Квест 17

Завдання 3: Скласти довідник із використанням текстових функцій Google Таблиць з прикладами 5-6 функцій.

Приклад довідника

1. **Функція** `LEN`

- **Призначення**: Визначає кількість символів у тексті.
- **Формат**: `=LEN(текст)`
- **Приклад**:
  В комірці A1 записано "Привіт".
  Формула: `=LEN(A1)`
  Результат: `6`.

2. **Функція** `LEFT`

- **Призначення**: Витягує певну кількість символів з початку тексту.
- **Формат**: `=LEFT(текст, кількість_символів)`
- **Приклад**:
  В комірці A1 записано "Google Таблиці".
  Формула: `=LEFT(A1, 6)`
  Результат: `Google`.

3. **Функція** `RIGHT`

- **Призначення**: Витягує певну кількість символів із кінця тексту.
- **Формат**: `=RIGHT(текст, кількість_символів)`
- **Приклад**:
  В комірці A1 записано "Google Таблиці".
  Формула: `=RIGHT(A1, 7)`
  Результат: `Таблиці`.

4. **Функція** `MID`

- **Призначення**: Витягує текст із середини рядка.
- **Формат**: `=MID(текст, початок, кількість_символів)`
- **Приклад**:
  В комірці A1 записано "Google Таблиці".
  Формула: `=MID(A1, 8, 7)`
  Результат: `Таблиці`.

matematika-doma.org

5. **Функція** `CONCATENATE` **або** `TEXTJOIN`

- **Призначення**: Об'єднує кілька текстових рядків в один.
- **Формат**:
  Для `CONCATENATE`: `=CONCATENATE(текст1, текст2, ...)`
  Для `TEXTJOIN`: `=TEXTJOIN(роздільник, пропуск_порожніх, текст1, текст2, ...)`
- **Приклад**:
  В комірках A1 записано "Google", а в B1 — "Таблиці".
  Формула: `=CONCATENATE(A1, " ", B1)`
  Результат: `Google Таблиці`.

6. **Функція** `TRIM`

- **Призначення**: Видаляє зайві пробіли з тексту.
- **Формат**: `=TRIM(текст)`
- **Приклад**:
  В комірці A1 записано " Google Таблиці ".
  Формула: `=TRIM(A1)`
  Результат: `Google Таблиці`.

Інструмент для довідника

Для створення довідника можна використати:

- Google Документи для текстового оформлення.
- Google Презентації для візуального представлення.
- Google Таблиці для інтерактивного демонстрування функцій.

1. Опис функцій у таблиці:

Функція	Призначення	Приклад	Результат
DAYS	Обчислює кількість днів між двома датами.	`=DAYS("01.09.2022"; "31.01.2022")`	213
YEARFRAC	Обчислює частку років між двома датами.	`=YEARFRAC("01.07.2016"; "31.12.2022")`	6,5
NOW	Виводить поточну дату й час.	`=NOW()`	Поточна дата й час
DAY	Визначає день місяця з вказаної дати.	`=DAY(D10)`	11
YEAR	Визначає рік із вказаної дати.	`=YEAR("01.09.2022")`	2022

matematika-doma.org

Завдання 4. Дослідження функцій для роботи з датами

1. **Проаналізувати корисність функцій:**
   - **DAYS:** зручна для підрахунку кількості днів між подіями.
   - **YEARFRAC:** ефективна для фінансових розрахунків, де потрібно обчислити частки року.
   - **NOW:** використовується для автоматичного відображення поточної дати й часу.
   - **DAY:** корисна для розрахунків, де потрібно виділити конкретний день з дати.
   - **YEAR:** підходить для визначення року у великих наборах даних.
2. **Висновок:**
   - Для аналізу дат найбільш корисними є `DAYS` і `YEARFRAC`, оскільки вони забезпечують обчислення різниць між датами.
   - `NOW` може використовуватись для динамічного відстеження змін у даних.
   - `DAY` і `YEAR` підходять для деталізації інформації про дати.

Квест 18

Завдання 1: Розгляд ситуацій застосування форматування, вирівнювання, перенесення та обертання текстових написів

Завдання 1: Розгляд ситуацій застосування форматування, вирівнювання, перенесення та обертання текстових написів

1. Форматування

- **Ситуація:** У таблиці необхідно виділити заголовки або важливі дані.
- **Приклад:** Виділення заголовків жирним шрифтом, курсивом або кольором.
- **Призначення:** Допомагає краще структуризувати таблицю та полегшує сприйняття.

2. Вирівнювання

- **Ситуація:** Текст у клітинках має бути зрозумілим і виглядати акуратно.
- **Приклад:** Вирівнювання заголовків по центру, а чисел — по правому краю.
- **Призначення:** Забезпечує гармонійний вигляд і логіку подання даних.

3. Перенесення тексту

- **Ситуація:** Текст у клітинці виходить за її межі.
- **Приклад:** Перенесення тексту в комірці з довгим описом.
- **Призначення:** Зберігає структуру таблиці та читаємість даних.

4. Обертання тексту

- **Ситуація:** Потрібно економити простір у таблиці.
- **Приклад:** Обертання назв стовпців на 90° (вертикальне відображення).
- **Призначення:** Робить таблицю компактнішою і більш зручною для використання.

matematika.com.ua

Завдання 2: Дослідження різниці між способами додавання зображень у Google Таблиці

Завдання 2: Дослідження різниці між способами додавання зображень у Google Таблиці

1. **Методи додавання зображень**

- **Додавання через функцію "Вставка → Зображення → У комірку"**
  - Зображення вставляється безпосередньо у клітинку (наприклад, A1).
  - Масштабування відбувається автоматично залежно від розмірів клітинки.
  - Зображення розглядається як частина даних клітинки, що впливає на її розмір і формат.
- **Додавання через "Редактор малюнків"**
  - Зображення додається поверх таблиці.
  - Його розташування не прив'язане до конкретної клітинки.
  - Малюнок можна переміщувати, масштабувати, але він не змінює властивостей клітинки.

2. **Порівняння**

Критерій	Зображення у клітинці	Зображення через редактор малюнків
Прив'язка до клітинки	Тісно прив'язане, залежить від її розміру	Незалежне, розташовується поверх
Редагування	Обмежене (масштаб автоматичний)	Можливе масштабування, зміна положення
Дані в клітинці	Займає клітинку, інші дані недоступні	Не заважає введенню тексту у клітинку
Застосування	Для статичних таблиць	Для презентацій чи схематичних зображень

3. **Висновок**

- Додавання зображення у **клітинку** зручно для структурованих таблиць, де потрібна інтеграція зображення і даних.
- Додавання через **редактор малюнків** більше підходить для візуальних презентацій або роботи із графічними елементами, які не залежать від структури таблиці.

Завдання 3: Порівняння прапорця та списку в електронній таблиці Google

matematika-doma.org

Завдання 3: Порівняння прапорця та списку в електронній таблиці Google

1. **Схожість**

- Обидва інструменти використовуються для організації даних та взаємодії з ними.
- Можуть застосовуватися для створення інтерактивних таблиць.

2. **Відмінності**

Критерій	Прапорець	Список
Призначення	Використовується для швидкого відмічання стану (так/ні, виконано/невиконано).	Застосовується для вибору одного значення зі списку.
Функціонал	Прапорець вмикається або вимикається (позначка √ або пусто).	Містить випадаючий перелік варіантів.
Приклад застосування	Відмічання виконання завдання у списку.	Вибір статусу (наприклад, "Планується", "Виконано").
Візуалізація	Лише два стани (позначений або ні).	Користувач бачить всі доступні варіанти вибору.

3. **Ситуації для використання**

- **Прапорець:**
  - Списки завдань або перевірка виконання.
  - Позначення завершених етапів проєкту.
- **Список:**
  - Категоризація даних (наприклад, "Рівень важливості": Низький, Середній, Високий).
  - Вибір опцій для фільтрації чи сортування.

Завдання 4: Дослідження об'єктів для додавання "розумних чипів" у Google Таблиці

matematika-dom.org

Завдання 4: Дослідження об'єктів для додавання "розумних чипів" у Google Таблиці

1. Що таке "розумні чипи"?

"Розумні чипи" (Smart Chips) у Google Таблицях — це інтерактивні елементи, які додають функціональність до таблиць. Вони дозволяють додавати інформацію або посилання на інші об'єкти в Google Workspace.

