірка взаємодії між компонентами програми

**Системне тестування**
Тестування програми в цілому

## Налагодження програм

**Налагодження (debugging)** — це процес виправлення помилок, які виявили під час тестування. Мета — забезпечити правильну роботу програми.

**Виявлення помилки**

Знайти та визначити, що не працює правильно

**Аналіз помилки**

Розуміння причин та наслідків помилки

**Виправлення помилки**

Зміна коду для усунення проблеми

Налагодження програм на **Python** може передбачати використання різних інструментів і технік для виявлення та усунення помилок, аналізу виконання коду та розв'язання інших проблем.

Ось кілька підходів до налагодження програм на **Python**:

### Використання вбудованих засобів

**Вбудований друк (print)**: вставте в код декілька друків для виведення значень змінних та повідомлень у консоль. Це дозволяє Вам відстежувати хід виконання програми та виявляти помилки.

**pdb (Python Debugger)**: це вбудований інструмент для відлагодження, який дозволяє Вам стежити за виконанням програми, встановлювати точки зупинки та досліджувати значення змінних у режимі виконання.

### Використання інструментів відлагодження

**IDE (Integrated Development Environment)**: використання IDE, таких як PyCharm, VSCode, які надають інтегровані інструменти відлагодження, зокрема подання змінних, встановлення точок зупинки тощо.

**pdb++ або ipdb**: розширені версії стандартного модуля **pdb** з додатковими можливостями.

### Використання інтерактивних середовищ

**IPython**: інтерактивний оболонковий інтерпретатор для **Python**, який надає додаткові можливості порівняно зі стандартним інтерпретатором.

**Jupyter Notebooks**: використання ноутбуків для інтерактивного виконання коду та відлагодження крок за кроком.

### Використання тестувальних фреймворків та аналізаторів коду

**pytest, unittest**: написання тестів для ізоляції та виправлення помилок. Тестове середовище допомагає переконатися, що ваш код працює, як очікується.

**pylint, flake8**: інструменти для аналізу коду та виявлення потенційних проблем.

Залежно від конкретної задачі та Ваших уподобань Ви можете вибрати певні інструменти або комбінувати їх для ефективного налагодження програм на **Python**.

Саме використання `print` та ввід різних даних і є найчастішим випадком тестування програм у школі.

Уміння налагоджувати програми є дуже важливим навиком для програміста. Процес налагодження вимагає великих інтелектуальних зусиль і концентрації уваги, проте це одне з найцікавіших занять.

Тестування та налагодження програм допомагає виявляти та виправляти помилки. Які ж помилки можуть виникати?

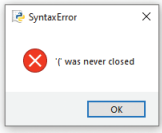
1. Синтаксичні
2. Логічні (семантичні)
3. Помилки виконання

## Синтаксичні помилки

Синтаксичні помилки виникають, коли програма не відповідає правилам граматики мови програмування. Приклади синтаксичних помилок у **Python**:

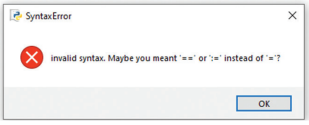
```
# Помилка: забули дужку
print("Hello, world"
```

```
# Помилка: забули дужку
print("Hello, world"
```




```
# Неправильне використання
# знаків в умовах та циклах
if x = 5:
    print('x дорівнює 5')
```

```
#Неправильне використання
#знаків в умовах та циклах
if x = 5:
    print('x is 5')
```




```
# Неправильне відступлення
if x == 5:
print('x дорівнює 5')
```

```
#Неправильне відступлення
if x == 5:
print('x is 5')
```


Ці приклади є лише кількома з багатьох можливих синтаксичних помилок. При виникненні таких помилок **Python** зазвичай надає повідомлення про помилку, яке вказує на місце, де вона виникла, а також її тип.

## Логічні помилки

Логічні помилки виникають, коли програма компілюється і запускається, але вона виконує неправильний алгоритм. Це може призводити до неточних результатів. Приклади логічних помилок:

```
# Помилка в умовному виразі
x = 10
y = 20

if x > y:
    print('x більше ніж y')
elif x < y:
    print('y більше ніж x')
```

```
# Помилка в обчисленні
radius = 5
area = 2 * 3.14 * radius
print(area)
# Неправильне обчислення площі круга
```

Правильний спосіб обчислення площі круга:

```
area = 3.14 * radius ** 2.
```

Сам код правильний, проте не повний. У цьому випадку, якщо **x** і **y** рівні, то жоден з виразів не буде виконаний. Можливо, потрібно включити додатковий випадок для рівності.

Варіантів виникнення логічних помилок є велика кількість.

Логічні помилки важливо виправляти, оскільки вони не завжди супроводжуються повідомленням про помилку, і програма може працювати, але видає неправильний результат.

Важливо уважно перевіряти ваш код на логічні помилки під час розробки та тестування.

**Помилки виконання (runtime errors)** в **Python** виникають тоді, коли програма запускається, але виникає проблема під час виконання. Це може бути пов'язано зі значеннями, які програма отримує під час виконання, неправильним доступом до елементів, діленням на нуль та іншими подібними ситуаціями.

Ось кілька прикладів помилок виконання:

```
# Помилка: неправильне додавання
a = "10"
b = 5
result = a + b
print(result)
```

```
Traceback (most recent call last):
   File "C:\Users\Admin\AppData\Local\
Programs\Python\Python312\сума чисел.py",
line 4, in <module>
    result = a + b
TypeError: can only concatenate str (not
"int") to str
```

```
# Ділення на нуль
result = 10 / 0
print(rezult)
```

```
Traceback (most recent call last):
   File "C:\Users\Admin\AppData\Local\
Programs\Python\Python312\сума чисел.py",
line 2, in <module>
    result = a + b
ZeroDivisionError: division by zero
```

У цьому випадку програма повідомить про помилку, що додали різні типи даних, число і рядок.

```
# Змінна r не визначена
radius = 5
area = 3.14 * r ** 2
print(area)
```

```
Traceback (most recent call last):
   File "C:\Users\Admin\AppData\
Local\Programs\Python\Python312\сума
чисел.py", line 3, in <module>
    area = 3.14 * r ** 2
NameError: name 'r' is not defined
```

Помилки виконання можуть виникати з різних причин, і для їх виявлення та виправлення слід використовувати засоби відлагодження, такі як виведення значень, обробка винятків та інші методи аналізу коду під час виконання програми.

## Покрокове виконання програми

Використовуючи інструменти для налагодження (наприклад, **pycharm**, **vscode** або **pdb**), Ви можете виконувати програму поетапно. Під час кожного етапу перевіряйте значення змінних, щоб переконатися, що вони правильні.

Розглянемо на прикладі застосування модуля **pdb** (python debugger). **Pdb** дозволяє Вам ставити точки зупинки, покроково виконувати код, переглядати значення змінних і взагалі вивчати виконання програми.

```
n = 321

a = n % 10
b = (n / 10) % 10
c = n // 100

z = a + b + c
print(z)

6.100000000000001
```

Знайдемо суму чисел трицифрового числа. Ви вже знаєте цей алгоритм. Проте відповідь у нас хибна. Перевіримо за допомогою **pgb**, в чому наша помилка.

Встановимо точку зупинки після обчислення значень **a**, **b** та **c**, і Ви зможете використовувати команди відлагодження для перевірки значень та виконання коду покроково.

Якщо Ви запустите цей код і відлагодження, то зможете використовувати команди, наприклад:

- **n** (next) для виконання наступного рядка коду;
- **p a**, **p b**, **p c** для виведення значень змінних **a**, **b**, **c**;
- **c** (continue) для виконання коду до кінця.

Це дозволить Вам систематично вивчати виконання коду та перевіряти значення змінних на кожному кроці.

```
import pdb

n = 321
a = n % 10
b = (n / 10) % 10
c = n // 100

# Встановлення точки зупинки
pdb.set_trace()

z = a + b + c
print(z)
```

```
b = (n // 10) % 10
```

```
> c:\users\admin\appdata\local\programs\python
e>()
-> z = a + b + c
(Pdb) p a
1
(Pdb) p b
2.1000000000000014
(Pdb) p c
```

Помітили, що саме число **b** є нецілим числом, а дробовим. І якщо подивитися у формулу знаходження змінної **b** — бачимо, що ділення у формулі використане не цілочисельне.

## Практична робота №25

1. *Синтаксичні помилки.* Створіть програму, яка запитує ім'я в користувача та виводить привітання на екран. Зробіть синтаксичну помилку (наприклад, забудьте закрити дужку або використайте некоректний оператор) та виправте її.
2. *Синтаксичні помилки.* Напишіть програму, що перевіряє, яким є введене число — парним чи непарним. Зробіть синтаксичну помилку в умові та виправте її.
3. *Логічні помилки.* Напишіть програму для обчислення середнього значення двох чисел. Зробіть логічну помилку (наприклад, неправильно обчисліть середнє значення) та виправте її.
4. *Логічні помилки.* Створіть програму для обчислення суми перших **n** натуральних чисел. Зробіть логічну помилку в циклі обчислення суми та виправте її.
5. *Виправлення помилок.* Перед Вами код програми, який визначає вид трикутника за його кутами. Проте в ньому є кілька помилок. Наберіть код в середовищі **Python**, віднайдіть та виправте помилки.

```
# Визначення виду трикутника за його кутами

# Введення кутів трикутника (числові дані)
angle1 = floar(input("Введіть перший кут: "))
angle2 = float(input("Введіть другий кут: "))
angle1 = input("Введіть третій кут: ")

# Сума кутів трикутника повинна дорівнювати 180 градусів
sum_angle = angle1 + angle2 - angle3

if suma_angle = 190:
    # Визначення виду трикутника
    if angle1 == angle2 == angle3:
        print("Рівносторонній трикутник)
    elif angle1 == angle2 or angle1 == angle3 or angle2 =
= angle3:
        print("Рівнобедрений трикутник")
    ekse:
        print("Звичайний трикутник")
else:
print(Трикутника з такими кутами не існує)
```

**6.** *Покрокове виконання.* До коду з попередньої задачі додайте рядки коду для покрокового виконання та запустіть на перевірку, використовуючи `n` для переходу до наступного кроку.

***Примітка.*** Рядок з початком відлагодження розмістити перед перевіркою умови `pdb.set_trace()`.

## Домашнє завдання

**1.** *Синтаксичні помилки.* Створіть програму для обчислення площі круга за формулою $S = \pi r^2$. Зробіть синтаксичну помилку у виразі для визначення площі та виправте її.

**2.** *Логічні помилки.* Напишіть програму, яка виводить найбільше з трьох чисел. Зробіть логічну помилку в умовній конструкції та виправте її.



# Динамічна графіка (анімації)

Теми 26-27

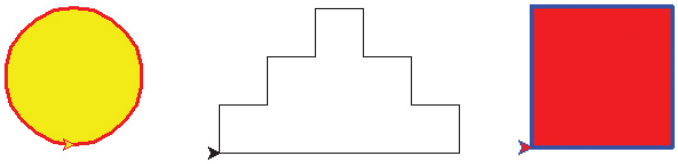
Ключові поняття

**Цикли в малюванні**
**Заповнення фігури**
**Візерунки**
**Повторення фігури**
**Циклічна зміна кольорів**
**Методи анімацій**

## Цикли в малюванні

Ви вже знаєте, що графічна бібліотека **Python turtle** надає простий інтерфейс для створення графічних програм. Вона дуже зручна для опанування основ програмування та візуалізації.

Ви навчилися створювати прості примітиви за допомогою лінійного алгоритму.



Проте для створення складних та динамічних малюнків доцільно використовувати умовні алгоритми та алгоритми з повторенням.

### Використання циклів для малювання

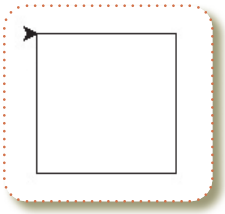
Цикл **`for`** дозволяє повторити блок коду певну кількість разів.

Щоб побудувати квадрат, Ви до цього використовували ось такий код. Проте можна помітити, що частина коду повторюється. Використаємо цикл **`for`**, щоб зробити код коротшим та лаконічним.

