МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Н. В. Матвіїшина

ОРГАНІЗАЦІЯ ТА ОБРОБКА ЕЛЕКТРОННОЇ ІНФОРМАЦІЇ

Навчальний посібник
для здобувачів ступеня вищої освіти бакалавра спеціальності
«Комп'ютерні науки» освітньо-професійної програми «Комп'ютерні науки»

Затверджено
вченою радою ЗНУ
Протокол № __ від _____

Запоріжжя
2025

УДК 004 (076)
М337

Матвіїшина Н. В. Організація та обробка електронної інформації : навчальний посібник для здобувачів ступеня вищої освіти бакалавра спеціальності «Комп'ютерні науки» освітньо-професійної програми «Комп'ютерні науки». Запоріжжя : Запорізький національний університет, 2025. 90 с.

Навчальний посібник «Організація та обробка електронної інформації» спрямований на закріплення студентами теоретичних знань та формування практичних навичок з ефективного застосування сучасних інструментів та методів роботи з даними.

Видання охоплює ключові питання, передбачені силабусом дисципліни, зокрема: робота з офісними додатками (MS Word, MS Excel) на поглибленому рівні; освоєння основ роботи зі спеціалізованим програмним забезпеченням (зокрема, елементи програмування та обчислення в MathCAD); опанування базових принципів розробки вебсайтів засобами мови HTML та CSS; введення в основи офісного програмування, включаючи роботу з редактором VBA.

Посібник містить необхідну теоретичну інформацію, приклади практичного застосування набутих знань, а також контрольні запитання та тестові завдання для самоконтролю до кожної теми.

Навчальний посібник призначений для здобувачів ступеня вищої освіти бакалавра спеціальності «Комп'ютерні науки», але може бути корисним для студентів інших спеціальностей, які прагнуть систематизувати знання у сфері обробки електронної інформації та прикладного програмування.

Рецензент
С. І. Гоменюк, доктор технічних наук, професор, декан математичного факультету

Відповідальний за випуск
О. С. Пшенична, кандидат педагогічних наук, доцент, в.о. завідувача кафедри комп’ютерних наук

ВСТУП

Опанування сучасними цифровими технологіями та інструментами автоматизації обчислень є надзвичайно важливим у контексті підготовки фахівців у сфері комп'ютерних наук та аналізу даних.

Метою вивчення дисципліни «Організація та обробка електронної інформації» є набуття студентами знань та практичних навичок з ефективної обробки, аналізу, автоматизації та представлення електронної інформації за допомогою сучасних прикладних програмних засобів та базових інструментів програмування.

Це передбачає оволодіння інструментарієм для вирішення розрахункових та інформаційно-аналітичних завдань в професійній сфері.

Основні завдання вивчення дисципліни є:

– набуття навичок ефективної організації текстової інформації;

– опанування методів обробки та аналізу табличних даних;

– оволодіння засобами математичного моделювання (використання спеціалізованого програмного забезпечення);

– отримання базових знань з вебпредставлення інформації;

– засвоєння основ офісного програмування та автоматизації;

– розвинення алгоритмічного мислення (проєктування та реалізація алгоритмів для вирішення обчислювальних та логічних задач.

Згідно з вимогами освітньої-професійної програми здобувачі освіти мають досягти таких **компетентностей**:

– здатність до абстрактного мислення, аналізу та синтезу

– здатність застосовувати знання у практичних ситуаціях

– здатність вчитися і оволодівати сучасними знаннями;

– здатність до пошуку, оброблення та аналізу інформації з різних джерел;

– здатність до логічного мислення, побудови логічних висновків, використання формальних мов і моделей алгоритмічних обчислень, проектування, розроблення й аналізу алгоритмів, оцінювання їх ефективності та складності, розв’язності та нерозв’язності алгоритмічних проблем для адекватного моделювання предметних областей і створення програмних та інформаційних систем

– здатність використовувати сучасні методи математичного моделювання об’єктів, процесів і явищ, розробляти моделі й алгоритми чисельного розв’язування задач математичного моделювання, враховувати похибки наближеного чисельного розв’язування професійних задач

– здатність проектувати та розробляти програмне забезпечення із застосуванням різних парадигм програмування: узагальненого, об’єктно-орієнтованого, функціонального, логічного, з відповідними моделями, методами й алгоритмами обчислень, структурами даних і механізмами управління;

та програмних **результатів навчання** :

– застосовувати знання основних форм і законів абстрактно-логічного

мислення, основ методології наукового пізнання, форм і методів вилучення, аналізу, обробки та синтезу інформації в предметній області комп'ютерних наук;

– використовувати сучасний математичний апарат неперервного та дискретного аналізу, лінійної алгебри, аналітичної геометрії, в професійній діяльності для розв’язання задач теоретичного та прикладного характеру в процесі проектування та реалізації об’єктів інформатизації;

– проектувати, розробляти та аналізувати алгоритми розв’язання обчислювальних та логічних задач, оцінювати ефективність та складність алгоритмів на основі застосування формальних моделей алгоритмів та обчислюваних функцій;

– розробляти програмні моделі предметних середовищ, вибирати парадигму програмування з позицій зручності та якості застосування для реалізації методів та алгоритмів розв’язання задач в галузі комп’ютерних наук.

Структура та зміст навчального посібника відповідають силабусу дисципліни «Організація та обробка електронної інформації».

1 ТЕКСТОВИЙ ПРОЦЕСОР. ІНСТРУМЕНТИ ТА ЗАСОБИ РОБОТИ З КОМПЛЕКСНИМИ ТЕКСТОВИМИ ДОКУМЕНТАМИ

1.1 Основні можливості текстового процесора

Офісні програмні продукти Microsoft, які традиційно об'єднані в пакет Microsoft Office (або сучасну хмарну підписку Microsoft 365), складають найбільш поширений набір інструментів для роботи з документами, даними та комунікаціями (Word, Excel, PowerPoint, Outlook тощо).

Офісні програмні продукти Microsoft мають відповідний інтерфейс – стрічковий. Стрічка – графічний елемент управління у вигляді набору панелей інструментів, розміщених на декількох вкладках. Типова структура стрічки складається з великих, вкладених панелей інструментів, заповнених графічними кнопками та іншими елементами управління, згрупованими за функціональністю. Контекстними є вкладки, які з'являються, коли користувач їх потребує. Наприклад, коли користувач вибирає зображення в документі Word, може з'являтися вкладка, пов'язана саме з зображеннями, дозволяючи користувачеві взаємодіяти з вибраним зображенням.

Одним з найбільш широко використовуваних засобів організації, обробки текстових статей, ділових паперів та інших документів є текстовий процесор MS Word, інтерфейс якого представлено на рисунку 1.1.

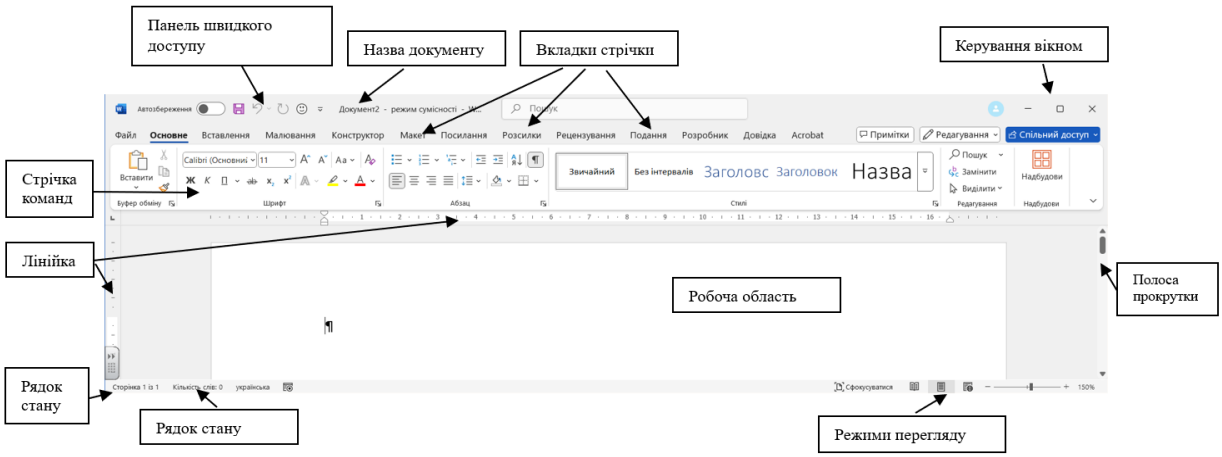


Рисунок 1.1 Інтерфейс MS Word

Microsoft Word – це багатофункціональний текстовий процесор у складі Microsoft Office, що забезпечує повний цикл роботи з документацією [1, 3, 4, 8]. Його ключові можливості включають:

– створення документів будь-якого профілю, професійних документів, незалежно від їхнього спрямування (юридичного, фінансового, науково-технічного чи художнього), завдяки широкому набору інструментів;
– проведення ефективного редагування та форматування для підвищення якості викладу та полегшення сприйняття інформації;

- інтеграція ілюстративних та графічних елементів (таблиць, діаграм, малюнків, фото) безпосередньо в текст документа;
- використання вбудованих засобів для роботи з графікою, таблицями та діаграмами, що дозволяє вирішувати значну частину завдань без залучення спеціалізованих зовнішніх програм;
- можливість збереження документів не лише у власному форматі, але й у загальнодоступних, таких як PDF або вебсторінки, для ефективного обміну та публікації;
- можливість застосування мови VBA для розробки додатків та автоматизації складних операцій через створення користувацьких екранних форм.

1.1.1 Ключові структурні елементи та об'єкти

Текстовий документ у Microsoft Word має чітку ієрархічну структуру. Він складається з різних об'єктів (елементів), які мають свої властивості та до яких можна застосовувати форматування.

Символ – найменша неподільна одиниця тексту.

Слово – послідовність символів, відокремлена від інших пробілами або розділовими знаками.

Абзац – головна структурна одиниця тексту, пов’язана смисловою єдністю і виділена виступом першого рядка, яка закінчується символом кінця абзацу ¶. Під абзацом у Word розуміють будь-яку кількість тексту і графіки (можуть існувати порожні абзаци),. До абзацу застосовуються основні параметри оформлення: вирівнювання, відступи та міжрядковий інтервал.

Розділ – частина документа, що має унікальні **параметри сторінки** (наприклад, різні поля, орієнтація – книжкова/альбомна, або унікальні колонтитули), відмінні від інших частин.

Сторінка – частина документа, що виводиться під час друку на один аркуш паперу

Документ – весь файл Word (**.docx**), що є найвищим рівнем ієрархії. Він містить усі вищеперелічені елементи та інші вбудовані об'єкти (таблиці, малюнки, діаграми).

Колонтитул – область сторінки, в якій розташовано довідкову інформацію (верхній колонтитул або нижній колонтитул).

Виноска – посилання на додаткову інформацію або джерела, що розміщуються внизу сторінки або в кінці документа/розділу.

Текстовий кадр (або **Надпис**) – це об'єкт, який дозволяє розмістити текст у будь-якому місці документа Word, незалежно від основного потоку тексту.

Шаблон Word – це спеціальний тип файлу (з розширенням **.dotx** або старіший **.dot**), який зберігає всі налаштування форматування та структуру, але при відкритті створює **новий документ** (з розширенням **.docx**).

Примітка (Коментар) – це функція, яка дозволяє додавати до документа Word зауваження, запитання, пропозиції або пояснення, не змінюючи при цьому основний текст документа.

1.1.2 Базові операції текстового процесора

Введення та редагування тексту. Введення та редагування тексту – це базові задачі, для вирішення яких призначений текстовий процесор. Після вводу тексту може виникнути необхідність редагувати його: ввести додатковий текст, видалити, перемістити або скопіювати частину вже введеного тексту, виправити слова, у яких є орфографічні помилки або пропущені букви.

Для виконання операцій з фрагментами тексту у текстовому процесорі MS Word використовуються стрічки **Основне, Вставлення.** Наприклад:

– **Основне/Виділити/Виділити все** − виділення вмісту всього документу;

– **Основне/Копіювати** (кнопка ), **Основне/Вирізати** (кнопка ), **Основне/Вставити** (кнопка ) − копіювання та переміщення фрагментів тексту за допомогою буферу обміну;

– **Вставлення/Рівняння/Вставити нове рівняння** − створення та розміщення в документі формули;

– кнопка  − скасування помилково виконаних дій.

Пошук і заміна. Текстовий процесор MS Word має механізм, що забезпечує пошук заданого слова, словосполучення, поєднання символів або тексту у заданому форматі. Текст, що знайдено, може бути автоматично замінений запропонованим фрагментом, або може бути виділеним для подальших дій користувача.

Для виконання операцій пошуку і заміни використовуються команди **Основне/Замінити** (рис. 1.2).



Рисунок 1.2 – Діалогове вікно *Пошук і замінювання*

Створення, відкриття та збереження документів. Можливість створення, відкриття та збереження документу в текстовому процесорі MS Word забезпечується меню **Файл**:

– **Файл/Створити** − створення нового документу;

– **Файл/Відкрити** − відкриття документу;

– **Файл/Зберегти, Файл/Зберегти як** − збереження документу (можливо, з іншим ім’ям або в іншій папці).

Можливість збереження документу з іншими форматами – одна з важливих функцій сучасного текстового процесору. Один і той же документ може бути потрібен користувачеві у різних представленнях, наприклад:

– *у вигляді тексту* – збереження тільки текстового вмісту;

– *у вигляді документу іншого формату* – для відкриття у текстовому процесорі іншого типу;

– *у форматі PDF* – для подальшого друку і читання в іншій операційній системі та на іншому пристрої;

– *у форматі HTML* – для публікації на вебсторінці.

Кожний документ базується на певному шаблоні. **Шаблон** визначає основну структуру документа і містить набір елементів форматування. Існують загальні шаблони і шаблони документів. До загальних належить, наприклад, шаблон *Normal.dot*, що містить настроювання, доступні для всіх документів.

Шаблон документу доступний тільки для документу, створеного на основі цього шаблону, наприклад, шаблони резюме або звітів, можна обрати командою **Файл/Створити/Обрати рекомендований шаблон** (рис. 1.3).

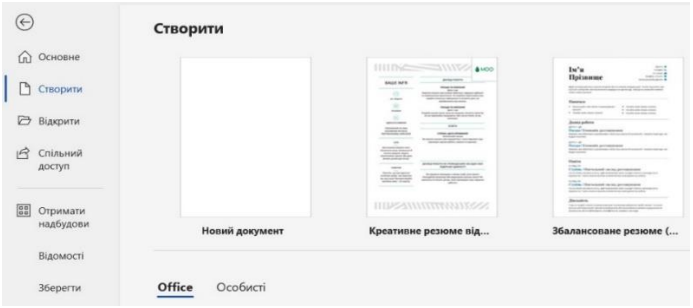


Рисунок 1.3 – Створення документу на основі шаблону

Документ Word можна відображати у *п'ятьох* режимах (стрічка **Подання**):

1) *режим розмітки сторінки* – екранне представлення документа повністю відповідає друкарському; режим зручний для більшості робіт, пов'язаних з форматуванням тексту, призначеного для друку;

2) *режим читання* – режим, призначений виключно для читання документа з екрану; редагування документа в режимі читання неможливе, але можна залишати примітки по тексту, які стають доступні в режимі розмітки;

3) *режим вебдокументу* – екранне представлення документа не відповідає друкарському; режим характерний для електронних публікацій;

4) *режим структури* – корисний, коли розробка документа починається зі створення плану змісту. При активізації цього режиму автоматично відкривається панель інструментів ***Структура***, кнопки якої дозволяють змінювати структуру документа;

5) *чернетка* – спрощена версія стандартного режиму розмітки, автор має можливість зосередитися на тексті, а не на тому, як у результаті буде виглядати документ.

1.1.3 Форматування тексту

Основними операціями форматування у MS Word є: форматування символів, форматування абзаців, створення списків.

Форматування символів. Шрифт – це повний комплект символів (літер, цифр, знаків пунктуації та спеціальних символів) певної **гарнітури** (дизайну), **розміру** та **стилю накреслення**.

Кожна **гарнітура** має свою назву, комплект шрифтів різних накреслень, насиченості, кегля і щільності, об'єднаних спільністю малюнка символів. MS Word надає для роботи кілька десятків гарнітур. Найбільш широкого поширення набули Times New Roman, Arial, Courier New.

Стиль шрифту – це графічний різновид шрифту в межах однієї гарнітури. У Word використовується три накреслення: звичайне (пряме), курсивне і напівжирне. При цьому можна використовувати поєднання напівжирний курсив.

Розмір (кегль) – це висота букви або символу. Кегль вимірюється в пунктах: 1 пт = 1/72 дюйма = 0,353 мм. 1 дюйм = 2,54 см. Мінімально можливий розмір шрифту дорівнює 1 пт, максимальний – 1638 пунктів. У редакторі Word розмір шрифту встановлюється з точністю до 0,5 пунктів. Зазвичай для оформлення основної частини тексту використовують шрифти розміром від 10 до 14 пунктів.

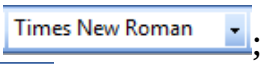
Підкреслення шрифту – виділення тексту горизонтальними лініями (прямі, пунктирні, хвилясті і ін.).

Зміна регістра – це зміна букв на рядкові та прописні.

Інтервал між символами (щільність) – відстань між символами в тексті. За замовчуванням використовується звичайний інтервал. Його можна замінити на розряджений або ущільнений.

Кернинг – зміна інтервалу між окремими парами символів.

Форматування символів здійснюється за допомогою команди **Основне/Шрифт** (рис. 1.4):

– вибір гарнітури шрифту: 

;

– вибір розміру шрифту: 

;

– вибір накреслення шрифту: 

.

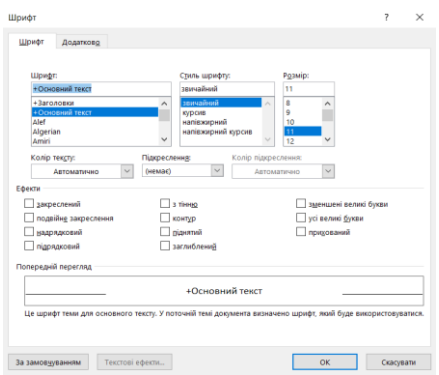
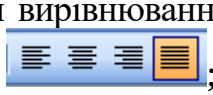


Рисунок 1.4 – Діалогове вікно *Шрифт*

Форматування абзацу – це зміна його параметрів як структурного елементу тексту. За допомогою команди **Основне/Абзац** можна змінити (рис. 1.5):

– тип вирівнювання: по правому краю, по лівому краю, по центру, по ширині ;
– відступи від полів сторінки: зліва, справа
– відступи першого рядка абзацу;
– міжрядковий інтервал: одинарний, 1,5 рядка тощо;
– відстань між абзацами;
– контроль положення абзацу на сторінці.

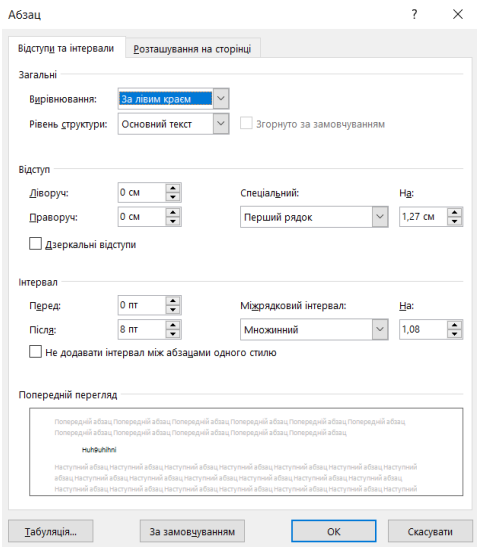


Рисунок 1.5 – Діалогове вікно *Абзац*

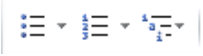
Побудова списків. Декілька послідовно розташованих абзаців можна оформити у вигляді списку: маркірованого, нумерованого, багаторівневого. Створення списку здійснюється за допомогою команди **Основне/Абзац**:  (рис. 1.6).



Рисунок 1.6 – Приклади списків

1.2 Засоби автоматизації розробки текстового документу

Інструментами, що автоматизують підготовку документа, є засоби для автоматичного створення змісту, покажчиків, списків ілюстрацій та використаної літератури, а також автоматичного форматування тексту. Однак, для того, щоб використати ці засоби, текст, що вводиться необхідно оформити відповідним стилем.

1.2.1 Стилі

Стилем називають іменований набір параметрів форматування, який застосовується до певних структурних одиниць документа. Стилі дозволяють за одну дію застосувати деякий набір елементів форматування.

Будь-який документ створюється на основі одного зі стилів, що зберігається в шаблоні цього документа. За замовчуванням використовується шаблон *Normal* і стиль *Звичайний*.

За допомогою команди **Основне/Стилі** або кнопки  вікна ***Стилі*** можна переглянути, застосувати і створити стилі: символів, абзацу, таблиці, списків. Для зміни вигляду тексту потрібно його виділити і застосувати існуючий (вбудований) стиль, вибираючи його зі списку у вікні ***Стилі*** (рис. 1.7).

Якщо потрібний стиль відсутній, його можна створити (рис. 1.8), а потім застосувати.

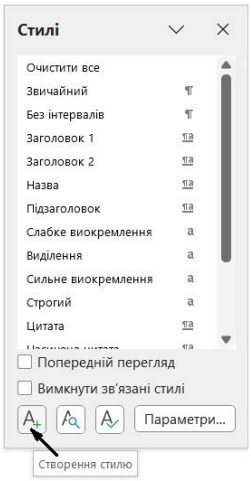


Рисунок 1.7 – Вибір стилю

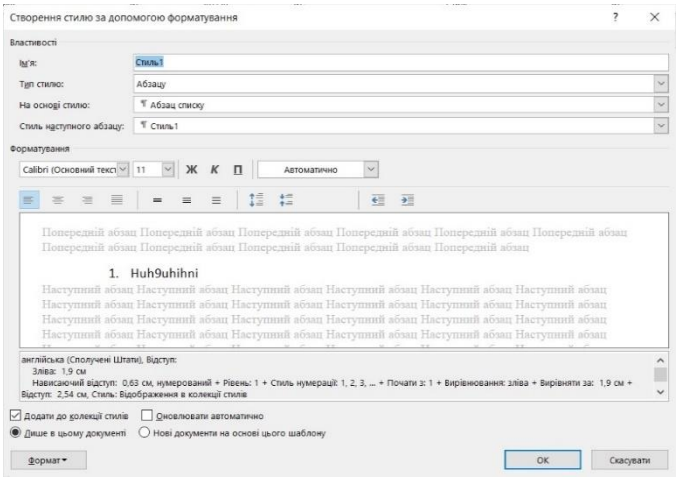


Рисунок 1.8 – Діалогове вікно *Створення стилю за допомогою форматування*

1.2.2 Поля

Поле – певна сукупність команд, реалізована спеціальними кодами, що застосовується для вставлення в документ додаткової інформації (номера сторінок, дати і т.д.). Поле можна ввести за допомогою стрічок **Вставлення, Посилання**.

Всі поля в електронному документі мають сіре заливання, тобто елемент обновлюваний. Деякі з полів обновляються автоматично, деякі – за допомогою команди контекстного меню **Оновити поле**.

Список ілюстрацій – це список, елементами якого є назви (наприклад, Рисунок 1), що додають до таблиці, рисунку, діаграми і т.д.

З метою автоматичного створення списку ілюстрацій у документі слід створити назви вищеперелічених об’єктів. Для цього виконується команда **Посилання/Вставити назву** (рис. 1.9).

Після створення назв ілюстрацій, необхідно виконати команду **Посилання/ Список ілюстрацій** (рис. 1.10), вибрати вид списку, встановити необхідні параметри і натиснути ОК.

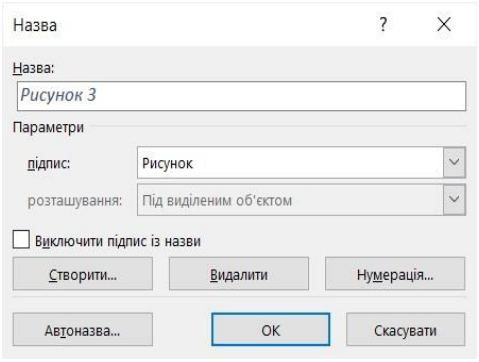


Рисунок 1.9 – Діалогове вікно *Назва*

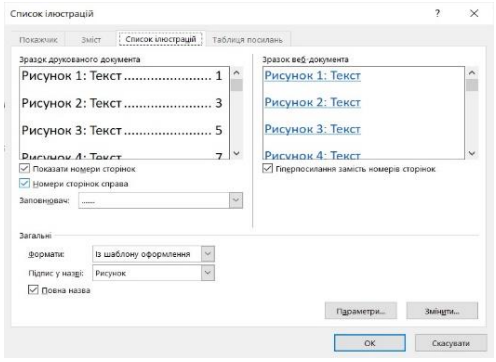


Рисунок 1.10 – Діалогове вікно *Список ілюстрацій*

Зміст – це список заголовків документа. Він використовується для перегляду тем, обговорюваних у документі, та спрощення переміщення від однієї теми до іншої безпосередньо зі змісту документа.

Найпростіший спосіб створення змісту ґрунтується на використанні стилів заголовків.

Для створення змісту всі заголовки, що становлять зміст, спочатку потрібно визначити стилем *Заголовок*. Заголовок можна створити тільки з цілого абзацу. Можна створювати до дев’яти рівнів заголовків.

Після використання команди **Посилання/Зміст/Настроюваний зміст** з’явиться відповідне діалогове вікно (рис. 1.11). У діалоговому вікні ***Параметри змісту*** вказуються стилі абзаців, з яких формується зміст та обирається рівень кожного стилю.

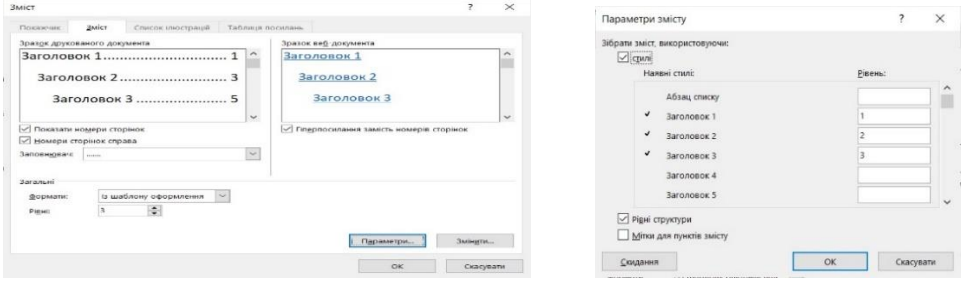


Рисунок 1.11 – Діалогові вікна *Зміст* та *Параметри змісту*

1.2.3 Таблиці

Щоб створити будь-яку таблицю, потрібно спочатку сформувати її структуру. Структура таблиці – це спосіб організації її комірок. Комірка – найменша структурна одиниця таблиці.