2. Типи об'єктів, які можна додати

- **Контакти**:
  - Додаються через @ім'я користувача.
  - Автоматично підтягують інформацію про користувача (ім'я, електронна пошта).
- **Файли з Google Диска**:
  - Можна додати документи, таблиці, презентації або будь-які файли з Google Диска.
  - Підтягнутий файл стає доступним для швидкого відкриття.
- **Події з Календаря**:
  - Можна додати події за допомогою @назва події.
  - Відображаються дата, час та учасники.
- **Місцезнаходження**:
  - Додаються через інтеграцію з Google Карти.
  - Показують адресу та посилання на карту.
- **Посилання на веб-сторінки**:
  - Можна створити чип із гіперпосиланням.
  - Автоматично відображає попередній перегляд сторінки.

3. Як додати "розумні чипи"?

1. Вибрати клітинку, де потрібно вставити чип.
2. Ввести символ "@" і обрати потрібний об'єкт із випадаючого списку.
3. Натиснути Enter для підтвердження.

4. Приклади використання

- **Командні таблиці**: Додавання контактів колег для швидкої комунікації.
- **Проєктний менеджмент**: Вставка файлів або подій для відстеження прогресу.
- **Звіти**: Додавання посилань на джерела або місцезнаходження.

matematika-doma.org

Завдання 5: Модель опитувальника для авторів учнівських робіт

Завдання 5: Модель опитувальника для авторів учнівських робіт

Модель опитувальника

1. **Основна інформація про автора:**
   - Прізвище автора: (Текстове поле)
   - Ім'я автора: (Текстове поле)
2. **Клас автора:**
   - Вибір з варіантів:
     - 5
     - 6
     - 7
3. **Жанр твору:**
   - Вибір одного варіанта з таких:
     - Рукоділля
     - Технічна творчість
     - Малюнок
     - Поробка
     - Фото твору
4. **Згода на виставлення твору на аукціоні:**
   - Чекбокс:
     - ☐ Так, погоджуюсь
5. **Оцінка складності виготовлення твору:**
   - Шкала оцінювання (1-5):
     - 1: Дуже легко
     - 2: Легко
     - 3: Середньо
     - 4: Складно
     - 5: Дуже складно

математика-дома.org

Реалізація в Google Формах

1. Відкрити Google Форми.
2. Створити питання за кожним пунктом моделі:
   - Текстові поля для імені й прізвища.
   - Варіанти вибору (радіокнопки) для класу.
   - Радіокнопки для вибору жанру твору.
   - Чекбокс для згоди.
   - Лінійна шкала для оцінювання складності (1–5).
3. Зберегти та поділитися посиланням для заповнення.

Завдання 6: Модель електронної таблиці для обліку книжок у шкільному куточку читання

Структура таблиці

Стовпець	Опис
Автор книги	Прізвище та ім'я автора книги (текстове поле).
Назва книги	Назва книги (текстове поле).
Жанр	Вибір із випадаючого списку (наприклад: художня література, фантастика, наукова, інше).
Прізвище та ім'я учня	Хто поділився книгою (текстове поле).
Клас	Вибір із варіантів: 5, 6, 7.
Дата взяття	Дата, коли книгу взято для читання (формат дати).
Чи взято книгу?	Чекбокс для позначення (✓ - взято, пусто - не взято).
Оцінка книги	Оцінка читача у формі шкали від 1 до 5.
Відгук	Коментар або відгук після прочитання (текстове поле).

matematika-doma.org

Кроки для створення таблиці

1. **Відкрити Google Таблиці:**
   - Створити нову таблицю.
2. **Додати заголовки стовпців:**
   - Ввести всі стовпці, зазначені вище.
3. **Налаштувати випадаючі списки:**
   - Для стовпця "Жанр":
     - Використати функцію "Дані → Перевірка даних → Список із елементами".
     - Внести жанри: художня література, фантастика, наукова, інше.
   - Для стовпця "Клас":
     - Використати список: 5, 6, 7.
4. **Додати чекбокс:**
   - У стовпці "Чи взято книгу?" використати "Вставка → Прапорець".
5. **Налаштувати формат дати:**
   - Для стовпця "Дата взяття" обрати формат дати.

Квест 19

Завдання 1: Форматування документа "Кореспонденція"

математика-дома.org

Завдання 1: Форматування документа "Кореспонденція"

Покрокова інструкція для форматування

1. **Відкриття документа:**
   - Завантажити файл "Кореспонденція" із папки **Текстовий процесор**.
   - Відкрити файл у текстовому редакторі (наприклад, Microsoft Word або Google Документи).
2. **Застосування стилів до заголовків:**
   - Виділити назву документа.
   - Застосувати стиль **Заголовок 1** (Heading 1).
3. **Форматування підзаголовків:**
   - Виділити всі підзаголовки у тексті.
   - Застосувати стиль **Заголовок 2** (Heading 2).
4. **Налаштування основного тексту:**
   - Виділити весь текст (крім заголовків і підзаголовків).
   - Встановити такі параметри:
     - Шрифт: **Arial**
     - Розмір шрифту: **14**
5. **Форматування виділеного тексту:**
   - Знайти у тексті слова або словосполучення, взяті у лапки.
   - Виділити їх і застосувати стиль **Підзаголовок** (або інший виділений стиль, якщо він доступний).
6. **Збереження документа:**
   - Зберегти відформатований документ у вихідну папку.
   - Перевірити правильність форматування перед передачею вчителю.

Завдання 2: Форматування документа "Кореспонденція"

Завдання 2: Створення документа "Поезія" у Google Документах

1. **Створіть документ "Поезія".**
2. **Додайте текст улюбленого вірша українського автора.**
3. **Форматування строф:**
   - Підвищення тону: **Заголовок 1.**
   - Зниження тону: **Звичайний текст.**
   - Рівний тон: **Заголовок 3.**
4. **Додайте посилання на вірш** (з інтернет-джерела) у спільний документ.
5. **Розміщення:** поділіться віршем із класом.
6. **Оцінка та порада:** запропонуйте, як покращити інтонацію чи стиль подання.

matematika-doma.org

Завдання 3: Виконання заміни у документі "Ой у лузі червона калина"

Завдання 3: Виконання заміни у документі "Ой у лузі червона калина"

1. **Відкрийте документ "Ой у лузі червона калина".**
2. **Виконайте заміни:**
   - Відповідно до зазначених в інструкціях вище (застосуйте стилі або редагуйте текст).
   - Використайте функцію **"Знайти та замінити"** для швидкого внесення змін.
     - **Комбінація клавіш:** Ctrl + H (Windows) або Command + H (Mac).
     - Введіть слово або фразу для заміни, та її новий варіант.
3. **Перевірте текст** після внесення змін.
4. **Збережіть документ.**

Завдання 4: Як швидко виправити помилки з пропусками у тексті

Завдання 4: Як швидко виправити помилки з пропусками у тексті

Швидкі способи виправлення:

1. **Використання функції "Знайти та замінити":**
   - **Кроки:**
     - Відкрити текстовий редактор.
     - Викликати інструмент **"Знайти та замінити"** (Ctrl + H або Command + H).
     - У поле "Знайти" ввести два або більше пробілів (наприклад,).
     - У поле "Замінити" ввести один пробіл.
     - Натиснути **"Замінити все"**.
   - **Результат:** Усі зайві пробіли будуть видалені.
2. **Використання інструмента перевірки орфографії:**
   - Активувати перевірку правопису у текстовому редакторі (вбудована функція в Google Документах або Microsoft Word).
   - Виправити помилки, запропоновані системою.
3. **Ручне виправлення (для невеликих текстів):**
   - Переглянути текст і видалити зайві пропуски вручну.

matematika-oma.org

Завдання 5: Порівняння способів додавання об'єктів у Google Документах та MS Word

Завдання 5: Порівняння способів додавання об'єктів у Google Документах та MS Word

Схожості

1. **Додавання зображень:**
   - Обидві платформи дозволяють вставляти зображення з локального пристрою, інтернету або іншого джерела.
   - Присутній варіант редагування та переміщення вставлених зображень.
2. **Додавання таблиць:**
   - Обидва редактори мають опцію створення таблиць із вибором розміру за кількістю рядків і стовпців.
   - Можливе налаштування стилю таблиці після вставлення.
3. **Додавання малюнків:**
   - Можна створювати малюнки всередині документів за допомогою інструментів малювання або імпортувати їх з інших джерел.

Відмінності

1. **Google Документи:**
   - Інтеграція з Google Диском для швидкого додавання об'єктів.
   - Можливість додавання зображень через URL-адресу або камеру пристрою.
   - Спрощені функції для створення та редагування малюнків.
2. **MS Word:**
   - Більш розширені інструменти для форматування та редагування зображень.
   - Наявність професійних шаблонів таблиць і малюнків.
   - Підтримка 3D-моделей і складних діаграм.