Результат буде той самий.

```
from turtle import *

t = Turtle()

t.forward(100)
t.right(90)
t.forward(100)
t.right(90)
t.forward(100)
t.right(90)
t.forward(100)
t.right(90
```

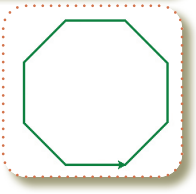
```
from turtle import *

t = Turtle()

for i in range(4):
    t.forward(100)
    t.right(90)
```



```
from turtle import *

t = Turtle()

t.width(3)
t.color('green')

for i in range(8):
    t.left(45)
    t.forward(75)
```

Цикли — це цікавий спосіб просто почати експериментувати, щоб побачити, що намалює програма.

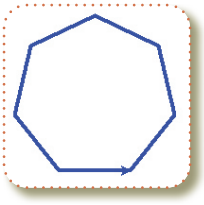
Тож спробуймо виконати кілька прикладів із різними ітераціями в циклах і різними значеннями, переданими у функції **`left()`**, **`right()`** і **`forward()`**, щоб побачити, що відбувається.



Якщо кількість ітерацій та кут повороту в добутку будуть 360 — то ми завжди отримаємо замкнений опуклий многокутник (3×120 — трикутник, 4×90 — квадрат, 8×45 — восьмикутник і т.д.).

Якщо використовувати функції **`input()`**, можна зробити програми, які будуть мати взаємодію із користувачем та робити різні варіації виконання програми.

У цьому прикладі програма запитує користувача ввести розмір, колір та кількість сторін многокутника і малює фігуру, яку замовить користувач.

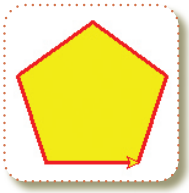


```
from turtle import *

t = Turtle()

w = int(input('Розмір олівця: '))
color = input('Колір олівця: ')
n = int(input('Кількість сторін многокутника: '))

t.width(w)
t.color(color)

for i in range(n):
    t.left(360/n) # вираховує кут повороту
    t.forward(75)
```

```
Розмір олівця: 4
Колір олівця: blue
Кількість сторін многокутника: 7
```

## Заповнення фігури

Також код можна доповнити заповненням фігури.



```
from turtle import *
t = Turtle()
n = int(input('Кількість сторін многокутника: '))
t.width(3)
t.color('red')
t.fillcolor('yellow')
t.begin_fill() # Початок заповнення
for i in range(n):
    t.left(360/n)
    t.forward(75)
t.end_fill() #Завершення заповнення
```

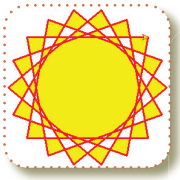
```
for i in range(n):
    t.begin_fill()
    t.left(360/n)
    t.forward(75)
    t.end_fill()
```

Уважно потрібно розміщати рядки початку та кінця заповнення фігури. В цьому коді неправильно розміщені **`begin_fill()`** та **`end_fill()`**.

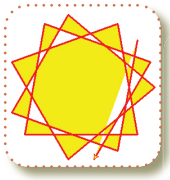
## Візерунки

Якщо поекспериментувати з різним поданням кутів — ми можемо отримати різні цікаві візерунки.

Проте треба зауважити: якщо побудова буде такою, що в результаті отримаємо незамкнену фігуру (Черепашка не повернеться в початкову точку) — то заповнення та сам візерунок виявляться не такими, якими очікували.



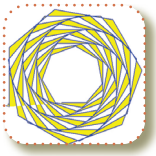
```
t.begin_fill()
for i in range(16):
    t.right(100)
    t.forward(200)
t.end_fill()
```

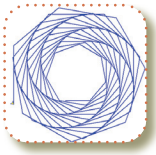


```
t.begin_fill()
for i in range(12):
    t.right(100)
    t.forward(200)
t.end_fill()
```

Якщо під час виконання ітерацій змінювати шлях переміщення або кут повороту, можна також отримати цікаві варіанти візерунків.

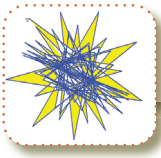
**1**





```
for i in range(100):
    t.forward(70 + i)
    t.left(50)
```

**2**



```
for i in range(75):
    t.right(20 + i)
    t.forward(1 + (i * 5))
    t.right(40 + i)
```

1. У цьому прикладі при кожній ітерації змінюється відстань переміщення Черепашки. Таким чином, з кожним кроком йде збільшення самої фігури.
2. У цьому прикладі з кожною ітерацією змінюється як кут повороту, так і переміщення Черепашки.

Оскільки ці фігури не зможуть бути замкнені, то потрібно уважно обирати чи використовувати заповнення фігури.

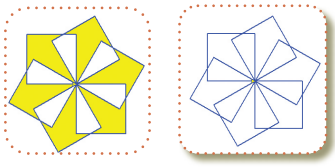
## Повторення фігури

Цікавим результатом також може бути повторення уже готових фігур чи візерунків.

```
for i in range(3): # кількість квадратів
    t.pendown()    # опускає олівець перед намалюванням квадрата
    # цикл, який малює квадрат
    for i in range(4):
        t.left(90)
        t.forward(100)
    t.penup()      # піднімає олівець
    t.forward(150) # переміщає черепашку в інше місце
```

У цьомуу прикладі використовується цикл всередині іншого циклу, що дає змогу скоротити код для створення кількох циклічних фігур (у даному випадку квадратів).

```
for i in range(6): # кількість квадратів
    # цикл, який малює квадрат
    for i in range(4):
        t.left(90)
        t.forward(100)
    t.right(60) # кут повороту черепашки
```

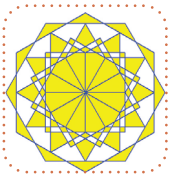


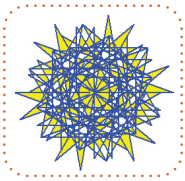
Якщо повертати фігуру всередині циклу, можна отримати такого роду візерунки.

```
for i in range(12): # кількість квадратів
  # цикл, який малює квадрат
  for i in range(4):
     t.left(90)
     t.forward(100)
  t.right(30) # кут повороту черепашки
  # цикл, який малює шестикутник
  for i in range(6):
     t.left(60)
     t.forward(100)
```

Навіть більше: всередині циклу можна розміщувати кілька фігур.

```
for i in range(6):
  for i in range(12):
    t.left(150)
    t.forward(100)
  t.right(60)
  for i in range(9):
    t.forward(120)
    t.left(160
```

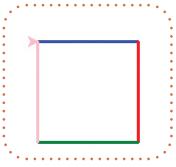




## Циклічна зміна кольорів

```
t.color('blue')
t.forward(100)
t.right(90)
t.color('red')
t.forward(100)
t.right(90)
t.color('green')
t.forward(100)
t.right(90)
t.color('pink')
t.forward(100)
t.right(90)
```

За допомогою циклів можна змінювати кольори ліній і не тільки. Щоб побудувати квадрат з чотирма різними кольорами сторін, можна зробити так:



```
for i in range(4):
    t.color('blue')
    t.forward(100)
    t.right(90)
```

Проте код має надто громіздкий вигляд, і якщо використати цикл, як показано на малюнку, то зміна кольору не працюватиме, оскільки в рядку, де задається колір, потрібно зробити так, щоб при кожній ітерації замість **`blue`** було інше значення.

Один із варіантів, як це зробити — використати список із кількох кольорів.

Детальніше із списками Ви ознайомитеся в старших класах.

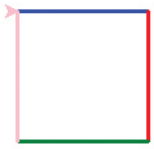
Список задається переліком елементів через кому у квадратних дужках:

```
colors = ['blue', 'red', 'green', 'pink']
```

```
colors = [0] => 'blue'
colors = [1] => 'red'
colors = [2] => 'green'
colors = [3] => 'pink'
```

І щоб звернутися до конкретного кольору, записують так: **colors[0]** — це перший елемент списку, **colors[3]** — це буде останній елемент нашого списку, де **colors** — це назва списку.

Тому в циклі додамо рядок коду, який при кожній ітерації обиратиме почергово колір зі списку.



```
colors = ['blue', 'red', 'green', 'pink']

for i in range(4):
    t.color(colors[i%4])
    t.forward(100)
    t.right(90)
```

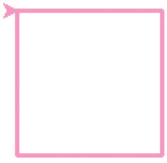
Чому пишемо **color[i%4]**, а не просто порядковий номер **color[i]**? А тому, що у випадку, коли ітерацій буде більше, ніж елементів у списку, то комп'ютер видасть помилку, бо такого елемента не існуватиме. І щоб такого не було, то писатимемо результат дії ділення з остачею ітерації і кількості елементів у списку.

Можна написати список із багатьох кольорів та різними математичними чи рандомними методами звернутися до певного елемента списку.

Ще один із варіатів зміни кольору — це використання **rgb-кольорів** та їх рандомного виклику.

```
color(r, g, b)
r - частина червоного (від 0 до 1)
g - частина зеленого (від 0 до 1)
b - частина блакитного (від 0 до 1)

Наприклад:

t.color(1, 0.6, 0.8)
```

```
for i in range(4):
    t.color(1, 0.6, 0.8)
    t.forward(100)
    t.right(90)
```



Побудуємо квадрат з цим кольором. Побачимо, що це рожевий колір.

Таким чином, за допомогою модуля **random** можна подати значення частин червоного, зеленого та блакитного кольорів рандомно. При цьому відбудеться зміна кольорів.

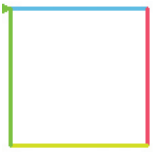
Виведемо спочатку у консоль значення рандомних чисел **r, g, b** і переконаємося, що отримаємо числа від **0** до **1**. Заокруглювати їх не обов'язково.

```
from random import *

r = random()
g = random()
b = random()
print(r, g, b)
```

```
0.10332674228877237
0.07394171401583371
0.18603715429455714
```

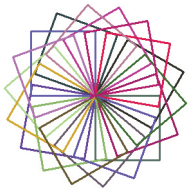
Якщо додамо зміну кольорів у цикл, то отримаємо квадрат з рандомними кольорами сторін.

```
for i in range(4):
    r = random()
    g = random()
    b = random()
    t.color(r, g, b)
    t.forward(100)
    t.right(90)
```



Об'єднавши зміни кольорів та положення фігур циклічно, можна отримати кольорові візерунки.

```
for i in range(18):
   t.color(random(), random(), random())
    for i in range(4):
        t.forward(100)
        t.right(90)
    t.right(100)
```

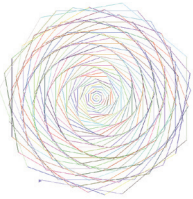


## Методи анімацій

Для кращої візуалізації показаних вище динамічних ефектів малювання в **Turtle** використовують методи, які відповідають за швидкість руху Черепашки та очищенням вікна.

Розглянемо використання цих методів на такому прикладі:

```
from turtle import *
from random import *
w = Screen()
t = Turtle()

for i in range(300):
   t.color(random(), random(), random())
    t.forward(i)
    t.right(50)
```



Ця програма малює візерунок такого типу.

Проте, записавши цей код, ми можемо побачити, що в циклі використовуватиметься багато ітерацій — 300 чи навіть більше. Тому логічно додати швидкість Черепашці або забрати затримку між кадрами малювання.

За швидкість Черепашки відповідає метод **speed()**.

```
t.speed('fastest') - найшвидше
t.speed('slowest') - найповільніше
t.speed(5) - швидкість черепашки від 1 до 10
```

За затримку між кадрами відповідає метод **_delay()**.

```
t._delay(n) - значення n від 0 і більше
```

Додавши дані рядки коду, наша візуалізація стане якнайшвидшою. При цьому можна ще збільшити кількість ітерацій.

```
t.speed('fastest')
t._delay(0)

for i in range(600):
   t.color(random(), random(), random())
    t.forward(i)
    t.right(50)
```

Є ще метод, який одразу виводить кінцеве зображення: **t._tracer(0)**.