Є декілька способів створення таблиць (рис. 1.12):

1) перетворення тексту в таблицю; виконується за допомогою команди **Вставлення/Таблиця/Перетворити на таблицю**;

2) формування таблиці виконанням команди **Вставлення/Таблиця/ Вставити таблицю** (кнопка );

3) створення таблиці шляхом використання команди **Вставлення/ Таблиця/Накреслити таблицю** (кнопка );

4) вставлення електронної таблиці MS Excel (кнопка );

5) вставлення вбудованих таблиць виконується за допомогою команди **Вставлення/Таблиця/Експрес-таблиці.**

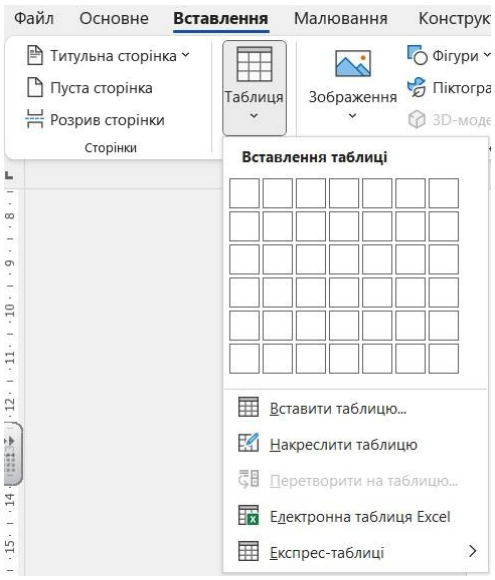


Рисунок 1.12 – Способи створення таблиць

При необхідності редагування та форматування структури та вмісту таблиць, використовуються контекстні вкладки Конструктор та Макет (рис. 1.13):

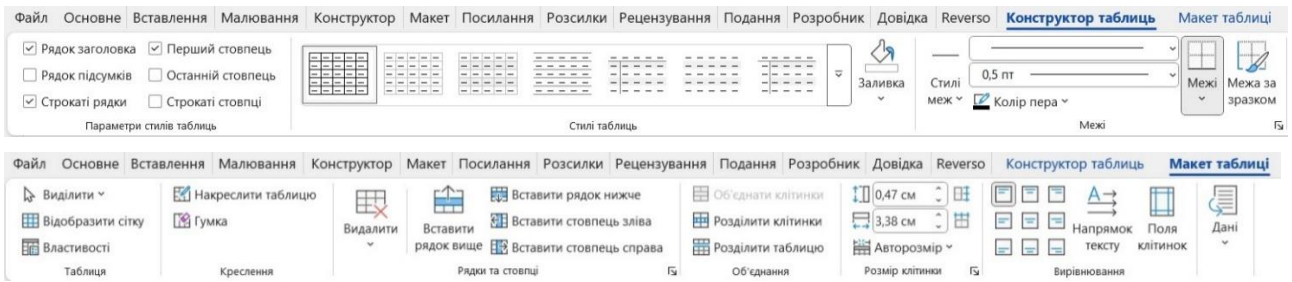


Рисунок 1.13 – Конструктор та Макет таблиць

1.2.4 Діаграми

За допомогою команди **Вставлення/Діаграма** процесор Word надає можливість представити табличні дані у вигляді діаграми, використовуючи для цього вбудований засіб *MS Graph* – редактор ділової графіки. Після виконання цієї команди в документі Word з'являється умовна таблиця MS Excel та її графічне зображення у вигляді стовпчикової діаграми (рис. 1.14).

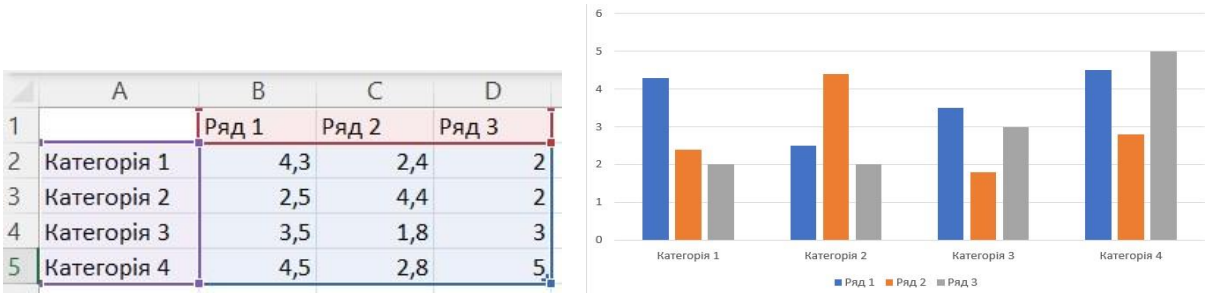


Рисунок 1.14 – Умовна таблиця MS Excel та її графічне зображення

В комірки таблиці вносять текст і числа, відповідно до тих, що необхідно відтворити на діаграмі. Побудова діаграми виконуватиметься автоматично в процесі заповнення таблиці.

Діаграму можна форматувати, використавши контекстне меню, або контекстні вкладки Конструктор, Формат під час роботи безпосередньо з діаграмою (рис. 1.15):

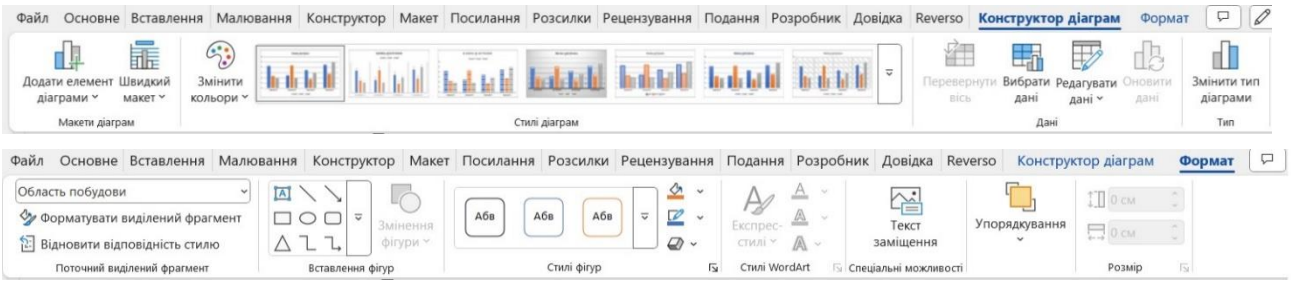


Рисунок 1.15 – Конструктор та Формат діаграми

1.2.5 Формули

В текстовому процесорі Word засобом створення математичних формул є редактор формул, для запуску якого необхідно виконати команду **Вставлення/Рівняння** (рис. 1.16). Формулу створюють, обираючи в контекстній вкладці *Структури* потрібні шаблони:

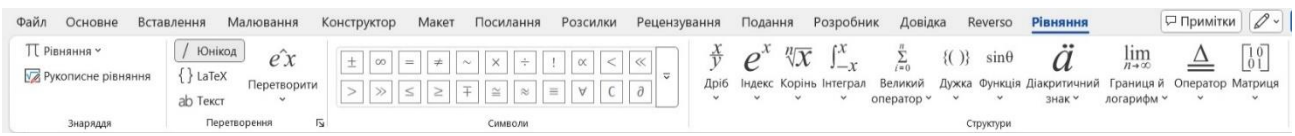


Рисунок 1.16 – Робота з формулами в MS Word

1.3 Форматування сторінок документа

1.3.1 Параметри сторінки

Форматування здійснюється автоматично з параметрами, що зберігаються в шаблоні *Normal.dotx*. Перед створенням нового документу, користувач має можливість приготувати середовище текстового процесора до роботи.

Щоб розмістити текст документа на папері, необхідно задати параметри сторінки за допомогою команди **Макет/Параметри сторінки** (рис. 1.17):

- розміри полів;
- орієнтацію сторінок документа;
- властивості сторінок;
- розмір паперу;
- графічне оформлення сторінок і т.д.

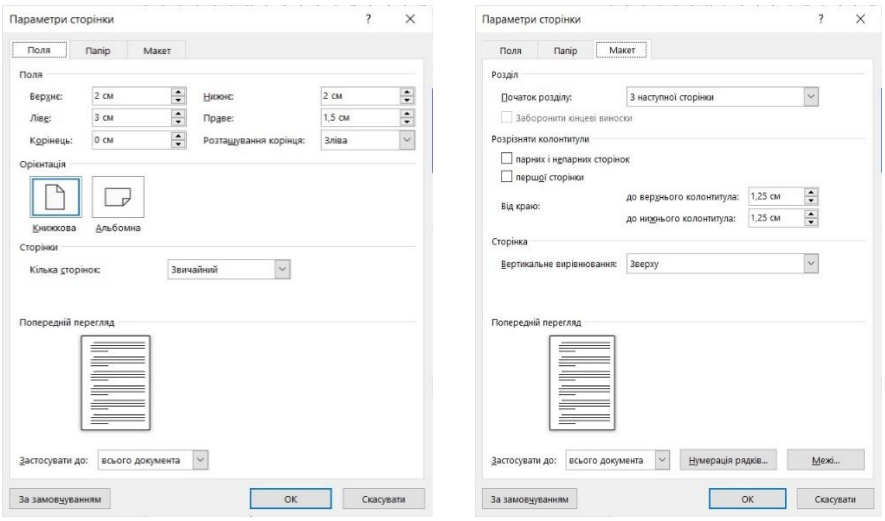


Рисунок 1.17 – Діалогове вікно *Параметри сторінки*

1.3.2 Настроювання параметрів текстового документа

Важливі параметри текстового документа встановлюють в діалоговому вікні, що викликається командою **Файл/Параметри** (рис. 1.18).

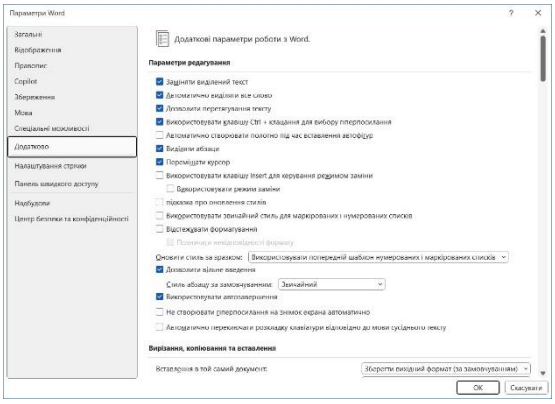


Рисунок 1.18 – Діалогове вікно *Параметри Word*

1.3.3 Розділи

Розділ – це структурна одиниця текстового документа, яка об’єднує певну його частину за змістом і зазвичай має свій заголовок. Щоб створити у тексті новий розділ, потрібно встановити текстовий курсор в кінці попереднього розділу і виконати команду **Макет/Розриви/Розриви розділів** (рис. 1.19). Новий розділ може мати власне форматування усіх рівнів.

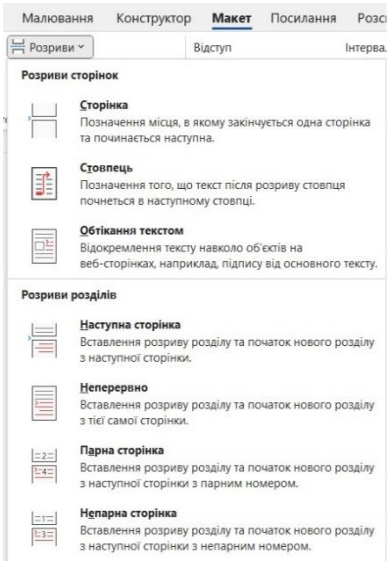


Рисунок 1.19 – Встановлення розривів сторінок та розділів

1.3.4 Виноски

Виноски – це символи, що використовуються у тексті у вигляді верхніх індексів, пов’язаних із певним, зазвичай роз’яснювальним текстом або текстом, який коментує (звичайні виноски), або як посилання на інші документи (в більшості випадків це кінцеві виноски).

Щоб додати виноску, потрібно встановити текстовий курсор у потрібному місці тексту і виконати команду **Посилання/Вставити виноску** (рис. 1.20).

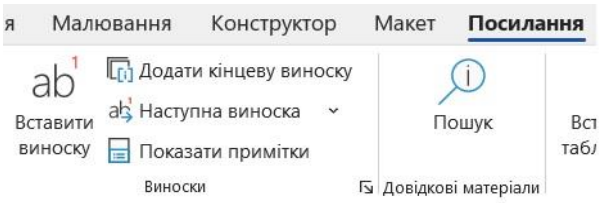


Рисунок 1.20 – Створення виноски

1.3.5 Колонтитули

Колонтитул – текст або графічні об’єкти, або те й інше, розташовані вгорі та внизу відносно основного тексту (в іншому шарі документа) і повторюються на кількох або всіх сторінках розділу чи всього документа. З метою створення колонтитулів потрібно виконати команду **Вставлення/Колонтитули** (рис. 1.21),

обрати вид колонтитулу (верхній або нижній) та, за необхідності, змінювати їх характеристики.

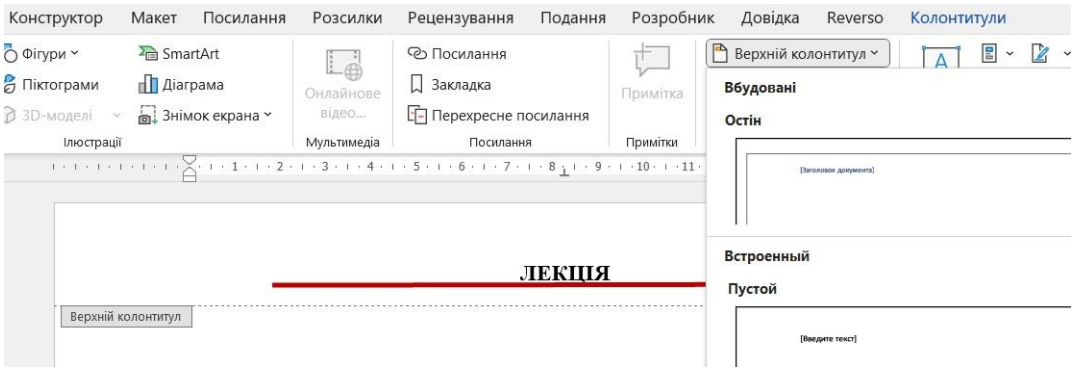


Рисунок 1.21 – Вигляд верхнього колонтитула

1.3.6 Нумерація сторінок

Перед друком документа його сторінки нумерують. Для цього використовується команда **Вставлення/Колонтитули/Номер сторінки** (рис. 1.22).

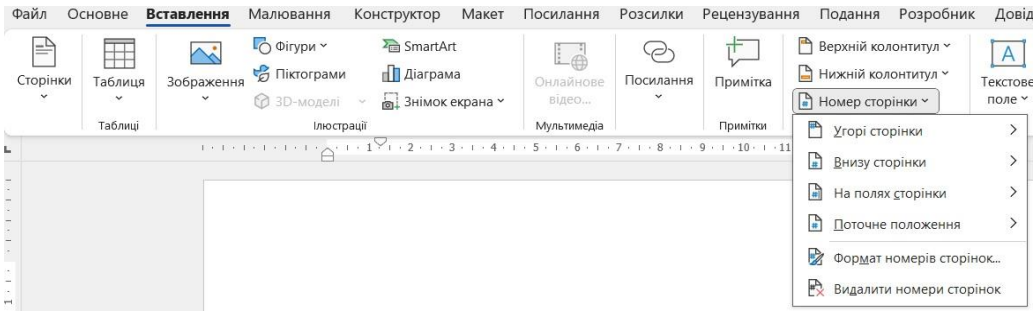


Рисунок 1.22 – Вибір нумерації сторінок

1.4 Рецензування тексту документа

Рецензування − це важливий етап роботи з документами, особливо під час спільної роботи над текстом. Стрічка **Рецензування** (рис. 1.23) у Microsoft Word містить набір інструментів, що дозволяють перевіряти, виправляти та коментувати вміст документа, зберігаючи при цьому оригінал незмінним.

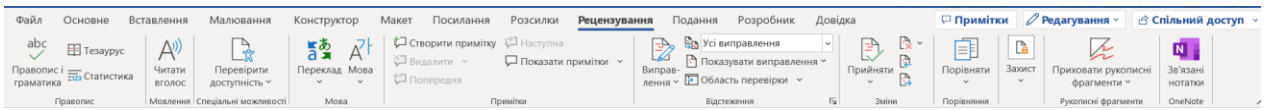


Рисунок 1.23 – Стрічка Рецензування

1.4.1 Мовні налаштування, перевірка правопису та граматики

Мова, на якій вводитиметься текст документа, обирається за допомогою команди **Рецензування/Мова** (рис. 1.24).

Для перевірки правопису вибирається команда **Рецензування/Правопис і граматика**, що запустить процедуру перевірки орфографії виділеного

фрагменту тексту, або всього документу. Розбіжності і пропоновані заміни Word дає у діалоговому вікні (рис. 1.25).

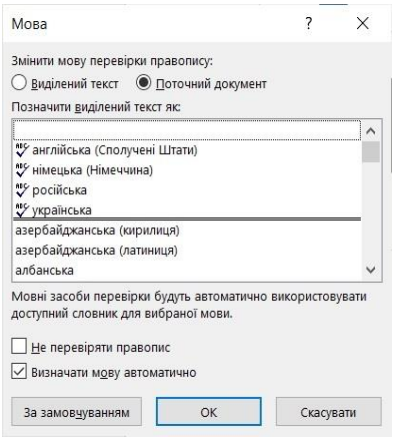


Рисунок 1.24 – Діалогове вікно *Мова*

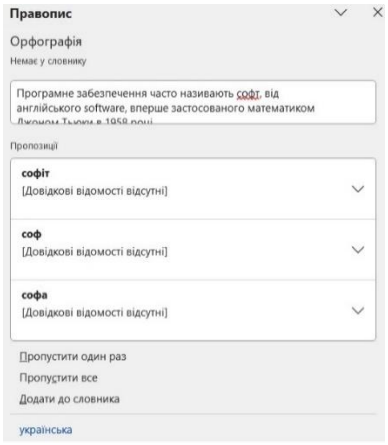


Рисунок 1.25 – Діалогове вікно *Правопис і граматика*

Тезаурус допомагає знаходити синоніми до слів. Для цього потрібно виділити слово та натиснути кнопку Тезаурус в групі *Правопис* (рис. 1.26, 1.27).

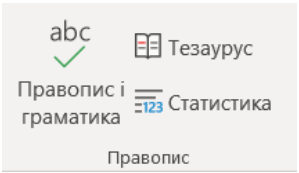


Рисунок 1.26 – Група *Правопис* на стрічці Рецензування

Статистика – інструмент для підрахунку кількості сторінок, слів, символів, абзаців і рядків у документі (рис. 2.26). Також можна дізнатися кількість символів з пробілами та без них.

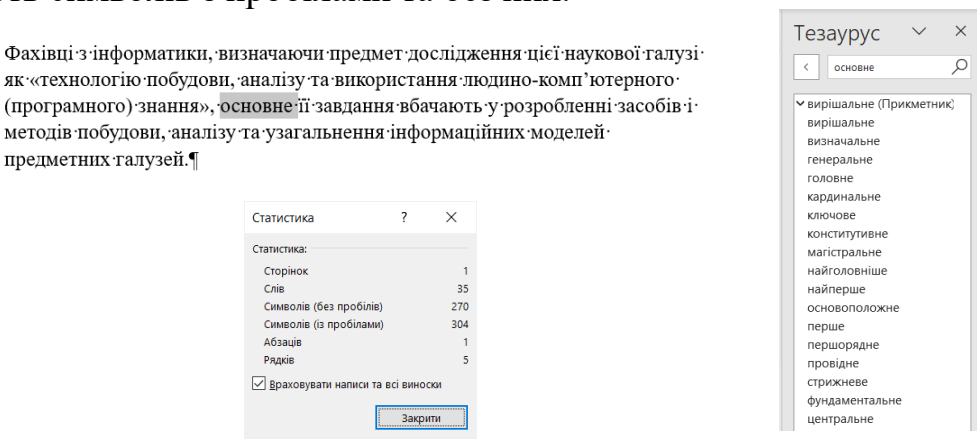


Рисунок 1.27 – Група Правопис на стрічці Рецензування

1.4.2 Переклад

Переклад – дозволяє перевести весь документ або окрему його частину (рис. 1.28). Текст відправляється в хмарну службу компанії Microsoft, а потім відкривається в уже перекладеному вигляді в окремому документ.

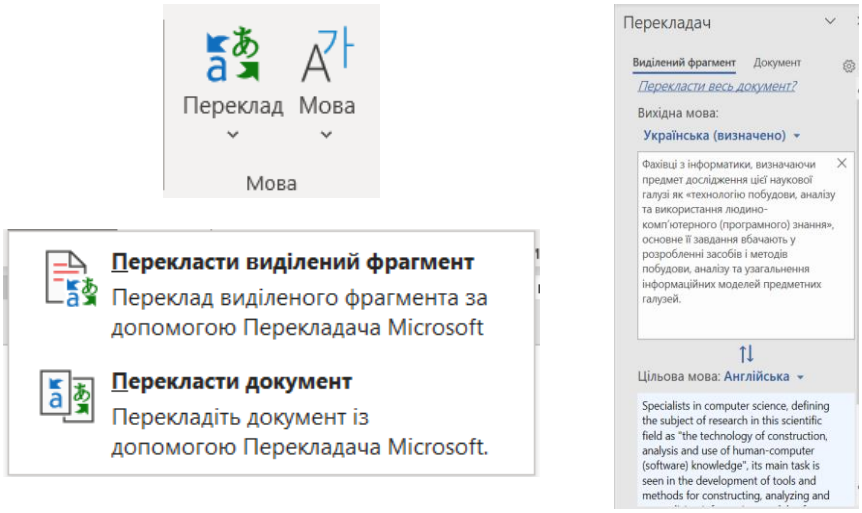


Рисунок 2.28 – Переклад тексту в MS Word

1.4.3 Примітки

Примітки – це позначки на полях, які містять в собі всі інструменти, які використовуються в редакторській або спільній роботі над документами (рис. 1.29). Це можливість вказати автору на неточності, внести зауваження, залишити побажання, підказки тощо, залишивши при цьому **незмінним початковий текст.**

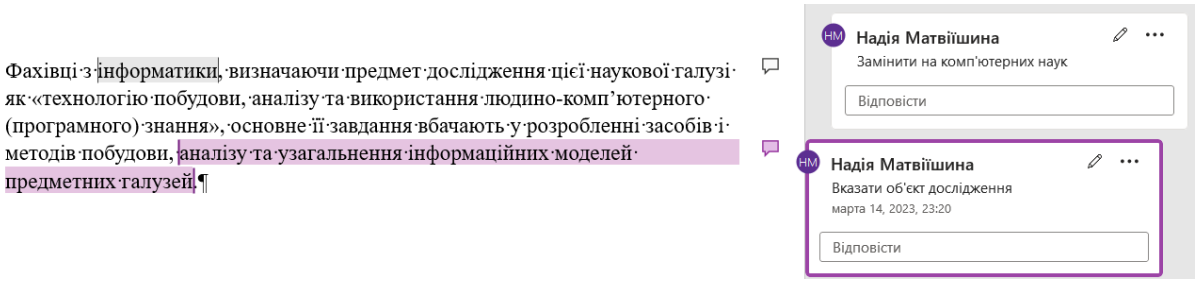


Рисунок 1.29 – Примітки при рецензуванні

1.4.5 Додавання та фіксування виправлень

Для додавання та фіксування виправлень у документі необхідно спочатку включити режим виправлень: **Рецензування/Відстеження/Виправлення** (рис. 1.30). Вихід з цього режиму – повторне натискання цієї кнопки.

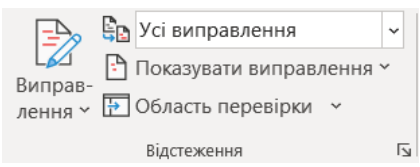


Рисунок 1.30 – Встановлення режиму Виправлення

Для налаштування параметрів виправлень при рецензуванні необхідно в блоці **Відстеження** обрати **Область перевірки/Додаткові параметри**, відкривається вікно (рис. 1.31), в якому налаштовують параметри редагування та форматування тексту й таблиць.

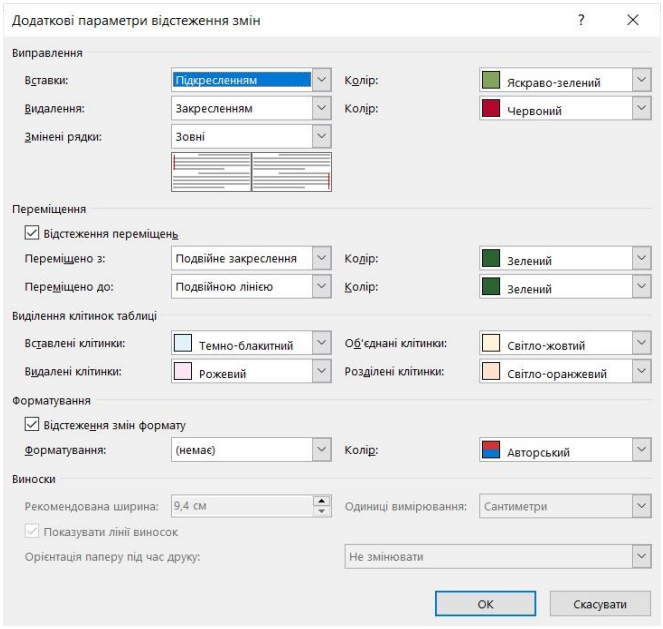


Рисунок 1.31 – Налаштування параметрів виправлень

Також користувач може управляти відображенням документу – змінений документ (без виправлень і з виправленнями) або вихідний документ (без виправлень і з виправленнями), показом правок, відображенням виносок та переглядати список рецензентів.

Внесені під час запису виправлення не видаляються з документа, тому за допомогою групи **Зміни** користувач може або прийняти, або відхилити виправлення (рис. 1.32).

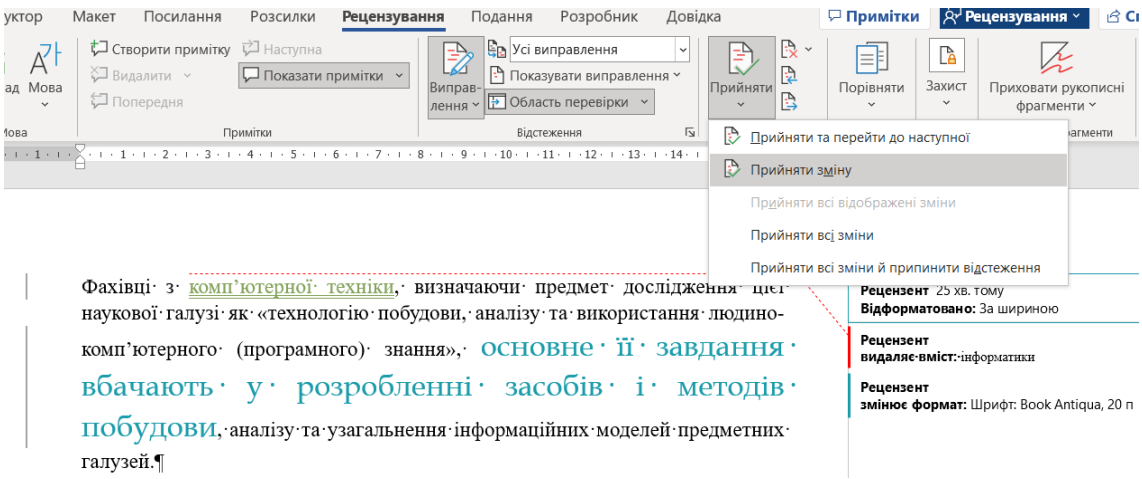


Рисунок 1.32 – Прийняття або відхилення виправлень

Користувач може порівняти різні версії документів, погодитися й об'єднати всі правки (рис. 1.33).

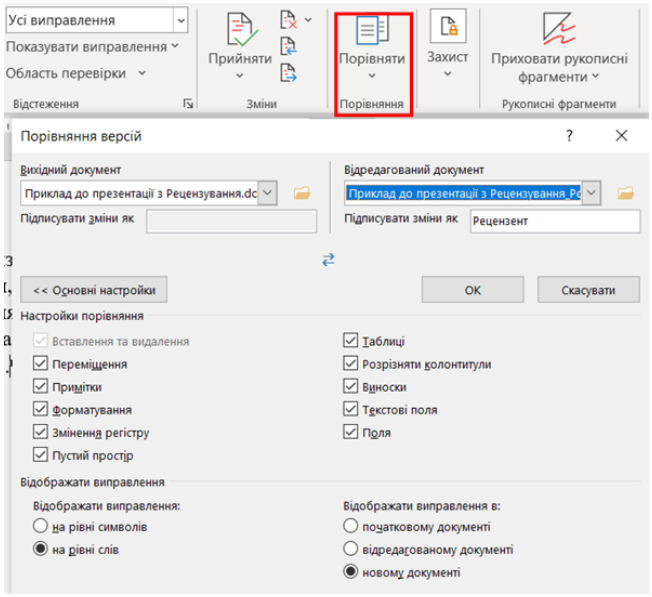


Рисунок 1.33 – Прийняття або відхилення виправлень

Дотримуючись наданого порядку дій, можна ефективно керувати правками, обговорювати зміни та фіналізувати документ разом із іншими користувачами.