Висновок

- **Google Документи** підходять для роботи з об'єктами в умовах інтеграції з хмарними сервісами та простих завдань.
- **MS Word** краще для професійного оформлення документації з розширеними можливостями для графічного оформлення.

Квест 20

Завдання 1: Огляд вебсайту навчального закладу та пропозиції щодо покращення

matematika-doma.org

Кроки виконання:

1. **Знайти вебсайт навчального закладу:**
   - Відкрити вебсайт свого закладу освіти.
   - Оглянути структуру сайту, його каталоги, навігацію та заставку.
2. **Аналіз структури:**
   - Оцініть зручність меню та підменю.
   - Перевірте доступність розділів (наприклад, новини, розклад, контакти).
   - Оцініть дизайн заставки (банери, кольори, зображення).
3. **Запропонувати ідеї покращення:**
   - **Структура каталогів:**
     - Додати або оптимізувати розділи (наприклад, "Для учнів", "Для батьків").
   - **Навігація:**
     - Зробити її більш інтуїтивною (спростити назви, додати іконки).
   - **Заставка:**
     - Використати сучасний дизайн або змінити кольорову схему для зручності читання.

Завдання 2: Аналіз структури сайтів та створення схеми

matematik-doma.org

Завдання 2: Аналіз структури сайтів та створення схеми

Кроки виконання:

1. **Виберіть сайт із запропонованого списку:**
   - Всеукраїнська школа онлайн.
   - Центр науки міста Тернополя.
   - Волинський краєзнавчий музей.
   - Стадіон "Арена Львів".
   - Івано-Франківський драматичний театр.
2. **Аналіз структури сайту:**
   - Огляньте меню сайту.
   - Запишіть основні розділи (наприклад, "Про нас", "Новини", "Контакти").
   - Вивчіть підрозділи в кожному розділі (наприклад, у "Новинах" — події, оголошення).
3. **Створення схеми структури:**
   - Намалюйте схему структури сайту, використовуючи інструменти (наприклад, Google Drawings, Canva, PowerPoint або будь-який графічний редактор).
   - Схема повинна включати:
     - Головну сторінку (корінь).
     - Основні розділи (гілки).
     - Підрозділи (гілки другого рівня).
4. **Збереження та надсилання:**
   - Збережіть файл схеми у форматі PDF або PNG.
   - Надішліть файл на електронну навчальну скриньку, вказану у завданні.

matematika-doma.org

Завдання 3: Аналіз доступності сайту Міністерства освіти і науки України

Завдання 3: Аналіз доступності сайту Міністерства освіти і науки України

Перевірка пристосованості сайту для людей із порушеннями зору

1. **Елементи, які сприяють доступності:**
   - **Контрастність тексту:** достатньо чітка, зручна для читання.
   - **Масштабування:** сайт підтримує збільшення шрифтів через браузер.
   - **Альтернативний текст:** присутній для зображень (допомагає при використанні екранних читачів).
   - **Структура заголовків:** коректно впорядкована, що полегшує навігацію через спеціальні програми.
2. **Недоліки (якщо присутні):**
   - Відсутність спеціальної кнопки для швидкого переходу до **версії для слабозорих**.
   - Не завжди оптимізовані елементи для екранних читачів (ARIA-теги можуть бути відсутніми або некоректними).
   - Можлива недостатня підтримка навігації лише за допомогою клавіатури.

Поради для покращення ергономічності сайту:

1. Додати **режим для слабозорих**, який змінює:
   - Колірну гаму на високо-контрастну.
   - Розмір шрифту на збільшений.
2. Поліпшити структуру **ARIA-атрибутів**, щоб екранні читачі коректно інтерпретували елементи сайту.
3. Забезпечити **чітку навігацію** для користувачів клавіатури (видимий фокус на активних елементах).
4. Додати кнопку для швидкого доступу до основного змісту (skip-to-content).
5. Тестувати сайт із використанням інструментів перевірки доступності, таких як **WAVE** чи **Lighthouse**.

математика-домашка.org

Завдання 4: Аналіз навігації та оформлення сайту "Мистецький Арсенал"

Завдання 4: Аналіз навігації та оформлення сайту "Мистецький Арсенал"

Навігація сайту

1. **Меню внизу сторінки:**
   - Включає основні посилання:
     - "Про нас" (інформація про установу).
     - "Події" (розклад заходів).
     - "Новини" (актуальні оголошення).
     - "Контакти" (адреса, телефони, карта проїзду).
   - Чітко структурована, дозволяє швидкий доступ до важливих розділів.
2. **Карта сайту:**
   - Дає змогу побачити повну структуру розділів.
   - Зручна для переходу до потрібних сторінок.
3. **Зручність навігації:**
   - Основні розділи доступні як з головного меню, так і з футера.
   - Логічна послідовність розташування посилань.

Оформлення сторінок

1. **Дизайн:**
   - Використано сучасний, мінімалістичний стиль.
   - Кольорова гама спокійна, зручна для читання.
2. **Функціонал:**
   - Сторінки завантажуються швидко.
   - Інтуїтивно зрозуміла структура для різних користувачів.
3. **Недоліки (якщо присутні):**
   - Можливе удосконалення дизайну меню внизу сторінки для кращого відображення на мобільних пристроях.
   - Додати іконки для швидкої ідентифікації розділів.

matematika-doma.org

Завдання 6: Розподіл етапів створення вебсайту між ШІ та ручною роботою

Завдання 6: Розподіл етапів створення вебсайту між ШІ та ручною роботою

Етапи, які можна доручити штучному інтелекту:

1. **Проєктування дизайну:**
   - Використання ШІ-інструментів (наприклад, Figma, Canva, Wix) для створення макетів вебсторінок.
   - Генерація стилів, шрифтів та кольорових схем на основі запитів.
2. **Створення текстового контенту:**
   - Генерація описів, блогів або інформаційних сторінок (за допомогою ChatGPT або аналогів).
3. **Розробка структури сайту:**
   - Автоматичне формування карти сайту чи шаблонів сторінок у конструкторах.
4. **Оптимізація зображень:**
   - Зменшення розміру файлів і покращення якості з використанням інструментів ШІ.
5. **Базовий код:**
   - Генерація HTML, CSS або JavaScript для типових компонентів (кнопки, форми тощо).

Етапи, які потребують ручної роботи:

1. **Збір вимог:**
   - Визначення цільової аудиторії, функціоналу та цілей сайту.
2. **Детальне налаштування:**
   - Інтеграція специфічних функцій (наприклад, баз даних, CRM-систем).
3. **Перевірка та тестування:**
   - Тестування на реальних пристроях і виправлення багів.
4. **Забезпечення безпеки:**
   - Установка SSL-сертифікатів, налаштування захисту даних.
5. **Остаточна публікація:**
   - Завантаження на сервер і запуск.

matematika-doma.org

Квест 21

Завдання 3: Панелі інструментів у векторному графічному редакторі Inkscape

Завдання 3: Панелі інструментів у векторному графічному редакторі Inkscape

Основні панелі інструментів і їх призначення

1. **Інструменти малювання:**
   - **Pen Tool (Перо):** Створення кривих і прямих ліній.
   - **Bezier Tool (Крива Безьє):** Побудова складних форм і шляхів.
   - **Pencil Tool (Олівець):** Вільне малювання контурів.
2. **Інструменти трансформації:**
   - **Move Tool (Переміщення):** Переміщення об'єктів по полотну.
   - **Scale Tool (Масштабування):** Зміна розмірів об'єкта.
   - **Rotate Tool (Обертання):** Поворот об'єктів.
3. **Інструменти роботи з кольором:**
   - **Fill and Stroke (Заливка та обведення):** Налаштування кольору фону та контуру об'єкта.
   - **Color Palette (Палітра кольорів):** Швидкий вибір кольорів.
4. **Інструменти тексту:**
   - **Text Tool (Текст):** Додавання та редагування текстових елементів.
5. **Інструменти вирівнювання та упорядкування:**
   - **Align and Distribute (Вирівнювання):** Розташування об'єктів за сіткою або відносно один одного.
   - **Group/Ungroup (Групувати/Розгрупувати):** Робота з групами об'єктів.
6. **Шари (Layers):**
   - Налаштування порядку об'єктів через розташування на різних шарах.

Завдання 6: Способи заливки об'єктів у векторній графіці та інструкція

Завдання 6: Способи заливки об'єктів у векторній графіці та інструкція

Способи заливки об'єктів:

1. **Суцільна заливка:**
   - Однорідний колір, заповнює весь об'єкт.
2. **Градієнтна заливка:**
   - Перехід між двома або більше кольорами.
   - Типи градієнтів: лінійний (від однієї точки до іншої), радіальний (від центру до краю).
3. **Візерунок:**
   - Повторюваний шаблон або текстура всередині об'єкта.
4. **Заливка з прозорістю:**
   - Додає прозорість, дозволяючи бачити шари під об'єктом.