Такі методи можна застосовувати не лише для Черепашки, а й для вікна (без нижньої рисочки): **w.delay(), w.tracer()**.

Функція **update()** використовується для оновлення вікна, що містить візуалізацію Черепахи. Вона активує відображення всіх змін, які сталися в процесі малювання, та показує їх на екрані.

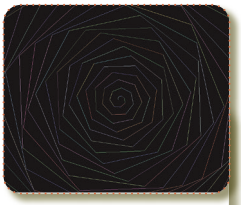
Тому додамо цю функцію в код, і побачимо, як наша фігура буде повторюватися багато разів та миттєво будуватися за допомогою **tracer()**.

```
t.speed('fastest')
t._delay(0)
t._tracer(0)

for k in range(100):
    for i in range(300):
        t.color(random(), random(), random())
        t.forward(i)
        t.right(50)
    w.update()  # оновлює вікно після кожного циклу
    t.home()    # повертає черепашку в початкове положення
    t.right(k)  # повертає черепашку на кут k
```

Щоб фігура рухалася та не накладалася, використаємо метод **clear()**.

Тому перед самою побудовою фігури додамо рядок, який очищає все, що намальовано Черепашкою. Та доповнимо зміною розміру і чорним фоном вікна.

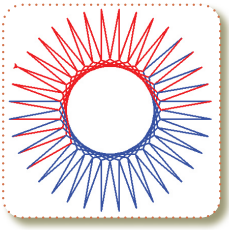
Запустивши код, побачимо анімацію. Проте зображення буде накладатися саме на себе.



```
for k in range(200):
    t.clear()
    for i in range(500):
        t.color(random(), random(), random())
        t.forward(i*k/50) # зміна розміру
        t.right(50)
    w.update()   # оновлює вікно після кожного циклу
    t.home()     # повертає черепашку в початкове положення
    t.right(k)   # повертає черепашку на кут k
```

Умови також можна використовувати для зміни положення чи для зміни кольорів.

```
for i in range(36):
    if t.ycor() > 0:
        t.color('red')
    else:
       t.color('blue')
    for i in range(3):
        t.forward(100)
        t.right(70)
    t.right(100)
```



У цьому прикладі програма щоразу перевіряє координату в Черепашки, і якщо її положення більше за нуль — то надає червоного кольору лінії, а якщо менше за нуль — то синього.

### 1. Алгоритм повороту Черепашки

Головний цикл **while True** використовується для нескінченного обертання трикутника навколо свого центра. У кожній ітерації циклу Черепашка встановлює новий кут **angle** за допомогою **t.setheading(angle)** та оновлюється екран за допомогою **w.update()**. Кут обертання збільшується на 1 під час кожної ітерації циклу, що спричиняє плавне обертання Черепахи.

**1**

```
from turtle import *

w = Screen()

t = Turtle()
t.shape("triangle")
t.shapesize(10)

angle = 0
while True:
    t.setheading(angle)
    angle += 1
    w.update()
```

**2**

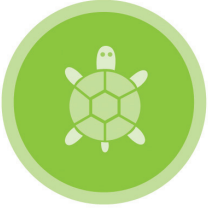
```
from turtle import *

w = Screen()
t = Turtle()

t.shape("triangle")
t.shapesize(5)

while t.xcor() < 400:
    t.forward(1)
    w.update()
```

### 2. Алгоритм руху Черепашки

У цьому коді використовується цикл **while**, який продовжується доти, доки координата **x** Черепашки **t.xcor()** є меншою, ніж **400**. Кожна ітерація циклу викликає метод **t.forward(1)**, щоб перемістити Черепаху вперед на **1** одиницю, та **w.update()**, щоб оновити екран. При цьому створюється рух по осі OX вправо на **400** одиниць.

***Висновок***

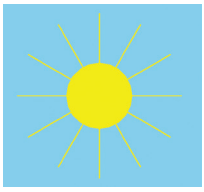
Створення анімацій в **Turtle** відкриває широкі можливості для творчого використання програмування. Використовуючи основні команди та алгоритми, можна створювати різноманітні та цікаві анімаційні ефекти. Робота з анімаціями в **Turtle** допомагає розвивати навички програмування та створювати власні графічні творіння.



## Практична робота №26

**1. Напишіть програму, яка зображує многокутники.**

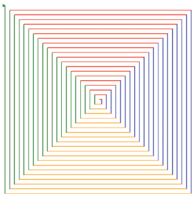
**2. Напишіть програму, яка зображує сонце на блакитному фоні.**



*Рекомендації:*

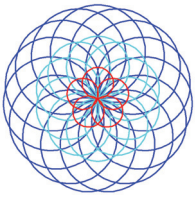
- створити 12 відрізків, які виходять з початкової точки, повернутих на кут 30 градусів (360 : 12 = 30);
- коло можна намалювати або задати тип Черепашки `circle`.

**3. Напишіть програму, яка зображує лабіринт такого типу:**



*Рекомендації:*

- перед головним циклом задати початкову довжину руху (в прикладі це 8);
- записати цикл (мінімум 80 ітерацій);
- усередині циклу записати рядок зміни кольорів;
- перемістити Черепашку вперед на задану довжину;
- повернути Черепашку на 90 градусів;
- переприсвоїти значення довжини руху на 8.



**4.** Зобразіть циклічні візерунки.

**5.** Побудова зображення квітки. Напишіть програму, яка зображує одну із поданих квіток.

*Рекомендації:*

- Пелюстка, яка складається з двох дуг кола.

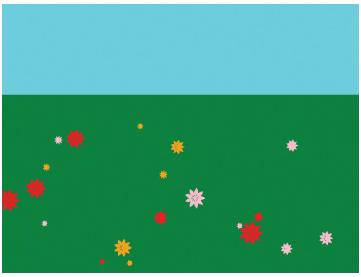
*Зауваження:* кут дуги кола та кут повороту в сумі повинні становити 180 градусів.

```
for i in range(2):
    t.circle(80, 100)
    t.left(80)
```

```
for i in range(4):
    t.forward(50)
    t.left(45)
    t.forward(50)
    t.left(135)
```

- Пелюстка, яка має вигляд ромба.

**6.** Квіткове поле. Складіть програму, яка зображує квіткове поле.



*Рекомендації:*

- створити вікно з розміром 800×600;
- задати колір фону: блакитний;
- намалювати прямокутник, використовуючи цикли, утворивши зелене поле;
- створити цикл для кількості квітів на полі (команди всередині циклу: 1) переміщення Черепашки в рандомне місце в межах зеленого поля; 2) створення квітки різного кольору);
- переміщення: задати координатам **x** та **y** рандомні значення в межах зеленого поля та задати рандомну величину пелюстки квітки;
- створеня квітки: в даному прикладі 9-пелюсткова.

```
x = randint(-380, 380)
y = randint(-280, 80)
size = randint(5, 20)

t.begin_fill()
for i in range(9):
  for i in range(2):
    t.circle(size, 80)
    t.left(100)
  t.right(360/9)
```

*Додатково:* задати рандомне значення кількості пелюсток квітки.

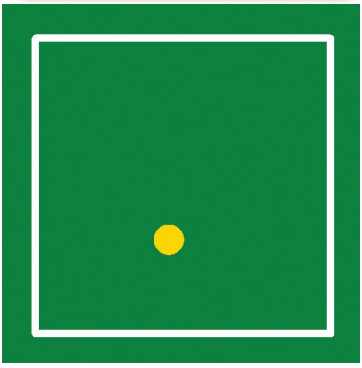
**7.** Рух кульки та її відбиття від стінок.

Створіть програму, в якій кулька рухатиметься у вікні.

***Порядок виконання задачі:***

**1.** Створити вікно з розмірами 600×600 та зеленим кольором фону.

**2.** Створити першу Черепашку, яка малюватиме рамку:



- перемістити Черепашку в точку **(-250, 250)**;
- задати розмір Черепашки **10**;
- задати білий колір Черепашки;
- побудувати квадрат зі стороною 500 за допомогою циклу.

**3.** Створити другу Черепашку — кульку:

- задати форму Черепашки **circle**;
- задати розмір Черепашки **2**;
- задати жовтий колір Черепашки;
- підняти перо, щоб при русі Черепашка не залишала сліду.

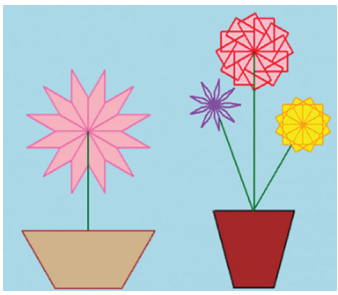
**4.** Створити анімацію кульки:

- **dx = 1.5:** встановлення приросту координат **x** для руху м'яча відносно його поточної позиції по горизонталі;
- **dy = 2.5:** встановлення приросту координат **y** для руху м'яча відносно його поточної позиції по вертикалі;
- **ballX = 0, ballY = 0:** ініціалізація початкових координат м'яча;
- **ball.speed('fastest'):** встановлення максимальної швидкості для м'яча;
- **ball.goto(ballX + dx, ballY + dy):** переміщення м'яча на нову позицію відносно його поточної позиції;
- **ballX = ball.xcor(), ballY = ball.ycor():** оновлення координат м'яча після переміщення;
- умовні оператори **if** перевіряють, чи м'яч дійшов до меж рамки, і змінюють напрямок руху відповідно до цього.

```
dx = 1.5
dy = 2.5
ballX = 0
ballY = 0
ball.speed('fastest')

while True:
    ball.goto(ballX + dx, ballY + dy)
    ballX = ball.xcor()
    ballY = ball.ycor()
    if ballX < -225 or ballX > 225:
        dx = dx * (-1)
    if ballY < -225 or ballY > 225:
        dy = dy * (-1)
```

## Домашнє завдання

1. Написати програму, яка намалює дане подане зображення.



2. Зоряне небо. Скласти програму, яка зображує зоряне небо.
   Рекомендації:
   - створити вікно з розміром 800×600;
   - задати колір фону: чорний;
   - задати колір Черепашки: жовтий;
   - створити цикл для кількості зірочок (команди циклу: 1) переміщення Черепашки в рандомне місце; 2) створення зірочки);
   - переміщення: задати координатам **x** та **y** рандомні значення в межах вікна програми, розмір зірочки — рандомне число:

```
x = randint(-380, 380)
y = randint(-280, 280)
size = randint(10, 30)
```

   - створення зірочки:

```
t.begin_fill()
for i in range(5):
    t.forward(size)
    t.left(720/5)
t.end_fill()
```

# Проєкт 1
# Автоматизований обмінник валют

Теми 28-29

## Етап 1. Організаційний

**Завдання.** Створити проєкт, який реалізує роботу обмінника валют, у якому користувач буде вводити один вид валюти, а програма конвертуватиме її в іншу та здійснюватиме різноманітну взаємодію обмінника з користувачем.

## Етап 2. Підготовчий

1. Визначте мету та функціонал обмінника валют. У цьому випадку мета — створення простого інтерфейсу для обміну грошей між гривнею, доларом та іншими валютами.
2. Визначте, як взаємодіяти з користувачем, які опції він має та як відбуватиметься обмін валют. У нашому випадку:
   - На початку апарат запитує в користувача перші дії: перейти до обмінника або вийти.
   - При переході до обмінника запитати в користувача, яку валюту буде міняти і на що.
   - Залежно від вибору опції обмінник буде конвертувати гроші та виводити інформацію для користувача про його рахунок.
3. Ознайомтеся та визначте початкові курси валют (курс долара, курс євро та ін.), які будете використовувати в проєкті. У реальному проєкті вони можуть бути отримані через зовнішні API, щоб отримати реальні національні курси валют, але в даному випадку ми задамо їх як константи.

## Етап 3. Проєктний

Із самого початку задамо значення констант, які використовуватимемо в проєкті:

```
курс_долара = 39.45
курс_євро = 42.24
```

Зауважимо, що деякі імена змінних та констант будемо записувати кирилицею для кращого розуміння, проте рекомендується задавати імена латиницею.

Запишемо головний цикл **while True**, що дозволяє обмінникові працювати доти, доки користувач не вибере опцію виходу. Весь код будемо розміщувати в тілі даного циклу.