Питання для самоконтролю до теми 1

1. Назвіть режими відображення документа Word і яке їх призначення.
2. Як задати параметри сторінки?
3. Як задати мову для перевірки правопису?
4. Як створити новий стиль?
5. Поняття колонтитул. Як створити колонтитул?
6. Яким чином створити в текстовому документі математичну формулу?
7. Як створити зміст документа, використовуючи засоби автоматизації Word?
8. Як побудувати діаграму на основі табличних даних?
9. Для чого використовуюся примітки під час рецензування документа?
10. Які дії необхідні для додавання та фіксування виправлень у документі під час рецензування?

Тестові завдання:

Для наведених завдань оберіть одну правильну відповідь:

1. Який елемент текстового документа є структурною одиницею, введення якої завершується натисканням клавіші Enter?

а) Символ
б) Слово
в) Абзац
г) Рядок

2. Який інструмент використовується в Word для додавання зауважень або пропозицій до конкретного фрагмента тексту без зміни основного вмісту документа?

а) Верхній колонтитул
б) Примітка
в) Виноска
г) Текстовий кадр

3. Для чого переважно використовується Шаблон Word (.dotx)?

а) Для створення резервної копії вже готового документа
б) Для збереження документа у хмарному сховищі
в) Для стандартизації структури, стилів та оформлення нових однотипних документів
г) Для перетворення документа на вебсторінку

4. Для чого в складному документі найчастіше використовують стилі Заголовок 1, Заголовок 2 та інші заголовки?

а) Виключно для зміни розміру шрифту тексту
б) Для створення підкреслених або курсивних фрагментів тексту
в) Для захисту тексту від несанкціонованого редагування
г) Для автоматичного створення змісту та навігації по структурі

5. Який елемент інтерфейсу Word містить вкладки (Головна, Вставлення, Макет тощо) та згруповані за тематикою команди?

а) Лінійка
б) Рядок стану
в) Панель швидкого доступу
г) Стрічка

6. Яка з перелічених властивостей належить до форматування символу, а не абзацу?

а) Міжрядковий інтервал
б) Відступ першого рядка
в) Вирівнювання по ширині
г) Накреслення

2 ОБРОБКА ТАБЛИЧНО-ОРГАНІЗОВАНОЇ ІНФОРМАЦІЇ

2.1 Основні поняття електронних таблиць

Для представлення даних в зручному вигляді часто використовують таблиці. Особливість електронних таблиць, у тому числі Excel, полягає в можливості використання формул для визначення зв'язків між значеннями різних комірок. Розрахунки за формулами виконуються автоматично.

Додаток Excel [1, 3, 4, 5, 8] дозволяє ефективно систематизувати дані, проводити їх автоматичну обробку, аналізувати і структурувати дані, а також представляти результат у наочному вигляді за допомогою різноманітних діаграм.

2.1.1 Робоча книга, робочий аркуш

Документ Excel називається **робочою книгою**, яка представляє набір **робочих аркушів**, що мають табличну структуру.

На перетині стовпчиків та рядків – **комірка** з відповідною адресою (наприклад, G15, A31) (рис. 2.1). Одна з комірок завжди є **активною** і виділяється рамкою активної комірки, яка в Excel є курсором. Комірка на неактивному робочому аркуші ідентифікується ім'ям аркуша та її адресою на аркуші, наприклад **Аркуш2!A1**

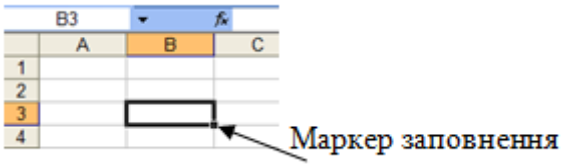


Рисунок 2.1 – Активна комірка

На дані, що розташовані у сусідніх комірках, можна посилатися у формулах як на єдине ціле. Таку групу комірок називають діапазоном (наприклад A3:F17 – суміжний діапазон, B4:D8; F5:G19 – несуміжний діапазон)

2.1.2 Форматування вмісту комірок

Окрема комірка може містити дані таких типів:

- текст;
- число;
- формула.

Тип даних, що розміщуються у комірка, визначається автоматично.

Якщо необхідно змінити формат даних у поточній комірці або діапазоні використовується команда **Основне/Формат/Формат клітинок** (рис. 2.2, рис. 2.3). Вкладки діалогового вікна **Формат клітинок** дозволяють обирати формат запису даних (кількість знаків після коми, визначення грошової одиниці, спосіб запису дати і т.д.), задавати напрям тексту і метод його

вирівнювання, визначати шрифт і накреслення символів, управляти відображенням і видом границь, задавати колір фону.

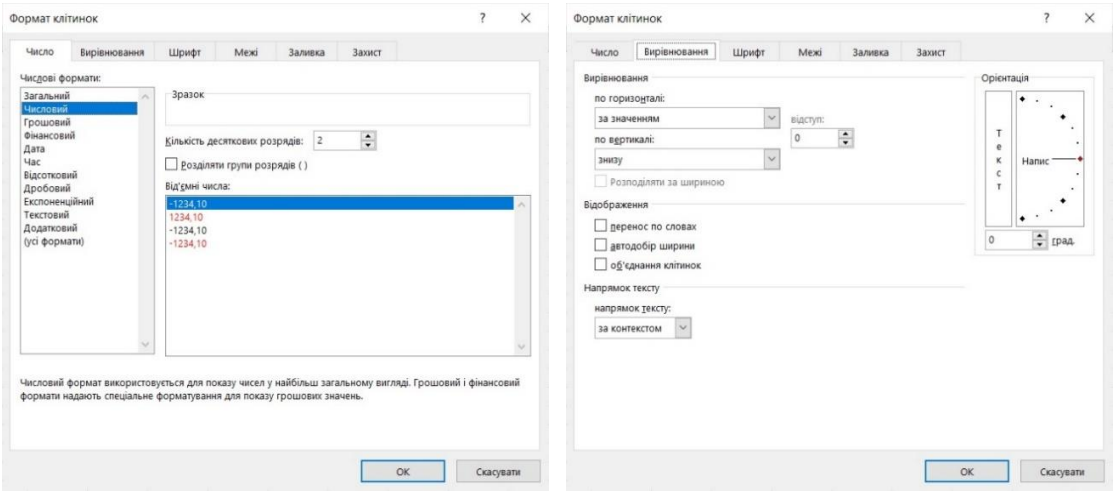


Рисунок 2.2 – Зміна формату комірки і методу вирівнювання вмісту комірки

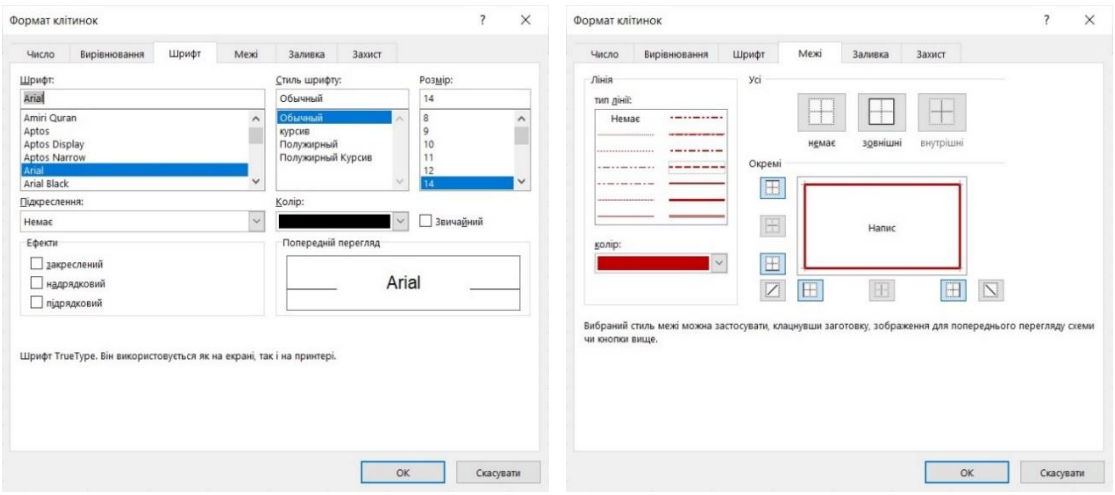


Рисунок 2.3 – Визначення шрифту, відображення границь

Приклад форматування діапазону комірок використанням вищенаведених вкладок:

	A	B	C	D
1	До форматування		Після форматування	
2	1,00		**1,00₴**	
3	2,00		**2,00₴**	
4	2,00		**2,00₴**	
5	5,00		**5,00₴**	
6	6,00		**6,00₴**	
7				

2.2 Обчислення в Excel

2.2.1 Формули, функції

Обчислення в таблицях Excel здійснюються за допомогою формул. **Формула** Excel завжди починається з символу **=**, і може містити числові константи, посилання на комірки і функції Excel, з'єднані знаками

математичних операцій. Дужки дозволяють змінювати стандартний порядок виконання дій. Якщо комірка містить формулу, на робочому аркуші відображається поточний результат обчислення цієї формули. Якщо зробити комірку поточною, то формула з'явиться у рядку формул (рис. 2.4)

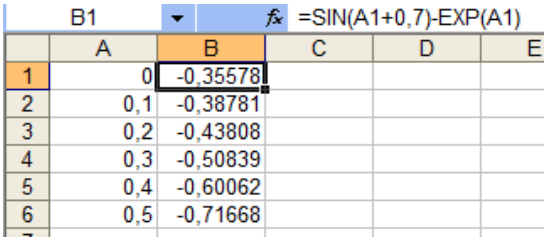


Рисунок 2.4 – Приклад формули

Стандартні (вбудовані функції) Excel призначені для виконання різноманітних обчислювальних та логічних процедур.

Всі функції мають однаковий стандартний формат:

Ім'я_функції(аргумент_1;аргумент_2;…аргумент_n)

В якості аргументів функції, як правило, виступають адреси комірок, що містять відповідні значення. Якщо в якості аргументу функції виступає інша функція, то говорять про **вкладену функцію**

Функції в Excel поділяються на категорії: математичні, статистичні, фінансові, дати і часу, текстові, логічні та ін. Також їх поділяють на такі види: без аргументу, з одним аргументом, з фіксованою кількістю аргументів, з невизначеною кількістю аргументів, з необов'язковими аргументами.

Суттєву допомогу користувачеві в процесі конструювання формул, може надати **Майстер функцій** (рис. 2.5): команда **Формули/Вставити функцію** або кнопка f_x у рядку формул.

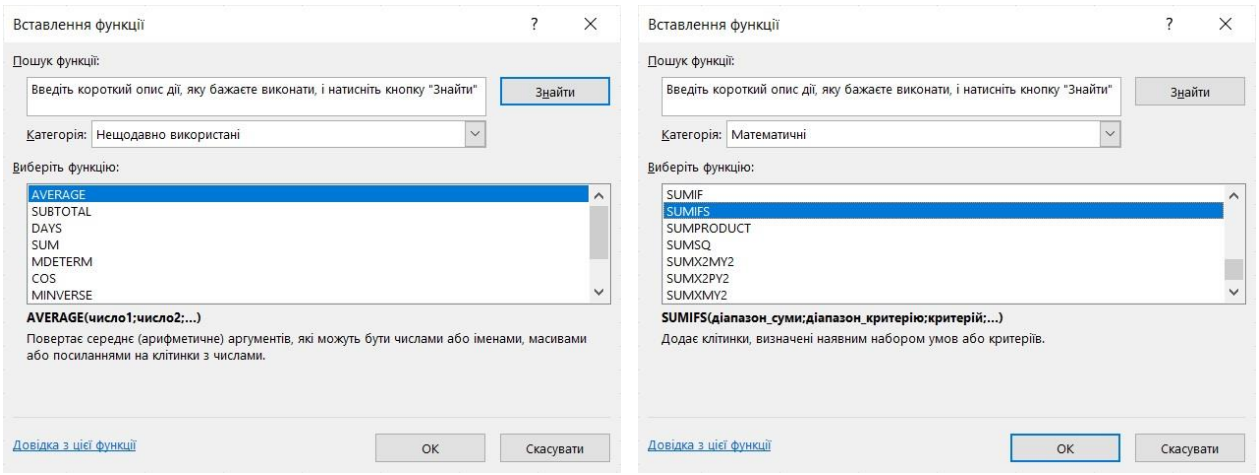


Рисунок 2.5 – Діалогові вікна Майстра функцій

На наступному кроці Майстра функцій з'являється діалогове вікно (рис. 2.6) для роботи з обраною функцією, наприклад, з функцією SUMIF:

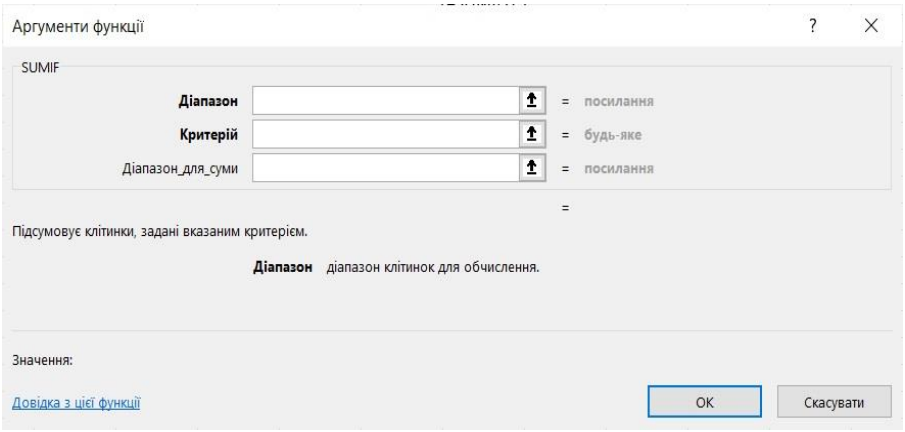


Рисунок 2.6 – Діалогове вікно «Аргументи функції»

В формулах Excel використовуються різноманітні **математичні** функції. Використання цих функцій дозволяє прискорити та спростити процес обчислень. В якості аргументів більшості функцій виступають числові значення, які можна представити в вигляді математичних виразів з використанням констант і посилань.

У табл. 2.1 представлено записи основних математичних функцій в Excel.

Таблиця 2.1 – Запис математичних функцій в Excel

Функція в Excel	**Математична функція**
ABS(число)	$\lvert число \rvert$
ACOS(число)	$arccos(число)$
ASIN(число)	$arcsin(число)$
ATAN(число)	$arctg(число)$
COS(число)	$cos(число)$
COSH(число)	$ch(число)$
EXP(число)	$e^{число}$
LN(число)	$ln(число)$
LOG(число;основа)	$\log_{a} число$
LOG10(число)	$lg(число)$
SIN(число)	$sin(число)$
SINH(число)	$sh(число)$
TAN(число)	$tg(число)$
SIGN(число) або ЗНАК(число)	$sign(число)$
SQRT(число) або КОРЕНЬ(число)	$\sqrt{число}$
POWER(число;ступінь) або СТЕПЕНЬ(число;ступінь)	$число^{n}$

У табл. 2.2 представлено приклади формул для обчислення математичних виразів в Excel. В записі формули Excel замість змінної ***x*** використано посилання на комірку **A2**.

Таблиця 2.2 – Приклади формул

Математичний запис	**Формула в Excel**
$\|x\|$	=ABS(A2)
$\sin x$	=SIN(A2)
$\cos(x-2)$	=COS(A2-2)
tgx	=TAN(A2)
$ctgx=\frac{\cos x}{\sin x}$	=COS(A2)/SIN(A2)
$\arccos x$	=ACOS(A2)
e^{5x-2}	=EXP(5*A2-2)
$\ln\|x-4\|$	=LN(ABS(A2-4))
$\sqrt{1-\cos^2(x)}=(1-\cos^2 x)^{\frac{1}{2}}$	=(1-COS(A2)^2)^(1/2) або =SQRT(1-COS(A2)^2)
$e^{\cos x \sin x+5}$	=EXP(COS(A2)*SIN(A2)+5)

Для проведення логічних обчислень в Excel використовують шість логічних функцій: IF, OR, AND, TRUE, FALSE, NOT.

Логічна функція **IF** (або **ЕСЛИ)** має наступний синтаксис:

IF (лог_вираз; значення_якщо_TRUE; значення_якщо_FALSE)

і повертає одне значення. Якщо **лог_виразу** – істина, то це результат обчислення **значення_якщо_TRUE**, у противному випадку це – **значення_якщо_FALSE**. Наприклад, на рисунку 2.7 представлено результат обчислення значення функції:

$$y=\begin{cases}\sin x, & \text{если } x>4\\ \cos x, & \text{если } x\le 4\end{cases}.$$

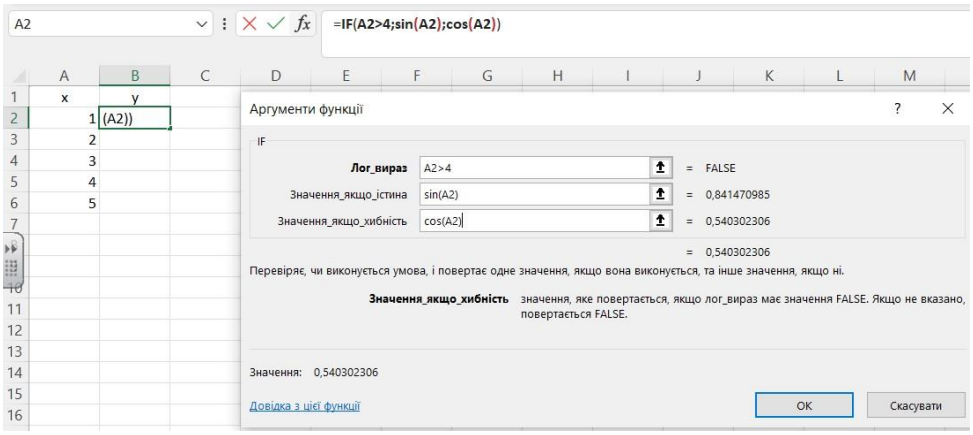


Рисунок 2.7 – Приклад використання функції IF

2.2.2 Посилання на комірки

Формула може містити **посилання**, тобто адреси комірок, вміст яких використовується в обчисленнях. Це означає, що результат обчислення формули залежить від числа, що знаходиться в іншій комірці.

Види посилань:

1. **Відносні** посилання (рис. 2.8, рис. 2.9). При копіюванні формули адреси в посиланнях автоматично змінюються відповідно до відносного розташування вихідної комірки і копії що створюється. Посилання **A1, A2, A3, B1, B2, B3, C1, D1** – відносні.

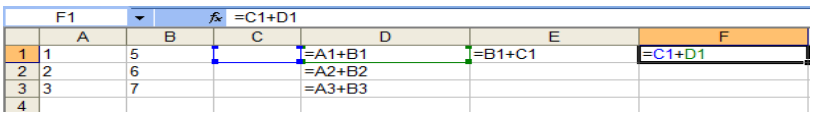


Рисунок 2.8 – Приклад відносних посилань

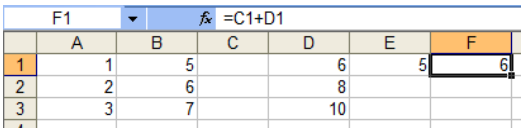


Рисунок 2.9 – Приклад роботи формули з відносними посиланнями

2. **Абсолютні** посилання (рис. 2.10, рис. 2.11). При абсолютній адресації адреси посилань при копіюванні не змінюються. Елементи номера комірки, що використовують абсолютні посилання, випереджаються символом $. **$A$1** – абсолютне посилання на комірку **A1**. Excel дозволяє використовувати в якості абсолютного посилання на комірку не тільки її адресу, але і привласнене цій комірці ім'я. Для присвоєння імені активній комірці виконується команда **Формули/Визначити ім'я**.

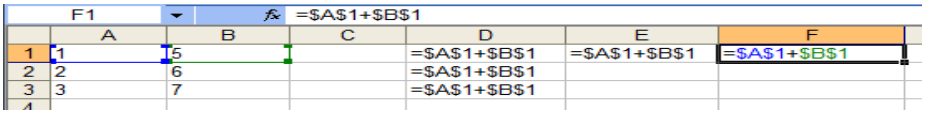


Рисунок 2.10 – Приклад абсолютних посилань

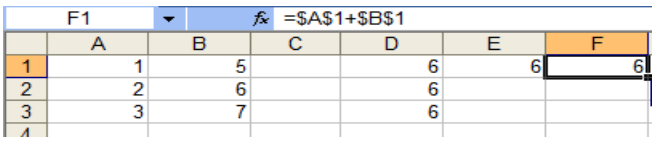


Рисунок 2.11 – Приклад роботи формули з абсолютними посиланнями

3. **Змішані** посилання (рис. 2.12, рис. 2.13). При копіюванні формули змінюється адреса або рядка, або стовпця. Наприклад, посилання **A$5**, **$A1** – змішані

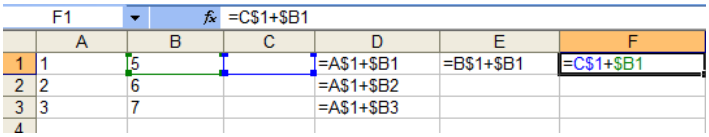


Рисунок 2.12 – Приклад змішаних посилань

F1 =C$1+$B1

	A	B	C	D	E	F
1	1	5		6	10	5
2	2	6		7		
3	3	7		8		

Рисунок 2.13– Приклад роботи формули зі змішаними посиланнями

Для зміни способу адресації при редагуванні формули треба виділити посилання на комірку і натиснути клавішу **F4**. Наприклад, при послідовному натисканні клавіші **F4** посилання на комірку **A1** буде записуватися: **A1**, **A1**, **A$1**, **$A1**.

2.3 Графічне представлення даних в Excel

В Excel термін діаграма використовується для позначення усіх видів графічного представлення числових даних. Побудова графічного зображення робиться на основі *ряду даних*. Так називають групу комірок із даними в межах окремого рядка або стовпця. На одній діаграмі можна відображати декілька рядів даних.

Категорії діаграми задають положення відповідних значень у ряді даних.

2.3.1 Побудова діаграм

Для побудови діаграм використовується команда **Вставлення/Діаграми**. Перед тим, як будувати діаграму можна виділити відповідний діапазон, але задати цю інформацію можна й в процесі роботи над діаграмою.

Процес побудови діаграми складається з декількох кроків, які можна здійснити за допомогою контекстних вкладок Конструктор, Формат (рис. 2.14):

– вибір типу і виду діаграми;

– зміна або вибір діапазону даних, на основі яких буде побудовано діаграму, визначення способу формування рядів діаграми;

– вибір оформлення діаграми;

– вибір розміщення діаграми (на поточному або на окремому аркуші).

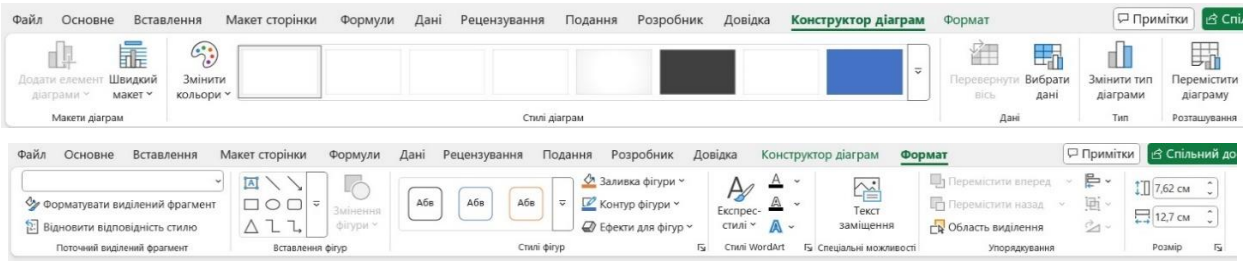


Рисунок 2.14 – Контекстні вкладки для побудови та оформлення діаграми

2.3.2 Редагування діаграм

Готову діаграму можна змінити. Вона складається з окремих елементів (рис. 2.15), таких, як самі графіки (ряди даних), осі координат, заголовок діаграми, область побудови, заголовки осей, легенда, підписи даних. При

натисканні на елемент діаграми, він виділяється маркерами, а при наведенні на нього курсору миші – описується спливаючої підказкою.

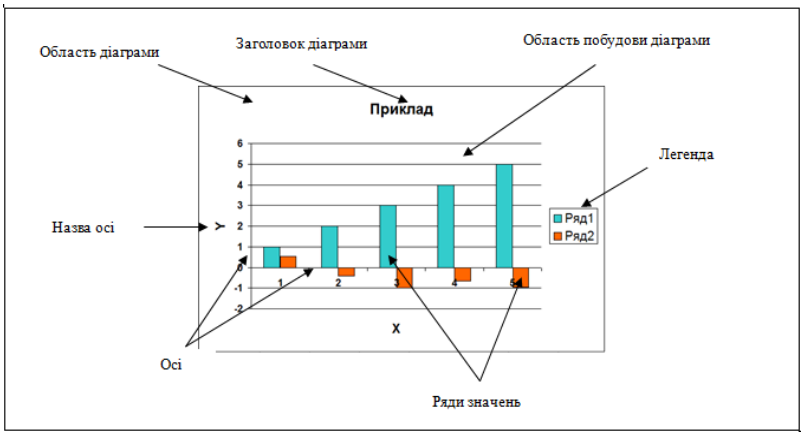


Рисунок 2.15 – Області діаграми

На діаграму можна додавати потрібні елементи та форматувати їх (рис. 2.16).

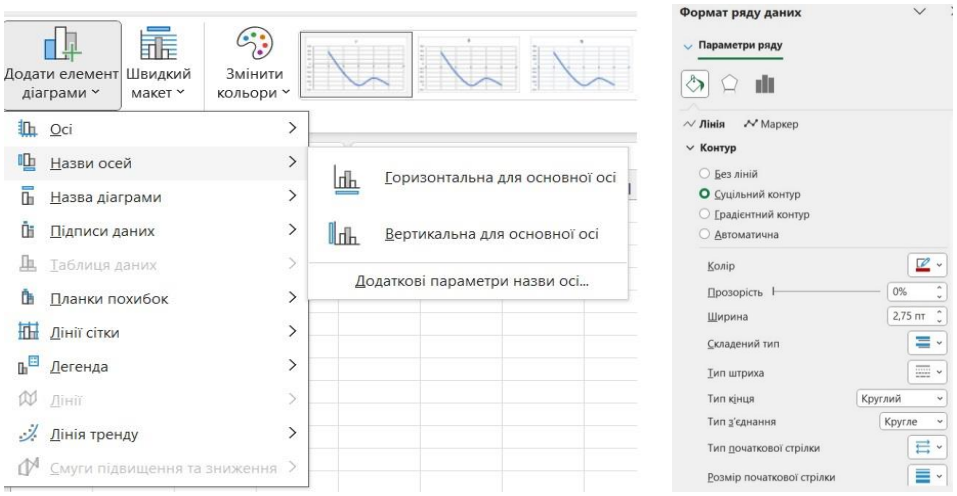


Рисунок 2.16 – Форматування елементів діаграми

2.4 Розв’язання задач зі змінними в Excel

Підбір параметра та **Пошук рішення** – це два вбудовані інструменти Excel, які дозволяють автоматизувати розв’язання задач, де потрібно знайти оптимальне значення або значення, яке задовольняє певним умовам.