Інформатика-дома.org

Інструкція для застосування заливки:

1. **Відкрийте об'єкт у редакторі (наприклад, Inkscape).**
2. **Обрати інструмент заливки:**
   - У панелі **"Fill and Stroke"** знайдіть налаштування заливки.
3. **Суцільна заливка:**
   - Оберіть вкладку **Fill**.
   - Виберіть однорідний колір із палітри.
4. **Градієнт:**
   - Оберіть вкладку **Gradient** у розділі заливки.
   - Налаштуйте кольори градієнта, переміщаючи вузли та змінюючи значення.
5. **Візерунок:**
   - У вкладці **Fill** оберіть опцію **Pattern**.
   - Виберіть доступний візерунок або імпортуйте свій.
6. **Прозорість:**
   - В панелі **Opacity** (у тій же вкладці) встановіть рівень прозорості для об'єкта.
7. **Застосування:**
   - Після налаштування натисніть Enter або просто клацніть за межі об'єкта для підтвердження.

Квест 22

Завдання 1: Алгоритм створення кривої лінії за допомогою інструмента "Крива Безьє"

математика-дома.org

Завдання 1: Алгоритм створення кривої лінії за допомогою інструмента "Крива Безьє"

Алгоритм створення кривої лінії:

1. **Виберіть інструмент "Крива Безьє":**
   - У редакторі (наприклад, Inkscape) оберіть інструмент "Крива Безьє" на панелі інструментів.
2. **Створення лінії:**
   - Клацніть мишкою на робочій області, щоб створити початкову точку.
   - Додайте другу точку, тримаючи ліву кнопку миші натиснутою, і перетягніть для створення вигину.
3. **Фіксація вузлів:**
   - Кожне клацання створює вузол. Після додавання останньої точки натисніть **Enter**, щоб завершити створення кривої.
4. **Корекція:**
   - Якщо потрібно, виберіть інструмент "Редагування вузлів" (Node Tool) і перемістіть вузли чи напрямні для точного налаштування.

Зміна форми кривої після побудови:

1. **Виберіть криву:**
   - Клацніть на лінії, щоб активувати її.
2. **Активуйте інструмент редагування вузлів:**
   - Виберіть "Редагування вузлів" (Node Tool).
3. **Налаштування вузлів:**
   - Переміщайте вузли для зміни форми кривої.
   - Використовуйте напрямні (ручки) для корекції вигинів.
4. **Додавання або видалення вузлів:**
   - Додайте новий вузол, клацнувши на лінії подвійним кліком.
   - Видаліть вузол, вибравши його і натиснувши **Delete**.

Завдання 4: Створення піктограм прогнозу погоди в Inkscape

matematika-doma.org

Алгоритм створення піктограми "Сонце за хмарою"

1. **Створення сонця:**
   - Виберіть **інструмент "Еліпс"**.
   - Намалюйте коло, утримуючи клавішу **Ctrl**, щоб зробити його ідеальним.
   - Виберіть жовтий колір для заливки.
   - Додайте промені:
     - Використовуйте **інструмент "Зірка/Багатокутник"**.
     - Задайте кількість кутів (наприклад, 8-12) і змінюйте довжину променів у налаштуваннях.
     - Розташуйте зірку за колом (сонцем).
2. **Створення хмари:**
   - Виберіть **інструмент "Еліпс"**.
   - Намалюйте кілька перекриваючихся кіл синього кольору.
   - Об'єднайте їх у єдиний об'єкт за допомогою **інструмента "Об'єднання" (Union)** у меню "Шляхи".
3. **Композиція:**
   - Розмістіть хмару частково перед сонцем.
   - Налаштуйте розміри та положення для гармонійного вигляду.

Алгоритм створення піктограми "Місяць і хмара"

1. **Створення місяця:**
   - Виберіть **інструмент "Еліпс"**.
   - Намалюйте одне коло, а потім друге, трохи менше, змістивши його праворуч.
   - Використовуйте **інструмент "Різниця" (Difference)** в меню "Шляхи", щоб створити форму півмісяця.
   - Залийте місяць жовтим кольором.
2. **Створення хмари:**
   - Виконайте ті ж дії, що й для піктограми "Сонце за хмарою".
3. **Композиція:**
   - Розмістіть хмару частково перед місяцем.
   - Відрегулюйте розміри.

Квест 25

Завдання 2

Які переваги та недоліки векторної анімації?

Переваги векторної анімації:

1. Масштабованість без втрати якості.

Зображення не розмиваються при збільшенні чи зменшенні розміру.

matematika-doma.org

2. Малий розмір файлів. Векторні файли займають менше місця порівняно з растровими.
3. Гнучкість у редагуванні. Легке внесення змін до зображення чи анімації.
4. Швидке завантаження. Підходить для веб–додатків завдяки економії пропускної здатності.
5. Сумісність із багатьма платформами. Підтримується більшістю сучасних інструментів розробки анімації.
Недоліки векторної анімації:
1. Обмежена деталізація. Не підходить для складних або фотореалістичних зображень.
2. Потреба в спеціальному програмному забезпеченні. Для створення та редагування потрібні відповідні інструменти.
3. Не всі браузери однаково підтримують складні векторні анімації. Можуть виникати проблеми із сумісністю.
4. Більш ресурсозатратно при рендерингу складних анімацій. Особливо для великої кількості об’єктів.
Висновок: Векторна анімація ідеально підходить для створення графіки, що потребує масштабованості та мінімального розміру файлів, але не завжди може задовольнити вимоги до деталізації чи реалістичності.

Завдання 8

Обговоріть, як впливає на анімацію зміна тривалості кадру. Знайдіть в інтернеті, що означає технологія 25–го кадру.

Вплив зміни тривалості кадру на анімацію:
Тривалість кадру визначається частотою кадрів за секунду (FPS – frames per second). Зміна цієї частоти безпосередньо впливає на сприйняття анімації:

• Збільшення FPS: При більшій кількості кадрів за секунду (наприклад, 60 FPS) анімація виглядає плавнішою та реалістичнішою. Однак це потребує більше ресурсів для створення та відтворення анімації.

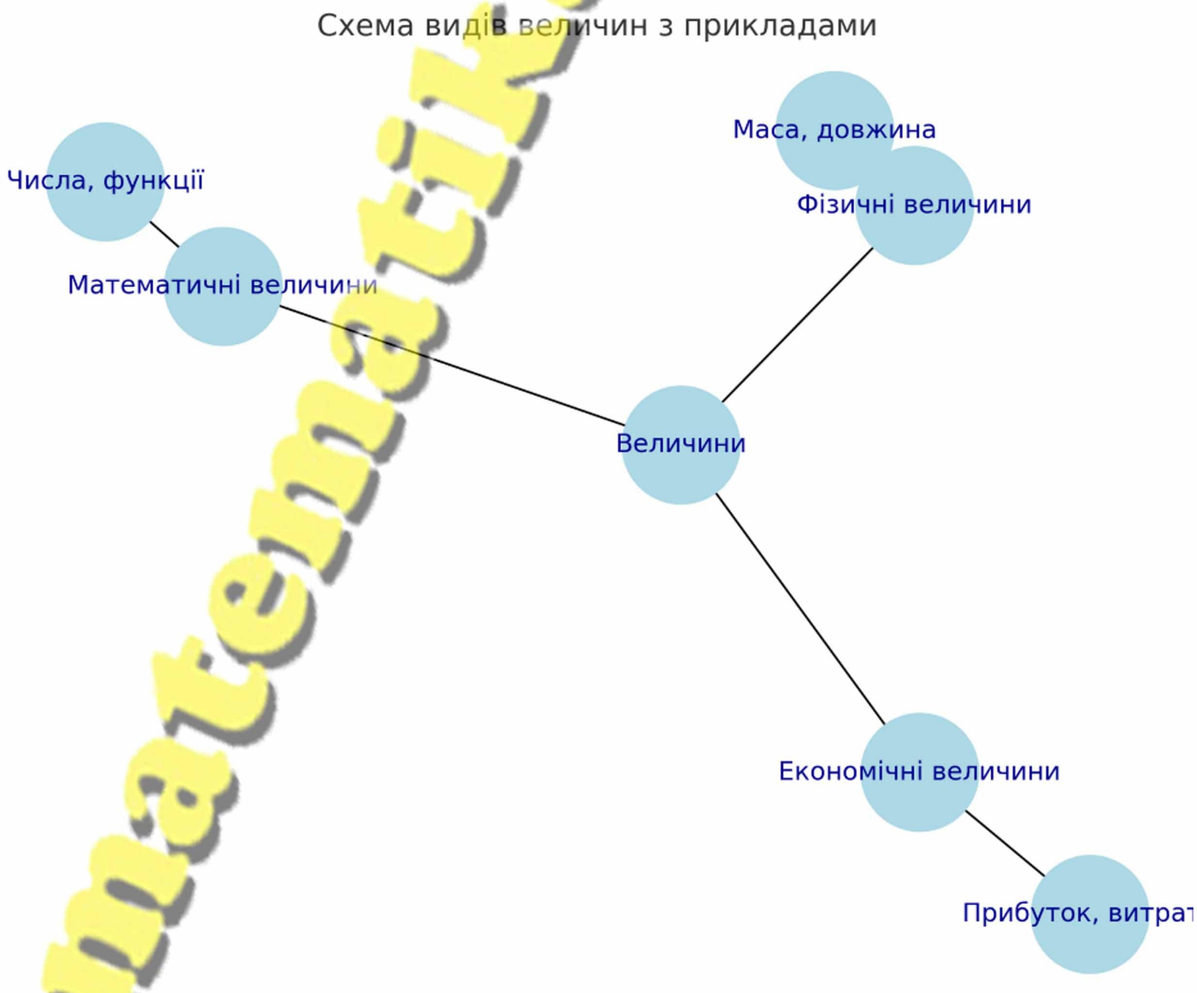
• Зменшення FPS: При меншій кількості кадрів за секунду (наприклад, 10 FPS) анімація може виглядати ривчастою та менш природною. Такий підхід може використовуватися для стилізованих ефектів або за обмежених ресурсів.