*Головне меню.*

Розпочнемо з початкових опцій **«Перейти до обміну валют»** або **"Вийти"**. При розширенні проєкту опції можуть доповнюватися.

```
while True:
    print('Вас вітає обмінник валют')
    print('1. Перейти до обміну валют')
    print('2. Вийти')
```

Після цього запишемо функцію **input()**, яка отримуватиме вибір опції від користувача. А також створимо змінну, яка буде містити інформацію про загальний рахунок користувача.

Ця змінна відповідатиме за вміст всіх коштів, які користувач буде конвертувати під час сесії використання обмінника.

```
choice = int(input('Введіть номер опції: '))
рахунок = 0
```

*Обробка вибору користувача.* Для реалізації вибору опцій використаємо оператор **if-else**.

```
if choice == 1:

elif choice == 2:

else:
    print('Неправильний вибір. Будь ласка, введіть
правильний номер опції.')
```

Оскільки маємо поки що дві опції, то будемо описувати дві умови, а також, якщо користувач введе дані, які не відповідатимуть номеру опції — видамо інформацію про неправильні дані.

*Створення коду для опції 1.*

Якщо користувач обрав опцію «Перейти до обміну валют», відбувається обмін валют, і переходимо до Меню обміну валют:

```
if choice == 1:
    print('1. Обміняти гривні на долари')
    print('2. Обміняти долари на гривні')
    print('3. Вийти')

    option_currency = int(input('Введіть номер опції: '))
```

Спочатку обмежимося тільки трьома опціями — користувач обирає опцію обміну гривні на долари, долари на гривні або виходу. Також визначимо значення вибраної опції **option_currency** (комбінація слів *опція* та *валюта* англійською мовою).

За аналогією уже описаних умов першого вибору опцій запишемо умови і для цього вибору опцій.

*Обробка вибору обміну валют.*

```
if currency == 1:
    # Обмін гривні на долари

elif currency == 2:
    # Обмін доларів на гривні

elif currency == 3:
    print('''Дякуємо за використання обмінника валют.
        До побачення!''')
    break

else:
   print('Неправильний вибір. Будь ласка, введіть правиль-
ний номер опції.')
```

*Обмін гривні на долари.* Спочатку запишемо рядок, який дозволяє користувачу вводити суму гривень, яку він хоче обміняти на долари, а щоб не обтяжувати користувача знову з вибором виходу з наступного циклу, зробимо так, щоб користувач вводив гроші та обмінював їх до тих пір, доки не введе **0**.

```
if option_currency == 1:
  гривні = int(input('Внесіть гривні до ячейки. 0. Вийти '))
  while гривні != 0:
```

Коли користувач вводить кошти, потрібно виконати математичні обчислення, щоб конвертувати гривні в долари. І заокруглити величину до сотих (два знаки після коми). Також до нашого рахунку, при кожній ітерації, будемо додавати отриману суму в доларах.

```
if option_currency == 1:
   гривні = int(input('Внесіть гривні до ячейки. 0. Вийти '))
   while гривні != 0:
      долари = round(гривні / курс_долара, 2)
      рахунок = рахунок + долари
```

Виведемо код виводу інформації в консоль для користувача:

```
if option_currency == 1:
   гривні = int(input('Внесіть гривні до ячейки. 0. Вийти '))
   while гривні != 0:
      долари = round(гривні / курс_долара, 2)
      рахунок = рахунок + долари
      print('Ви отримали', долари, 'доларів')
      print('На вашому рахунку', рахунок, 'доларів')
```

Для повторення запиту на введення гривень або виходу продубльоємо написану вище функцію **`input()`** всередину циклу. А також виведемо повідомлення про фінальну кількість доларів, яку користувач отримав за обмін гривні, якщо він вирішить вийти.

```
if option_currency == 1:
   гривні = int(input('Внесіть гривні до ячейки. 0. Вийти '))
   while гривні != 0:
      долари = round(гривні / курс_долара, 2)
      рахунок = рахунок + долари
      print('Ви отримали', долари, 'доларів')
      print('На вашому рахунку', рахунок, 'доларів')
      гривні = int(input('Внесіть гривні до ячейки. 0. Вийти '))
   print('Отримайте ваші', рахунок, 'дол. в щілині')
```

*Обмін доларів на гривні.* Аналогічно пишемо код для обміну доларів на гривні, змінюючи формулу для конвертації та повідомлення, які будуть виводитися в консоль:

```
elif option_currency == 2:
   долари = int(input('Внесіть долари до ячейки. 0. Вийти '))
   while долари != 0:
      гривні = round(долари * курс_долара, 2)
      рахунок = рахунок + гривні
      print('Ви отримали', гривні, 'гривень')
      print('На вашому рахунку', гривні, 'гривень')
      долари = int(input('Внесіть долари до ячейки. 0. Вийти '))
   print('Отримайте ваші', рахунок, 'грн. в щілині')
```

*Вихід із програми.* Якщо користувач в Меню обміну валют обере опцію **Вийти**, то виведемо інформацію в консоль, та за допомогою оператора **`break`** вийдемо з програми:

```
elif option_currency == 3:
   print('''Дякуємо за використання обмінника валют.
         До побачення!''')
   break
else:
   print('Неправильний вибір. Будь ласка, введіть
   правильний номер опції.')
```

Аналогічний код продублюємо та зробимо зміни при виборі опції **Вийти** в головному меню:

```
elif choice == 2:
   print('''Дякуємо за використання обмінника валют.
        До побачення!''')
   break
else:
   print('Неправильний вибір. Будь ласка, введіть
   правильний номер опції.')
```

### Етап 4. Тестувальний

Випробуйте різні сценарії використання програми для перевірки правильності роботи коду. Впевніться, що програма коректно реагує на введені дані та обчислює результати обміну валют.

```
Вас вітає обмінник валют
1. Перейти до обміну валют
2. Вийти
Введіть номер опції: 1
1. Обміняти гривні на долари
2. Обміняти долари на гривні
3. Вийти
Введіть номер опції: 2
Внесіть долари до ячейки. 0.Вийти 100
Ви отримали 3945 гривень
На вашому рахунку 3945 гривень
Внесіть долари до ячейки. 0.Вийти 150
Ви отримали 5918 гривень
На вашому рахунку 9863 гривень
Внесіть долари до ячейки. 0.Вийти 0
Отримайте ваші 9863 дол. в щілині
Вас вітає обмінник валют
1. Перейти до обміну валют
2. Вийти
Введіть номер опції: 2
Дякуємо за використання обмінника валют.
                  До побачення!
```

***Додатково:***

### Етап 5. Оптимізація та Розширення

Оптимізуйте код та розширте функціонал. Наприклад:

- додайте підтримку обміну гривень на євро, євро на гривні, долари на євро, євро на долари та інші валюти, такі як злоті, фунти, ліри і т.д.;
- змініть текстову інформацію;
- додайте рядок, який повідомлятиме користувачу про курс валют.

### Етап 6. Документація

Створіть короткий опис функцій програми, коментарі до коду та інструкцію користувача для правильного використання обмінника валют.

## Проєкт 2. Черепашачі перегони

Теми 30-31

### Етап 1. Організаційний

**Завдання.** Створити проєкт, який реалізує схему перегонів Черепашок.

### Етап 2. Підготовчий

**1.** Підключення модулів:

- Модуль **`turtle`** — для створення вікна та змоги працювати з графічними об'єктами.
- Підключаємо модуль **`random`** — для вибору рандомних чисел, які відповідатимуть за рандомну швидкість Черепашок.

```
from turtle import *
from random import randint
```

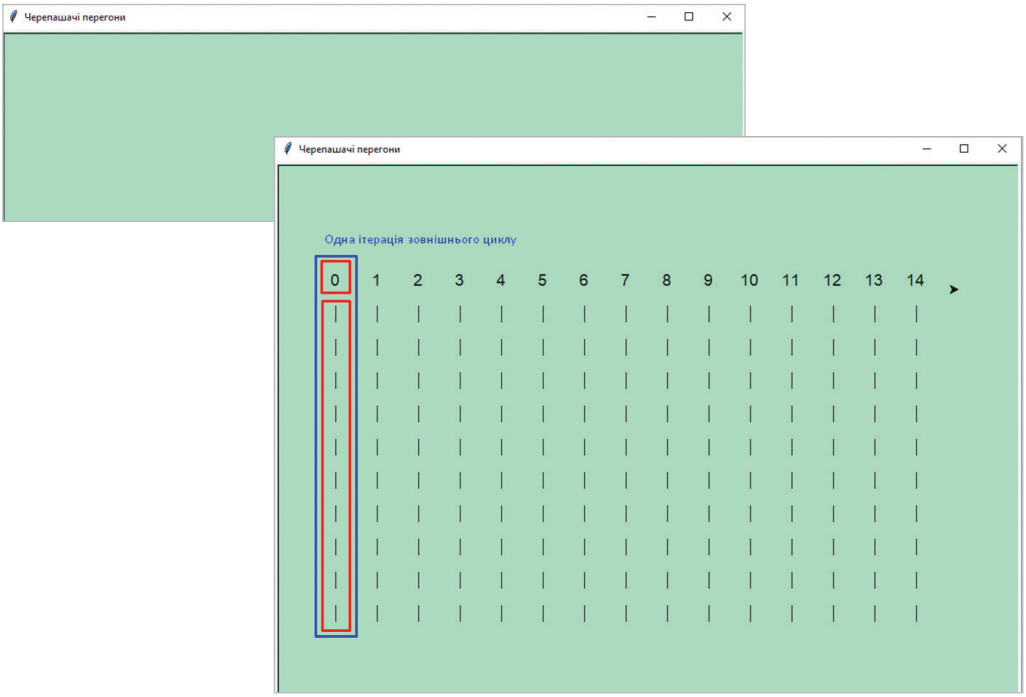
**2.** Створення вікна:

- Задати розміри вікна (900, 700).
- Задати заголовок вікна **Черепашачі перегони**.
- Задати колір фону вікна.

```
window = Screen()
window.setup(900, 700)
window.title('Черепашачі перегони')
window.bgcolor('#A4F0C8')
```

Ви можете обрати свій колір фону.

**3.** Створення траси для перегонів. У нашому випадку траса для перегонів складатиметься з 15-ти ліній-етапів.




Щоб реалізувати цю трасу, знадобиться цикл з 15-ти ітерацій, кожна з якої буде містити:

- текстове поле з текстом, який буде містити значення ітерації;
- цикл з 10-ти ітерацій, який малюватиме 10 вертикальних відрізків.

Тому робимо такі дії:

- створюємо Черепашку **t**, яка малюватиме трасу;
- піднімаємо олівець і переміщаємо **t** в точку **(-380, 200)** — верхній лівий куток, з якого буде починатися створення траси;
- задаємо швидкість Черепашки **5**;
- створюємо цикл, який відповідатиме за 15 етапів;
- усередині циклу задаємо команди створення одного етапу: текстове поле **write**, текст якого дорівнюватиме номеру ітерації, розмір тексту **14**;
- цикл, який буде створювати відрізки з пропуском, а саме: піднімає олівець, переміщається на **20** кроків, опускає олівець, переміщається на **20** кроків, малюючи за собою слід.

```
t = Turtle()
t.penup()
t.goto(-380, 200)
t.speed(5)
```

Після цього створюється перший етап. Щоб перейти до наступного етапу, потрібно підняти олівець, повернутися вгору на 400 кроків, повернути Черепашку вправо, рухати прямо на 50 кроків (відстань між етапами). Тоді починається виконання 2 ітерації зовнішнього циклу і т.д.

```
for i in range(15):
        t.write(i, align = 'centre', font = ('Arial',
        14, 'normal'))
        t.right(90)

        for i in range(10):
                t.penup()
                t.forward(20)
                t.pendown()
                t.forward(20)

        t.penup()
        t.backward(400)
        t.left(90)
        t.forward(50)
```

## Етап 3. Проєктний

**Завдання.** Додаємо учасників перегонів — чотирьох Черепашок.

У модулі, як Ви знаєте, є форма Черепашки **`turtle`**. Для створення учасника оберемо саме таку форму. Задамо колір червоний та розмір Черепашки 2.