2.4.1 Інструмент Добір параметра

Інструмент **Добір параметра** Excel – це потужний засіб, який дозволяє знайти конкретне значення в одній комірці, яке призведе до певного результату в іншій комірці. Іншими словами, він допомагає «підібрати» таке значення для однієї змінної, щоб отримати бажаний результат у формулі.

Щоб підібрати потрібне значення комірки, необхідно:

– вибрати команду **Дані/Підбір параметра** – відкриється діалогове вікно ***Підбір параметра*** (рис. 2.17);

– у поле введення «**Установити у клітинці:**» ввести посилання на комірку, що містить формулу;

– у поле введення **Значення:** ввести необхідне значення;

– у поле введення **Змінюючи значення клітинки:** ввести посилання на комірку, значення якої треба підібрати;

– натиснути **ОК**.

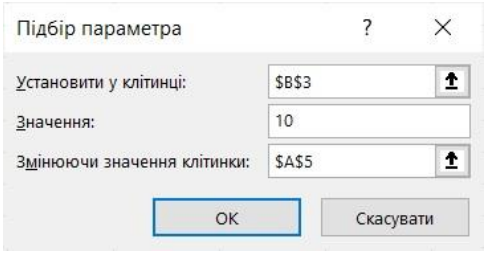


Рисунок 2.17 – Діалогове вікно Підбір параметра

2.4.2 Надбудова Пошук рішення

Надбудова **Пошук рішення** Excel є інструментом, який дозволяє автоматично знаходити оптимальне значення для формули в одній комірці (цільова функція) за певних обмежень, що накладаються на значення в інших комірках.

Для того щоб застосувати **Пошук рішення** необхідно сформулювати задачу в математичному вигляді (скласти математичну модель). Необхідно визначити цільову комірку, комірки, значення яких змінюються, і ті обмеження, що потрібно враховувати в процесі рішення.

Цільова комірка (або цільова функція) – це комірка робочого аркуша, для якої потрібно знайти максимальне, мінімальне або задане значення.

Комірки, що змінюються, – це комірки, від яких залежить значення цільової комірки. Пошук рішення підбирає значення комірок, що змінюються, доти, поки не буде знайдено рішення.

Обмеження – це значення комірки, яке повинно знаходитися у визнаних межах та задовольняти цільовому значенню. Обмеження можна накладати на цільову комірку і на комірки, що змінюються.

Для запуску надбудови **Пошук рішення** необхідно:

– виконати команду **Дані/Розв'язувач** – відкриється діалогове вікно ***Параметри розв'язувача*** (рис. 2.18);

– у полі **Оптимізувати цільову функцію:** задати посилання на комірку (що обов'язково повинна містити формулу), значення якої потрібно змінити в процесі пошуку рішення;

– поставити перемикач «**До:**» в положення:

Максимум – для пошуку максимально можливого значення;

Мінімум – для пошуку мінімально можливого значення;

Значення – для пошуку значення, рівного числу, що варто ввести в поле;

– ввести у поле **Змінюючи клітинки змінних:** посилання на комірки або діапазони комірок, значення яких варто змінювати в процесі пошуку рішення;

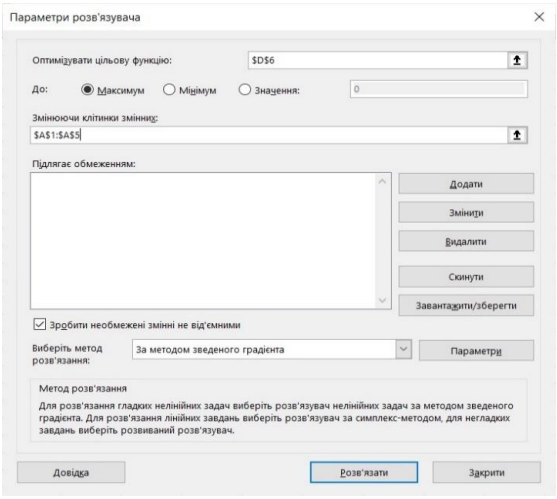


Рисунок 2.18 – Діалогове вікно Параметри розв’язувача

– для введення обмежень на значення натиснути кнопку **Додати**:

а) у діалоговому вікні ***Додати обмеження*** (рис. 2.19) ввести потрібні посилання у поле **Посилання на клітинку:**;

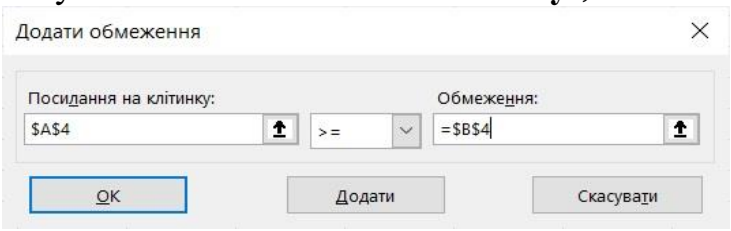


Рисунок 2.19 – Діалогове вікно Додати обмеження

б) зі списку, що розкривається, (>=, =, <=, целый, двоич) вибрати потрібний знак;

в) ввести обмеження у поле **Обмеження:** для значень комірок;

г) якщо ще потрібно додати обмеження, то натискають кнопку **Додати**;

д) якщо всі обмеження задані то натискають кнопку ОК;

– у діалоговому вікні ***Параметри розв’язувача*** натиснути кнопку **Розв’язати**.

Слід додати що **Пошук рішення** може працювати з кількома змінними одночасно, на відміну від інструменту **Підбір параметра**.

2.5 Обробка статистичних даних в Excel

2.5.1 Описова статистика

Описова статистика (Descriptive Statistics) – це сукупність методів, що використовуються для узагальнення, організації та опису основних характеристик набору даних. В контексті електронної інформації, застосування описової статистики дозволяє швидко оцінити ключові параметри великих масивів даних (табл.. 2.3).

Таблиця 2.3 – Основні вибіркові статистичні характеристики, які обчислюються в MS Excel

Характеристика	Призначення	Функція MS Excel
Середнє арифметичне	Міра центральної тенденції, що показує "середнє" значення вибірки	СРЗНАЧ / AVERAGE
Медіана	Значення, що ділить упорядковану вибірку навпіл (50% значень менші, 50% – більші)	МЕДІАНА / MEDIAN
Мода	Значення, яке найчастіше зустрічається у вибірці	МОДА.ОДН / MODE.SNGL
Дисперсія	Міра розкиду даних, показує, наскільки значення відхиляються від середнього	ДИСП.В / VAR.S
Середньоквадратичне відхилення	Стандартизована міра розкиду, корінь квадратний з дисперсії (вимірюється в тих самих одиницях, що й дані)	СТАНДВ.В / STDEV.S
Середнє арифметичне	Міра центральної тенденції, що показує "середнє" значення вибірки	СРЗНАЧ / AVERAGE

Для обчислення багатьох характеристик одночасно доцільно використовувати надбудову **Пакет аналізу (Data Analysis)**, а саме інструмент **Описова статистика (Descriptive Statistics)**.

2.5.2 Кореляційний аналіз

Кореляційний аналіз досліджує взаємозв'язок (ступінь залежності) між двома або більше змінними.

Коефіцієнт кореляції Пірсона (r) – це числове значення в діапазоні від -1 до +1, яке описує силу та напрямок лінійного взаємозв'язку між двома змінними (табл. 2.5)

Таблиця 2.5 – Властивості коефіцієнта кореляції

Значення r	Інтерпретація
r = +1	Ідеальна пряма (позитивна) кореляція
0 < r < +1	Пряма (позитивна) кореляція (зі зростанням однієї змінної зростає інша)
r = 0	Лінійний зв'язок відсутній
-1 < r < 0	Обернена (негативна) кореляція (зі зростанням однієї змінної інша спадає)
r = -1	Ідеальна обернена (негативна) кореляція

В MS Excel для обчислення коефіцієнта кореляції використовується функція **КОРЕЛ (CORREL)** або інструмент **Кореляція** з надбудови **Пакет аналізу**.

2.5.3 Лінійна регресія та прогнозування

Регресійний аналіз використовується для моделювання зв'язку між змінною, яку потрібно спрогнозувати (залежна змінна Y), та однією або декількома незалежними змінними (факторами X).

Лінійна регресія описується рівнянням прямої:

$$Y = a + bX$$

де

Y – прогнозоване значення (залежна змінна).

X – незалежна змінна (фактор).

B – **Нахил (Slope)**: показує, на скільки одиниць зміниться Y при зміні X на одну одиницю. Обчислюється функцією **НАКЛОН** (SLOPE).

a – **Відрізок (Intercept)**: точка перетину лінії регресії з віссю Y (значення Y, коли X=0). Обчислюється функцією **ОТРЕЗОК** (INTERCEPT).

Для безпосереднього прогнозування значення Y за заданим значенням X на основі лінійної регресії використовується функція: **ПРОГНОЗ.ЛІНІЙНИЙ (FORECAST.LINEAR)**.

Коефіцієнт детермінації ($\mathbf{R^2}$) – це значення в діапазоні від 0 до 1, яке вказує, яка частка дисперсії залежної змінної (Y) пояснюється моделлю регресії.

Якщо $\mathbf{R^2} \to 1$ (зокрема, в межах 0,9–1,0), це свідчить про високу адекватність моделі та можливість її використання для прогнозування.

Цей показник автоматично відображається при побудові лінії тренду на діаграмі (рис. 2.20).

2.5.4 Типи ліній тренду для моделювання даних

MS Excel підтримує кілька типів ліній тренду (**Trendlines**), які дозволяють моделювати різні види залежностей між даними, що відображаються на діаграмі (рис. 2.20). Вибір правильного типу лінії тренду критично важливий для точного прогнозування (табл. 2.6).

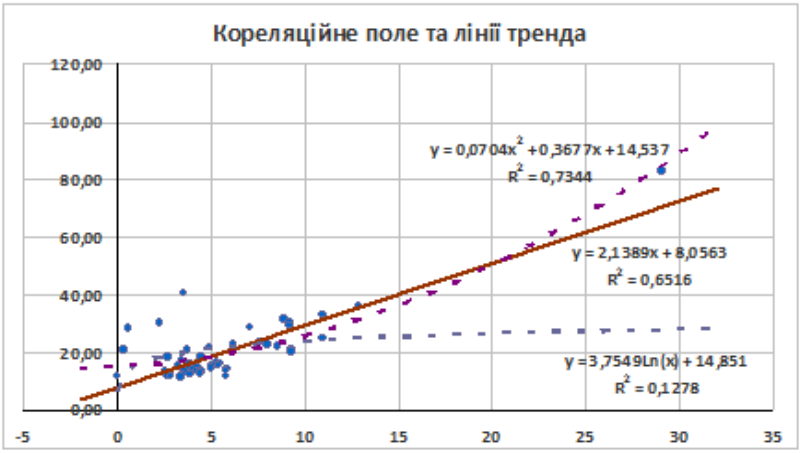


Рисунок 2.20 – Використання ліній тренду

Таблиця 2.6 – Типи ліній тренду

Тип лінії тренду	Вид залежності	Рівняння моделі	Призначення та особливості
Лінійна (Linear)	Проста лінійна залежність	$Y = a + b \cdot X$	Використовується, коли дані зростають або спадають із постійною швидкістю. Найпростіша та найпоширеніша
Логарифмічна (Logarithmic)	Різке зростання або спадання на початку, що потім вирівнюється	$Y = a \cdot \ln(X) + b$	Підходить для даних, де швидкість зміни швидко зростає або спадає, а потім стабілізується
Поліноміальна (Polynomial)	Дані з коливаннями (піками та спадами)	$Y = b_1X + b_2X^2 + \ldots. + a$	Використовується для моделювання складних нелінійних залежностей
Степенева (Power)	Залежність, коли швидкість зростання або спадання змінюється прискорено	$Y = a \cdot X^b$	Використовується для даних, які порівнюють вимірювання, що зростають з певною швидкістю (наприклад, продажі відносно зростання населення). Вимагає позитивних даних
Експоненційна (Exponential)	Різке зростання або спадання зі швидкістю, що постійно збільшується або зменшується	$Y = a \cdot e^{b \cdot X}$	Часто використовується для моделювання приросту, що прискорюється (наприклад, зростання інвестицій)

При виборі нелінійної лінії тренду слід звертати увагу на коефіцієнт детермінації (R^2). Чим він ближчий до 1, тим краще обрана модель описує реальні дані, і тим точнішим буде прогноз.

Питання для самоконтролю до теми 2

1. Що таке робочий аркуш, комірка, діапазон?
2. Як відтворити формули в комірках?
3. Як задати абсолютне посилання на комірку?
4. Яке призначення має кнопка  ?
5. Чи змінюються відносні та абсолютні посилання при копіюванні формули?
6. Які є різновиди числових форматів у MS Excel?
7. Який синтаксис має вбудована функція Excel?

8. Яким чином можна побудувати діаграму?
9. Які задачі можна розв’язувати, використовуючи надбудову **Пошук рішення**?
10. Які типи ліній тренду використовуються для моделювання даних?

Тестові завдання:

Для наведених завдань оберіть одну правильну відповідь:

1. Що є основним призначенням табличного процесора Microsoft Excel?

а) Створення та форматування багатосторінкових текстових документів
б) Керування базами даних та генерація звітів
в) Обробка та аналіз числових даних, виконання обчислень, побудова діаграм
г) Створення та редагування відеопрезентацій

2. Яку назву (адресу) має комірка, розташована на перетині п'ятого рядка та другого стовпця?

а) A5
б) 5B
в) B5
г) 5A

3. Що відбувається з відносним посиланням (наприклад, A1) у формулі, коли цю формулу копіюють до іншої комірки?

а) Посилання залишається незмінним (A1)
б) Посилання перетворюється на абсолютне (A1)
в) Посилання автоматично змінюється відповідно до нового розташування
г) Excel видає помилку копіювання

4. Як називається функція, яка дозволяє автоматично застосовувати форматування до комірок (наприклад, змінювати колір тла), якщо їхні значення відповідають певній умові (наприклад, більше 100)?

а) Перевірка даних
б) Умовне форматування
в) Валідація формули
г) Фільтрація

5. Для чого використовується інструмент Фільтр на робочому аркуші?

а) Для сортування даних за кількома стовпцями одночасно
б) Для зміни формату дати та часу
в) Для приховування всіх числових даних і відображення лише тексту
г) Для відображення лише тих рядків даних, які відповідають заданим критеріям

6. Який тип має файл макросів (файл, що містить код Visual Basic for Applications) у Excel?

а) .xlsx
б) .xlsm
в) .xltx
г) .csv

3 ОСНОВНІ ІНСТРУМЕНТИ ТА ЗАСОБИ РОБОТИ З МАТЕМАТИЧНИМ ПАКЕТОМ MATHCAD

3.1 MathCAD та Mathcad Prime

Програмні комплекси Mathcad та Mathcad Prime є не просто обчислювальними інструментами, а інтегрованими середовищами для документування, моделювання та аналізу даних, які відповідають вимогам інженерних та наукових стандартів. Mathcad (і його сучасна версія Prime) забезпечує повний цикл роботи з даними: від коректного введення та перевірки даних до складної обробки, візуалізації та формування остаточного, повністю задокументованого інженерного звіту. Mathcad та Mathcad Prime є унікальними інструментами, оскільки вони дозволяють обробляти дані безпосередньо у формі інженерного або наукового документа.

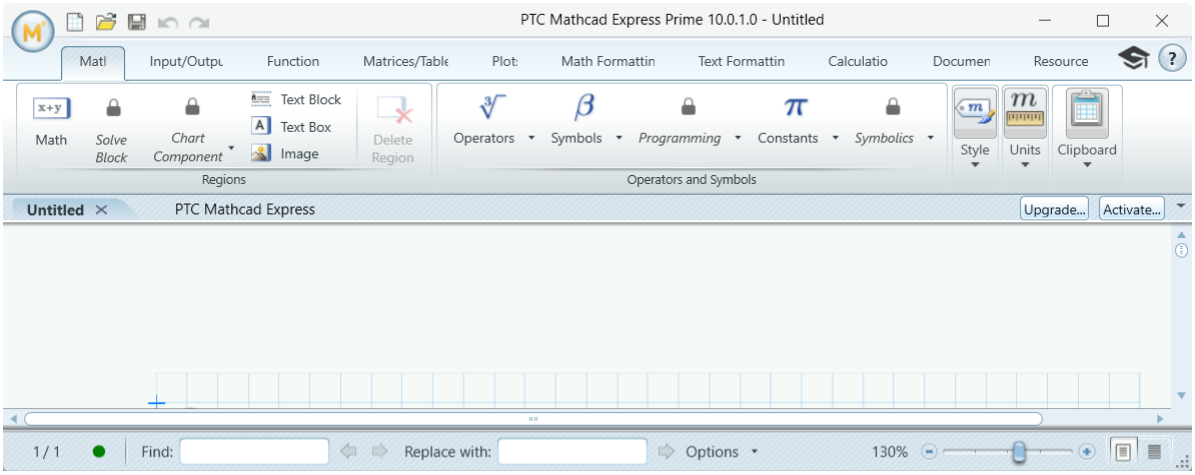


Рисунок 3.1 – Вікно MathCAD Prime

Щоб активувати повнофункціональну 30-денну версію Mathcad, необхідно:

- натиснути «Активувати» у верхній правій частині екрану;
- запросити 30-денну повнофункціональну версію;
- створити обліковий запис PTC.COM;
- вибрати продукт та отримати ліцензію.

В даному навчальному посібнику розглядаються деякі аспекти роботи з класичним Mathcad.

3.2 Інтерфейс користувача математичного пакету MathCAD

Класичний Mathcad (до версії 15) – це історично потужний інструмент, який досі має своїх прихильників завдяки деяким унікальним можливостям програмування та більш вільному формату документа

Додаток MathCAD [2, 3] – один з найбільш популярних комп‘ютерних математичних пакетів. З його допомогою можна вирішувати найрізноманітніші математичні задачі і оформляти результати розрахунків на високому професійному рівні. За допомогою цього пакета здійснюються не

тільки прості і допоміжні обчислення але і доволі складні розрахунки і наукові дослідження, що використовують комбінації різних чисельних алгоритмів і аналітичних перетворень.

Основні елементи вікна MathCAD представлено на рисунку 3.2.

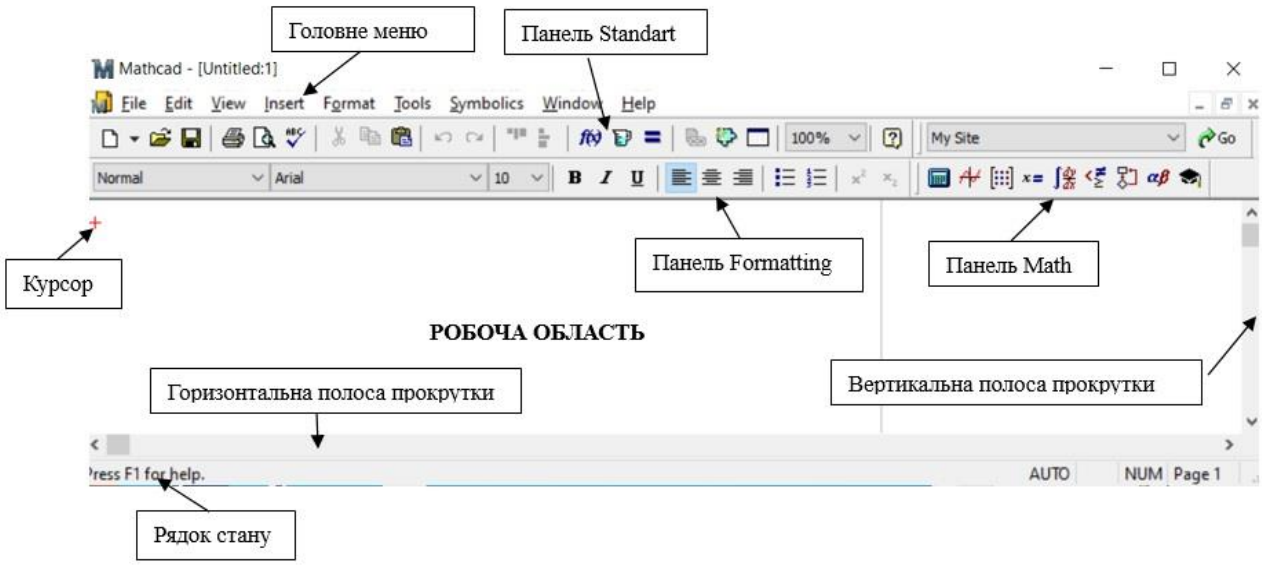


Рисунок 3.2 – Елементи вікна MathCAD

Головне меню MathCAD містить команди, які об‘єднані у відповідності з функціями, які виконуються:

File – загальні команди роботи з документом (створення, відкриття, зберігання, друк тощо);

Edit – команди редагування документу (копіювання, видалення, вставки тощо);

View – команди управління виглядом інтерфейсу (показ, приховування панелей інструментів, зміна масштабу тощо)

Insert – команди вставки в документ різних об‘єктів (графіки, таблиці, функції тощо);

Format – команди управління форматом відображення всіх об‘єктів MathCAD;

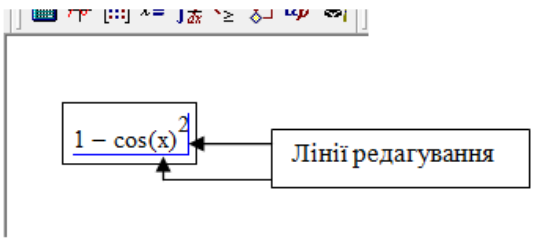
Tools – допоміжні команди (перевірка орфографії, зміна налаштувань документу тощо);

Symbolics – команди, які управляють символьними обчисленнями;

Window – стандартні команди управління вікнами відкритих документів;

Help – довідкова система MathCAD.

Робоча область і курсор документу MathCAD. Робоча область займає більшу частину вікна MathCAD і саме тут можна вводити формули, графіки та інші об‘єкти MathCAD. Курсор показує те місце на сторінці документу, де буде починатися наступне введення інформації. В процесі введення формули курсор MathCAD набуває вигляд двох ліній синього кольору.



Панелі інструментів MathCAD. Панелі інструментів призначені для швидкого виконання різних команд. Показати або приховати панелі можна за допомогою команди **View/Toolbars** (рис. 3.3).

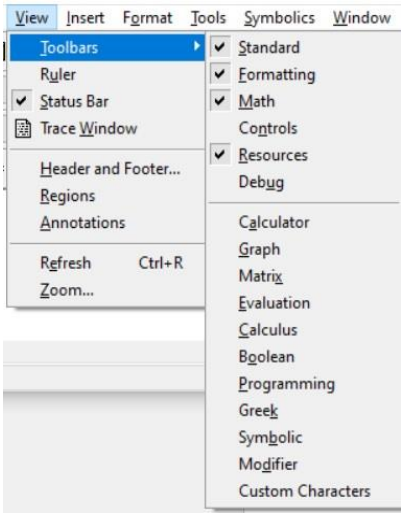


Рисунок 3.3 – Меню панелей інструментів

За замовчуванням в MathCAD показано декілька панелей інструментів: **Formatting**, **Standard**, **Resources**, а також панель **Math,** яка містить посилання на панелі зі специфічними елементами.

Панель інструментів Math. Панель Math призначена для відтворення панелей інструментів, які необхідні для роботи з формулами, графіками тощо (рис. 3.4, Рисунок , рис. 3.6).

Панель **Calculator** – введення в документ основних математичних символів та операцій.

Панель **Graph** – вставка в документ шаблонів для графіків різного вигляду.

Панель **Matrix** – операції з векторами і матрицями.

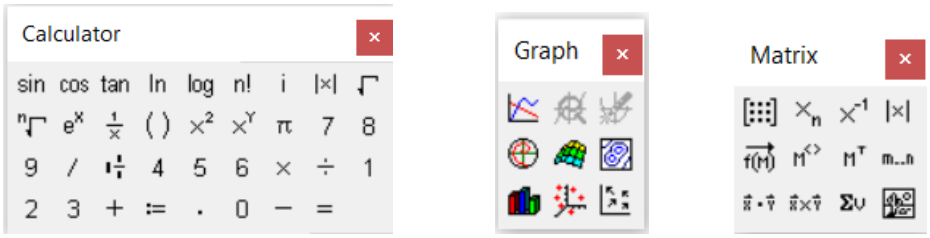


Рисунок 3.4 – Вигляд панелей інструментів **Calculator**, **Graph**, **Matrix**

Панель **Evaluation** – доступ до команд, які управляють процесом обчислення (присвоювання значень, виведення результатів розрахунків).

Панель **Calculus** – вставка в документ шаблонів диференціювання, інтегрування тощо

Панель **Boolean** – доступ до команд, які задають рівняння, нерівності, а також логічні операції.

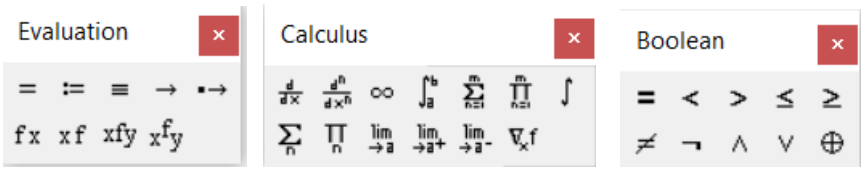


Рисунок 3.5 – Вигляд панелей інструментів **Evaluation**, **Calculus**, **Boolean**

Панель **Programming** – доступ до операторів, необхідних для створення програмних модулів.

Панель **Greek** – вставка в документ грецьких символів.

Панель **Symbolic** – вставка в документ ключових слів для управління символьними операціями.

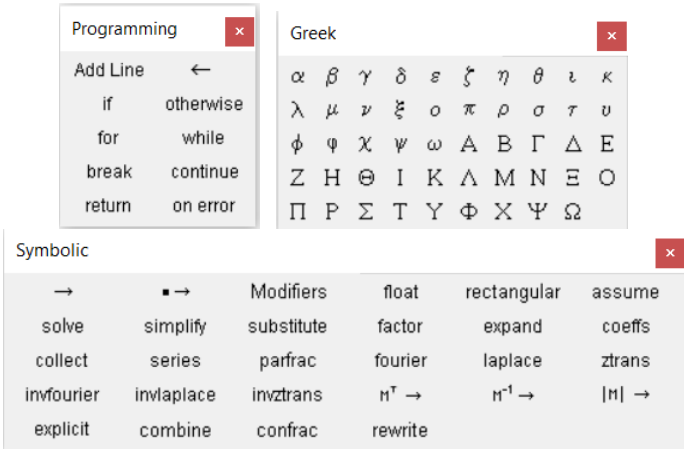


Рисунок 3.6 – Вигляд панелей інструментів **Programming**, **Greek**, **Symbolic**

3.3 Введення та виведення інформації

3.3.1 Створення формул

Виконання обчислень – це основне призначення пакету MathCAD. Кожному математичному оператору відповідає та або інша кнопка на панелях математичних інструментів або клавіатурне поєднання для швидкого вводу з клавіатури.

Прості арифметичні вирази можуть бути введені безпосередньо з клавіатури, як показано на рисунку 3.7. Наприклад, необхідно ввести вираз $\frac{1}{3}+5\cdot\left(\frac{2}{5}-7\right)$ та отримати результат обчислень.