Технологія 25–го кадру: Технологія "25–го кадру", також відома як сублімінальна реклама, передбачає вставку додаткового кадру в відеоряд, який демонструється настільки швидко, що свідомість його не сприймає, але, як вважалося, підсвідомість може його обробити. У 1957 році Джеймс Вайкері стверджував, що під час показу фільму вставляв кадри з повідомленнями "Їжте поп–корн" та "Пийте кока–колу", що призвело до зростання продажів цих продуктів. Однак пізніше Вайкері зізнався,

що результати його експериментів були сфабриковані, і ефективність цієї технології не підтвердилася.
Висновок: Зміна тривалості кадру суттєво впливає на плавність та якість анімації, визначаючи її сприйняття глядачем. Що стосується технології 25–го кадру, численні дослідження не підтвердили її ефективність, і вона вважається міфом.

Квест 27

Завдання 5. Подайте інформацію про види величин у вигляді схеми. Доповніть її прикладами. Збережіть схему малюнком.



Завдання 7. Складіть програму (Python), у якій І послідовно змінним a, b, c, d надається значення 2, 4, 6, 8, а на екрані

математика.com.ua

результат виводиться так: 1) 2468 2) 8642 3) 22446688.

```
# Програма для виконання Завдання 7


# Присвоєння змінним значень
a, b, c, d = 2, 4, 6, 8


# Вивід результатів без використання додаткових змінних
print("1)", f"{a}{b}{c}{d}")
print("2)", f"{d}{c}{b}{a}")
print("3)", f"{a}{a}{b}{b}{c}{c}{d}{d}")
```

Завдання 9. Складіть програму, в якій для заданих значень змінних *x* та *y*, дібраних самостійно, виводиться відповідний результат.

```
# Значення змінних
x = 20
y = 3

# Виведення результатів операцій
print("Оператор +:", x + y)  # Додавання
print("Оператор -:", x - y)  # Віднімання
print("Оператор *:", x * y)  # Множення
print("Оператор /:", x / y)  # Ділення
print("Оператор //:", x // y)  # Цілочисельне ділення
print("Оператор %:", x % y)  # Залишок від ділення
```

Таблиця, яку потрібно заповнити, буде виглядати так:

Оператор	+	–	*	/	//	%
Результат	23	17	60	6.6667	6	2

Ця програма обчислює результати для всіх базових математичних операторів.

підтримка.org.ua

Завдання 10. Переконайтесь, що ви розумієте, як використовувати оператори +=, –=, "=, /=, //=, %=, **=.

```
# Початкове значення змінної
x = 10

# Використання оператора += (додавання та присвоєння)
x += 5  # еквівалентно: x = x + 5
print("Після x += 5:", x)  # Результат: 15

# Використання оператора -= (віднімання та присвоєння)
x -= 3  # еквівалентно: x = x - 3
print("Після x -= 3:", x)  # Результат: 12

# Використання оператора *= (множення та присвоєння)
x *= 2  # еквівалентно: x = x * 2
print("Після x *= 2:", x)  # Результат: 24

# Використання оператора /= (ділення та присвоєння)
x /= 4  # еквівалентно: x = x / 4
print("Після x /= 4:", x)  # Результат: 6.0

# Використання оператора //= (цілочисельне ділення та присвоєння)
x //= 2  # еквівалентно: x = x // 2
print("Після x //= 2:", x)  # Результат: 3.0

# Використання оператора %= (залишок від ділення та присвоєння)
x %= 2  # еквівалентно: x = x % 2
print("Після x %= 2:", x)  # Результат: 1.0

# Використання оператора **= (піднесення до степеня та присвоєння)
x = 3  # Оновлюємо значення для демонстрації
x **= 3  # еквівалентно: x = x ** 3
print("Після x **= 3:", x)  # Результат: 27
```

matematika-doma.org

Пояснення:

- `+=` : додає значення до змінної.
- `-=` : віднімає значення від змінної.
- `*=` : множить змінну на значення.
- `/=` : ділить змінну на значення.
- `//=` : виконує цілочисельне ділення.
- `%=` : залишок від ділення.
- `**=` : підносить до степеня.

Завдання 11

```
# Початкове значення змінної
x = 20

# Використання операторів та виведення результатів
x += 3  # Додавання і присвоєння
print("Оператор +=:", x)  # Результат: 23

x = 20  # Повертаємо початкове значення
x -= 3  # Віднімання і присвоєння
print("Оператор -=:", x)  # Результат: 17

x = 20  # Повертаємо початкове значення
x *= 3  # Множення і присвоєння
print("Оператор *=:", x)  # Результат: 60

x = 20  # Повертаємо початкове значення
x /= 3  # Ділення і присвоєння
print("Оператор /=:", x)  # Результат: 6.666666666666667

x = 20  # Повертаємо початкове значення
x //= 3  # Цілочисельне ділення і присвоєння
print("Оператор //=:", x)  # Результат: 6

x = 20  # Повертаємо початкове значення
x %= 3  # Залишок від ділення і присвоєння
print("Оператор %=:", x)  # Результат: 2
```

matematika-doma.org

Таблиця результатів:

Оператор	+=	-=	*=	/=	//=	%=
Результат	23	17	60	6.6667	6	2

Ця програма дозволяє виконати всі необхідні операції та отримати відповідні результати для таблиці.

Завдання 13

Складіть програму, в якій для заданих значень змінної *x*, дібраної самостійно, виводиться відповідний результат.

```
# Початкове значення змінної
x = 15.647

# Використання різних форматів округлення і виведення результатів
print("Результат з округленням до цілого числа:", round(x))
print("Результат з округленням до 1 знака після коми:", round(x, 1))
print("Результат з округленням до 2 знаків після коми:", round(x, 2))
print("Результат з підвищенням до наступного цілого числа:", int(x) + 1)
```

Результати:

Оператор	Округлення до цілого	Округлення до 1 знака	Округлення до 2 знаків	Підвищення до наступного
Результат	16	15.6	15.65	20

Програма дозволяє адаптувати точність округлення та отримати всі потрібні значення для таблиці.

matematika-dom.org

Завдання 16

Семикласники придумали гру. Якщо у квитку на сеанс фільму, який містить тризначний номер, сума цифр цього номера виявиться рівною 20, то квиток щасливий. Складіть програму, яка за введеним значенням номера обчислює його суму цифр.

```
# Введення тризначного номера
ticket_number = int(input("Введіть тризначний номер квитка: "))

# Перевірка, чи введене число є тризначним
if 100 <= ticket_number <= 999:
    # Обчислення суми цифр номера квитка
    digit_sum = sum(int(digit) for digit in str(ticket_number))
    print("Сума цифр номера квитка:", digit_sum)

    # Перевірка на "щасливий" квиток
    if digit_sum == 20:
        print("Квиток щасливий!")
    else:
        print("Квиток не є щасливим.")
else:
    print("Введене число не є тризначним. Спробуйте ще раз.")
```

Як працює програма:

1. Користувач вводить тризначний номер квитка.

2. Програма перевіряє, чи число є тризначним.

3. Якщо умова виконана, обчислюється сума цифр.

