ВІДОКРЕМЛЕННИЙ СТРУКТУРНИЙ ПІДРОЗДІЛ

«ОПТИКО-МЕХАНІЧНИЙ ФАХОВИЙ КОЛЕДЖ КИЇВСЬКОГО НАЦІОНАЛЬНОГО УНІВЕРСИТЕТУ ІМЕНІ ТАРАСА ШЕВЧЕНКА»

Циклова комісія спеціальності «Інженерія програмного забезпечення»

**З В І Т**

**З НАВЧАЛЬНОЇ ПРАКТИКИ**

спеціальність 121 Інженерія програмного забезпечення

освітньо-кваліфікаційний рівень «фаховий молодший бакалавр»

|  |  |  |
| --- | --- | --- |
| ПІДСУМКОВА ОЦІНКА  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (оцінка цифрою та прописом)  Керівники практики від коледжу:  \_\_\_\_\_\_\_\_\_\_\_\_\_\_ А. С. Нікітенко  (оцінка) (підпис)  \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Т. І. Лумпова  (оцінка) (підпис)  \_\_\_\_\_\_\_\_\_\_\_\_\_\_ М. С. Ходжаєв  (оцінка) (підпис) |  | Виконав: |
| студент 2-го курсу групи ІПЗ-23 |
| Костенков Глєб Анатолійович  залікова книжка № \_\_\_\_\_  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (підпис виконавця)  «\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2024 року |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_ А. В. Степанюк  (оцінка) (підпис) |  |  |
| «\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2024 року |  |  |

м. Київ — 2024

ЗАТВЕРДЖУЮ

Директор ОМФК КНУ

імені Тараса Шевченка  
 Борис ГАПРІНДАШВІЛІ

«17» травня 2024 р.

**ІНДИВІДУАЛЬНЕ ЗАВДАННЯ**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| на період | *навчальної* | | практики | |
|  |  | |  | |
| *Костенкова Глєба Анатолійовича* | | | | |
|  | | | | |
| Спеціальність | *121 Інженерія програмного забезпечення* | Група | | *ІПЗ-23* |

|  | **Питання для вивчення** | **Відмітки про виконання** |
| --- | --- | --- |
| **1** | **Вступне заняття.** Вивчення питань охорони праці. Ознайомлення з електронною бібліотекою коледжу. Вивчення правил оформлення звітної документації. |  |
| **2** | **Розділ "Інформатика**" |  |
| 2.1 | Засобами MS Word розробити документ складної структури та зберегти у різних форматах |  |
| 2.2 | Розв’язування рівнянь |  |
| 2.3 | Розв’язування оптимізаційних задач |  |
| 2.4 | Основи статистичного аналізу даних. Ряди даних. Кореляційний аналіз даних |  |
| 2.5 | Обчислення основних статистичних характеристик вибірки |  |
| 2.6 | Візуалізація рядів і трендів даних |  |
| 2.7 | Розв‘язок задач за допомогою засобів табличного редактора MS Excel |  |
| 2.8 | Проектування бази даних та створення бази даних *MS ACCESS* |  |
| 2.9 | Основні поняття реляційних баз даних |  |
| 2.10 | Введення, коригування, вилучення інформації з бази даних *MS ACCESS* |  |
| 2.11 | Вибірка, пошук, фільтрація даних в *MS ACCESS* |  |
| 2.12 | Виконання практичних завдань за допомогою засобів табличного редактора MS Access |  |