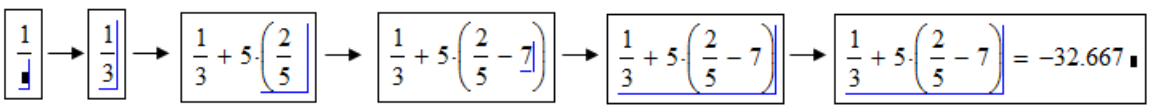
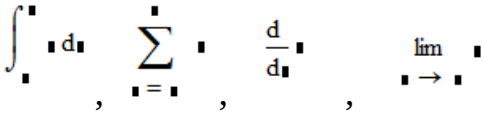
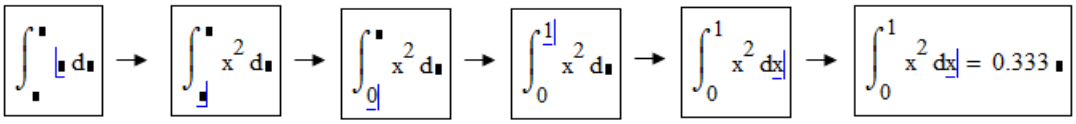


Рисунок 3.7 – Введення простих арифметичних виразів

Для створення складних математичних конструкцій використовуються шаблони відповідного оператора, які обираються на панелі інструментів **Calculus**:



Після вставки шаблона в документ необхідно заповнити місцезаповнювачи відповідними значеннями:



3.3.2 Вставка функцій

Крім операторів, вирази, що обчислюються можуть містити функції. В MathCAD є дві групи функцій:

1) функції користувача;

2) вбудовані функції.

Функція характеризується своїм іменем та аргументами (змінні, що вказуються в дужках):

```
ім‘я_функції(змінна_1, змінна_2, …, змінна_n)
```

Загальний вигляд функції користувача:

```
ім‘я_функції(змінна_1, змінна_2, …, змінна_n):= вираз
```

Функцію користувача можна, наприклад, задати таким чином:

$a := 1 \qquad b := 3$

$f(x,y) := a \cdot \sin(x) + b \cdot \cos(y)$

$f(0,0) = 3$ значення фунції при x=0 та y=0

Вбудовані функції – це функції, що задані в MathCAD за замовчуванням. Для того, щоб їх використати, достатньо коректно ввести ім‘я функції з клавіатури, скористатися панеллю **Calculator** для вставки найбільш поширених функцій, або скористатися бібліотекою функцій MathCAD (команда **Insert**/**Function**) (рис. 3.8).

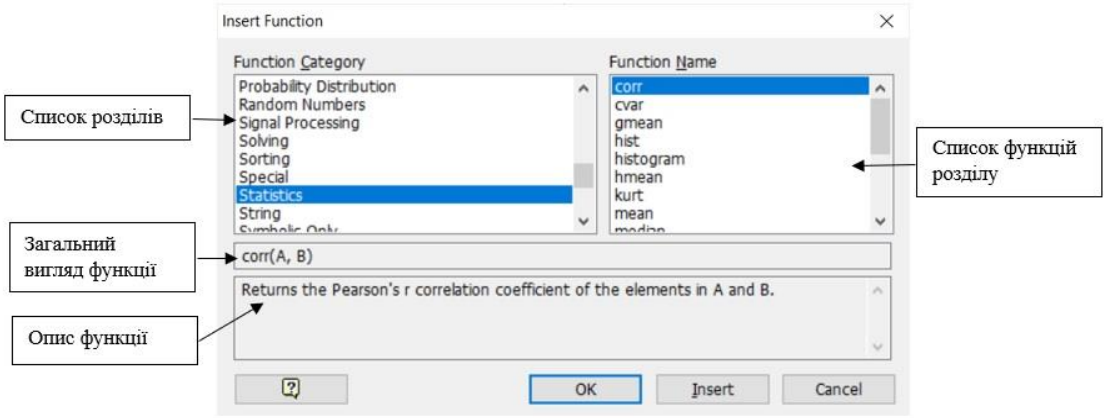
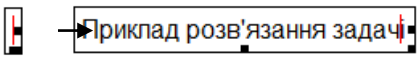


Рисунок 3.8 – Діалогове вікно для вставки вбудованої функції

Слід зазначити, що імена функцій користувача не повинні співпадати з іменами вбудованих функцій.

3.3.3 Створення тексту

Для того, щоб створювати будь-які коментарі, а також якісно оформлювати задачі, що розв‘язуються, корисно використовувати текстовий режим MathCAD. Щоб ввести текстову область, можна використати комбінацію клавіш Shift+ «’», або команду **Insert/Text Region**. Текст розміщується в області, що спеціально створено:



3.3.4 Введення/виведення даних в текстові файли

Часто аналіз даних пов‘язаний з імпортом даних із зовнішніх джерел та експортом на зовнішні носії. Для роботи із зовнішніми файлами даних в MathCAD передбачено декілька функцій (рис. 3.9, рис. 3.10):

`READPRN(file)` – читання даних в матрицю з текстового файлу;

`WRITEPRN(file)` – запис даних в текстовий файл;

`APPENDPRN(file)` – дозапис даних в текстовий файл, що вже існує;

`READFILE(file, type,[colwidths, rows, cols, emptyfill])` – читання даних з файлу (більш універсальна функція):

`file` – назва файлу;

`type` – тип файлу (“`delimited`”, “`fixed`”, “`Excel`”);

`colwidths` – ширина стовпця даних, якщо зчитування з файлу проводиться з параметром “`fixed`” (з фіксованою шириною);

`rows` – початковий рядок імпорту даних;

`cols` – початковий стовпець імпорту даних;

`emptyfill` – значення, яке використовується для заміни даних, що відсутні.

x := READPRN("d:\mnv\z_f.txt")

$x^T = (4.2 \quad 4.4 \quad 5.5 \quad 4.7 \quad 5.09 \quad 6.2)$

p := READFILE("d:\mnv\z_f.txt" , "delimited")

$p^T = (4.2 \quad 4.4 \quad 5.5 \quad 4.7 \quad 5.09 \quad 6.2)$

k := (90) → значення повинно бути вектором або матрицею!!!

APPENDPRN("d:\mnv\z_f.txt") := k

y := READPRN("d:\mnv\z_f.txt")

$y^T = (4.2 \quad 4.4 \quad 5.5 \quad 4.7 \quad 5.09 \quad 6.2 \quad 90)$

Рисунок 3.9 – Використання функцій READPRN, READFILE, APPENDPRN

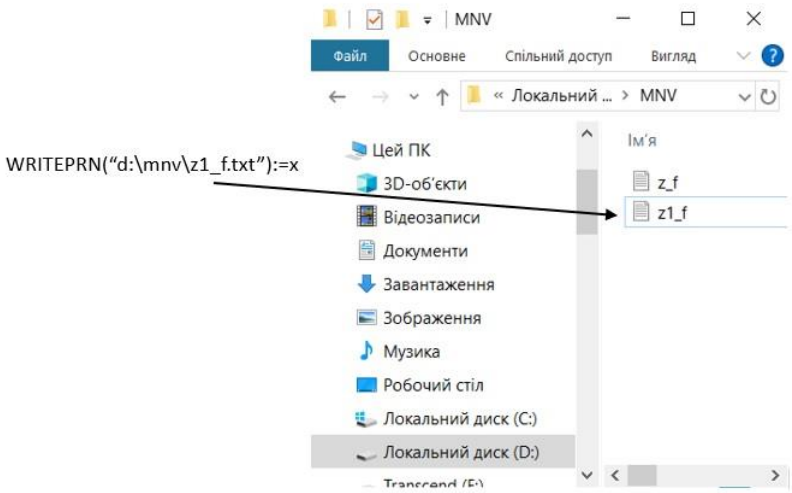


Рисунок 3.10 – Використання функції WRITEPRN

3.4 Типи даних. Обчислення в MathCAD

В MathCAD існує три типи даних:
1) числа (дійсні та комплексні);
2) рядки (будь-який текст в лапках);
3) масиви (матриці, вектори, таблиці, тобто будь-яка впорядкована послідовність елементів)

3.4.1 Змінні

Змінна – це іменований об'єкт, має певне значення, яке може змінюватися в процесі виконання програми.

Для того, щоб визначити деяку змінну, необхідно використати символ :=, або символ ≡ з панелі інструментів **Evaluation** (рис. 3.11). Якщо присвоювання здійснюється за допомогою :=, то змінна – **локальна**, якщо за допомогою ≡, то змінна – **глобальна**.

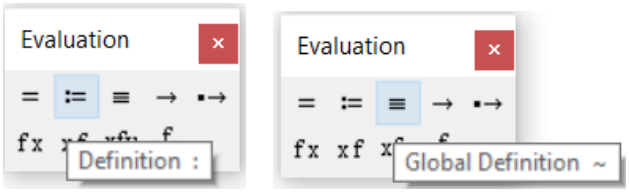
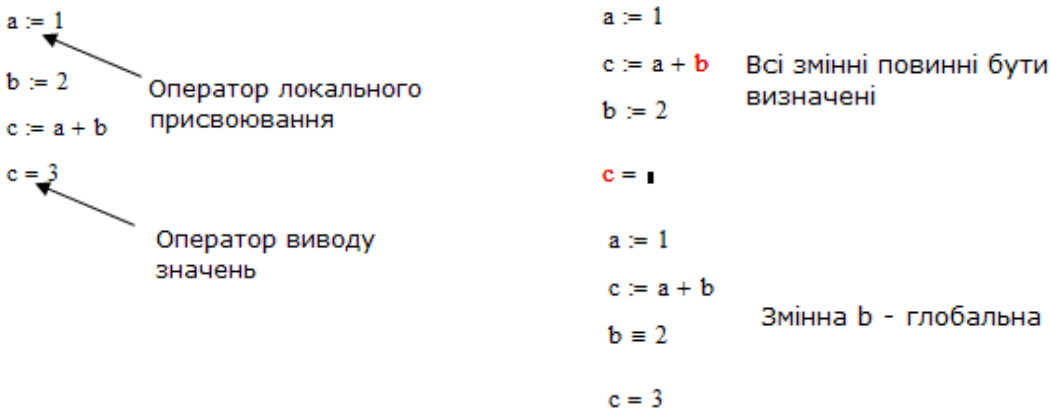


Рисунок 3.11 – Локальне та глобальне присвоювання

MathCAD прочитує документ **двічі**: зліва направо і зверху вниз. Спочатку виконуються всі дії, що приписані глобальним оператором присвоювання (в цьому випадку визначення змінної можна здійснювати в будь-якій точці документу), а потім дії, пов'язані з локальним присвоюванням, а також виводяться всі необхідні результати обчислень за допомогою оператору чисельного виводу (=):



Змінні можуть набувати значення із заданого інтервалу (рис. 3.12). Локальне присвоювання для такої змінної виглядає так:

1) значення змінної змінюються з шагом 1

```
Name:=Nbegin .. Nend
```

2) значення змінної змінюються з шагом `Step`

```
Name:=Nbegin, Nbegin+Step.. Nend
```

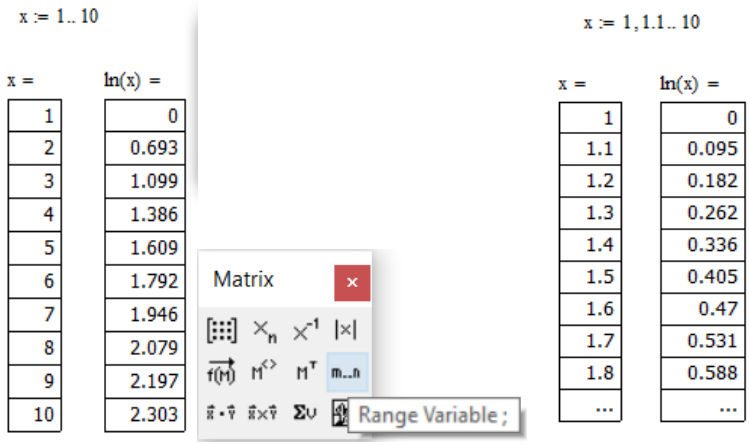


Рисунок 3.12 – Приклад набуття змінною x значень з інтервалу

3.4.2 Символьні обчислення

Значення багатьох функцій або виразів може бути обчислено як символьно такі чисельно. В системі MathCAD існує два оператори виводу: чисельний та символьний. В якості оператора чисельного виводу виступає = (див. п. 1.3.1).

Символьне обчислення пов'язано з отриманням результату аналітичним методом.

Для того, щоб задіяти символьний процесор MathCAD, необхідно використати спеціальний оператор виводу → з панелі інструментів **Evaluation** (рис. 3.13, рис. 3.14).

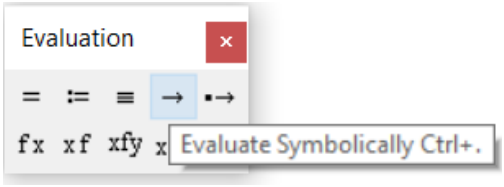


Рисунок 3.13 – Символьний оператор виводу

$$\frac{1}{3}+\frac{5}{9}\rightarrow\frac{8}{9} \qquad \frac{1}{3}+\frac{5}{9}=0.889$$

$$\cos\left(\frac{\pi}{6}\right)\rightarrow\frac{\sqrt{3}}{2} \qquad \cos\left(\frac{\pi}{6}\right)=0.866$$

$$\int x^2\,dx\rightarrow\frac{x^3}{3} \qquad \int_0^1 x^2\,dx\rightarrow\frac{1}{3} \qquad \int_0^1 x^2\,dx=0.333$$

Рисунок 3.14 – Приклад символьного та чисельного розрахунку

3.5 Побудова та форматування графіків

Графік – це зручний спосіб візуалізації інформації. MathCAD має значну кількість засобів для побудови графіків різних типів, і всі основні типи графіків та інструменти роботи з ними розташовані на панелі **Graph**, що включено до панелі інструментів **Math:**

- графік кривої у двовимірній системи координат (X-Y Plot);
- графік кривої у полярній системі координат (Polar Plot);
- поверхня (Surface);
- контурний графік (Contour Plot);
- стовпчаста тривимірна діаграма (3D Bar Plot);
- точковий тривимірний графік (3D Scatter Plot);
- векторне поле (Vector Field).

Список всіх типів графіків MathCAD можна отримати за допомогою команди **Insert/Graph** (рис. 3.15).

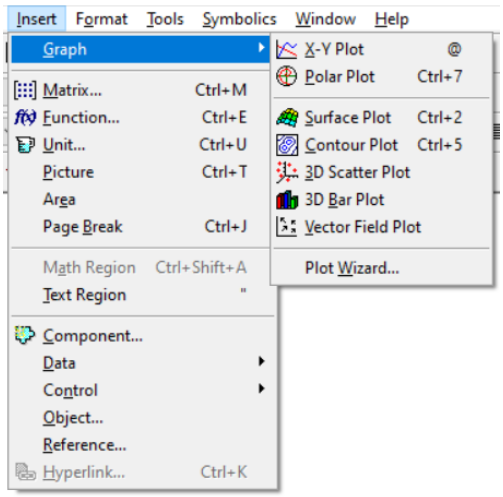


Рисунок 3.15 – Типи графіків MathCAD

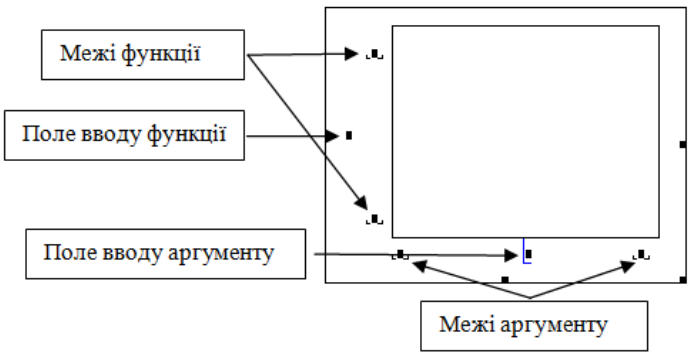
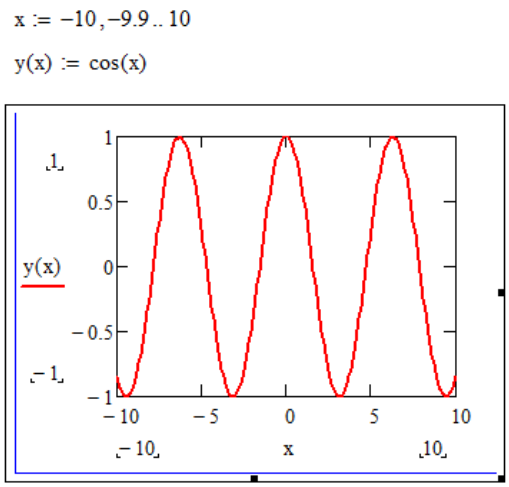
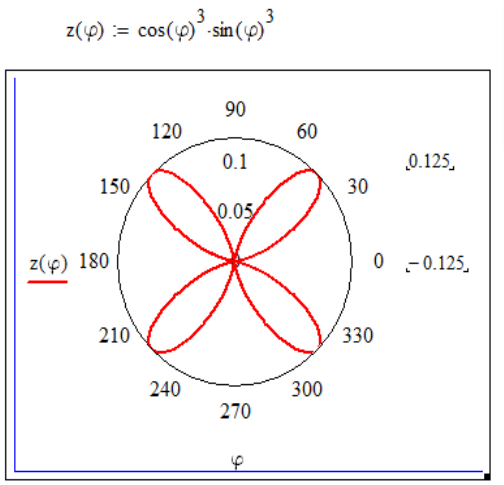


Рисунок 3.16 – Шаблон графіка MathCAD в декартовій системі координат

Приклади графіків функцій, що задано у декартових та полярних координатах :





В MathCAD на одній площині можна зобразити декілька графіків функцій. Для цього необхідно ввести їх в поле вводу функції через **кому**. Кожна функція може залежати від свого аргументу, тому, за необхідністю, можна ввести через **кому** аргументи в поле вводу аргументів (рис. 3.17).

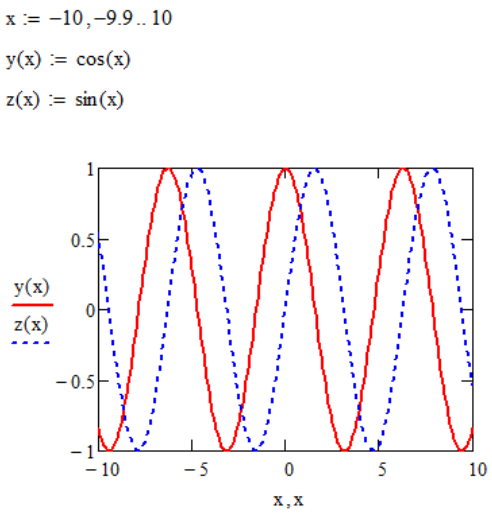


Рисунок 3.17 – Побудова декількох графіків в MathCAD

Після того, як графік побудовано, можна налаштувати його зовнішній вигляд. Для цього використовується команда **Format/Graph**. Вікно форматування будь-якого графіку можна визвати з контекстного меню командою **Format…**.

Питання для самоконтролю до теми 3

1. Які команди містить головне меню MathCAD?
2. Які основні елементи містить вікно MathCAD.
3. Для чого призначена панель інструментів «Математика»?
4. Які функції використовуються для читання/запису текстових файлів?
5. Розкрити сутність локального та глобального присвоювання. Як визначити локальну та глобальну змінну в MathCAD?
6. Яким чином в MathCAD здійснюються символьні обчислення?
7. За допомогою якого інструменту та які типи графіків можна побудувати в MathCAD?
8. Який загальний вигляд має функція користувача?
9. Яким чином побудувати в MathCAD декілька графіків функцій?

Тестові завдання:

Для наведених завдань оберіть одну правильну відповідь:

1. Яка із вказаних функцій Mathcad є критично важливою для інженерного документування, оскільки забезпечує візуальну прозорість розрахунків?

а) Редагування розміру шрифту
б) Інтеграція математичних формул, тексту та графіки в єдиному документі
в) Використання кольору фону
г) Можливість створення гіперпосилань

2. Для чого використовується функція графічної візуалізації (2D-графіки, 3D-графіки) в процесі обробки даних?

а) Виключно для естетичного в) Для швидкого виявлення трендів,

оформлення документа
б) Для автоматичного розв'язання інтегралів
перевірки відповідності моделі експериментальним даним тощо
г) Для зменшення розміру файлу

3. Чи є файли робочих аркушів, створені в класичному Mathcad (.xmcd), прямо сумісними та придатними для відкриття в Mathcad Prime?

а) Так, вони відкриваються без жодних змін
б) Ні, Prime взагалі не підтримує старі формати
в) Ні, вони вимагають обов'язкового використання вбудованого конвертера
г) Так, якщо класична версія була 15

4. Яке основне призначення функції Given у поєднанні з функцією Find у Mathcad?

а) Створення блоку програмування для циклічних обчислень
б) Побудова графіків із залежністю від кількох параметрів
в) Символьне розв'язання диференціальних рівнянь
г) Чисельне розв'язання систем нелінійних рівнянь або оптимізаційних задач

5. Яка із вказаних змінних, що використовується у Mathcad, не може бути візуалізована безпосередньо як точковий графік (X-Y Plot)?

а) Вектор A (набір значень)
б) Скалярна змінна C := 5
в) Функція f(x)
г) Оператор діапазону (Range Variable)

6. Який оператор у Mathcad використовується для чисельного розв'язання рівняння, що містить одну невідому змінну?

а) Оператор символьного обчислення (→)
б) Оператор пошуку кореня (root(f(x), x))
в) Оператор чисельного обчислення (=)
г) Оператор присвоєння (:=)

4 РОЗРОБКА WEB-САЙТУ ЗАСОБАМИ МОВИ HTML

4.1 Елементи, атрибути HTML-документа

HTML (Hypertext Markup Language) – мова гіпертекстової розмітки. Ця мова дозволяє задавати спосіб представлення на Web-сторінках тексту, зображень, відео файлів або електронної інформації будь-якого іншого типу [7].

Будь-який HTML-документ – це набір елементів, що описують окремі складові документа, такі як заголовки, списки, абзаци тексту, таблиці та ін.

Найчастіше елемент розмітки складається з трьох частин: початкового і кінцевого компонентів, між якими розміщуються текст або інші елементи документа. Цими компоненти є спеціальні конструкції, що управляють, для розмітки вмісту HTML-документу і називаються **тегами**.

Основний принцип змін у **HTML5** – відокремлення структури від оформлення. Усе, що стосується зовнішнього вигляду (колір, розмір шрифту, вирівнювання, рамки), має бути реалізовано за допомогою **CSS** (Каскадних Таблиць Стилів), а не за допомогою HTML-тегів чи атрибутів.

HTML-елементом називається комбінація начального тегу, кінцевого тегу та вмісту. Приклади HTML-елементів:

```
<p> це абзац </p>
<em> це текст з акцентом </em>
```

Згідно з рекомендаціями HTML5, для чистоти коду та сумісності з іншими стандартами теги та їх атрибути слід писати лише в нижньому регістрі (`<p>`, `<h1>`, `href`, `class`).

Загальний синтаксис написання тегів:

`<тег атрибут1="значення" атрибут2="значення">` – (одинарний тег, який не має вмісту (наприклад, `<img src="image.png" alt="Опис">`)

`<тег атрибут1="значення" атрибут2="значення">...</тег>` – парний тег (контейнер)

Одинарний тег використовується самостійно, а парний може включати інші теги або текст. Щоб розширити можливості окремих тегів і гнучкіше управляти вмістом контейнерів застосовуються **атрибути** тегів, які розділяються між собою пробілами.

Атрибути задаються в початковому тегу елементів і складаються з імені і значення, наприклад в атрибуті `href="http://www.microsoft.com"` `href` є ім'ям, а `http://www.microsoft.com` – значенням.

Згідно специфікації HTML5 для всіх атрибутів слід використовувати повний синтаксис з подвійними лапками ("приклад").

Порядок атрибутів у тегу HTML, як правило, не має значення і не впливає на відображення чи функціональність сторінки.

Наприклад:

```
<!-- Порядок 1 --> <img src="photo.jpg" alt="Моє фото" width="200" height="150">
<!-- Порядок 2 (переставлено) --> <img width="200" height="150" alt="Моє фото" src="photo.jpg">
```

Обидва приклади повністю ідентичні з точки зору браузера. Зображення буде завантажено, відображено з тими ж розмірами, і матиме той самий альтернативний текст.

Для полегшення читання та підтримки коду, а також для уніфікації проєктів розробники часто використовують узгоджений порядок та поширені домовленості (особливо в React, Angular та загальному Front-end):

1. Основні та обов'язкові атрибути: `src`, `href`, `class`.
2. Специфічні атрибути HTML: `id`, `name`, `type`.
3. Атрибути розмірів: `width`, `height`.
4. Атрибути доступності (Accessibility): `alt`, `role`, `aria-*`.

Наприклад, більшість команд прагнуть розмістити `class` на початку, оскільки це найпоширеніший атрибут, що керує стилями та поведінкою.

4.2 Структура HTML-кода

Сучасний HTML-документ має максимально спрощену декларацію типу документа та використовує семантичні теги для кращої доступності та індексації (табл. 4.1).

Приклад базового HTML-документа:

```
<!DOCTYPE html>
<html lang="uk">
    <head>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <title>Приклад Web-сторінки (HTML5)</title>
        <link rel="stylesheet" href="styles.css">
    </head>
    <body>
        <header>
            <h1>Заголовок Web-сторінки</h1>
        </header>
        <main>
            <p>Перший абзац.</p>
            <!-- Коментар -->
            <p>Другий абзац.</p>
        </main>
        <footer>
            <p>&copy; 2024. Усі права захищено.</p>
        </footer>
    </body>
</html>
```

Таблиця 4.1 – Пояснення базової структури HTML-документа

Рядок коду	**Назва/Тег**	**Призначення**
`<!DOCTYPE html>`	Декларація DOCTYPE	Обов'язковий рядок, який вказує браузеру, що документ написаний відповідно до стандарту HTML5. Це найкоротша і найсучасніша декларація, яка замінила довгі DTD попередніх версій.
`<html lang="uk">`	Кореневий елемент `<html>`	Позначає початок HTML-документа. Атрибут `lang="uk"` вказує мову контенту сторінки (українська), що важливо для доступності та SEO.
`<head>`	Заголовок документа	Контейнер для метаданих сторінки, які не відображаються безпосередньо в тілі документа (`<body>`), але використовуються браузером, пошуковими системами та іншими сервісами.
`<meta charset="UTF-8">`	Кодування символів	Встановлює кодування символів для коректного відображення тексту, включаючи кирилицю та спеціальні символи. UTF-8 є стандартом.
`<meta name="viewport" content="width=device-width, initial-scale=1.0">`	Адаптивний дизайн	Критично важливий для мобільних пристроїв. Він вказує браузеру, що ширина вікна перегляду повинна дорівнювати ширині пристрою, а початковий масштаб має бути 1.0. Це забезпечує коректну роботу адаптивного дизайну (Responsive Web Design).
`<title>Приклад Web-сторінки (HTML5)</title>`	Заголовок вікна/вкладки	Текст, який відображається у вкладці браузера. Це один із найважливіших елементів для SEO.
`<link rel="stylesheet" href="styles.css">`	Підключення CSS	Пов'язує HTML-документ із зовнішнім файлом стилів (`styles.css`), що є найкращою практикою для відокремлення структури від оформлення.
`</head>`	Закриваючий тег `<head>`	
`<body>`	Тіло документа	Містить весь видимий контент сторінки: текст, зображення, таблиці, посилання тощо.
`<header>`	Семантичний тег `<header>`	Використовується для вступного контенту, зазвичай містить заголовки, логотипи або навігаційні елементи.
`<h1>Заголовок Web-сторінки</h1>`	Головний заголовок	Визначає основний заголовок сторінки (повинен бути лише один на сторінку).
`</header>`	Закриваючий тег `<header>`	
`<main>`	Семантичний тег `<main>`	Містить основний, унікальний контент документа. Контент всередині `<main>` не повинен повторюватися на інших сторінках (наприклад, бічні панелі чи нижні колонтитули).
`<p>Перший абзац.</p>`	Абзац тексту	Стандартний блоковий елемент для

		текстового абзацу.
`<!-- Коментар -->`	Коментар	Текст, який ігнорується браузером. Використовується для пояснень у коді.
`<p>Другий абзац.</p>`	Абзац тексту	
`</main>`	Закриваючий тег `<main>`	
`<footer>`	Семантичний тег `<footer>`	Містить інформацію про автора, копірайт, посилання на умови використання або контактну інформацію. Розташовується в нижній частині сторінки або секції.
`<p>© 2024. Усі права захищено.</p>`	Інформація про авторське право	
`</footer>`	Закриваючий тег `<footer>`	
`</body>`	Закриваючий тег `<body>`	
`</html>`	Закриваючий тег `<html>`	

Елемент `<!DOCTYPE>` призначений для того, щоб вказати тип поточного документа – DTD (document type definition, опис типу документа). Це необхідно, щоб браузер розумів, як слід інтерпретувати поточну Web-сторінку, адже HTML існує в декількох версіях.