4. Якщо сума цифр дорівнює 20, квиток вважається щасливим; інакше – ні.

matematika-dlya-vsih.org.ua

Завдання 1

Обговоріть переваги та недоліки використання модулів у мові програмування Python.

Переваги модулів у Python:

1. Реюзабельність: Код можна використовувати повторно.

2. Структурованість: Програма стає зрозумілішою.

3. Готові рішення: Велика стандартна бібліотека.

4. Легкість тестування: Модулі легко перевіряти окремо.

5. Командна робота: Зручно ділити проект на частини.

Недоліки:

1. Залежність: Від сторонніх модулів.

2. Конфлікти: Можливі дублікати назв.

3. Продуктивність: Зниження швидкості через імпорт.

4. Розмір: Збільшення розміру програми.

Висновок: Модулі роблять розробку зручнішою, але потребують обережного використання.

Завдання 5. Складіть програму обчислення значень виразів для деякого до данного значення x:

Підтримка-студента.org

```python
import math

# Введення значення x
x = float(input("Введіть значення x: "))

# Обчислення виразу а
y_a = math.sqrt(x) + math.sqrt(x + 10) + math.sqrt(x + 100)

# Обчислення виразу б
y_b = (math.sqrt(x**2 + 5)) / (2 * math.sqrt(x))

# Виведення результатів
print("Результат для виразу а):", y_a)
print("Результат для виразу б):", y_b)
```

Завдання 6. Складіть програму «Решта», у якій знаходять різницю між округленням деякого дійсного числа із проміжку від 1 до 10 та його округленням до найближчого більшого числа.

```python
import math

# Введення дійсного числа з проміжку [1, 10]
number = float(input("Введіть дійсне число з проміжку [1, 10]: "))

# Перевірка, чи число в межах [1, 10]
if 1 <= number <= 10:
    # Округлення до найближчого цілого числа
    rounded_number = round(number)

    # Округлення до найближчого більшого числа
    ceil_number = math.ceil(number)

    # Знаходження різниці
    difference = ceil_number - rounded_number

    # Виведення результату
    print("Різниця між округленням і округленням до більшого:", difference)
else:
    print("Число не в межах [1, 10]. Спробуйте ще раз.")
```

matematika.org

Пояснення:

1. Користувач вводить число з проміжку [1, 10].
2. Програма перевіряє, чи введене число знаходиться в цьому проміжку.
3. Використовуються функції:
   - `round()` для звичайного округлення.
   - `math.ceil()` для округлення до найближчого більшого цілого.
4. Різниця між результатами обчислюється і виводиться.

> **Наприклад, для числа `5.3`:**
>
> - **Округлення (`round`) → 5**
> - **Округлення до більшого (`ceil`) → 6**
>
> **Різниця → 1.**

Завдання 7. Складіть програму, яка обчислює факторіал випадкового числа з проміжку від 1 до 6.

```python
import random
import math

# Генеруємо випадкове число з проміжку [1, 6]
random_number = random.randint(1, 6)

# Обчислюємо факторіал числа
factorial = math.factorial(random_number)

# Виводимо результати
print(f"Випадкове число: {random_number}")
print(f"Факторіал числа {random_number}: {factorial}")
```

matematika-doma.org

Як працює програма:

1. `random.randint(1, 6)` генерує випадкове ціле число з проміжку [1, 6].
2. `math.factorial()` обчислює факторіал цього числа.
3. Програма виводить згенероване число та його факторіал.

> **Наприклад:**
>
> - Випадкове число: `5`
> - Факторіал числа `5`: `120`.

Завдання 9

Складіть програму «Чарівне число» мовою Python та реалізуйте її в середовищі програмування. У програмі комп'ютер має «Загадати» деяке ціле число від 1 до 10. Виконати такі дії: до задуманого числа додати 3; від того, що вийшло, відняти 1; до того, що вийшло, додати 2; до того, що вийшло, додати 1; до того, що вийшло, додати 2, від того, що вийшло, відняти задумане число. Вивести результат. Перевірте, який результат отримано. Чи можна після повторного запуску програми отримати те ж саме число? Чи означає це, що комп'ютер задумав одне й те саме число?

matematyka.org.ua

```python
import random

# Комп'ютер загадує число від 1 до 10
secret_number = random.randint(1, 10)

# Виконання послідовності дій
result = secret_number
result += 3
result -= 1
result += 2
result += 1
result += 1
result += 2
result -= 2
result -= secret_number

# Виведення результату
print("Загадане число:", secret_number)
print("Результат після виконання дій:", result)
```

Як працює програма:

1. **Комп'ютер загадує число** з діапазону [1, 10] за допомогою `random.randint(1, 10)`.
2. **Послідовно виконуються дії** згідно із завданням: додавання, віднімання тощо.
3. **Результат виводиться на екран.**

Пояснення:

- Після повторного запуску програми ви завжди отримаєте однаковий результат (число 6), незалежно від того, яке число було загадане. Це означає, що виконувані дії є частиною математичного алгоритму, який повертає одне й те саме значення.

Завдання 12. Складіть програми для отримання зображень (мал. 28.6).

matematika-doma.org

```python
import matplotlib.pyplot as plt

# Функція для малювання фігур
def draw_shapes():
    # Налаштування полотна
    fig, axes = plt.subplots(1, 4, figsize=(12, 3))

    # Трикутник
    triangle_x = [0, 1, 0.5, 0]
    triangle_y = [0, 0, 1, 0]
    axes[0].plot(triangle_x, triangle_y, color="blue")
    axes[0].set_title("Трикутник")
    axes[0].axis("equal")

    # Прямокутник
    rectangle_x = [0, 2, 2, 0, 0]
    rectangle_y = [0, 0, 1, 1, 0]
    axes[1].plot(rectangle_x, rectangle_y, color="orange")
    axes[1].set_title("Прямокутник")
    axes[1].axis("equal")
```

matematika-doma.org

```
# Паралелограм
parallelogram_x = [0, 1.5, 2.5, 1, 0]
parallelogram_y = [0, 0, 1, 1, 0]
axes[2].plot(parallelogram_x, parallelogram_y, color="red")
axes[2].set_title("Паралелограм")
axes[2].axis("equal")

# Ромб
rhombus_x = [0, 1, 0, -1, 0]
rhombus_y = [1, 0, -1, 0, 1]
axes[3].plot(rhombus_x, rhombus_y, color="black")
axes[3].set_title("Ромб")
axes[3].axis("equal")

# Відображення
for ax in axes:
    ax.axis("off")
plt.tight_layout()
plt.show()
```

```
# Виклик функції
draw_shapes()
```

informatika-doma.org

Квест 29

Завдання 4

У програмі, написаній мовою програмування Python з використанням значення текстової змінної s1, отримали з2(табл. 29.3). Складіть відповідний програмний код. Перевірте, чи одинакові засоби отримання значення змінної s2 із змінної s 1 використав кожен із вас. Зробіть висновки.

```python
# Задаємо значення змінної s1
s1_1 = "Слово – не спіймаєш, вилетить – не горобець"
s1_2 = "Світнебездобрихлюдей"
s1_3 = "Вік учись – вік живи"

# Перетворення для отримання s2
s2_1 = "Слово – не горобець, вилетить – не спіймаєш"
s2_2 = s1_2.replace("небез", " не без ").replace("людей", " людей")
s2_3 = "Вік живи – вік учись"

# Виведення результатів
print("Значення s1:", s1_1)
print("Значення s2:", s2_1)
print("-----")
print("Значення s1:", s1_2)
print("Значення s2:", s2_2)
print("-----")
print("Значення s1:", s1_3)
print("Значення s2:", s2_3)
```

Пояснення:

1. **s1_1 → s2_1:**
   - Ручна перестановка фрагментів тексту.
2. **s1_2 → s2_2:**
   - Використовується `replace` для вставки пробілів та слів у тексті.
3. **s1_3 → s2_3:**
   - Ручна перестановка частин тексту у зворотному порядку.

Завдання 6. Складіть програму, за допомогою якої зі слова s1 можна отримати слово s2 (табл. 29.5)

informatika-doma.org

```python
# Словник відповідностей між s1 та s2
word_mapping = {
    "рекомендація": "оцінка",
    "університет": "турист",
    "чиновник": "новини",
    "спортсмен": "метро"
}

# Функція для отримання s2 зі слова s1
def get_s2(s1):
    return word_mapping.get(s1, "Невідоме слово")

# Тестування програми
s1_list = ["рекомендація", "університет", "чиновник", "спортсмен"]

for s1 in s1_list:
    s2 = get_s2(s1)
    print(f"Слово s1: {s1} -> Слово s2: {s2}")
```

Пояснення:

1. **Словник** `word_mapping`:
   - Зберігає відповідності між словами `s1` та `s2`.
2. **Функція** `get_s2`:
   - Приймає слово `s1` і повертає відповідне слово `s2` зі словника.
   - Якщо слово `s1` немає у словнику, повертається "Невідоме слово".
3. **Тестування**:
   - Всі слова `s1` з таблиці передаються до функції, а результати відображаються.