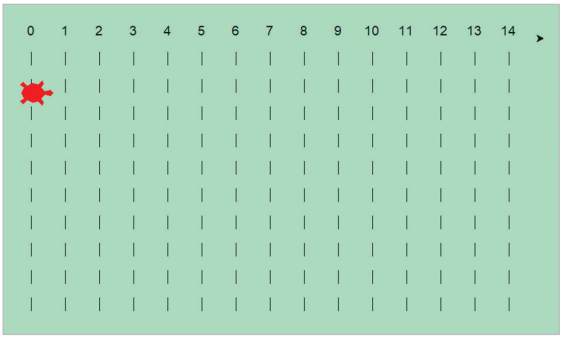
Перемістимо учасника на початок траси для перегонів в точку **`(-380, 120)`**.

Задамо ефект, як з'являтиметься учасник на початку, а саме робить оберт по колу.

```
player_1.penup()
player_1.goto(-380, 120)
player_1.pendown()
```

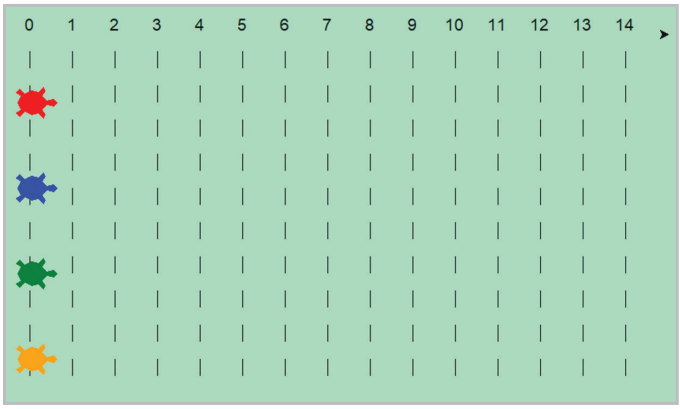
```
player_1 = Turtle()
player_1.color('red')
player_1.shape('turtle')
player_1.shapesize(2)
```

```
for i in range(10):
  player_1.right(360/10)
```



Аналогічно створіть самостійно ще трьох учасників:

- **player_2**, Черепашка синього кольору, з початковим розміщенням **-380, 20**. Ефект появи — змінити кількість ітерацій;
- **player_2**, Черепашка зеленого кольору, з початковим розміщенням **-380, -80**. Ефект появи — змінити кількість ітерацій;
- **player_2**, Черепашка оранжевого кольору, з початковим розміщенням **-380, -180**. Ефект появи — змінити кількість ітерацій.




Можете самостійно змінити кількість учасників, їхні кольори та відстань між ними.

**Створюємо рух учасників перегонів.**

Щоб Черепашка рухалася, ми циклічно будемо переміщати при кожній ітерації вперед на рандомну відстань, наприклад, від **1** до **5**.

```
forward(randint(1, 5))
```

Проте, щоб перегони закінчилися, нам потрібно отримати координату **x** кожної з Черепашок.

```
xp1 = player_1.xcor()
xp2 = player_2.xcor()
xp3 = player_3.xcor()
xp4 = player_4.xcor()
print(xp1, xp2, xp3, xp4)
```

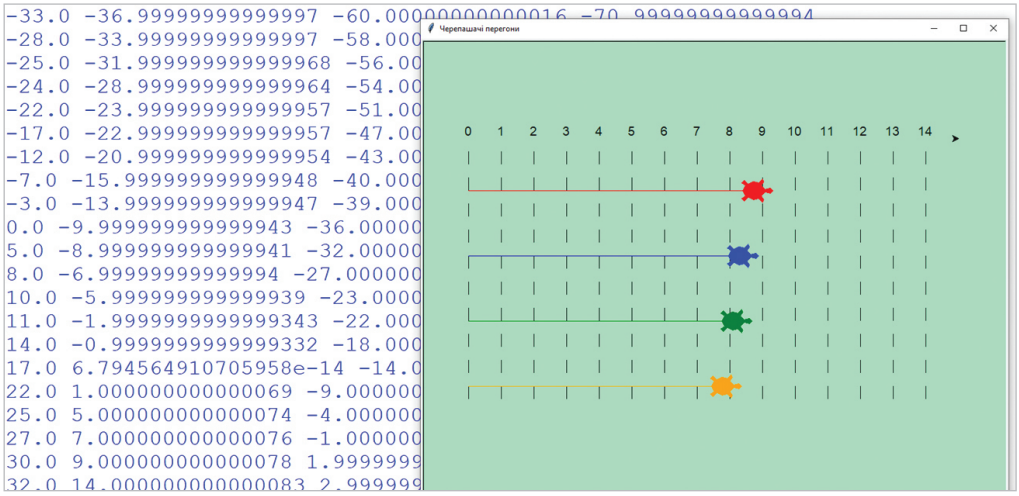
Виведемо ці значення в консоль, щоб протестувати, чи дійсно все правильно працює на цьому етапі програми. Зачекавши, доки виконання дійде до певного рядка, ми отримаємо потрібний результат в консолі.

```
-380 -380 -380 -380
```

Зробимо ще один тест для перевірки, яка координата по **x** нам потрібна для Черепашки, щоб рух Черепашок зупинився. Для цього запустимо цикл на близько 220-230 ітерацій, у якому при кожній ітерації кожна Черепашка рухатиметься вперед, і в консоль будуть виводитися координати кожної Черепашки. Щоб зміни координат змінювалися — то цю частину коду з визначенням координат перемістимо в тіло циклу:

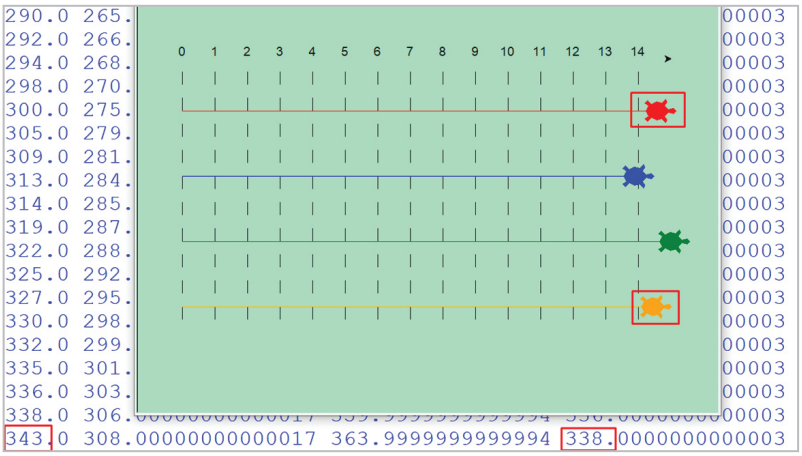
```
for i in range(220):
    xp1 = player_1.xcor()
    xp2 = player_2.xcor()
    xp3 = player_3.xcor()
    xp4 = player_4.xcor()
    print(xp1, xp2, xp3, xp4)
    player_1.forward(randint(1, 5))
    player_2.forward(randint(1, 5))
    player_3.forward(randint(1, 5))
    player_4.forward(randint(1, 5))
```

Запустивши код на виконання, ми побачимо, як в консолі змінюватимуться числа — координати наших Черепашок.




Запускаємо код і тестимо до тих пір, доки не визначимо потрібну координату **x**.

Після кількох тестів у поданому випадку помічаємо, що координати червоної та оранжевої Черепашок більш-менш підходять нам для перемоги. В консолі їхні координати по **x** дорівнюють **343** та **338**. Проте для перемоги ми візьмемо число **340**.




Це значення нам дозволить запустити цикл уже **while**, а не **for**, щоб наші учасники рухалися до тих пір, доки координати у всіх будуть менші за **340**. Як тільки умова перестане бути істинною — рух зупиниться.

```
xp1 = player_1.xcor()
xp2 = player_2.xcor()
xp3 = player_3.xcor()
xp4 = player_4.xcor()

while xp1 < 340 and xp2 < 340 and xp3 < 340 and xp4 < 340:
    xp1 = player_1.xcor()
    xp2 = player_2.xcor()
    xp3 = player_3.xcor()
    xp4 = player_4.xcor()
    print(xp1, xp2, xp3, xp4)
    player_1.forward(randint(1, 5))
    player_2.forward(randint(1, 5))
    player_3.forward(randint(1, 5))
    player_4.forward(randint(1, 5))
```

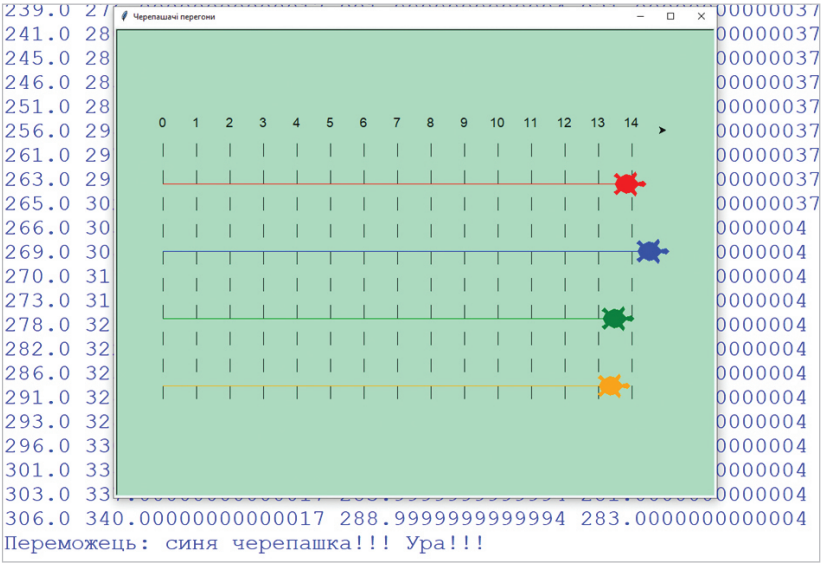
Як тільки побачили, що все працює належним чином — команду **print()** можна вилучити з коду.

Отже, ми використали змінні і всередині циклу, і попереду нього. Попереду — щоб ініціалізувати змінні, які використовуємо в умові. Всередині — переприсвоєння значень змінних.

**Визначаємо переможця**

Додаємо умови, які визначають переможця. Тобто, який із учасників матиме координату більшу або що дорівнюватиме 340 — то та Черепашка буде переможцем.

```
if xp1 >= 340:
    print('Переможець: червона черепашка!!! Ура!!!')
elif xp2 >= 340:
    print('Переможець: синя черепашка!!! Ура!!!')
elif xp3 >= 340:
    print('Переможець: зелена черепашка!!! Ура!!!')
elif xp4 >= 340:
    print('Переможець: оранжева черепашка!!! Ура!!!')
```




Отримали повідомлення, що перемогла синя Черепашка. Наша програма працює.

### Етап 4. Тестувальний

Запустити програму кілька разів, щоб протестувати, чи працює програма так, як потрібно, при цьому змінюючи різні значення, наприклад, швидкостей Черепашок. Якщо буде віднайдено баг — виправити його.

***Додатково:***

1. Вивести повідомлення про перемогу не в консоль, а у вікно програми за допомогою функції **`write()`**.

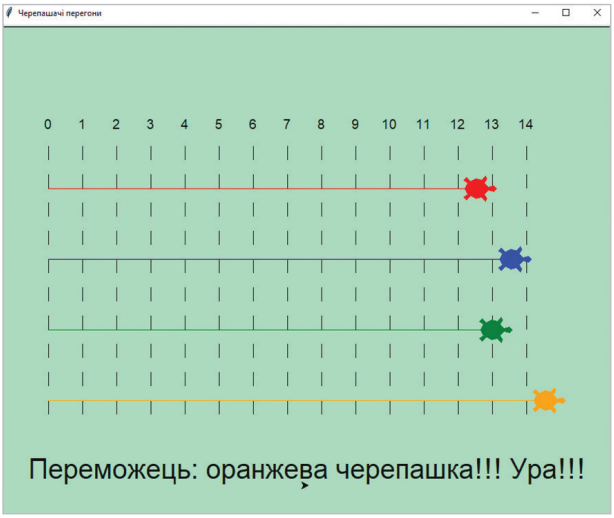
   *Примітка:*

   В умовах не використовувати функцію **`print()`**, а присвоїти значення змінній.

```
if xp1 >= 340:
    text = 'Переможець: червона черепашка!!! Ура!!!'
```

- Перемістити Черепашку **t** в місце, де хочете розмістити текст, і по аналогії до уже використаної функції **write()** вище в коді — вивести інформацію у вікні.