Тег `<html>` визначає початок HTML-документу, усередині нього зберігається заголовок (`<head>`) і тіло документа (`<body>`).

4.3 Дизайн тексту та семантична розмітка

В HTML5 для форматування тексту використовується семантична розмітка, а всі стилі виносяться у CSS (табл. 4.2).

Таблиця 4.2 – Дизайн тексту

Призначення	**Семантичний тег (HTML5)**	**Стилізація (CSS-аналог)**
Абзац	`<p>...</p>`	Відступи та вирівнювання задаються CSS
Примусовий перехід на новий рядок	` `	Використовувати мінімально.
Горизонтальна лінія (тематичний роздільник)	`<hr>`	Стилізація (товщина, колір) задається CSS
Жирний шрифт (Важливість)	`<strong> текст </strong>`	`font-weight: bold;`
Курсив (Акцент/Виділення)	`<em> текст </em>`	`font-style: italic;`
Підкреслення (Стилістичне виділення)	`<span style="text-decoration: underline;"> текст </span>`	`text-decoration: underline;`
Заголовки	`<h1>...</h1>` до `<h6>...</h6>`	Розмір шрифту задається CSS (`font-size`)

Вирівнювання елементів: Вирівнювання тексту або блокових елементів завжди здійснюється за допомогою CSS:

– по центру: `text-align: center` або `margin: 0 auto`;

– по лівому/правому краю: `text-align: left / text-align: right.`

Для створення рухомого тексту у сучасному HTML5 використовуються CSS-анімації або CSS-трансформацій замість застарілого тегу `<marquee>`. Хоча `<marquee>` досі може працювати в деяких браузерах, він не є стандартним і вважається порушенням принципів веб-доступності (Accessibility).

При створенні рухомого тексту за допомогою CSS-анімації можна використовувати:

1. `<div class="marquee-container">` – це контейнер; його властивість `overflow: hidden;` приховує ту частину тексту, яка знаходиться за його межами, створюючи ефект «вікна»

```
<div class="marquee-container">
       <p class="moving-text">
      *** Академічна доброчесність - це основа якісної освіти та наукових досліджень! ***
       </p>
</div>
```

Класи `marquee-container та moving-text` можуть виглядати, наприклад, так:

```
.marquee-container {
     /* Контейнер обмежує видиму область руху */
         width: 80%;
         overflow: hidden;
         border: 2px solid #1e88e5; /* Синя рамка */
         border-radius: 8px;
         padding: 10px 0;
         background-color: #fff;
         box-shadow: 0 4px 10px rgba(0, 0, 0, 0.1);
                  }
.moving-text {
     /* Сам рухомий текст */
         display: inline-block;
         white-space: nowrap; заборона перенесення рядків */
         padding-left: 100%; /* Початкова позиція тексту за межами контейнера */
         font-size: 24px;
         font-weight: bold;
         color: #d81b60; /* червоний колір */
```

2. `@keyframes marquee-motion` – це визначення самої анімації:

– `0%` – текст починає рух;

– `100%` – текст зміщується ліворуч на велику відстань (translate(-100%, 0)), щоб повністю пройти через екран

```
    @keyframes marquee-motion {
    /* 0% - Початок анімації: текст знаходиться за правим краєм
*/
    0% {
          transform: translate(0, 0);
       }
    /* 100% - Кінець анімації: текст повністю пройшов через екран
*/
    100% {
    /* Зміщення на -100% від ширини контейнера + ширина самого
тексту */
          transform: translate(-100%, 0);
        }
                              }
```

3. застосування анімації до тексту, наприклад, може виглядати так: `animation: marquee-motion 15s linear infinite`

– `marquee-motion` – ім'я визначених ключових кадрів;

– `15s` – тривалість одного циклу руху (15 секунд);

– `linear` – швидкість руху постійна;

– `infinite` – анімація повторюється нескінченно.

4.4 Створення списків та їх дизайн

Списком називається взаємозв'язаний набір окремих фраз або пропозицій, які починаються з маркера або цифри. Списки надають можливість упорядкувати і систематизувати різні дані і представити їх в наочному і зручному для користувача вигляді.

Маркований список утворюється за допомогою парних тегів `<ul>…</ul>` та `<li>…</li>`. Приклад:

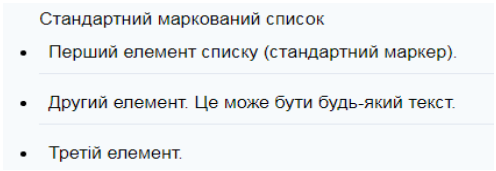
```
    <h2>Стандартний маркований список</h2>
         <ul class="marker-disc">
              <!-- Кожен елемент списку - це тег <li> -->
              <li>Перший елемент списку (стандартний
маркер).</li>
              <li>Другий елемент. Це може бути будь-який
текст.</li>
              <li>Третій елемент.</li>
        </ul>
```

Класи для різних типів маркерів:

```
    .marker-disc { list-style-type: disc; } /* Стандартний маркер
(заповнене коло) */
    .marker-circle { list-style-type: circle; } /* Порожнє коло
*/
    .marker-square { list-style-type: square; } /* Квадрат */
```

Властивість CSS `list-style-type: square;` відповідає за зміну зовнішнього вигляду маркера (замість стандартної чорної точки буде чорний квадрат).

Результат:



Нумерований список утворюється за допомогою тегів `<ol>…</ol>` і `<li>…</li>`.

Приклад:

```
<h1>Створення нумерованого списку (HTML5)</h1>
<p>Нумерований список, оформлений римськими цифрами за допомогою CSS:</p>
<!--
  Створення списків:
  Тег <OL> (Ordered List) створює контейнер для нумерованого списку.
-->
<ol>
   <!-- Тег <li> (List Item) створює кожен елемент списку -->
    <li>Створити декларацію <!DOCTYPE html>.</li>
    <li>Додати кореневий тег <html> з атрибутом lang="uk".</li>
    <li>Визначити секцію заголовка &lt;head&gt; з мета-даними.</li>
    <li>Визначити секцію тіла документа &lt;body&gt; з контентом.</li>
    <li>Застосувати CSS для вирівнювання та стилізації елементів.</li>
</ol>
```

Результат:



Стиль списку змінюється за допомогою властивості `list-style-type` у CSS, наприклад: `list-style-type: hi-roman;`

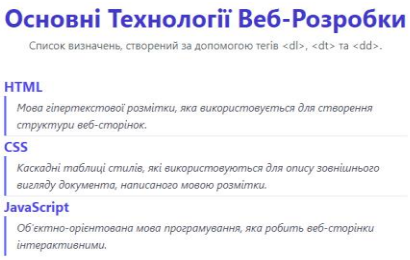
Список визначень. Список визначень складається з термінів і їх визначень. Такий список задається за допомогою тегів `<dl>…</dl>`. Терміни в такому списку задаються тегом `<dt>`, а їх визначення тегом `<dd>`. Нижче приведена структура списку визначень.

```
<dl>
    <!-- ПЕРШИЙ ЕЛЕМЕНТ -->
    <dt>HTML</dt>
    <dd>Мова гіпертекстової розмітки, яка використовується для створення структури веб-сторінок.</dd>
```

```
    <!-- ДРУГИЙ ЕЛЕМЕНТ -->
    <dt>CSS</dt>
    <dd>Каскадні таблиці стилів, які використовуються для опису зовнішнього вигляду документа, написаного мовою розмітки.</dd>

    <!-- ТРЕТІЙ ЕЛЕМЕНТ -->
    <dt>JavaScript</dt>
    <dd>Об'єктно-орієнтована мова програмування, яка робить веб-сторінки інтерактивними.</dd>
</dl>
```

Результатом може бути:



В сучасному HTML використовують такі елементи для заголовків списків або груп елементів (табл. 4.3):

– заголовок усередині контейнера `<div>` або `<section>` – це найпоширеніший спосіб; використовуються стандартні теги заголовків (`<h1>` - `<h6>`) безпосередньо перед списком `<ul>` або `<ol>`;

– тег `<legend>` для елементів форми – створює заголовок або підпис для групи елементів форми, яка об'єднана в тег `<fieldset>`, він часто асоціюється з візуальним обрамленням групи;

– тег `<figcaption>` для малюнків та діаграм – використовується для підпису вмісту всередині тегу `<figure>`.

Наприклад:

```
/* Стиль для рамки Fieldset */
        fieldset {
          border: 2px solid #3b82f6; /* Синя рамка */
          border-radius: 0.5rem;
          padding: 1.5rem;
          margin-top: 2rem;
                 }
/* Стиль для заголовка Legend */
        legend {
          font-size: 1.25rem;
          font-weight: 700;
          color: #3b82f6;
          padding: 0 0.5rem;
          background-color: #f3f4f6;
          border-radius: 0.25rem;
               }
```

Таблиця 4.3 – Опис стилю елемента за допомогою CSS

Властивість CSS	Значення	Пояснення
font-size	1.25rem	Розмір шрифту. 1.25rem – це приблизно 20 пікселів (якщо базовий розмір браузера 16px), що робить текст помітно більшим за стандартний
font-weight	700	Насиченість шрифту. 700 – це стандартне числове значення для жирного накреслення
color	#3b82f6	Колір тексту. #3b82f6 – це яскравий, насичений синій колір
padding	0 0.5rem	Внутрішній відступ. Встановлює простір навколо вмісту елемента. 0 (зверху і знизу) та 0.5rem (зліва і справа). Тобто, горизонтальний відступ присутній, а вертикальний – відсутній
background-color	#f3f4f6	Колір фону. Встановлює світло-сірий колір фону для елемента
border-radius	0.25rem	Радіус рамки. Трохи заокруглює кути елемента

4.5 Створення таблиць та їх дизайн

У HTML5 використовуються лише семантичні теги (<table>, <thead>, <tbody>, <tr>, <th>, <td>), а все оформлення переноситься в зовнішній або внутрішній **CSS**.

Таблиці створюють за допомогою наступних обов'язкових тегів:

- <table ...> </table>: Контейнер, який визначає саму таблицю.
- <caption> Заголовок таблиці </caption>: Визначає **заголовок** або **назву** таблиці, який має бути першим елементом всередині <table>. Це забезпечує семантичну доступність.
- <thead> ... </thead>: Групує вміст, що є заголовком таблиці (наприклад, назви стовпців).
- <tbody> ... </tbody>: Групує основний вміст (тіло) таблиці.
- <tr> ... </tr>: Визначає **рядок** таблиці (Table Row).
- <th> Текст </th>: Формує **комірку-заголовок** (Table Header). Використовується всередині <thead> або для заголовків рядків у <tbody>.
- <td> Текст </td>: Формує **комірку даних** (Table Data).

Приклад створення таблиці 2x2:

```
<table>
     <caption>Приклад таблиці 2x2</caption>
        <thead>
           <tr>
             <th>Заголовок А</th>
             <th>Заголовок Б</th>
           </tr>
        </thead>
```

```
        <tbody>
                <tr>
                  <td>Дані 1</td>
                  <td>Дані 2</td>
                </tr>
                <tr>
                  <td>Дані 3</td>
                  <td>Дані 4</td>
                </tr>
         </tbody>
      </table>
```

Весь дизайн таблиці, включаючи рамки, відступи, ширину та колір, переноситься в CSS.

Таблиця 4.4 – Властивості для стилізації таблиці

Властивість CSS	**Пояснення**
`width: ...;`	Задає ширину елемента. Можна вказати її у пікселях (`px`), відсотках (`%`), або інших одиницях
`height: ...;`	Задає висоту елемента. Рідко використовується для всієї таблиці, частіше – для окремих комірок або рядків
`border: ...;`	Скорочена властивість для встановлення товщини, стилю та кольору рамки одночасно
`border-color: ...;`	Встановлює лише колір рамки
`background-color;`	Встановлює колір фону елемента (таблиці, рядка, комірки)
`background-image: ...;`	Встановлює зображення як фон елемента
`padding: ...;` для `<th>` та `<td>`	Внутрішній відступ. Визначає простір між вмістом комірки (текстом) та її рамкою
`border-collapse: collapse;`	Об'єднує рамки сусідніх комірок в одну спільну тонку рамку
`margin: 0 auto;`	Центрування блочних елементів (яким є таблиця) по горизонталі

Приклад стилізації таблиці:

```
/* 1. Стилізація всієї таблиці */
   table {
     width: 50%; /* Замість width="200" */
     border-collapse: collapse; /* Для однієї тонкої рамки */
     margin: 20px auto; /* Для центрування таблиці, замість
align="center" */
     font-family: Arial, sans-serif;
         }

/* 2. Стилізація заголовка */
   caption {
       font-weight: bold;
       padding: 10px;
           }
```

```
/* 3. Стилізація комірок-заголовків */
   th {
     background-color: #4CAF50;
     color: white;
     padding: 12px 15px; /* Замість cellpadding */
     border: 1px solid #ddd;
     text-align: left;
      }

/* 4. Стилізація комірок даних */
   td {
     border: 1px solid #ddd;
     padding: 10px 15px;
      }

/* 5. Додавання смугастого оформлення для читабельності */
   tr:nth-child(even) {
     background-color: #f2f2f2;
}
```

4.6 Додавання графічних файлів

Комбінуючи зображення і звичайний текст можна представляти інформацію максимально ефективно.

Тег `<img>` призначений для відображення на Web-сторінці зображень в графічному форматі ***.gif***, ***.jpeg*** або ***.png***. Адреса файлу з рисунком задається за допомогою атрибуту `src`. Якщо необхідно, то рисунок можна зробити посиланням на інший файл, розмістивши тег `<img>` до контейнеру `<a>`.

Синтаксис тегу `<img>`:

```
<img src="адреса графічного файлу" alt="альтернативний текст">
```

Основні атрибути тегу `<img>`:

– `alt` – альтернативний текст, який виводитиметься замість зображення, якщо браузер не може прийняти графічний файл або якщо режим відображення графіки вимкнено

```
<img alt="текст">;
```

– `width` – задає розмір графічного об’єкту за шириною в пікселах або відсотках сторінки

```
<img width="значення">;
```

– `height` – задає розмір графічного об’єкту за висотою в пікселах або відсотках сторінки

```
<img height="висота">;
```

Приклад відображення зображень:

```
<a href="https://google.com.ua" target="_blank">
    <!-- Тег зображення, вкладений у посилання -->
<img
```

```
    src="placeholder.png"
    alt="Натисніть тут для переходу на Google"
    width="150"
    height="100"
    style="border: 0;"
   >
</a>
```

4.7. Додавання гіперпосилань

Посилання є основою гіпертекстових документів і дозволяють переходити з однієї Web-сторінки на іншу. Особливість їх полягає в тому, що посилання може вести не лише на HTML-файл, але і на файл будь-якого типа, причому цей файл може розміщуватися зовсім на іншому сайті.

Для створення посилання необхідно повідомити браузеру, що є посиланням, а також вказати адресу документа, на який слід зробити посилання. Обидві дії виконуються за допомогою тегу `<a>`.

Загальний синтаксис створення посилань:

```
<a href="URL"> текст посилання  </a>
```

Атрибут `href` визначає URL (Universal Resource Locator, універсальний покажчик ресурсу), тобто, адресу документа, на яку слід перейти, а вміст контейнера `<a>` є посиланням.

Адреса посилання може бути як абсолютною, так і відносною. Абсолютні адреси повинні починатися з назви протоколу (зазвичай http://) і містити ім'я сайту. Відносні посилання ведуть відлік від кореня сайту або поточного документа.

Наприклад, посилання виду `<a href="http://www.microsoft.com>Сайт Microsoft</a>` є абсолютним і веде на головну сторінку сайту Microsoft.

Відносне посилання вказує на адресу сторінки відносно поточного документа і має, наприклад, вигляд `<a href="page.html">Задача</a>`.

За замовчуванням, при переході за посиланням, документ відкривається в поточному вікні або фрейм. При необхідності ця умова може бути змінена атрибутом `target` – вказує ім’я вікна або фрейму, куди браузер завантажуватиме документ

```
<a target="ім’я вікна">…</a>
```

В якості значення використовується ім'я вікна або фрейму, що задається атрибутом `name`. Якщо встановлено неіснуюче ім'я, то буде відкрито нове вікно. В якості зарезервованих застосовуються наступні імена:

- `_self` – завантажує сторінку в поточне вікно (це значення задається за замовчуванням, тобто `target="_self"`);
- `_parent` – завантажує сторінку у батьківський фрейм; якщо фреймів немає, то це значення працює як `_self`;
- `_top` –відміняє всі фрейми і завантажує сторінку в повному вікні
- `_blank` – завантажує сторінку в нове вікно браузера.

В якості гіперпосилань можна використовувати картинки. Для цього усередині тегу <a>… необхідно додати тег <img>: Наприклад,

```
<a href="page.html"><img src=”image.jpg”></a>.
```

4.8 Оформлення HTML-документу за допомогою CSS

Cascading Style Sheets (Каскадні Таблиці Стилів – CSS) – це спеціальна мова опису стилів, яка володіє набагато багатшим і функціональним, в порівнянні з HTML, набором засобів форматування і управління стилями елементів документа.

Після введення CSS деякі теги і атрибути в HTML стали вважатися застарілими (за допомогою CSS можна застосовувати форматування набагато ефективніше).

Таблиці стилів складаються з набору правил. Кожне правило складається з одного або декількох селекторів і блоку визначення, що виділяється фігурними дужками.

Блок визначення може містити одне або декілька властивостей відокремлених крапкою з комою (;) (після останньої властивості крапка з комою необов'язкова). Кожна властивість повинна мати значення відокремлене двокрапкою (:).

Таблиця 4.5 – Методи використання CSS

Спосіб	Де розміщується CSS-код
1. Вбудований стиль	В атрибуті `style` HTML-тегу (`<p style="...">`)
2. Внутрішній стиль	У тегу `<style>` у секції `<head>`
3. Зовнішній стиль	В окремому файлі `.css`, який підключається тегом `<link>`

Отже, CSS (Cascading Style Sheets) є необхідним доповненням до HTML, яке відповідає за візуальне оформлення.

Питання для самоконтролю до теми 4

1. Назвіть теги, що створюють заголовки.
2. Яку загальну структуру має HTML-документ?
3. Наведіть засоби дизайну тексту на мові HTML.
4. Чи можна зробити гіперпосиланням графічний об’єкт?
5. Як створити ефект руху для тексту?
6. Як створити на Web-сторінці маркований список?
7. Для чого призначені CSS?
8. Для чого використовується семантичний тег `<footer>`?
9. Чи має значення порядок атрибутів у тегу HTML?
10. Наведіть приклади тегів в HTML5, які не вимагають кінцевого тегу.

Тестові завдання:

Для наведених завдань оберіть одну правильну відповідь:

1. Що означає «семантична розмітка» в контексті HTML5?

а) Використання CSS для поліпшення зовнішнього вигляду елементів
б) Використання тегів для опису призначення вмісту (наприклад, `<header>` для заголовка)
в) Використання лише парних тегів (з початковим і кінцевим тегом)
г) Обов'язкове використання JavaScript для динамічної зміни вмісту

2. Який атрибут можна використовувати в тегу <img> для надання браузеру підказки про розміри зображення, яке він буде завантажувати, щоб уникнути «стрибків» вмісту?

а) `size`
б) `layout`
в) `dimensions`
г) `width i height`

3. Як виглядає мінімально необхідна і коректна декларація типу документа (DTD) для HTML5?

а) `<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 5.0//EN">`
б) `<!DOCTYPE html5>`
в) `<!DOCTYPE html>`
г) `<!DOCTYPE html SYSTEM "about:legacy-compat">`

4. Яке основне призначення атрибута `alt` (альтернативний текст) у тегу `<img>`?

а) Забезпечити SEO-оптимізацію, надаючи пошуковим системам ключові слова
б) Вказати URL-адресу зображення, яке має завантажитися
в) відобразити текст, якщо зображення не може бути завантажене, та покращити доступність для програм-читачів
г) Встановити розміри зображення перед його завантаженням

5. Який тег використовується для вбудовування правил CSS безпосередньо в секцію `<head>` HTML-документа?

а) `<style>`
б) `<css>`
в) `<link>`
г) `<format>`

6. У якому місці в структурі списку (наприклад, `<ol>` або `<ul>`) має бути розміщений вкладений список?

а) Між двома тегами `<li>...</li>`
б) Всередині тегу `<li>` елемента, до якого він відноситься
в) Після кінцевого тегу списку (`</ol>` або `</ul>`)
г) Безпосередньо перед початковим тегом списку (`<ol>` або `<ul>`)

5 ЕЛЕМЕНТИ ОФІСНОГО ПРОГРАМУВАННЯ

Пакет офісних програм – це не просто інструмент для документів, а потужна платформа для автоматизації.

Офісне програмування – це набір методів та інструментів, що дозволяють автоматизувати рутинні завдання, розширювати функціонал та інтегрувати офісні програми (як-от Microsoft Office, Google Workspace чи LibreOffice) з іншими системами [4, 6].

Для фахівця у сфері інформаційних технологій, володіння Microsoft Excel є ключовою навичкою. Розуміння його можливостей – від роботи з формулами і аналізом даних до макропрограмування на VBA (Visual Basic for Applications), дозволяє:

– автоматизувати трудомісткі, повторювані операції (наприклад, генерацію звітів);

– розширювати стандартний функціонал Excel, створюючи індивідуальні (спеціалізовані) інструменти;

– ефективно аналізувати та управляти великими масивами даних.

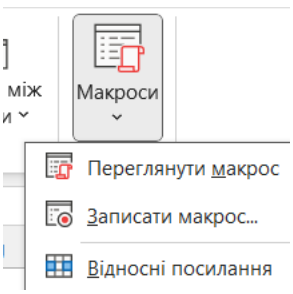
Таким чином, практичне застосування VBA в Excel є невід'ємною частиною інформаційної культури, яка безпосередньо підвищує продуктивність та якість роботи IT-фахівця.

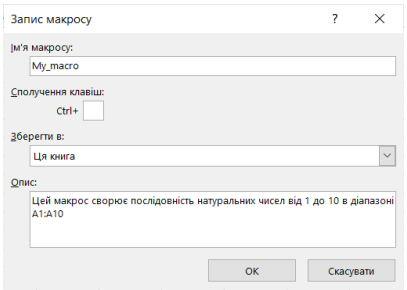
5.1 Створення макросів

Середовище MS Office пропонує два способи створення програм, що відрізняються підходом до процесу: використання макрорекордера і ручне кодування (на мові VBA). Ці підходи орієнтовані на різні категорії: безпосередньо користувачів і програмістів відповідно. Макрорекордер (Macrorecorder) – це програмний інструмент, що записує дії користувача при роботі з документами і додатками, із збереженням запису у вигляді **макросу** − вихідного коду на мові VBA.

Розглянемо побудову макросу, що виконує заповнення діапазону A1:A10 послідовністю натуральних чисел від 1 до 10. Нижче представлено процес створення подібного макросу.

Вибрати команду **Подання/Макроси/Записати макрос…**. В результаті розкриється діалогове вікно:





Після натискання на кнопку ОК починається запис макросу.

Алгоритм запису макросу:

1. Ввести число від 1 до 10 в діапазон A1:A10 за допомогою маркеру заповнення (рис. 5.1, рис. 5.2)

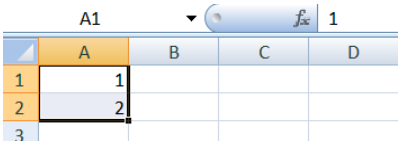


Рисунок 5.1 – Внесення чисел до комірок A1 та A2

2. Виділити комірку B1. Результат, що отримано, показано на Рисунку 5.3.

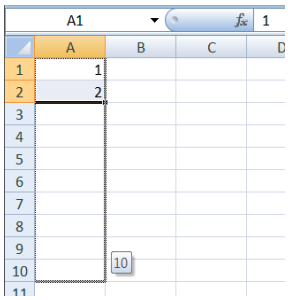


Рисунок 5.2 – Заповнення діапазону A1:A10 за допомогою маркеру заповнення

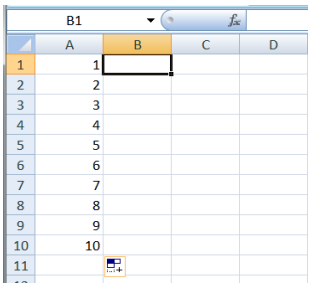
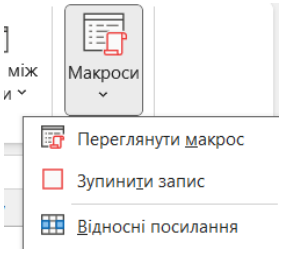


Рисунок 5.3 – Вигляд робочого аркуша перед завершенням запису макросу

3. Наприкінці слід натиснути на кнопку Зупинити макрос:



В подальшому, для запуску макросу необхідно вибрати команду **Подання/Макроси/Переглянути макрос**. Відкриється діалогове вікно *Макрос* (рис. 5.4), в якому потрібно вибрати необхідний для роботи макрос

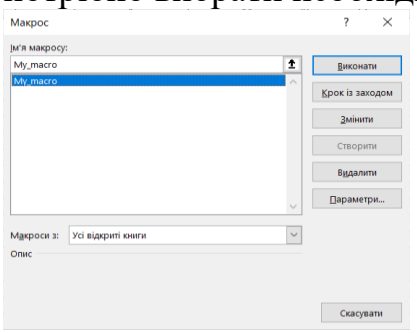
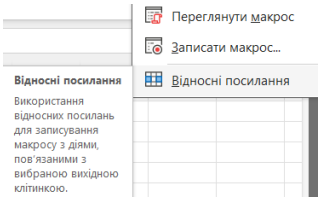


Рисунок 5.4 – Вікно *Макрос*, що можна використовувати для запуска макросів

У наведеному прикладі діапазон комірок, що заповнюється не залежить від того, яка комірка на момент запуску макросу є активною, тобто посилання

на вказаний діапазон є абсолютним. При створенні макросу корисно використовувати не абсолютні, а відносні посилання.

Для того, щоб посилання на комірки, або на діапазони комірок інтерпретувались як відносні, при записі макросу необхідно натиснути кнопку **Відносні посилання**:



5.2 Редактор VBA

Microsoft Excel − це дуже потужний інструмент, який має багатий набір можливостей стандартного інтерфейсу користувача і який можна використовувати для управління даними, їх аналізу і візуалізації. Але інколи, може знадобитися знайти нескладний спосіб виконувати повсякденні завдання, що повторюються, або якісь завдання, які не можна вирішити за допомогою інтерфейсу користувача. Майже в усі програми Microsoft Office (Excel, Word, Access, PowerPoint, Outlook) інтегрований Visual Basic для додатків (VBA), мова програмування, що дозволяє розширювати можливості цих додатків. Найпоширенішою причиною використання VBA в Excel є автоматизація завдань, що повторюються.

VBA працює, виконуючи макроси, покрокові процедури, написані на мові Visual Basic.

Відповідні коди створюються та редагуються в спеціальному редакторі, який визивається з вікна додатку Excel. Для цього достатньо вибрати команду **Розробник/Visual Basic** (рис. 5.5).