Результат:

Для введених слів `s1` програма виведе відповідні слова `s2`.

Завдання 8

У програмі Розрядні одиниці вводять чотиризначне ціле число і отримують запис числа у вигляді суми розрядних одиниць. Наприклад, для введеного числа 5843 отримують 5*1000+8"100+4* 10+3*1'.

matematika-doma.org

Реалізуйте відповідну програму засобами мови програмування.

```
# Введення чотиризначного числа
number = input("Введіть чотиризначне число: ")

# Перевірка, чи є число чотиризначним
if len(number) == 4 and number.isdigit():
    # Розрядні одиниці
    units = [1000, 100, 10, 1]

    # Формування розрядного запису
    result = " + ".join(f"{int(digit)}*{unit}" for digit, unit in zip(number,

    # Виведення результату
    print("Розрядний запис числа:", result)
else:
    print("Помилка: введіть дійсне чотиризначне число.")
```

Як працює програма:

1. **Введення числа:**
   - Користувач вводить чотиризначне число.
2. **Перевірка:**
   - Програма перевіряє, чи число має 4 цифри та є цілим.
3. **Розрядний запис:**
   - Використовується `zip` для з'єднання кожної цифри з відповідною розрядною одиницею (1000, 100, 10, 1).
   - Формується рядок, що відповідає запису числа у вигляді суми розрядних одиниць.
4. **Виведення:**
   - Результат виводиться у вигляді, наприклад, `5*1000 + 8*100 + 4*10 + 3*1`.

Завдання 9. У програмі Код учня перетворюють введене прізвище, ім'я учня, рік народження у код за таким шаблоном (мал. 29.2).

informatika-doma.org

```python
# Введення даних учня
surname = input("Введіть прізвище: ").strip()
name = input("Введіть ім'я: ").strip()
birth_year = input("Введіть рік народження: ").strip()

# Перевірка, чи рік народження має 4 цифри
if len(birth_year) == 4 and birth_year.isdigit():
    # Формування коду
    last_two_digits = birth_year[-2:]  # Останні дві цифри року народження
    first_letter_name = name[0].upper()  # Перша літера імені у верхньому регістрі
    code = f"{last_two_digits}{first_letter_name}{surname}_u"

    # Виведення результату
    print("Код учня:", code)
else:
    print("Помилка: введіть коректний рік народження у форматі YYYY.")
```

Як працює програма:

1. **Введення даних:**
   - Користувач вводить прізвище, ім'я та рік народження.
2. **Перевірка року народження:**
   - Програма перевіряє, чи рік народження має 4 цифри.
3. **Формування коду:**
   - Беруться останні дві цифри року народження.
   - Витягується перша літера імені та перетворюється на верхній регістр.
   - Додається прізвище та суфікс `_u`.
4. **Виведення результату:**
   - Код виводиться у форматі, наприклад, `05JSmith_u`.

Приклад:

Вхідні дані:

- Прізвище: `Smith`
- Ім'я: `John`
- Рік народження: `2005`

Завдання 10

Обговоріть, у чому схожість процесу налагодження програми у середовищі програмування та редагування тексту в середовищі текстового процесора. Чи можна вважати редагування та

matematika-doma.org

налагодження синонімами?

Схожість процесів:

1. Виправлення помилок:

• У програмуванні процес налагодження (debugging) полягає у виявленні та виправленні помилок у коді.

• У текстовому редакторі редагування тексту також передбачає пошук і виправлення помилок (граматичних, стилістичних, друкарських).

2. Оптимізація:

• У програмуванні налагодження часто включає оптимізацію коду, щоб програма працювала ефективніше.

• У текстовому процесорі редагування може включати покращення структури тексту чи формулювання речень.

3. Циклічний процес:

• Налагодження та редагування є ітеративними: після виправлення однієї помилки або недоліку можуть бути виявлені інші.

4. Засоби підтримки:

• У програмуванні для налагодження використовуються спеціальні інструменти, такі як дебагери.

• У текстових редакторах є вбудовані засоби перевірки правопису та граматики.

Відмінність:

• Сутність помилок:

- У програмуванні помилки технічні (синтаксичні, логічні), що впливають на виконання програми.

- У текстовому процесорі помилки переважно мовні та стилістичні.

Чи можна вважати синонімами?

Редагування та налагодження мають багато схожих рис, проте вони не є синонімами, оскільки:

• Налагодження фокусується на технічних аспектах виконання програми.

• Редагування зосереджене на змістовній та мовній якості тексту.

Висновок: Обидва процеси мають схожість у виправленні помилок, але їхні цілі та застосування різні, тому вважати їх синонімами некоректно.

Завдання 11. Ідея гри: "Збери слово"

У грі гравець отримує випадковий набір букв, з яких потрібно скласти слово. Гравець вводить своє слово, і програма перевіряє, чи воно є валідним.

matematika.org.ua

```python
import random

# Список слів для гри
words = ["програмування", "комп'ютер", "алгоритм", "мова", "інформація"]

# Вибір випадкового слова
chosen_word = random.choice(words)

# Перемішування літер слова
shuffled_word = "".join(random.sample(chosen_word, len(chosen_word)))

# Інструкція для гравця
print("Збери слово з поданих літер!")
print("Перемішані літери:", shuffled_word)

# Отримання слова від гравця
user_word = input("Введіть своє слово: ").strip().lower()

# Перевірка відповіді
if user_word == chosen_word:
    print("Вітаємо! Ви правильно склали слово!")
else:
    print(f"На жаль, це неправильне слово. Правильне слово: {chosen_word}")
```

Як працює програма:

1. **Випадковий вибір слова**:
   - Програма вибирає випадкове слово зі списку.
2. **Перемішування літер**:
   - Використовується `random.sample`, щоб перемішати літери.
3. **Ввід гравця**:
   - Гравець вводить слово, яке на його думку відповідає перемішаним літерам.
4. **Перевірка**:
   - Програма порівнює введене слово з оригіналом і виводить результат.

matematika-doma.org

Як вдосконалити:

1. **Словник:**
   - Додати більше слів для гри.
2. **Рівні складності:**
   - Вводити обмеження за часом або кількістю спроб.
3. **Бали:**
   - Додавати систему підрахунку балів за правильні відповід
4. **Багатокористувацький режим:**
   - Організувати гру на декількох гравців.

Приклад виконання:

```
Збери слово з поданих літер!
Перемішані літери: арумгнованпир
Введіть своє слово: програмування
Вітаємо! Ви правильно склали слово!
```

matematika-doma.org

Квест 30

Завдання 1

Складіть у парі 5 виразів на порівняння значень величин різних типів мовою програмування Python. Запропонуйте їх іншій парі. Перевірте правильність отриманих відповідей.

```
# 1. Порівняння чисел
a = 10
b = 20
print("1. Числа:", a, ">", b, "?", a > b)

# 2. Порівняння рядків
str1 = "Python"
str2 = "Programming"
print("2. Рядки:", str1, "==", str2, "?", str1 == str2)

# 3. Порівняння списків
list1 = [1, 2, 3]
list2 = [1, 2, 3]
print("3. Списки:", list1, "==", list2, "?", list1 == list2)

# 4. Порівняння кортежів
tuple1 = (1, 2, 3)
tuple2 = (3, 2, 1)
print("4. Кортежі:", tuple1, ">", tuple2, "?", tuple1 > tuple2)

# 5. Порівняння словників
dict1 = {"a": 1, "b": 2}
dict2 = {"a": 1, "b": 2}
print("5. Словники:", dict1, "==", dict2, "?", dict1 == dict2)
```

Пояснення:

1. Числа: Порівнюються за їх значеннями (наприклад, більше, менше або рівність).

2. Рядки: Порівнюються за алфавітним порядком або на рівність.

3. Списки: Порівнюються елементи списків, починаючи з першого.

matematika.dom.ua

4. Кортежі: Порівняння відбувається аналогічно спискам.
5. Словники: Порівнюються пари "ключ–значення" на рівність.
Приклад результатів:

```
1. Числа: 10 > 20 ? False
2. Рядки: Python == Programming ? False
3. Списки: [1, 2, 3] == [1, 2, 3] ? True
4. Кортежі: (1, 2, 3) > (3, 2, 1) ? False
5. Словники: {'a': 1, 'b': 2} == {'a': 1, 'b': 2} ? True
```

Ця програма демонструє порівняння значень різних типів у Python. Ви також можете спробувати інші варіанти, змінюючи значення або типи даних.