|  | **Питання для вивчення** | **Відмітки про виконання** |
| --- | --- | --- |
| **3** | **Розділ "Програмування С++"** |  |
| 3.1 | Алгоритмізація обчислювальних процесів. Розробка постановки задачі, визначення вхідних та вихідних даних, оформлення блок-схем. |  |
| 3.2 | Типи даних в мові С/С++, внутрішнє представлення даних, операції перетворення типів даних. |  |
| 3.3 | Оператор присвоювання. Операції введення- виведення даних |  |
| 3.4 | Програмування лінійних та розгалужених обчислювальних процесів |  |
| 3.5 | Програмування циклічних обчислювальних процесів |  |
| 3.6 | Складання програм |  |
| 3.7 | Виконання тестів |  |
| 4 | **Розділ "** **Практична робота з С++"** |  |
| 4.1 | Тема 1. Введення та виведення у базових консольних програмах |  |
| 4.2 | Тема 2. Структури даних для управління станом програми |  |
| 4.3 | Тема 3. Робота з файлами: запис та читання |  |
| 4.4 | Тема 4. Формати файлів та їх обробка |  |
| 4.5 | Тема 5. Системи контролю версій (VCS) |  |
| 4.6 | Тема 6. Мова Markdown для опису проекті |  |
| 4.7 | Розроблення ігрової програми |  |
| 5 | Розділ ”Веб-дизайн та HTML” |  |
| 5.1 | Мова розмітки HTML та структура Веб-сторінки |  |
| 5.2 | Розмітка тексту за допомогою HTML. Зображення та посилання |  |
| 5.3 | Каскадні таблиці стилів CSS та їх структура |  |
| 5.4 | Спадковості і каскадування за допомогою CSS |  |
| 5.5 | Розробка та оформлення Постановки задачі |  |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | **Питання для вивчення** | | | | **Відмітки про виконання** | | |
| 5.9 | | Розробка HTML документу | | | |  | | |
| **6** | | Систематизація матеріалів, оформлення звіту з навчальної практики | | | |  | | |
| Голова циклової комісії | |  |  | Савєльєва І.В. | |  |  |
|  | | (підпис) |  | (ініціали, прізвище) | |  | (дата) |
| Керівники практики від коледжу | |  |  |  | |  |  |
|  | |  |  | Нікітенко А.С. | |  | 17.05.2024р. |
|  | | (підпис) |  | (ініціали, прізвище) | |  | (дата) |
|  | |  |  | Лумпова Т.І. | |  | 17.05.2024р. |
|  | | (підпис) |  | (ініціали, прізвище) | |  | (дата) |
|  | |  |  | Нікітенко А.С. | |  | 17.05.2024р. |
|  | | (підпис) |  | (ініціали, прізвище) | |  | (дата) |
|  | |  |  | Лумпова Т.І. | |  | 17.05.2024р. |
|  | | (підпис) |  | (ініціали, прізвище) | |  | (дата) |
|  | |  |  |  | |  |  |
|  | |  |  |  | |  |  |

Змн.

Арк.

№ докум.

Підпис

Дата

Арк.

4

Розроб.

Перевір.

Реценз.

В.Я

Н. Контр.

В.Я

Затверд.

Звіт з навчальної практики

Літ.

Аркушів

ІПЗ-23

ОМФК.0121 НП 20

Зміст

[ВСТУП 7](#_Toc168575140)

[**2.1** **Розроблення засобами MS Word документа складної структури** 8](#_Toc168575141)

[**2.2 Розв‘язок задач за допомогою засобів табличного редактора MS Excel** 9](#_Toc168575142)

[**2.3 Виконання** **практичних завдань за допомогою засобів табличного редактора MS Access** 10](#_Toc168575143)

[3 РОЗДІЛ "ПРОГРАМУВАННЯ С++" 12](#_Toc168575144)

[**3.1 Теоретичні відомості** 12](#_Toc168575145)

[3.1.1 Загальні поняття. Елементи мови С++ - константи, змінні, операції, перетворення типів 12](#_Toc168575146)

[3.1.2 Арифметичні операції 14](#_Toc168575147)

[3.1.3 Поняття вхідного та вихідного потоку, найпростіші математичні функції 15](#_Toc168575148)

[3.1.4 Програмування лінійних та розгалужених обчислювальних процесів 17](#_Toc168575149)

[3.1.5 Програмування циклічних обчислювальних процесів 21](#_Toc168575150)

[**3.2 Практичне завдання** 22](#_Toc168575151)

[3.2.1 Завдання 22](#_Toc168575152)

[3.2.2 Постановка задачі 22](#_Toc168575153)

[3.2.3 Код програми 24](#_Toc168575154)

[3.2.4 Скріншот виконання програми 26](#_Toc168575155)

[**3.3 Виконання тестів** 26](#_Toc168575156)

[4 РОЗДІЛ " ПРАКТИЧНА РОБОТА З С++" 26](#_Toc168575157)

[**4.1 Тема 1. Введення та виведення у базових консольних програмах** 26](#_Toc168575158)

[**4.2 Тема 2. Структури даних для управління станом програми** 28](#_Toc168575159)

[**4.3 Тема 3. Робота з файлами: запис та читання** 28](#_Toc168575160)

[**4.4 Тема 4. Формати файлів та їх обробка** 30](#_Toc168575161)

[**4.5 Тема 5. Системи контролю версій (VCS)** 31](#_Toc168575162)

[**4.6 Тема 6. Мова Markdown для опису проекті** 33](#_Toc168575163)

[**4.7 Розроблення ігрової програми** 34](#_Toc168575164)

[4.7.1 Код програми 35](#_Toc168575165)