*Наприклад:*




**2.** Справа від ліній траси перегонів створити ще одну лінію — червоного кольору, яка відповідатиме за переможну стрічку. При цьому поміняти значення координат перемоги, бо логічно, що вони змістяться вправо.

**3.** Додати п'ятого учасника.

**4.** Запитувати в користувача, яка Черепашка переможе. І в кінці видати повідомлення, чи відгадав користувач переможця.

**5.** (*) Гра для кількох користувачів:
- запитує, скільки користувачів братиме участь;
- кожен вводить своє ім'я та номер Черепашки, яку вважає майбутнім переможцем;
- у кінці видати повідомлення, яка Черепашка перемогла;
- звіритися з консоллю, де вводили ім'я, хто з користувачів правильно вгадав.

Тема 32

# Поняття мультимедіа та його складові. Апаратна та програмна складова мультимедіа. Кодування аудіо- та відеоданих, формати аудіо-, відеофайлів. Медіаконтейнери. Кодеки

Ключові поняття

**Мультимедіа**
**Мультимедійні кодеки**
**Кодування аудіо- та відеоданих**
**Пошук відео**
**Збереження відео**

## Мультимедіа

**Мультимедіа** — технологія, що об'єднує різні форми медіаконтенту, таких як текст, звук, графіка, відео та анімація, для передачі інформації та взаємодії з користувачем.

Слово «мультимедіа» походить від латинських слів *multi* (багато) та *media* (засоби масової інформації).

Текст | Графіка | Аудіо | Відео | Анімація

**Текст** — інформація, яка може бути представлена у вигляді слів, речень або параграфів. Текст використовується як основний елемент для надання контексту та пояснень.

**Графіка** — зображення та графічні елементи, такі як фотографії, малюнки, діаграми, схеми тощо. Графіка використовується для візуалізації інформації та залучення уваги користувача.

**Аудіо** — звукові елементи, такі як музика, звукові ефекти, голосові коментарі. Аудіоелементи можуть створювати атмосферу, надавати додаткову інформацію або покращувати загальне враження від контенту.

**Відео** — рухомі зображення, які можуть містити комбінацію звуку та зображення. Відео широко використовується для надання динамічної інформації та вражень.

**Анімація** — рухливі зображення чи об'єкти, що створюються за допомогою послідовності кадрів. Анімація додає динаміку та привертає увагу до певних елементів.

**Апаратна складова мультимедіа** передбачає різноманітні пристрої та компоненти, які забезпечують обробку, відтворення та передачу різних видів медіаконтенту.

1. **Процесор (Центральний процесор, CPU)** — відповідає за обчислення та обробку даних, що стосуються мультимедійного контенту.
2. **Графічна картка (відеокарта)** — відповідає за відтворення та обробку графіки. Вона містить графічний процесор (GPU), який дозволяє відтворювати великі обсяги графічних даних, таких як відео та 3D-графіка.
3. **Аудіокарта** — обробляє аудіосигнали та контролює відтворення звуку.
4. **Динаміки** або **навушники** — відтворюють звуковий вихід. Динаміки вбудовані в пристрої (наприклад, у смартфонах або ноутбуках), а навушники можуть бути підключені ззовні.
5. **Камеру** та **мікрофон** використовують для створення відео та аудіо; застосовують у вебкамерах, смартфонах, ноутбуках тощо.

## Мультимедійні кодеки

**Програмна складова мультимедіа** передбачає різноманітні програмні засоби та програми, які дозволяють створювати, редагувати, відтворювати та обробляти різні види медіаконтенту.

### Мультимедійні кодеки

**Кодеки** — це програмне забезпечення, яке дозволяє стискати та розпаковувати відео та аудіо. Наприклад, кодеки **h.264**, **mp3**, **aac**.

### Програми відтворення мультимедіа

Такі як **Windows Media Player**, що відтворюють відео- та аудіофайли різних форматів.

### Графічні редактори

**Adobe Photoshop** та інші — для створення та редагування графічних елементів. Можна використовувати онлайн графічні редактори — **Canva** або **VistaCreate**.

### Мультимедійні платформи

Такі як **YouTube**, **Spotify**, **Netflix** — для потокової передачі відео та аудіо через мережу «Інтернет».

### Відеоредактори та аудіоредактори

Наприклад, **Adobe Premiere**, **iMovie** — для монтажу та редагування відео. Наприклад, **Adobe Audition** — для обробки та редагування аудіосигналів.

## Кодування аудіо- та відеоданих

**Кодування аудіо- та відеоданих** — процес перетворення сигналів з аналогового або нестиснутого цифрового вигляду в стиснутий цифровий формат для ефективнішої передачі та зберігання.


Цей процес дозволяє зменшити обсяг файлів, не суттєво втрачаючи якість.

## Формати аудіофайлів та їхні кодеки

**MP3 (MPEG-1 Audio Layer III)**

Один з найпопулярніших форматів для стиснення аудіо. Кодек MPEG Layer III використовується для створення MP3-файлів.

**WAV (Waveform Audio File Format)**

Формат для зберігання аудіоданих в нередагованому вигляді. Використовується для високоякісного аудіо.

## Формати відеофайлів та їхні кодеки

**MKV (Matroska Multimedia Container)**

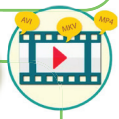
Універсальний контейнер для відео та аудіо, який підтримує різні кодеки, такі як H.264, H.265, VP9 та інші.

**AVI Audio Video Interleave)**

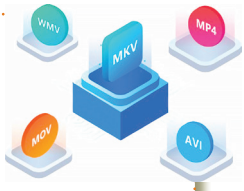
Контейнерний формат для зберігання відео та аудіо. Кодеки можуть варіюватися, включаючи DivX та XviD.

**H.264 (MPEG-4 Part 10 AVC)**

Один з найпопулярніших стандартів стиснення відео для високоякісної передачі в інтернеті та на пристроях.



## Медіаконтейнер




Це дозволяє зручно зберігати та передавати різноманітні типи мультимедійної інформації.

Файловий формат, який об'єднує різні типи медіаданих, такі як відео, аудіо, субтитри, графіка в одному файлі.

## Кодек (від compression-decompression)

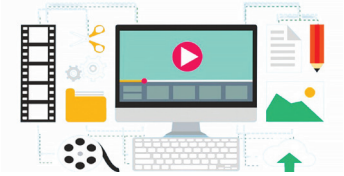
Кодеки визначають, як саме дані будуть стиснуті та відновлені.

Програмне забезпечення або апаратне обладнання, яке використовується для стиснення та розпакування медіаданих в контейнерах.

## Пошук відео

Використовуйте легальні платформи для пошуку та перегляду відео, такі як **Youtube**, **Vimeo**, або інші, які мають право на розміщення відповідного контенту.

Перед тим як почати пошук, чітко визначте тему чи ключове слово, що Вас цікавить. Це допоможе зорієнтуватися та отримати більш релевантні результати.



### Популярні відеоплатформи

**YouTube** — найбільша відеоплатформа, яка пропонує широкий вибір контенту від різних авторів та каналів.

**Vimeo** — некомпресований формат для зберігання аудіоданих в нередагованому вигляді. Використовується для високоякісного аудіо.

**TED Talks** — спеціалізована платформа для виступів від визначених експертів у різних галузях.

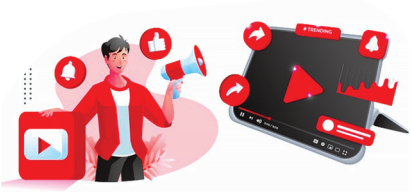
## Збереження відео

Завантажуйте відео лише з відповідних джерел. Уникайте використання з неофіційних сховищ.

**YouTube** дозволяє завантажувати відео офлайн для подальшого перегляду без доступу до інтернету. Ця функція доступна на мобільних пристроях через офіційний додаток.

### Оцінювання відео

Оцінка може передбачати технічні аспекти, такі як якість зображення та звуку, а також творчі та вмістові елементи, такі як сценарій, розкриття теми та взаємодія з аудиторією.



**Залишайте коментарі та відгуки**

Залишайте свої коментарі та відгуки. Це може бути корисно для авторів та інших глядачів.

**Дотримання правил спільноти**

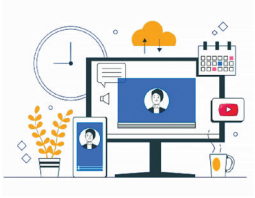
Уникайте публікації образливого чи непристойного вмісту. Дотримуйтеся правил та етики платформи.

**Вказання авторства**

Якщо Ви використовуєте вміст, вказуйте авторство та надавайте посилання на оригінальний вміст, де це можливо.

## Орієнтовний алгоритм оцінки відео

1. **Технічні аспекти**

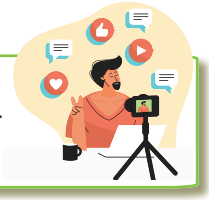


Чи має відео чітке зображення та чіткий звук?

2. **Творчі аспекти**

Чи є цікавий сценарій, що розкриває відео від початку до кінця? Якщо відео призначено для навчання, чи вдало воно передає інформацію? Чи зрозуміло і цікаво?

3. **Взаємодія та залучення**



Перевір реакції глядачів! Які реакції має відео? Чи є коментарі, вподобайки, репости?

4. **Загальне враження**

Яке загальне враження від відео? Чи воно відповідає твоїм очікуванням?

## Практична робота №27

### Частина 1

**Завдання.** Створити відео за допомогою графічного онлайн-редактора **Canva**.
Тема «Ефективні методики навчання».

1. Авторизуйтеся в сервісі **Canva**.
2. Оберіть шаблон **Мобільні відео**.
3. Оберіть шаблон, який Вам подобається.
4. Наповніть контентом.
5. Налаштуйте анімацію до всіх об'єктів.
6. Завантажте відео у форматах **mp4** та **gif**.
7. Створіть свою папку на **Google диску** та завантажте відео в папку.
8. Надайте доступ Вашій вчительці/Вашому вчителю за допомогою електронної пошти.

### Частина 2

**Завдання.** Створити відео за допомогою **ClipChamp**.
Тема «Мої лайфхаки вчасного виконання завдань».

1. Оберіть шаблон — **Instagram**.
2. Оберіть шаблон, який Вам подобається.
3. Додайте свій контент.
4. Налаштуйте шаблон для Вас.
5. Завантажте відео у форматах **mp4** та **gif**.
6. Завантажте відео в папку на **Google диску**.
7. Надайте доступ Вашій вчительці/Вашому вчителю за допомогою електронної пошти.

### Частина 3

**Завдання.** Здійснити пошукову роботу цікавих відео для навчання та саморозвитку.

Зробити колективну презентацію з цікавими відео для навчання.

- Підписати свій слайд.
- Написати кілька слів про відео.
- Додати покликання на відео.

Здійснити пошук відео на трьох різних платформах:

**Vimeo** **TED** **YouTube**

З кожної платформи додати одне відео в колективну презентацію.

*Примітка для вчительки/вчителя:*

Надати доступ учням/ученицям до колективної презентації.

## Домашнє завдання

Створити власний ТОП 10 **YouTube** каналів для навчання, самоосвіти та особистісного розвитку.

Для пошуку обрати різні платформи відеохостингу.

Результати пошукової роботи.

Сервіс — **Canva**.

Шаблон — **Презентація**.

Надати доступ для Вашої вчительки/Вашого вчителя за електронною поштою.