Завдання 2. Як визначити значення логічного виразу, що містить логічні операції and, or та not? Як для пошуку відповіді на це запитання можна використати таблиці істинності?

математика-онлайн.org

Як визначити значення логічного виразу?

Логічний вираз з операціями `and`, `or` та `not` оцінюється відповідно до правил булевої алгебри:

1. `and` **(логічне "і"):**
   - Результат істинний (`True`), якщо **обидва** операнди істинні.
   - Інакше результат хибний (`False`).
   - Приклад: `True and False → False`.
2. `or` **(логічне "або"):**
   - Результат істинний (`True`), якщо **хоча б один** операнд істинний.
   - Інакше результат хибний (`False`).
   - Приклад: `True or False → True`.
3. `not` **(логічне "ні"):**
   - Інвертує значення: `True` стає `False`, а `False` стає `True`.
   - Приклад: `not True → False`.

Таблиці істинності допомагають систематично визначати значення логічного виразу для всіх можливих комбінацій вхідних значень.

Приклад для A and B:

A	B	A and B
True	True	True
True	False	False
False	True	False
False	False	False

mistecka-doma.org

Приклад для A or B:

A	B	A or B
True	True	True
True	False	True
False	True	True
False	False	False

Приклад для not A:

A	not A
True	False
False	True

Як використати таблиці істинності?

1. **Розкласти вираз:**
   - Для складного виразу, наприклад, `(A and B) or not C`, розділіть його на частини:
     - Обчисліть `A and B`.
     - Обчисліть `not C`.
     - Об'єднайте результати за допомогою `or`.
2. **Заповнити таблицю:**
   - Запишіть всі можливі значення змінних `A`, `B`, `C`.
   - Вирахуйте результати для кожної частини виразу.

Таблиця істинності для `(A and B) or not C`:

A	B	C	A and B	not C	(A and B) or not C
True	True	True	True	False	True
True	True	False	True	True	True
True	False	True	False	False	False
True	False	False	False	True	True
False	True	True	False	False	False
False	True	False	False	True	True
False	False	True	False	False	False
False	False	False	False	True	True

математика.дома.org

Висновок:

Для визначення значення логічного виразу потрібно:

1. Використати правила операцій and, or, not.

2. Скласти таблицю істинності, щоб перевірити всі можливі комбінації значень змінних.

Таблиці істинності допомагають побудувати систематичний підхід до оцінки складних виразів.

Завдання 3. Визначте, якого логічного і значення набуває логічний вираз, записаний мовою програмування Python:

```
# 1) False and False
result_1 = False and False
print("1)", result_1)  # Результат: False

# 2) 1 == 1 and 2 == 1
result_2 = 1 == 1 and 2 == 1
print("2)", result_2)  # Результат: False

# 3) 'test' == 'testing'
result_3 = 'test' == 'testing'
print("3)", result_3)  # Результат: False

# 4) 'test' != 'testing'
result_4 = 'test' != 'testing'
print("4)", result_4)  # Результат: True

# 5) True and 0 != 0
result_5 = True and 0 != 0
print("5)", result_5)  # Результат: False

# 6) not (1 == 1 and 0 != 1)
result_6 = not (1 == 1 and 0 != 1)
print("6)", result_6)  # Результат: False
```

matematika.org.ua

Як визначено значення:

• and: Обидва операнди мають бути істинними, щоб результат був True.

• or: Хоча б один операнд має бути істинним для True.

• not: Інвертує значення (істинне стає хибним і навпаки).

Завдання 6. Складіть програму мовою Python, у якій із двох введених цілих чисел A та B (< 1000) виводиться найбільше з них.

```python
# Введення двох чисел
A = int(input("Введіть перше число A (<1000): "))
B = int(input("Введіть друге число B (<1000): "))

# Перевірка, чи числа менші за 1000
if A < 1000 and B < 1000:
    # Визначення більшого числа
    max_number = max(A, B)
    print(f"Найбільше число: {max_number}")
else:
    print("Помилка: обидва числа мають бути меншими за 1000.")
```

Як працює програма:

1. Користувач вводить два числа A і B.
2. Програма перевіряє, чи обидва числа менші за 1000.
3. Якщо умова виконується, використовується функція `max()` для визначення більшого числа.
4. Якщо числа перевищують 1000, виводиться повідомлення про помилку.

Завдання 7. Складіть програму, за допомогою якої можна визначити, з одного слова складається введений текст чи з декількох.

Інформатика-олімпіада.org

```python
# Введення тексту
text = input("Введіть текст: ").strip()

# Перевірка на кількість слів
if " " in text:
    print("Текст складається з декількох слів.")
else:
    print("Текст складається з одного слова.")
```

Як працює програма:

1. Введення тексту: Користувач вводить текст.

2. Перевірка:

• Якщо текст містить пробіл (" "), програма визначає, що в тексті кілька слів.

• Якщо пробілів немає, текст складається з одного слова.

3. Результат: Програма виводить відповідне повідомлення.

Завдання 9. Обговоріть, який оператор розгалуження варто використати у програмі, за допомогою якої описують кількість n горобців, що сидять на гілці (n < 100). Наприклад, для n = 1 – отримують 1 горобець, n = 2 – отримують 2 горобці, n = 2 – отримують 8 горобців. Перевірте власні міркування у програмі в середовищі програмування.

Інформатика.org

```python
# Введення числа n
n = int(input("Введіть кількість горобців n (n < 100): "))

# Перевірка умови
if n < 1 or n >= 100:
    print("Число n повинно бути від 1 до 99.")
elif n == 1:
    print(f"{n} горобець.")
elif n == 2:
    print(f"{n * 4} горобців.")  # Задача припускає, що при n=2 дає 8
else:
    print(f"{n} горобців.")
```

Як працює програма:

1. **Перевірка діапазону:** Якщо n не у межах 1–99, програма виводить повідомлення про помилку.
2. **Умови:**
   - Якщо $n = 1$, програма виводить "1 горобець".
   - Якщо $n = 2$, програма множить n на 4 та виводить результат.
   - Для інших значень n виводиться загальне повідомлення.
3. **Розгалуження:** Використовується оператор `if-elif-else`, що дозволяє додати гнучкі умови.

Завдання 10. Обговоріть, які зміни потрібно внести у програму Максимум із двох, щоб впорядкувати три введені числа за зростанням. Побудуйте блок–схему, створіть відповідну програму та протестуйте її в середовищі програму-вання.

matematika-doma.org

```python
# Введення трьох чисел
a = int(input("Введіть перше число: "))
b = int(input("Введіть друге число: "))
c = int(input("Введіть третє число: "))

# Впорядкування чисел за зростанням
if a > b:
    a, b = b, a
if b > c:
    b, c = c, b
if a > b:
    a, b = b, a

# Виведення результату
print(f"Числа у порядку зростання: {a}, {b}, {c}")
```

Як працює програма:

1. Порівнюються числа попарно:

• Якщо a > b, вони міняються місцями.

• Якщо b > c, вони міняються місцями.

• Повторно перевіряється a > b після перестановки.

2. У результаті числа a,b,c упорядковуються за зростанням.

Блок–схема:

1. Початок.
2. Введення чисел a, b, c
3. Перевірка:

• Якщо a > b, поміняти місцями a і b.

• Якщо b > c, поміняти місцями b і c.

• Якщо a > b (після перестановок), поміняти місцями a і b.

4. Виведення чисел a, b, c.
5. Кінець.

Підтримка-дома.org

Завдання 13. За допомогою умовного оператора мовою програмування Python запишіть фрагмент програмного коду для:

1) заміни значення змінної s дійсного типу його абсолютною величиною;

```
s = float(input("Введіть число: "))
s = abs(s)  # Заміна значення на абсолютну величину
print(f"Абсолютне значення змінної s: {s}")
```

2) присвоєння змінній k значення 0, якщо її початкове значення належало інтервалу (0; 5);

```
k = float(input("Введіть значення змінної k: "))
if 0 < k < 5:
    k = 0  # Присвоєння 0, якщо k належить інтервалу (0; 5)
print(f"Значення змінної k: {k}")
```

3) надання змінній t значення True, якщо введені координати x та y належать першій або третій координатній чверті.

```
x = float(input("Введіть координату x: "))
y = float(input("Введіть координату y: "))

# Перевірка, чи координати належать першій або третій чверті
t = (x > 0 and y > 0) or (x < 0 and y < 0)
print(f"Значення змінної t: {t}")
```

matematika-doma.org