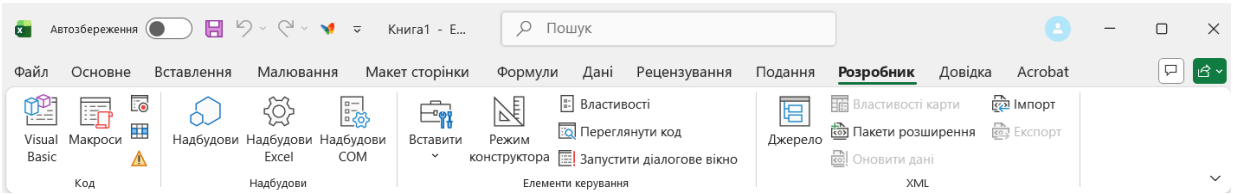
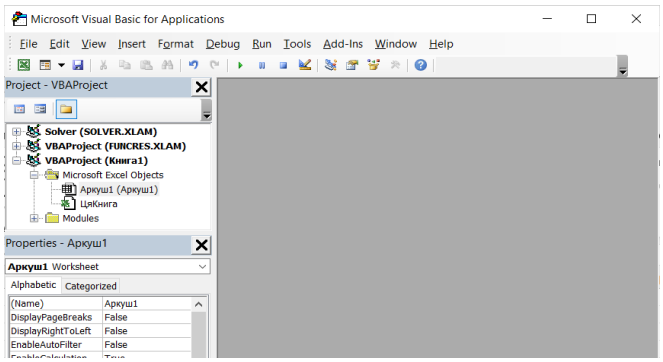


Рисунок 5.5 – Виклик редактора VBA

В результаті відкривається вікно редактора VBA:



Модуль – місце, де зберігається програмний код.

Форма – діалогове вікно з різними елементами управління.

Проект – сукупність модулів, що створюють програму.

Для редагування програмного коду макросу (або функції) слід відкрити той модуль, у якому він був записаний. В якості прикладу наведено код макросу, який був створений в п. 5.1 (рис. 5.5)

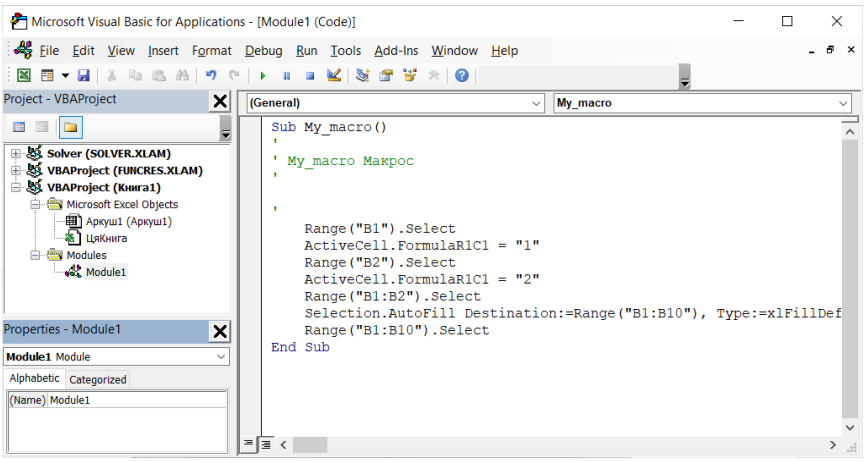


Рисунок 5.6 – Вікно з кодом макросу

В цей код, вже після створення макросу, можна вносити зміни.

5.3 Поняття об‘єкту, властивості, методу, події

Visual Basic for Applications (VBA) використовує об'єктно-орієнтований підхід (ООП), що є основою для автоматизації в додатках Microsoft Office. Хоча VBA не є повноцінною ООП-мовою (як C# чи Java), вона ефективно застосовує ключові концепції: Об'єкти, Класи, Властивості, Методи та Події.

Кожен додаток Office (Excel, Word, Access) розширює базові можливості VBA власним набором об'єктів – це його Об'єктна модель.

Об'єкт – це програмна одиниця, що містить як дані (властивості), так і процедури їх обробки (методи).

У Excel об'єктами є майже всі видимі та невидимі елементи:

- сам додаток Excel (`Application`);
- робоча книга (`Workbook`);
- робочий аркуш (`Worksheet`);
- діапазон або комірка (`Range`);
- кнопки, форми, діаграми, фільтри.

Клас – це шаблон, який визначає структуру, властивості та поведінку групи об'єктів. Кожен конкретний об'єкт є **екземпляром** певного класу.

Наприклад, кожен робочий аркуш MS Excel – це об'єкт, створений на основі **класу** `Worksheet`.

Об'єкти VBA вкладені один в одного, формуючи сувору ієрархію (**об'єктна модель**). Щоб отримати доступ до об'єкта, необхідно пройти цей ланцюжок від батьківського об'єкта до потрібного нащадка.

Приклад (звернення до комірки): щоб занести число 13 у комірку A5 аркуша "Аркуш1" книги "MyBook.XLS", необхідно вказати шлях:

```
Workbooks("MyBook.XLS").Sheets("Лист1").Range("A5").Value = 13
```

Тут послідовно обираються:

– `Workbooks(...)` – об'єкт-книга (екземпляр класу `Workbook`);

– `.Sheets(...)` – об'єкт-аркуш (екземпляр класу `Worksheet`) усередині книги;

– `.Range(...)` – об'єкт-діапазон (екземпляр класу `Range`) усередині аркуша;

– `.Value` – властивість об'єкта `Range`, якій присвоюється значення.

Таблиця 5.1 – Ієрархія та приклади вбудованих об'єктів

Об'єкт	Опис	Приклади Звернення
`Workbook` (Книга)	Файл Excel. Містить об'єкти **`Worksheet`**	**`ThisWorkbook`**, **`Workbooks("Звіт.xlsx")`**
`Worksheet` (Аркуш)	Один робочий аркуш. Містить об'єкти **`Range`**	**`Sheets("Дані"), Sheets(3)`**
`Range` (Діапазон)	Діапазон комірок, може включати одну або багато комірок	**`Range("A1:C5"), Range("B1")`**
`Cell` (Комірка)	Одна комірка. Зручний спосіб звернення за індексами	**`Cells(2, 4) (D2), Cells(i,j)`**
`Row` (Рядок)	Цілий рядок робочого аркуша	**`Rows(10)`**
`Column` (Стовпець)	Цілий стовпець робочого аркуша	**`Columns("A"), Columns(5)`**

Властивості – це значення, константи або змінні, що належать об'єкту та описують його стан, характеристики чи дані (табл. 5.2).

Звернення:

```
Об'єкт.Властивість
```

Встановлення значення властивості:

```
Об'єкт.Властивість = Вираз
```

Отримання значення властивості:

```
Змінна = Об'єкт.Властивість
```

Приклади:

```
Application.Workbooks("File.xlsx").Worksheets("Sheet1").Range("A1
                    ").Value = 100
                  Range("A1").Value = 100
                   Xn = Cells(1, 2).Value
                   Xn = Range("B1").Value
```

Таблиця 5.2 – Основні властивості ключових об'єктів VBA

Об’єкт	Властивість	Опис	Приклад VBA
Range	**.Value**	Значення, записане або обчислене в комірці	`Range("B1").Value`
Range	**.Formula**	Формула, записана в комірці	`Range("C1").Formula = "=A1+B1"`
Range	**.Font**	Об'єкт, що містить атрибути шрифту (колір, розмір, жирність)	`Range("A1").Font.Bold = True`
Range	**.Borders**	Об'єкт, що керує рамками діапазону	`Range("A1:B2").Borders.LineStyle = xlContinuous`
Range	**.Interior**	Об'єкт, що керує кольором заливки	`Range("A1").Interior.Color = RGB(255, 0, 0)`
Range	**.NumberFormat**	Задає формат відображення числових даних (наприклад, відсотки, дата)	`Range("B3").NumberFormat = "0.00%"`
Workbook	**.Name**	Назва файлу робочої книги	`MsgBox Workbooks(1).Name`
Workbook	**.Sheets**	Колекція всіх аркушів (Worksheets, ChartSheets тощо) у книзі	`Workbooks(1).Sheets.Count`
Worksheet	**.Name**	Назва аркуша, яка відображається на вкладці	`Sheets("Sheet1").Name = "Дані"`
Worksheet	**.UsedRange**	Діапазон комірок, які містять дані або форматування	`Set usedArea = ActiveSheet.UsedRange`
Worksheet	**.Cells**	Колекція всіх комірок на аркуші. Використовується для посилань за номерами рядка та стовпця	`ActiveSheet.Cells(1,1).Value = "A1"`
Chart	**.ChartType**	Тип діаграми (наприклад, лінійчата, стовпчаста, кругова)	`ActiveChart.ChartType = xlLine`

Якщо властивості описують стан об'єкта (колір, назва, розмір), то методи – описують дії, які об'єкт може виконати.

Методи – це процедури (макроси або функції), пов'язані з об'єктом, які виконують над ним певні дії (табл. 5.3). Вони змінюють властивості об'єкта або змушують його виконувати функцію.

Виклик:

```
Об'єкт.Метод
```

Наприклад:

`ThisWorkbook.Close` – закрити книгу

`Range("A1").ClearContents` – очистити вміст комірки

`Application.Workbooks.Add.SaveAs "НоваКнига.xlsx"` – створення нової книги та збереження

`Worksheets("Звіт").Range("B5").Activate` – активація комірки на конкретному аркуші

Таблиця 5.3 – Основні методи ключових об'єктів VBA

Об’єкт	Метод	Опис	Приклад VBA
Application	**.Quit**	Завершує роботу програми Excel.	`Application.Quit`
Application	**.Calculate**	Примусово перераховує всі відкриті робочі книги.	`Application.Calculate`
Workbook	**.Save**	Зберігає робочу книгу	`ActiveWorkbook.Save`
Workbook	**.Close**	Закриває робочу книгу, опціонально зберігаючи зміни	`ThisWorkbook.Close SaveChanges:=True`
Workbook	**.Activate**	Робить цю робочу книгу активною (видимою для користувача)	`Workbooks("Звіт.xlsx").Activate`
Worksheets	**.Add**	Додає новий аркуш у колекцію (у робочу книгу)	`Worksheets.Add After:=Worksheets(Sheets.Count)`
Worksheet	**.Delete**	Видаляє робочий аркуш.	`Sheets("Лист2").Delete`
Worksheet	**.Copy**	Копіює аркуш в іншу робочу книгу або в новий файл	`Sheets(1).Copy Before:=Workbooks("Нова.xlsx").Sheets(1)`
Range	**.Select**	Виділяє діапазон (переводить на нього курсор)	`Range("A1:C10").Select`
Range	**.ClearContents**	Видаляє лише вміст комірок (значення та формули), зберігаючи форматування	`Range("E:E").ClearContents`
Range	**.Copy**	Копіює вміст діапазону в буфер обміну	`Range("A1").Copy`
Range	**.PasteSpecial**	Вставляє дані з буфера обміну зі спеціальними параметрами (наприклад, лише значення)	`Range("B1").PasteSpecial xlPasteValues`
Range	**.Sort**	Сортує діапазон за вказаними критеріями	`Range("A1:D10").Sort Key1:=Range("B1")`

Подія (Events) – це зовнішній вплив або дія, на який об'єкт може реагувати шляхом запуску відповідного коду (обробника подій) (табл. 5.4).

Обробник події (Event Handler) – це спеціальна процедура `Sub`, назва якої відповідає об'єкту (Application, Workbook, Worksheet, UserForm) та події (наприклад, `Worksheet_Change`).

В VBA обробники подій завжди мають стандартний синтаксис:

```
Private Sub Об'єкт_Подія(Аргументи)
    ' Код, який виконується
End Sub
```

Таблиця 5.4 – Основні події до ключових об'єктів VBA

Подія	Опис
`Workbook_SheetActivate(Sh As Object)`	Запускається, коли користувач переходить на будь-який аркуш цієї книги. Використовується для оновлення даних або елементів інтерфейсу при зміні аркуша
`Workbook_NewSheet(Sh As Object)`	Запускається при створенні нового аркуша. Використовується для автоматичного налаштування нового аркуша (наприклад, встановлення заголовків чи формату)
`Workbook_Open`	Запускається при відкритті книги. Використовується для ініціалізації: налаштування інтерфейсу, перевірка даних, запуск форм, підключення до баз даних
`Worksheet_Change(ByVal Target As Range)`	Запускається, коли користувач змінює вміст однієї або кількох комірок. `Target` – це діапазон змінених комірок. Використовується для динамічної зміни форматування, перевірки даних, автоматичних розрахунків
`Worksheet_SelectionChange(ByVal Target As Range)`	Запускається, коли змінюється виділений діапазон. Використовується для відображення контекстної інформації, підсвічування рядків/стовпців або створення ефектів "фокусу" при роботі з даними.
`UserForm_Initialize`	Запускається при завантаженні форми в пам'ять, але до її відображення
`UserForm_Activate`	Запускається, коли форма стає активною (після `Initialize` або після повернення до неї з іншого вікна)
`UserForm_QueryClose(Cancel As Integer, CloseMode As Integer)`	Запускається перед закриттям форми. Використовується для підтвердження виходу або перевірки, чи всі необхідні дані були введені. Встановлення `Cancel = 1` скасовує закриття
`CommandButton1_Click`	Запускається, коли користувач натискає на кнопку. Використовується для запуску будь-якої основної функції: виконання макросу, збереження даних, відкриття іншої форми
`App.WorkbookOpen(Wb As Workbook)`	Запускається при відкритті будь-якої книги в цьому екземплярі Excel
`App.NewWorkbook(Wb As Workbook)`	Запускається при створенні нової книги

У ООП-моделі Excel часто використовується поняття колекції.

Колекція – це об'єкт, який містить групу однотипних об'єктів. Колекції дозволяють зручно перебирати (ітерувати) усі об'єкти групи:

- колекція `Workbooks` містить усі відкриті об'єкти `Workbook`;
- колекція `Worksheets` містить усі об'єкти `Worksheet` у поточній книзі;
- колекція `Charts` містить усі діаграми на аркуші.

Приклад перебору усіх аркушів у поточній книзі (цикл `For Each`):

```
Dim ws As Worksheet
For Each ws In ThisWorkbook.Worksheets
    ws.Visible = xlSheetVisible ' Зробити аркуш видимим
Next ws
```

Використання колекції дозволяє застосовувати методи та властивості до великої кількості об'єктів одночасно.

5.4 Представлення даних: типи даних, константи та змінні, масиви

Константи – це явно задані або означені дані. Вони об'являються зі словом `Const`.

Для завдання константи достатньо записати службове слово `Const`, після якого її ім'я, потім знак "=" і саме значення. Текстові константи беруться у лапки. Наприклад:

```
Const Пі_5 = 3.14159
Const Екзамен = "Інформатика"
Дні_тижня = ("Понеділок", "Вівторок", "Середа", "Четвер", "П‘ятния", "Субота", "Неділя")
```

Змінні – це дані, значення яких можна змінювати при виконанні програми. Для завдання змінних використовується службове слово `Dim`, після якого ставиться її ім'я, потім службове слово `As` і вказується тип змінної.

Наприклад:

```
Dim Площа As Double
Dim Ширина
Dim i, g23, k3_a As Integer
Dim X As Byte, Z As Integer, C, Слово As String
```

Тип `Variant` використовується, якщо його задано явно, або коли інше не вказано. Змінні цього типу можуть приймати значення замість будь-якого з інших простих типів.

Окремим видом змінних є **масиви**. При їх завданні після імені змінної ставиться у дужках список інтервалів зміни індексів (у простішому випадку – кількість елементів у масиві).

Наприклад:

```
Dim AA(50)
Dim BB3(1 To 5, 4 To 9, 3 To 5) As Double
```

Якщо треба створити масив з наперед заданими елементами, можна скористатися функцією `Array`. Але слід пам'ятати, що вона повертає не справжній масив, а змінну типу `Variant`, що містить у себе заданий масив у символьному вигляді.

Типи даних розділяються на прості (скалярні) і складні (структуровані). В простих типах даних можливі значення даних єдині і неділимі. Складні ж типи мають структуру, в яку входять різні прості типи даних. Скалярні типи даних представлені в Таблиці 5.5.

Таблиця 5.5 – Скалярні типи даних VBA

Ім‘я типу	Можливі значення
Boolean (логічний)	True, False
Byte	0…255
Integer (ціле)	-32768…+32767
Long (довге ціле)	-2147483648…+2147483647
Single (число з плаваючою точкою)	-3,4E38…-1,4E-45 для негативних значень; 1,4E-45…3,4E38 для позитивних значень
Double (число з плаваючою точкою подвійної точності)	-1,7E308…-4,9E-324 для негативних значень; 4,9E-324…1,7E308 для позитивних значень
String (рядковий)	рядки фіксованої довжини (до 216 символів); рядки змінної довжини (до 231символов). Дані записуються в лапках
Object (об‘єкт)	Посилання на об‘єкт (покажчик)
Variant	Універсальний тип; значенням можуть бути дані будь-якого з перерахованих вище типів, значення NULL і значення помилок ERROR

5.5 Керуючі конструкції VBA: оператори розгалуження, вибору та циклу

Керуючі конструкції мови програмування − це інструкції та групи інструкцій, застосування яких дозволяє змінювати послідовність виконання інших інструкцій програми. Керуючі конструкції підрозділяються на розгалуження і цикли.

5.5.1 Розгалуження

Оператор розгалуження перевіряє задану умову та відповідно до її істинності виконує перший або другий операнди. У Visual Basic розгалуження обчислень можна організовувати трьома ос-новними способами із застосовуванням ключових слів: `If`, `Elseif`, `SelectCase`

конструкція `If… Then`

```
If  умова  Then
    оператор
    . . .
    оператор
End if
```

конструкція `If… Then…Else`

```
If  умова  Then
    оператор
    . . .
    оператор
Else
    оператор
    . . .
    оператор
End if
```

конструкція `If...ElseIf...EndIf`

```
If  умова1  Then
    оператор
    . . .
    оператор
ElseIf  умова2  Then
```

конструкція `Select Case`

```
Select Case вираз, що перевіряється
   Case діапазон значень1
        оператор
        . . .
        оператор
```

```
        оператор
        . . .
        оператор
    . .
    Else
        оператор
        . . .
        оператор
    End if
```

```
    Case діапазон значень2
        оператор
        . . .
        оператор
    Case Else
        оператор
        . . .
        оператор
End Select
```

Оператор `Select Case` працює за принципом порівняння: значення початкового `виразу` обчислюється, а потім перевіряється на відповідність умовам, зазначеним після кожного ключового слова `Case`. Виконується код лише того блоку, чий діапазон або умова відповідає значенню `виразу`.

Діапазони значень можуть бути задані трьома основними способами:

– переліком (через кому);

– проміжком (від... до...) за допомогою `To` (наприклад, `10 To 20`);

– логічною умовою за допомогою `Is` та знака порівняння (наприклад, `Is > 200`).

Приклад. Записати процедуру VBA для обчислення значення функції. В комірку A1 (Аркуш 1) табличного процесору Excel записати значення змінної ***x***, в комірку C1 (Аркуш 1) – значення функції (***y***):

$$y = \begin{cases} sinx + |cosx|, & x \le 0,5 \\ x^2 + 1, & 0,5 < x < 0,7 \\ cos^5 x, & x \ge 0,7 \end{cases}$$

Результат:

```
    Sub CalculateFunctionY()
        Dim x As Double   ' Змінна для вхідного значення (з
комірки A1)
        Dim y As Double  ' Змінна для результату функції
        Dim ws As Worksheet ' Змінна для посилання на Аркуш 1
           ' Встановлення посилання на Аркуш 1
        Set ws = ThisWorkbook.Sheets("Аркуш1")
            ' Зчитуємо значення x з комірки A1
          x = ws.Range("A1").Value
           If x <= 0.5 Then
                 y = Sin(x) + Abs(Cos(x))
          ElseIf x < 0.7 Then
                 y = x ^ 2 + 1
          Else
                 y = Cos(x) ^ 5
        End If
            ' Відображаємо обчислене значення y в діалоговому
вікні
        MsgBox y
            'Записуємо результат y в комірку C1
        ws.Range("C1").Value = y

    End Sub
```

5.5.2 Цикли

Цикл – це керуюча конструкція, яка дозволяє виконувати послідовність інструкцій багаторазово, доки не буде виконана певна умова або доки не буде пройдено задану кількість кроків.

У VBA існують декілька типів циклів:

1. Цикл **`For...Next`** (цикл з лічильником). Використовується, коли кількість повторень відома заздалегідь.

Синтаксис:

```
For Лічильник = Початок To Кінець [Step Крок]
      оператор
      . . .
      оператор
Next [Лічильник]
```

Приклад: Заповнення комірок та використанням `Step`

```
Sub Example_For_Next_Loop()
 'Приклад 1: Просте заповнення стовпця A
      Dim i As Long
        For i = 1 To 5
 'Заповнюємо комірки A1, A2, A3, A4, A5
        Worksheets("Sheet1").Cells(i, 1).Value =
"Значення: " & i
        Next i
 'Приклад 2: Використання ключового слова Step (крок)
      Dim j As Long
        'Цикл виконається для j=1, 3, 5, 7, 9
        For j = 1 To 10 Step 2
 'Заповнюємо B1, B3, B5, B7, B9
        Worksheets("Sheet1").Cells(j, 2).Value = "Крок 2:
" & j
        Next j
 MsgBox "Комірки A1:A5 та B1, B3, B5, B7, B9 заповнені."
End Sub
```

2. Цикл **`Do While / Do Until`** (цикл з передумовою). Використовується, коли кількість повторень невідома заздалегідь, і цикл залежить від виконання (або невиконання) певної логічної умови.

Синтаксис **`Do While`**:

```
Do While Умова
  оператор
   . . .
  оператор
Loop
```

Цикл виконується, доки умова істинна, інакше цикл завершується.

Синтаксис **`Do Until`**:

```
Do Until Умова
  оператор
   . . .
  оператор
Loop
```

Цикл виконується, якщо умова хибна, інакше цикл завершується.

Приклад. Знайти перше ціле число (починаючи з 1), сума якого та всіх попередніх чисел перевищить 50.

Результат:

```
Sub FindSum_DoWhile()
    Dim Sum As Variant
    Dim i As Integer
    Const Sum1 As Integer = 50
      Sum = 0
      i = 0
      ' Цикл Do While: виконується, ДОКИ умова ІСТИННА
      ' Умова: Sum <= Sum1 (сума не перевищила 50)
         Do While Sum <= Sum1
             i = i + 1
             Sum = Sum + i
         Loop
     ' виведення результату
     MsgBox Sum
End Sub
```

Якщо використовувати цикл **Do Until**, то в представленому коді зміниться лише частина перевірки умови:

```
         Do Until Sum >= Sum1
             i = i + 1
             Sum = Sum + i
         Loop
```

3. Цикл **Do...Loop While / Do...Loop Until** (цикл з постумовою). Використовується, коли гарантовано необхідно виконати тіло циклу хоча б один раз (наприклад, для отримання вхідних даних, які потім будуть перевірені).

Для попереднього приклада фрагменти програмного кода в частині перевірки умови виглядатимуть:

```
Do                              Do
   i = i + 1                       i = i + 1
   Sum = Sum + i                   Sum = Sum + i
   Loop Until Sum >= Sum1          Loop While Sum <= Sum1
```

4. Цикл **For Each...Next** є одним із зручніших інструментів у VBA, особливо коли необхідно працювати з об'єктами Excel.

Ця структура призначена для перебору елементів у колекції (структурі даних) без необхідності використовувати числовий лічильник.

Синтаксис:

```
  For Each елемент In колекція_об'єктів
    ' Блок операторів, які виконуються для кожного елемента
  Next елемент
```

Наприклад, необхідно встановити ширину стовпця А на всіх аркушах у поточній робочій книзі Excel.

Результат:

```
Sub FormatAllSheetsColumnA()
   Dim ws As Worksheet ' змінна ws - об'єкт Worksheet
    ' Цикл: Для кожного аркуша (ws) в колекції Worksheets поточної книги
    For Each ws In ThisWorkbook.Worksheets
          ' 1. Встановлюється ширина стовпця A
        ws.Columns("A").ColumnWidth = 15
         ' 2. Назва аркуша записується у комірку A1
        ws.Range("A1").Value = ws.Name
    Next ws ' Переходимо до наступного об'єкта 'ws' у колекції
    MsgBox "Ширина стовпця A була змінена на всіх аркушах."
End Sub
```

5.6 Введення-виведення повідомлень в діалогових вікнах

5.6.1 MsgBox

Функція `Msgbox` здійснює виведення інформації в діалоговому вікні і встановлює режим чекання натиснення кнопки користувачем (рис. 5.7). У VBA функція `MsgBox` може використовуватися двома способами, і це впливає на те, чи потрібні круглі дужки.

1. Використання як Підпрограма (Statement)

Якщо необхідно показати повідомлення і не цікавитися відповіддю користувача, дужки не використовуються. У цьому режимі MsgBox працює як команда і не повертає значення.

Синтаксис:

```
MsgBox Prompt, [Buttons], [Title]
```

2. Використання як Функція (Function)

Якщо необхідно отримати відповідь від користувача (наприклад, "Так", "Ні", "Скасувати"), дужки використовуються навколо всіх аргументів, а результат роботи функції має бути присвоєний змінній.

У цьому режимі `MsgBox` повертає значення (тип `VbMsgBoxResult`).

Синтаксис:

```
Змінна = MsgBox(Prompt, [Buttons], [Title])
```

– `Prompt` (повідомлення) – обов'язковий аргумент, який задає у вікні інформаційне повідомлення, що виводиться. Може складатися з декількох текстових рядків, об'єднаних знаком &;

– `Buttons` (кнопки) – значення цього аргументу визначає категорії кнопок, що з'являються у вікні. Від значення аргументу кнопки залежить також, чи з'являється у вікні який-небудь значок. Якщо не вказано, які кнопки необхідно відображувати у вікні повідомлень, то використовується значення за замовчуванням, відповідне кнопці ОК;

– `Title` (заголовок) – задає заголовок вікна.

Кнопки поділяються на групи, і їх можна комбінувати знаком +:

vbOKOnly – тільки кнопка ОК;

vbOKCancel – кнопки ОК та Скасувати;

vbYesNo – кнопки Так та Ні;

vbYesNoCancel – кнопки Так, Ні та Скасувати;

vbCritical – критична помилка (значок **X**);

vbQuestion – запит (значок **?**);

vbExclamation – попередження (значок **!**);

vbInformation – інформація (значок **i**);

vbDefaultButton1 – перша кнопка стає кнопкою за замовчуванням;

vbDefaultButton2 – друга кнопка стає кнопкою за замовчуванням.

Якщо MsgBox використовується як функція для отримання відповіді, вона поверне одну з наступних констант:

vbOK – ОК;

vbCancel – Скасувати;

vbAbort – Перервати;

vbRetry – Повторити;

vbIgnore – Ігнорувати;

vbYes – Так;

vbNo – Ні.