# Поняття про відеохостинг. Основні можливості: пошук відео, збереження. Авторське право, види ліцензій на мультимедійний контент. Рівні доступу до опублікованого відео

Тема 33

**Ключові поняття**

**Відеохостинг**
**Авторське право**
**Рівні доступу до опублікованого відео**

## Відеохостинг

**Відеохостинг** — онлайн-платформа, яка дозволяє користувачам завантажувати, переглядати, коментувати та обмінюватися відео.

Основною метою відеохостингу є надання зручного способу розміщення та спільного використання відеоматеріалів у мережі «Інтернет».

### Основні можливості відеохостингу

**1 Завантаження та збереження відео**

Користувачі можуть завантажувати свої відеофайли на відеохостинг, щоб поділитися ними з іншими користувачами.

**2 Пошук та огляд відео**

Відеохостинги надають можливість користувачам шукати відео за ключовими словами, категоріями чи тегами.

Вбудовані інструменти перегляду дозволяють глядачам комфортно переглядати відео в різних режимах.

**3** **Закритий та відкритий доступ**

Користувачі можуть встановлювати на свої відео обмеження доступу, роблячи їх приватними або доступними для широкої громадськості.

**4** **Взаємодія**

Користувачі можуть залишати коментарі, ставити вподобайки, відправляти відео друзям/подругам чи вбудовувати їх на інші вебсайти. Функції взаємодії роблять відеоплатформу соціальним середовищем для обміну думками та досвідом.

**5** **Монетизація**

Деякі відеохостинги дозволяють користувачам заробляти гроші на своєму вмісті через рекламу, спонсорські відео чи платний контент.

## Авторське право

**Авторське право** — правовий статус, який надає авторам та іншим власникам права на використання та поширення їхніх творів.

Це охоплює різні види творчих робіт, включаючи текст, зображення, музику, відео та інше.

**Текстові графічні зображення**

**Право на власні твори**

Автор має право визначати, як його твір буде використовуватися.

**Право на відшкодування**

Автор має право отримувати відшкодування за використання свого твору.

## Рівні доступу до опублікованого відео

Рівні доступу до опублікованого відео визначають, хто має можливість переглядати це відео та наскільки вільно воно доступне глядачам.

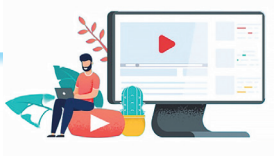
Зазвичай ці сайти створюються з метою представлення особистості, просування особистого бренду або надання спрощеної контактної інформації.

**1** **Публічний (для всіх)**

Відео доступне для будь-якого користувача інтернету без будь-яких обмежень. Відмінно підходить для контенту, який призначений для широкої аудиторії, такої як відео на **YouTube** або відеоблоги.

**2** **Обмежений (не для всіх)**

Відео може бути доступним лише для обраних осіб, які мають спеціальне посилання. Підходить для осіб, які хочуть обмежити доступ до свого контенту або поділитися ним з конкретною аудиторією.



**3** **Приватний (обмежений)**

Відео доступне лише для обраних осіб, які отримали спеціальне запрошення від автора або за допомогою електронної адреси. Цей рівень доступу використовується для осіб, які хочуть зберегти контент тільки для обмеженої групи друзів чи колег.

**4** **Для спонсорів**

Тип рівня доступу «для спонсорів» на платформі YouTube вказує на наявність спеціального рівня або переваг для тих глядачів, які підтримують канал фінансово через програму «Спонсорство». Цей функціонал дозволяє власникам каналів створювати ексклюзивний контент та переваги для осіб, які обирають спонсорську підтримку.

## Практична робота №28

### Частина 1

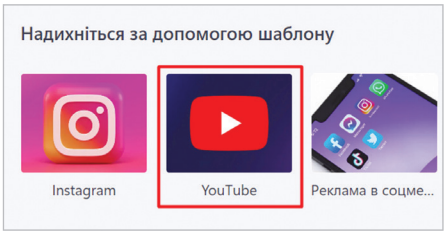
**Завдання.** Створити відео за допомогою онлайн-відеоредактора **ClipChamp**.

- Оберіть шаблон — слайд-шоу.
- Додайте візуальні ефекти, переходи, графічні елементи, титри, фрагменти відео, вкладання зображення, інфографіки.
- Створіть відео по визначних місцях обраної Вами країни.
- Збережіть готове відео у форматі **mp4**.
- Завантажте на свій **Google диск** та надайте покликання для перегляду педагогу.

### Частина 2

**Завдання.** Створити відео — початок та кінцівку для **YouTube.**

1. Оберіть шаблон **YouTube**.
2. Редагуйте шаблон.
3. Збережіть відео у форматі **mp4**.




## Домашнє завдання

- Створити відео за допомогою онлайн-відеоредактора **ClipChamp**.
- На Вашу тематику.
- Завантажте в свою папку на **Google диску** на надайте доступ Вашій вчительці/Вашому вчителю.

Тема 34

# Власний цифровий відеобраз (культура комунікації, культура відеоконтенту, інфосимуляція, дипфейк)

Ключові поняття

**Власний цифровий відеобраз**
**Культура комунікації**
**Культура відеоконтенту**
**Дипфейк (deepfake)**
**Інфосимуляція (infosimulation)**

## Власний цифровий відеобраз

**Власний цифровий відеобраз** — унікальний індивідуальний стиль або образ, який створюється чи відображається в цифровому відеоконтенті, такому як власні відео, анімації чи інші мультимедійні роботи.

**Оригінальний вміст**

Твої власні ідеї та контент, який відрізняється від інших. Спробуй додавати свій особистий погляд та творчий підхід.

**Яскраве та впізнаване зображення**

Роби свої обкладинки та логотипи такими, щоб їх було легко впізнати. Використовуй унікальні кольори та елементи дизайну.

**Особистий стиль ведення**

Визнач свій стиль комунікації. Це може бути твій голос, тон голосу, жести чи будь-які інші особливості, які роблять тебе унікальним.

**Особливі ефекти чи графіка**

Додавай у свої відео особливі ефекти чи графічні елементи.

## Культура комунікації

**Культура комунікації** — система норм, правил і цінностей, які визначають, як люди взаємодіють та спілкуються між собою. Основною метою культури комунікації є забезпечення ефективного обміну інформацією та ідеями.

### Уважність та повага

Використовуйте яскраві та привабливі зображення, які відображають сутність Вашого відео. Вони мають бути цікавими та релевантними.

### Ввічливість та етикет

Робіть свої обкладинки та логотипи такими, щоб їх було легко впізнати. Використовуйте унікальні кольори та елементи дизайну.

### Особистий стиль ведення

Визначте свій стиль комунікації. Це може бути Ваш голос, тон голосу, жести чи будь-які інші особливості, які роблять Вас унікальним.

### Особливі ефекти чи графіка

Додавайте у свої відео особливі ефекти чи графічні елементи.

## Культура відеоконтенту

**Культура відеоконтенту** — унікальний індивідуальний стиль або образ, який створюється чи відображається в цифровому відеоконтенті, такому як власні відео, анімації чи інші мультимедійні роботи.

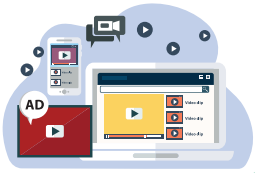
## Банер каналу на YouTube

**Банер каналу на YouTube** — велике зображення, яке відображається у верхній частині сторінки каналу. Ефективний банер може привертати увагу глядачів і робити канал більш привабливим.

| | |
|---|---|
| Логотип та ім'я каналу | Додайте свій логотип та ім'я каналу. Зробіть їх помітними, оригінальними та читабельними. |
| Графічний елемент або зображення | Використовуйте яскраві та привабливі зображення, які відображають тематику Вашого каналу або його контент. |
| Короткий опис каналу | Додайте короткий опис, який швидко пояснює, що можна очікувати на Вашому каналі. |
| Заклик до дії (CTA) | Додайте заклик до дії, наприклад, «Підписуйся!», «Став вподобайку!» або «Дивися нове відео!». |

## Обкладинка відео на YouTube

**Обкладинка відео на YouTube** — важливий елемент, який привертає увагу глядачів та збільшує ймовірність того, що вони клікнуть на Ваш контент.




| | |
|---|---|
| Зображення, що привертає увагу | Використовуйте яскраві та привабливі зображення, які відображають сутність Вашого відео. Вони мають бути цікавими та релевантними. |
| Заголовок відео | Додається на обкладинці відео, повинен бути лаконічним та виразним. Забезпечте йому читабельність, використовуючи контрастні кольори та шрифти, що легко читаються. |
| Логотип та ім'я каналу | Додається для визначення авторства та брендингу. Розмістіть їх так, щоб вони не заважали розумінню зображення обкладинки. |
| Елементи дизайну та фільтри | Використовуйте елементи дизайну, які додають унікальність обкладинці. Фільтри чи ефекти можуть також допомогти привернути увагу. |

## Дипфейк (deepfake). Інфосимуляція (infosimulation)

**Дипфейк (deepfake)** — технологія, яка використовує штучний інтелект для створення монтажу або модифікації мультимедійного контенту, такого як відео, таким чином, що результат виглядає дуже реалістично і може вводити в оману глядача.


**Інфосимуляція (infosimulation)** — термін, який використовується для опису процесу створення або поширення інформації, яка може бути сприйнята як реальна, але фактично є модельованою або вигаданою. Інфосимуляція може містити використання вигаданих подій, статистики чи фактів з метою впливу на думки або переконання аудиторії.

**Фейк** — означає підробку, вигадку, щось несправжнє. Часто вживається в інформаційній сфері, коли йдеться про новини.


### Як відрізнити фейк від правди?

Перевірити достовірність інформації можна шляхом звернення до офіційних сайтів.

**Переглянути першоджерело.** Якщо Ви побачили сумнівну новину, в першу чергу дізнайтеся, звідки пішла інформація.

**Авторство.** Якщо стаття підписана як admin, несправжнім ім'ям або взагалі не підписана, швидше за все, Ви читаєте фейк.

**Емоційна мова та лексика.** Якщо лексика емоційна, викликає обурення або, навпаки, сильні позитивні емоції — ймовірно, перед Вами фейк.

**Наявність фактів, даних.** Відсутність конкретики, яку можна перевірити — місце, час, ім'я, назва установи тощо.

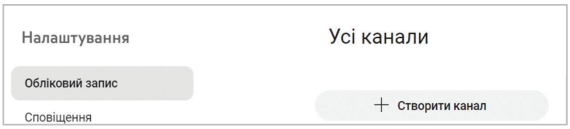
**Дата публікації.** Обов'язково перевірити дату публікації.

**Непоширення такої ж інформації серйозними медіа.** Непоширення такої ж інформації серйозними медіа, до яких є довіра.

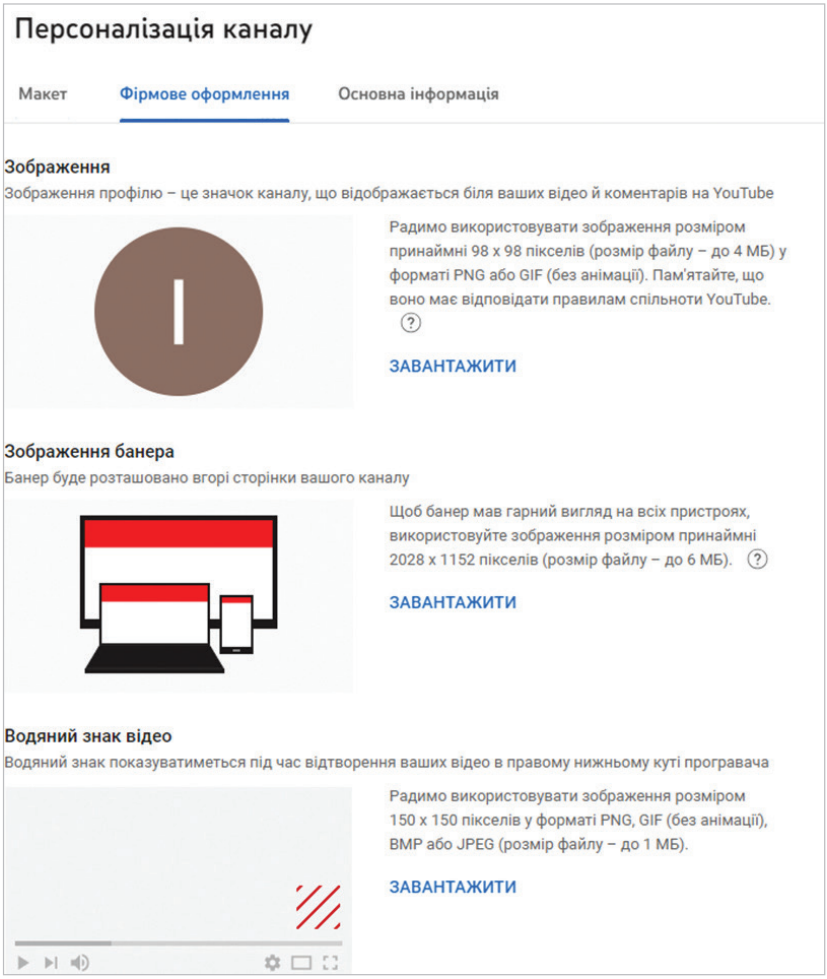
## Практична робота №29

**Завдання.** Створення власного **YouTube**-каналу та публікація відео.

1. Увійдіть в свій обліковий запис **Google**.
2. Оберіть **Змінити обліковий запис** та **Створити канал**.



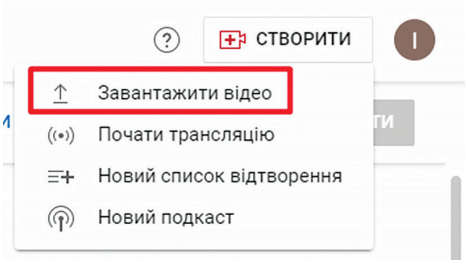

3. Назвіть канал своїм прізвищем та ім'ям.
4. Оберіть **Персоналізувати канал** та налаштуйте канал.



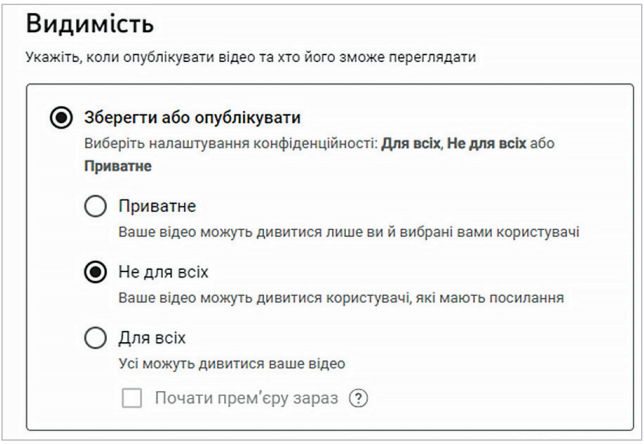
5. За допомогою графічного онлайн-редактора **Canva** створіть:
   - зображення банера з шаблона;
   - водяний знак відео.
6. Ви вже створювали свій віртуальний аватар — завантажте його на зображення профілю.
7. Додайте в розділ **Основна інформація** — інформацію про себе та свої хобі.

### Публікація відео в YouTube

1. Відео, яке ми створили минулого уроку, публікуємо на свій новий **YouTube** канал.




2. За допомогою графічного онлайн-редактора **Canva** — створюємо обкладинку відео.
3. Вказуємо заголовок відео та опис відео.
4. Якщо є потреба — додаємо відео в список відтворення.
5. Вкажіть видимість — не для всіх.
6. Відправте покликання Вашому вчителю/Вашій вчительці за допомогою електронної адреси.



***Пам'ятка.*** Що повинно бути на банері каналу в **YouTube**:

- Логотип та ім'я каналу.
- Графічний елемент або зображення.
- Короткий опис каналу.
- Соціальні мережі та лінки.
- Заклик до дії (CTA).
- Контактна інформація.

Що повинно бути на обкладинці відео в **YouTube**:

- Яскраве та привабливе зображення.
- Заголовок відео.
- Логотип та ім'я каналу.
- Заклик до дії (CTA).

## Домашнє завдання

Створити мобільні відео за шаблоном.
В графічному редакторі **Canva**.
Шаблон — мобільне відео.
Тематика — на Ваш вибір.
Опублікувати в свій **YouTube**-канал, приватність не для всіх.
Надіслати покликання Вашій вчительці/Вашому вчителю за допомогою покликання.

**Тема 35** **Публікація відео в інтернеті, види відеоподкастів. Моушн-дизайн. Зворотний зв'язок та цільова аудиторія**

Ключові поняття

**Публікація відео**
**Підкаст**
**Моушн-дизайн**
**Цільова аудиторія**
**Зворотний зв'язок**

## Публікація відео

**Публікація відео** — процес завантаження та розміщення відеофайлів в інтернеті для доступу аудиторії.

### Створення відео та вибір платформи

- Продумайте цікавий сценарій.
- Забезпечте високу якість відео.
- Оберіть платформу.

### Створення обкладинки та опису

- Створіть цікаву обкладинку для відео.
- Додайте опис, який чітко описує зміст відео та містить ключові слова.

### Оптимізація для пошукових систем

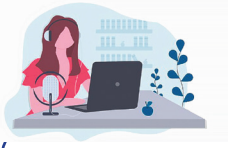
- Використовуйте ключові слова у заголовку, описі та тегах.
- Додайте посилання на Ваші соціальні мережі та поширюйте однокласникам/однокласницям за допомогою месенджерів.

### Взаємодія з глядачами та відстеження аналітики

- Відповідайте на коментарі та питання глядачів.
- Аналізуйте дані про перегляди, вподобайки, коментарі.

## Підкаст

**Підкаст** — цифровий медіаформат, який складається з аудіо- або відеофайлів, зазвичай використовується для стрімінгу, доступно для завантаження через інтернет та прослуховування офлайн.



### Аудіопідкаст

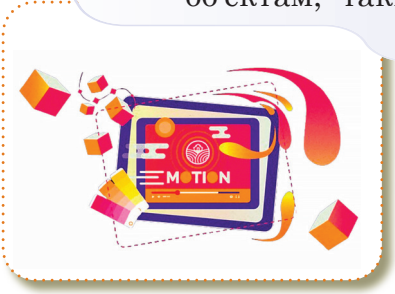
Оскільки аудіопідкасти базуються на звуці, важливо забезпечити високу якість аудіо. Аудіопідкасти можна слухати в будь-який час, навіть коли користувачі зайняті і не можуть вдивлятися в екран.

### Відеопідкаст

Відеопідкасти дозволяють долучити візуальний контент, такий як графіка, демонстрації, графічні ефекти тощо. Редагуйте відео так, щоб воно було цікавим, але не перебільшувати з ефектами.

## Моушн-дизайн

**Моушн-дизайн** — вид дизайну, суть якого полягає в тому, щоб надати рух та динаміку різноманітним об'єктам, таким як текст, картинки, логотипи чи ілюстрації.




Ця техніка використовується в різних галузях, включаючи вебдизайн, рекламу, відеопродакшн, ігрову індустрію та багато інших.

**Рух та трансформація**

Сайт повинен бути зрозумілим та не перевантаженим інформацією.

**Графічні ефекти**

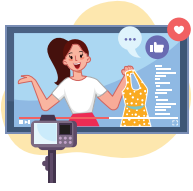
Додавання спеціальних ефектів, таких як світлові блискітки, тіні, частинки для поліпшення візуальної привабливості.

**Використання часу**

Керування часом для створення різних темпів анімації, затримок та ефектів.

## Цільова аудиторія

**Цільова аудиторія** — група людей, на яку спрямований Ваш контент або кому буде цікаво дивитися Ваш контент.




## Зворотний зв'язок

**Зворотний зв'язок** — інформація, яку отримуєте від своєї аудиторії щодо Вашого контенту.


## Практична робота №30

### Частина 1. Створення відео за допомогою сервісу Canva

Обираємо шаблон **Відео**.

Обрати тип **Відеоповідомлення**.

- Налаштовуємо анімацію до об'єктів.
- Налаштовуємо переходи.
- Налаштовуємо час та тривалість.
- Додаємо ефекти.

Зберегти у форматі **.mp4** та опублікувати на своєму **YouTube**-каналі, вказавши приватність — не для всіх.

Покликання надіслати Вашій вчительці/Вашому вчителю та однокласницям/однокласникам за допомогою електронної пошти.

Прокоментувати відео однокласників та поставити реакції.

### Частина 2. Створення відео за допомогою сервісу Canva

Обираємо шаблон **Відео**.

Обираємо тип **Відеоколаж**.

- Налаштовуємо анімацію до об'єктів.
- Налаштовуємо переходи.
- Налаштовуємо час та тривалість.
- Додаємо ефекти.

Зберегти у форматі **.mp4** та опублікувати на своєму **YouTube**-каналі, вказавши приватність — не для всіх.

Покликання надіслати Вашому вчителю/Вашій вчительці та однокласникам/однокласницям за допомогою електронної пошти.

Прокоментувати відео однокласників та поставити реакції.

## ЗМІСТ

### РОЗДІЛ 1. ПЕРСОНАЛЬНИЙ ЦИФРОВИЙ ПРОСТІР



### РОЗДІЛ 2. ЦИФРОВЕ СЕРЕДОВИЩЕ ДЛЯ НАВЧАННЯ ТА СПІВПРАЦІ

## РОЗДІЛ 3. ВІЗУАЛЬНИЙ КОНТЕНТ



## РОЗДІЛ 4. ТЕКСТИ ТА ПУБЛІКАЦІЇ



## РОЗДІЛ 5. ГРАФІЧНЕ ПРОГРАМУВАННЯ

**Відомості про стан підручника**

| № | Прізвище та ім'я учня/учениці | Навчальний рік | Стан підручника | |
|---|---|---|---|---|
| | | | на початку року | в кінці року |
| 1 | | | | |
| 2 | | | | |
| 3 | | | | |
| 4 | | | | |
| 5 | | | | |

*Навчальне видання*

ТРІЩУК Інна Володимирівна
ЛАЗАРЕЦЬ Олександр Юрійович

# ІНФОРМАТИКА

**Підручник для 7 класу закладів загальної середньої освіти**

*Рекомендовано Міністерством освіти і науки України*

**Видано за рахунок державних коштів. Продаж заборонено**

Підручник відповідає Державним санітарним нормам і правилам «Гігієнічні вимоги до друкованої продукції для дітей»

*У підручнику використано ілюстрації та фотографії з таких інтернет-джерел:*
*www.freepik.com www.wikipedia.org*

Головний редактор *Богдан Будний*
Редактор *Володимир Дячун*
Обкладинка *Ростислава Крамара*
Комп'ютерна верстка *Зоряни Сидор*
Художній редактор *Ростислав Крамар*
Технічна редакторка *Неля Домарецька*

Підписано до друку 10.04.2024. Формат 70×100/16. Папір офсетний.
Гарнітура CentSchbook Win95BT. Друк офсетний. Умовн. друк арк. 23,40.
Умовн. фарбо-відб. 93,60. Обл.-вид. арк. 12,60. Тираж 41 689 пр. Зам. 228/04.

Видавництво «Навчальна книга – Богдан»
Свідоцтво про внесення суб'єкта видавничої справи до Державного реєстру видавців, виготівників і розповсюджувачів видавничої продукції
ДК № 4221 від 07.12.2011 р.

Навчальна книга – Богдан, просп. С. Бандери, 34а, м. Тернопіль, 46002
Навчальна книга – Богдан, а/с 529, м. Тернопіль, 46008
тел./факс (0352)52-06-07
*office@bohdan-books.com* www.bohdan-books.com

Надруковано на ПП «Юнісофт»
вул. Морозова, 13б, м. Харків, 61036
Свідоцтво серія ДК №3461 від 14.04.2009 р.