Приклад процедури з використанням оператора Msgbox (просте сповіщення та для отримання відповіді користувача):

```
Sub MsgBox_Demo()
  ' Випадок 1: MsgBox як Підпрограма - БЕЗ ДУЖОК
  ' Виводимо інформацію, не очікуючи на повернення результату
    MsgBox "Починаємо процес обробки даних...", , "Старт"

    ' Випадок 2: MsgBox як Функція (Function) - З ДУЖКАМИ
    ' Отримуємо відповідь користувача
    Dim Response As VbMsgBoxResult ' Спеціальний тип даних у
VBA, який називається перерахуванням
    Response = MsgBox("Продовжити? (Так/Ні)", vbYesNo +
vbQuestion, "Запит")
  ' Аналіз відповіді, використовуючи функціональний результат
    If Response = vbYes Then
  ' Повідомлення про продовження (як Підпрограма - БЕЗ ДУЖОК)
        MsgBox "Добре, продовжуємо роботу.", vbInformation,
"Продовження"
    Else
  ' Повідомлення про завершення (як Підпрограма - БЕЗ ДУЖОК)
        MsgBox "Роботу завершено.", vbExclamation, "Вихід"
    End If
End Sub
```

Результат роботи процедури:

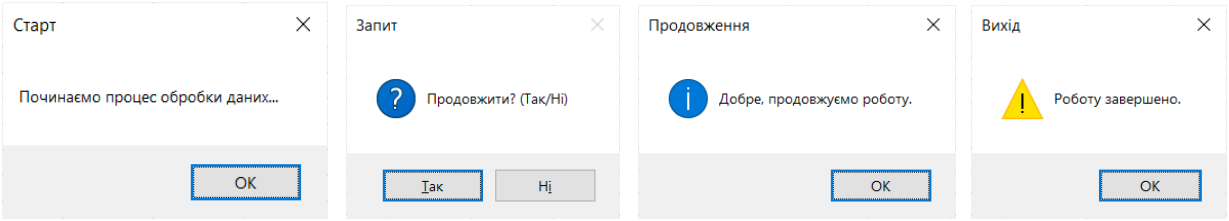


Рисунок 5.7 – Результат роботи процедури `MsgBox_Demo`

5.6.2 InputBox

Функція `Inputbox` здійснює введення значень змінних за допомогою діалогового вікна вводу і має синтаксис:

```
Ім‘я_змінної = InputBox(Prompt, [Title], [Default])
```

Аргументи:

– `Prompt` (повідомлення) – обов'язковий аргумент. Текстовий рядок, що відображається у вікні (запитання до користувача);

– `Title` (заголовок) – задає заголовок вікна;

– `Default` (за замовчуванням) – значення, яке заздалегідь відображається у полі введення (значення за замовчуванням).

Приклад: Ввести значення змінної N з клавіатури, передбачивши значення за замовчуванням рівне 10.

`N = InputBox("Введіть N", "Введення вхідних даних", 10)` В результаті буде виведено наступне вікно для вводу значення змінної N:

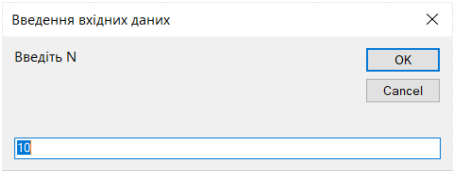


Рисунок 5.8 – Результат роботи `InputBox`

Отже, InputBox служить основним засобом, що дозволяє програмам VBA отримувати (запитувати) інформацію безпосередньо від користувача.

5.7 Функції та процедури

Процедури (`Sub`) та функції (`Function`) є фундаментом будь-якої програми на VBA, оскільки вони дозволяють розбивати великий код на логічні, багаторазові блоки.

Процедурою називається фрагмент коду (мінімальна семантично закінчена програмна конструкція), розміщений між операторами `Sub` і `End Sub`.

Спрощений синтаксис процедури такий:

```
[Private|Public] Sub ім'я_процедури (аргумент_1, ... ,аргумент N)
                    оператор
                    ...
```

```
    оператор
End Sub
```

Основною відмінністю між процедурою та функцією є те, що функція повертає значення. Синтаксис функцій трохи відрізняється від синтаксису процедур, тому що для функції необхідно вказати тип значення, що повертається нею:

```
[Private|Public] Function ім'я_функції (аргумент_1, ... ,
аргумент N) As тип_значення_що_повертається>
    оператор
      ...
    оператор
    ім'я_функції = обчислене_значення
End Function
```

Ключове слово `Private` обмежує область видимості процедури або функції, для якої воно задане, модулем, у якому вона описана. У результаті її можуть викликати тільки процедури або функції цього ж модуля.

Ключове слово `Public`, навпаки, оголошує процедуру або функцію доступною для всіх модулів проекту. За замовчуванням будь-яка процедура або функція загальнодоступна, тобто має статус `Public`.

Приклад 1. Обчислити площу прямокутника та повернути результат; застосувати функцію.

Результат:

```
Function Calc_Area(Length As Double, Width As Double) As
Double
    Dim Area As Double
    Area = Length * Width
    Calc_Area = Area
End Function
```

Приклад 2. Записати процедуру обчислення площі прямокутника. отримати вхідні дані, викликати функцію для обчислення, та вивести результат користувачеві.

Результат:

```
Sub Proc_Area_1 ()
    Dim L As Double
    Dim W As Double
    Dim ResultArea As Double
    L = InputBox("Введіть довжину:", "Введення даних")
    W = InputBox("Введіть ширину:", "Введення даних")
    ' Виклик функції та використання її повернутого значення
    ' Значення, яке повертає функція C_Area, зберігається у
змінній ResultArea
    ResultArea = Calc_Area(L, W)
    ' Виведення результату
    MsgBox "Площа прямокутника зі сторонами " & L & " та " &
W & " дорівнює: " & ResultArea, vbInformation, "Результат
обчислення"
```

```
End Sub
```

Однією з можливостей VBA є створення нової функції MS Excel, яку потім можна використовувати аналогічно вбудованим функціям (SUM, IF тощо). Створену функцію можна обрати з категорії ***Визначені користувачем*** (рис. 5.9) або ввести ім'я функції безпосередньо у рядок формул, та отримати значення функції.

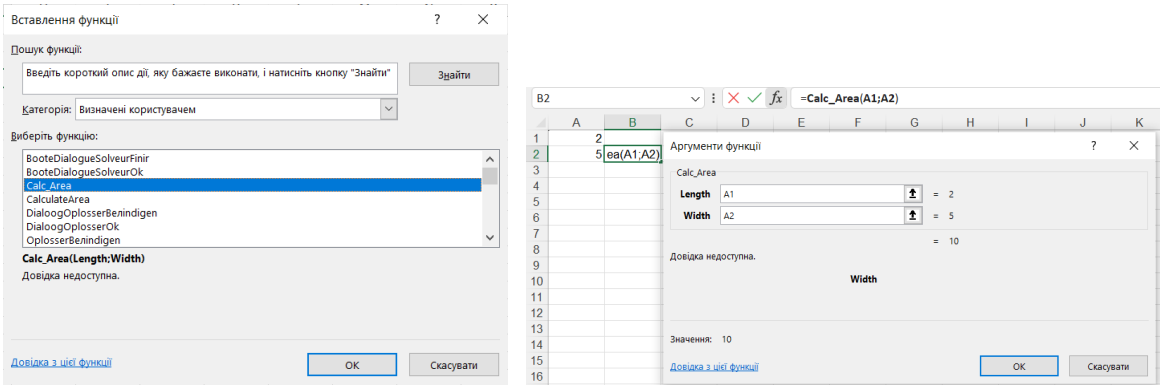


Рисунок 5.9 – Використання функції, визначеної користувачем

5.8 Створення UserForm

`UserForm` це спеціальний об'єкт у VBA, який є порожнім діалоговим вікном, на якому можна розміщувати різні **елементи управління** (такі як кнопки, текстові поля, списки), щоб створити власний інтерфейс для взаємодії з користувачем.

Для створення `UserForm` необхідно:

– відкрити Редактор VBA (Alt + F11 в Excel);

– додати UserForm; у вікні *Project Explorer* (ліворуч) натиснути правою кнопкою миші на книгу (`VBAProject(Книга1.xlsm)`, обрати **Insert/UserForm** (рис. 5.10);

– з'являться два нових вікна:

- `UserForm1` – порожнє діалогове вікно;
- `Toolbox` (Панель інструментів) – містить елементи керування (кнопки, текстові поля тощо).

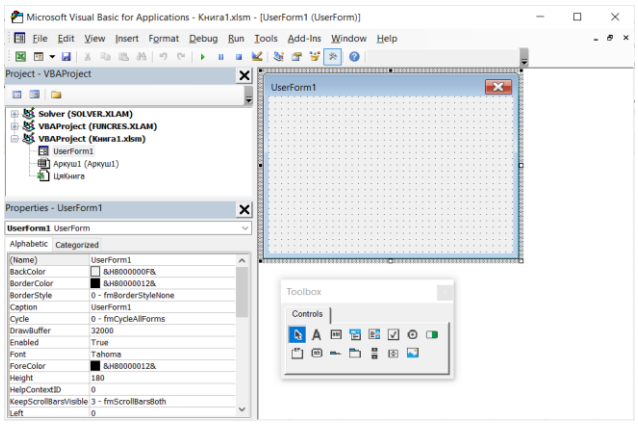


Рисунок 5.10 – Редактор VBA з `UserForm`

Кожному об'єкту `UserForm` притаманні певні **властивості** (табл. 5.6), **методи** та **події**, які він успадковує від класу об'єктів `UserForms`.

`UserForm` має сотні властивостей, але для створення робочих та естетичних форм достатньо знати лише кілька ключових. Вони поділяються на властивості, що впливають на саму форму (контейнер), та властивості, що впливають на її елементи керування (кнопки, поля).

Таблиця 5.6 – Основні властивості форми

Властивість	Опис
(Name)	Унікальне ім'я, яке використовується для посилання на форму в коді (наприклад, **frmInput.Show**)
Caption	Текст, який відображається у рядку заголовка форми. Це те, що бачить користувач (наприклад, "Введення даних клієнта").
Height & Width	Визначається висота та ширина форми у пунктах
BackColor	Колір фону форми. Зазвичай встановлюється для естетики (наприклад, &**H8000000F**& – стандартний світло-сірий колір системи).
ShowModal	Визначає, чи є форма модальною (користувач не може взаємодіяти з Excel (клітинами, меню), доки форма відкрита; `True` – за замовчуванням) чи немодальною (Користувач може працювати в Excel, поки форма відкрита; `False`)

5.9 Елементи керування UserForm

Редактор Visual Basic надає спеціальну панель інструментів `Toolbox`, з елементами керування. Також він надає можливість візуального визначення розмірів елементів керування та їх розміщення у визначеній області форми (рис. 5.11).

Елемент керування **Напис** (`Label`) використовують для виведення необхідного тексту у певній області форми для пояснення інших елементів керування або виведення повідомлень користувачеві.

Елемент керування **Текстове поле** (`TextBox`) використовується для введення даних користувачем, які, зазвичай, оброблюються у процедурах модуля форми.

Елемент керування **Кнопка** (`CommandButton`) безумовно є найчастіше використовуваним елементом керування. Взаємодія з ним, зазвичай, запускає програму обробки введених користувачем даних у інші елементи керування, а також ініціює різноманітні дії, такі як закриття форми, скасування обробки даних тощо.

Елементи керування **Прапорець** (`CheckBox`) та **Кнопка з фіксацією** (`ToggleButton`) використовуються, коли необхідно забезпечити вибір варіантів із деякої їх кількості, при цьому можливий вибір від одного до усіх варіантів одночасно.

Елементи керування **Перемикач** (`OptionButton`) та **Рамка** (`Frame`). На відміну від Прапорців та Кнопок з фіксацією, Перемикач використовують для обрання одного з варіантів, при цьому декілька Перемикачів об'єднують у групу за допомогою Рамки.

Елементи керування **Смуга прокрутки** (`ScrollBar`) та **Лічильник** (`SpinButton`). Зі Смугами прокрутки ми вже стикалися при вивченні властивостей Текстового поля: його властивість `ScrollBars` дозволяє додавати або горизонтальну, або вертикальну, або обидві Смуги прокрутки, що є корисним у разі, якщо введений к поле текст не вміщується на екрані.

Додання рисунків до форм підвищує естетичність їх сприйняття і може бути використане для реалізації певних функцій. Розміщувати Рисунок в одному з поширених форматів (`.bmp`, `.cur`, `.gif`, `.ico`, `.jpg`, `.wmf`) на формі дозволяє елемент керування **Рисунок** (`Image`).

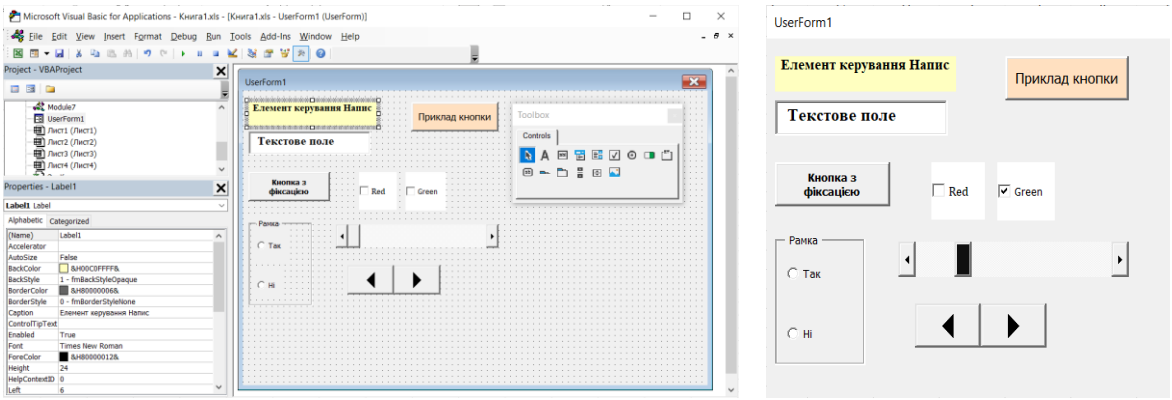


Рисунок 5.11 Використання інструментів **`Toolbox`**

Властивості елементів управління є спільними для більшості елементів, таких як `TextBox`, `Label` та `CommandButton`.

Таблиця 5.7 – Основні властивості елементів керування

Властивість	Опис
`(Name)`	Унікальне ім'я, яке використовується для читання чи запису значень у коді
`Value`	Значення елемента: для **`TextBox`** – текст, який ввів користувач; для **`CheckBox`** – `True/False`; для **`OptionButton`** – `True/False` (стан вибору)
`Caption`	Підпис/Текст: для **`Label`** – текст, що описує поле поруч (наприклад, "Введіть Ім'я:"); для **`CommandButton`** – текст кнопки (наприклад, "Зберегти", "Скасувати")
`Left & Top`	Визначають горизонтальне та вертикальне положення елемента на формі
`Enabled`	Визначає, чи може користувач взаємодіяти з елементом (`True/False`)
`Locked`	Визначає, чи можна **редагувати** вміст елемента (`True/False`). Якщо `Locked = True`, користувач може бачити вміст `TextBox`, але не може його змінити
`Visible`	Визначає, чи відображається елемент на формі (`True/False`)

5.10 Методи та події об’єктів UserForm

Метод – це процедура, пов'язана з об'єктом, яка виконує певну дію над цим об'єктом або від його імені. Якщо властивість описує об'єкт (наприклад, колір, розмір), то метод змушує об'єкт щось робити (наприклад, сховатися, змінити положення).

Об'єкт `UserForm` (сама форма) має методи, які керують його відображенням і станом (табл. 5.8).

Таблиця 5.8 – Основні методи об’єкта форми `UserForm`

Метод	Опис
`.Show`	Відображає форму на екрані. Може приймати аргумент `vbModal` (форма блокує Excel) або `vbModeless` (Excel залишається активним): `UserForm1.Show vbModal`
`.Hide`	Приховує форму, але не вивантажує її з пам'яті. Це дозволяє зберігати дані, введені користувачем: `Me.Hide`
`.PrintForm`	Друкує вміст форми на принтері за замовчуванням: `UserForm1.PrintForm`
`.Move`	Переміщує форму на екрані та/або змінює її розмір: `UserForm1.Move 100, 50, 400, 300` (ліворуч=100, верх=50, ширина=400, висота=300)

Різні елементи керування (`TextBox`, `ListBox`, `CommandButton` тощо) мають свої унікальні методи (табл. 5.9).

Таблиця 5.9 – Основні методи елементів керування форми `UserForm`

Метод	Опис
`.AddItem`	Додає новий елемент (рядок) до списку: `ListBox1.AddItem "Нове значення"`
`.RemoveItem`	Видаляє елемент за його індексом (нумерація починається з 0): `ListBox1.RemoveItem 2` (видаляє третій елемент)
`.Clear`	Видаляє всі елементи зі списку `ListBox1.Clear`
`.SetFocus`	Встановлює фокус на елемент керування, роблячи його активним (наприклад, текстове поле готове до введення) `TextBox1.SetFocus`
`.Move`	Переміщує або змінює розмір контрола в межах форми (аналогічно методу форми) `CommandButton1.Move 5, 5`

Подія (Event) – це дія, яка відбувається в об'єкті (наприклад, натискання кнопки, зміна тексту, завантаження форми), і яку VBA може розпізнати. Кожна подія має пов'язану з нею процедуру-обробник (Event Handler), куди програміст поміщає код, який має бути виконаний у відповідь на цю дію.

Синтаксис процедури події:

```
Private Sub Об'єкт_Подія([Аргументи])
        ' Програмний код
        End Sub
```

Події, які керують життєвим циклом форми представлено в Таблиці 5.10.

Таблиця 5.9 – Основні події об’єкта `UserForm`

Подія	Опис
`Initialize`	Ініціалізація змінних, заповнення списків (ListBox, ComboBox) початковими даними; спрацьовує один раз, коли форма завантажується в пам'ять (але ще не відображається).
`Activate`	Оновлення даних, якщо форма була неактивною; спрацьовує кожного разу, коли форма стає активним вікном
`Terminate`	Звільнення об'єктів або збереження даних перед закриттям; спрацьовує коли форма вивантажується з пам'яті (наприклад, після `Unload Me`)
`QueryClose`	Дозволяє скасувати закриття форми або виконати перевірку даних

Кнопки (`CommandButton`) – найпростіші елементи з точки зору подій. Ключовою подією кнопок є подія `Click`, яка спрацьовує коли користувач натискає і відпускає ліву кнопку миші на кнопці. Типовим використанням є запуск обчислень, збереження даних, перехід на іншу форму.

Приклад: створено форму, на якій розміщено `TextBox`, `ListBox`, `CommandButton`. Необхідно додати значення з текстового поля у список та встановити курсор назад у текстове поле.

```
Private Sub CommandButton1_Click()
    ' Перевірка на порожнє поле
        If TextBox1.Value <> "" Then
            ' Додаємо значення з текстового поля до ListBox
              ListBox1.AddItem TextBox1.Value
            ' Очищаємо текстове поле після додавання
              TextBox1.Value = ""
       End If
    ' Встановлюємо курсор назад у текстове поле
    TextBox1.SetFocus
End Sub
```

Отже, створення інтерактивних `UserForm` у VBA є фундаментальним кроком в офісному програмуванні, перетворюючи статичні документи на динамічні додатки.

Питання для самоконтролю до теми 5

1. Які об’єкти має об’єктна модель Excel?
2. Як розробити функцію користувача?
3. Методи об’єкту VBA – це…
4. Яким чином можна створити модуль робочого аркуша, модуль робочої книги?
5. Як називаються процедури, пов’язані з подіями робочого аркуша?
6. Об’єкти VBA – це…
7. Властивості об’єкту VBA– це…
8. Яким чином можна записати макрос?

9. Як звернутися у процедурі до діапазону комірок, окремої комірки?
10. Назвіть керуючі структури VBA.

Тестові завдання:

Для наведених завдань оберіть одну правильну відповідь:

1. Який об'єкт є найвищим у ієрархії Об'єктної Моделі Excel VBA?

а) Workbook
б) Worksheet
в) Application
г) Range

2. Що в об'єктно-орієнтованому підході VBA описує характеристики, стан або дані об'єкта (наприклад, колір або значення)?

а) Метод
б) Властивість (Property)
в) Подія (Event)
г) Клас

3. Як називається контейнер у Редакторі VBA (VBE), де зберігаються загальні процедури Sub та функції Function, доступні з будь-якого місця книги?

а) Клас
б) Колекція
в) Форма Користувача (UserForm)
г) Модуль

4. Який синтаксис є правильним для оголошення змінної `i` як цілого числа у VBA?

а) `Set i As Integer`
б) `Dim i : Integer`
в) `Dim i As Integer`
г) `i = Integer`

5. Яка функція VBA використовується для відображення простого повідомлення користувачеві з можливістю натискання OK?

а) `Alert()`
б) `InputBox()`
в) `MsgBox()`
г) `ShowMessage()`

6. Яка подія запускається, коли користувач змінює вміст будь-якої комірки на робочому аркуші?

а) `Worksheet_Click`
б) `Worksheet_SelectionChange`
в) `Worksheet_Open`
г) `Worksheet_Change`

ВИКОРИСТАНА ЛІТЕРАТУРА

1. Басюк Т.М., Думанський Н.О., Пасічник О.В. Основи інформаційних технологій: навч. посібник. Львів : «Новий Світ-2000», 2024. 390 с.
2. Бойко Т.В., Квітка О.О., Шахновський А.М. Комп'ютерна техніка та організація обчислювальних робіт: Організація розрахунків у середовищі MathCAD : навч. посібник. Київ : КПІ ім. Ігоря Сікорського, 2019. 64 с.
3. Гуржій А. М., Возненко Л. І., Поворознюк Н. І., Самсонов В. В. Основи інформаційних технологій : навч. посібник. Київ : Літера ЛТД, 2023. 288 с.
4. Трофименко О.Г., Прокоп Ю.В., Логінова Н.І., Чанишев Р.І. Офісні технології : навч. посібник. Одеса : Фенікс, 2019. 207 с.
5. Alexander M., Kusleika R., Walkenbach J. Excel 2019 Bible. John Willey & Sons, Inc., 2018. 1120 p.
6. Mansfield R. Mastering VBA for Microsoft Office 2019. John Wiley & Sons, Inc., 2019. 914 p.
7. Tittel E., Minnick C. Beginning HTML5 and CSS3 For Dummies. For Dummies, 2013. 384 p.
8. Weverka P. Office 365 All-in-One For Dummies (For Dummies (Computer/Tech)) 1st Edition. John Wiley & Sons, Inc., 2019. 848 p.

РЕКОМЕНДОВАНА ЛІТЕРАТУРА

1. Глинський Я.М. Інформатика : підручник. Львів : Видавництво Львівської політехніки, 2023. 394 с
2. Зачек О. І., Сеник В. В., Магеровська Т. В. Інформаційні технології : навч. посібник / за ред. О. І. Зачека. Львів : Львівський державний університет внутрішніх справ, 2022. 432 с.
3. Коломоєць Г. П. Офісне програмування мовою VBA : навч. посібник. Запоріжжя : Запорізький національний університет, 2023. 271 с.
4. Кравченко І. В., Микитинко В.І. Інформаційні технології. Київ : КПІ ім. Ігоря Сікорського, 2022. 447 с.
5. Meyer J. The Essential Guide to HTML5 Using Games to Learn HTML5 and JavaScript. Apress, 2022. 436 p.

Інформаційні ресурси

1. Портал Дія.Цифрова освіта. URL : https://osvita.diia.gov.ua
2. Розробка вебсайтів для початківців HTML - CSS - JAVASCRIPT. URL : https://nvkarta.com/project/library/uploads/engineering/programming/(uk)_rozrobka_veb-saitiv_dlia_pochatkivtsiv_html_css_javascript.pdf
3. LibreOffice. URL https://libreoffice.readthedocs.io/ru/latest/
4. MathCAD. URL : https://mathcad.com.ua/
5. Microsoft 365.: https://www.office.com/
6. OpenOffice. URL : https://www.open-of-course.org/courses/mod/url/view.php?id =826
7. Prometheus. URL : https://courses.prometheus.org.ua/

ЗМІСТ

ВСТУП 3
1 ТЕКСТОВИЙ ПРОЦЕСОР. ІНСТРУМЕНТИ ТА ЗАСОБИ РОБОТИ З КОМПЛЕКСНИМИ ТЕКСТОВИМИ ДОКУМЕНТАМИ 5
1.1 Основні можливості текстового процесора 5
1.1.1 Ключові структурні елементи та об'єкти 6
1.1.2 Базові операції текстового процесора 7
1.1.3 Форматування тексту 9
1.2 Засоби автоматизації розробки текстового документу 11
1.2.1 Стилі 11
1.2.2 Поля 11
1.2.3 Таблиці 13
1.2.4 Діаграми 14
1.2.5 Формули 14
1.3 Форматування сторінок документа 15
1.3.1 Параметри сторінки 15
1.3.2 Настроювання параметрів текстового документа 15
1.3.3 Розділи 16
1.3.4 Виноски 16
1.3.5 Колонтитули 16
1.3.6 Нумерація сторінок 17
1.4 Рецензування тексту документа 17
1.4.1 Мовні налаштування, перевірка правопису та граматики 17
1.4.2 Переклад 18
1.4.3 Примітки 19
1.4.5 Додавання та фіксування виправлень 19
Питання для самоконтролю до теми 1 21
Тестові завдання: 21
2 ОБРОБКА ТАБЛИЧНО-ОРГАНІЗОВАНОЇ ІНФОРМАЦІЇ 23
2.1 Основні поняття електронних таблиць 23
2.1.1 Робоча книга, робочий аркуш 23
2.1.2 Форматування вмісту комірок 23
2.2 Обчислення в Excel 24
2.2.1 Формули, функції 24
2.2.2 Посилання на комірки 28
2.3 Графічне представлення даних в Excel 29
2.3.1 Побудова діаграм 29
2.3.2 Редагування діаграм 29
2.4 Розв’язання задач зі змінними в Excel 30
2.4.1 Інструмент Добір параметра 30
2.4.2 Надбудова Пошук рішення 31
2.5 Обробка статистичних даних в Excel 32
2.5.1 Описова статистика 32
2.5.2 Кореляційний аналіз 33

2.5.3 Лінійна регресія та прогнозування 34
2.5.4 Типи ліній тренду для моделювання даних 34
Питання для самоконтролю до теми 2 35
Тестові завдання: 36
3 Основні інструменти та засоби роботи з математичним пакетом MathCAD 37
3.1 MathCAD та Mathcad Prime 37
3.2 Інтерфейс користувача математичного пакету MathCAD 37
3.3 Введення та виведення інформації 40
3.3.1 Створення формул 40
3.3.2 Вставка функцій 41
3.3.3 Створення тексту 42
3.3.4 Введення/виведення даних в текстові файли 42
3.4 Типи даних. Обчислення в MathCAD 43
3.4.1 Змінні 43
3.4.2 Символьні обчислення 45
3.5 Побудова та форматування графіків 45
Питання для самоконтролю до теми 3 47
Тестові завдання: 47
4 РОЗРОБКА WEB-САЙТУ ЗАСОБАМИ МОВИ HTML 49
4.1 Елементи, атрибути HTML-документа 49
4.2 Структура HTML-кода 50
4.3 Дизайн тексту та семантична розмітка 52
4.4 Створення списків та їх дизайн 54
4.5 Створення таблиць та їх дизайн 57
4.6 Додавання графічних файлів 59
4.7. Додавання гіперпосилань 60
4.8 Оформлення HTML-документу за допомогою CSS 61
Питання для самоконтролю до теми 4 61
Тестові завдання: 62
5 ЕЛЕМЕНТИ ОФІСНОГО ПРОГРАМУВАННЯ 63
5.1 Створення макросів 63
5.2 Редактор VBA 65
5.3 Поняття об‘єкту, властивості, методу, події 66
5.4 Представлення даних: типи даних, константи та змінні, масиви 71
5.5 Керуючі конструкції VBA: оператори розгалуження, вибору та циклу . 72
5.5.1 Розгалуження 72
5.5.2 Цикли 74
5.6 Введення-виведення повідомлень в діалогових вікнах 76
5.6.1 MsgBox 76
5.6.2 InputBox 78
5.7 Функції та процедури 78
5.8 Створення UserForm 80
5.9 Елементи керування UserForm 81
5.10 Методи та події об’єктів UserForm 83
Питання для самоконтролю до теми 5 84

Тестові завдання: 85
ВИКОРИСТАНА ЛІТЕРАТУРА 86
РЕКОМЕНДОВАНА ЛІТЕРАТУРА 86

Навчально-методичне видання
(українською мовою)

Надія Вікторівна Матвіїшина

ОРГАНІЗАЦІЯ ТА ОБРОБКА ЕЛЕКТРОННОЇ ІНФОРМАЦІЇ

Навчальний посібник
для здобувачів ступеня вищої освіти бакалавра спеціальності
«Комп'ютерні науки» освітньо-професійної програми «Комп'ютерні науки»

Рецензент *С. І. Гоменюк*
Відповідальний за випуск *О. С. Пшенична*
Коректор *Н.В.Матвіїшина*