[4.7.2 Скріншоти виконання програми 40](#_Toc168575166)

[5 РОЗДІЛ ”ВЕБ-ДИЗАЙН ТА HTML” 41](#_Toc168575167)

[**5.1 Мова розмітки HTML та структура Веб-сторінки** 41](#_Toc168575168)

[**5.2 Розмітка тексту за допомогою HTML. Зображення та посилання** 42](#_Toc168575169)

[**5.3 Каскадні таблиці стилів CSS та їх структура** 44](#_Toc168575170)

[**5.4 Спадковості і каскадування за допомогою CSS** 45](#_Toc168575171)

[**5.5 Практична частина** 46](#_Toc168575172)

[5.5.1 Постановки задачі 46](#_Toc168575173)

[5.5.2 HTML документ 46](#_Toc168575174)

[ВИСНОВКИ 47](#_Toc168575175)

[***4 тема:*** 47](#_Toc168575176)

[***5 тема:*** 49](#_Toc168575177)

[ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ ІНФОРМАЦІЇ 51](#_Toc168575178)

[ДОДАТКИ 52](#_Toc168575179)

ОМФК.0121 НП 20

**ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ І ТЕРМІНІВ**

* <html> – позначає початок і кінець HTML-документа.
* <head> – містить метадані веб-сторінки.
* <title> – задає заголовок веб-сторінки, який відображається в заголовку браузера.
* <meta> – використовується для вказання метаданих, таких як кодування символів.
* <style> – містить CSS-стилі для веб-сторінки.
* <body> – визначає основний вміст веб-сторінки.
* <header> – визначає верхній колонтитул документа або розділу.
* <nav> – містить навігаційне меню.
* <ul> – створює ненумерований список.
* <li> – визначає елемент списку.
* <a> – створює посилання на іншу веб-сторінку або документ.
* <main> – визначає головний вміст документа.
* <section> – визначає розділ у документі.
* <h1> до <h6> – визначає заголовки різного рівня на веб-сторінці.
* <p> – створює абзац тексту.
* <br> – вставляє розрив рядка.
* <hr> – вставляє горизонтальну лінію (роздільник).
* <footer> – визначає нижній колонтитул документа або розділу.
* <table> – створює таблицю.
* <thead> – задає заголовок таблиці.
* <tbody> – задає тіло таблиці.
* <tfoot> – задає підвал таблиці.
* <tr> – створює рядок таблиці.
* <th> – створює заголовок стовпця таблиці.
* <td> – створює клітинку таблиці.
* <form> – створює форму для введення даних користувача.
* <input> – створює поле для введення даних.
* <label> – прив'язує мітку до елемента введення.
* <em> – виділяє текст курсивом.
* <b> – виділяє текст жирним.
* <i> – виділяє текст курсивом.
* <u> – підкреслює текст.
* <strong> – виділяє текст жирним шрифтом, підкреслюючи його важливість.
* <small> – відображає текст меншим шрифтом.
* <blockquote> – визначає цитату з відступом.
* <code> – відображає текст у вигляді коду.
* <pre> – відображає попередньо відформатований текст, зберігаючи пробіли та розриви рядків.
* <textarea> – створює багаторядкове поле введення тексту.
* <button> – створює кнопку.
* <select> – створює випадаючий список.
* <option> – створює елемент списку в <select>.
* <article> – визначає самостійний фрагмент вмісту.
* <aside> – визначає побічний вміст, наприклад, бокову панель.
* <img> – вбудовує зображення в HTML-документ.
* <audio> – додає аудіо-контент.
* <source> – вказує джерело мультимедіа для <audio> або <video>.
* <video> – додає відео-контент.

# ВСТУП

У цьому звіті представлені результати навчальної практики, яка охоплює різні аспекти роботи з комп’ютером та програмування. Звіт має кілька розділів, кожен з яких присвячений окремій темі.

У розділі "Інформатика" детально описано створення документів складної структури у MS Word, вирішення задач у MS Excel та роботу у MS Access. Особлива увага приділяється автоматизації та оптимізації робочих процесів для підвищення ефективності і точності виконання завдань.

Розділ, що стосується програмування на C++, включає теоретичні аспекти мови, такі як основні поняття, операції, робота з константами, змінними та перетворення типів. Також розглядаються методи програмування лінійних, розгалужених та циклічних обчислювальних процесів. Практична частина включає написання коду, розробку алгоритмів і виконання тестів для закріплення та розвитку навичок.

Практична робота з C++ включає введення та виведення у консольних програмах, роботу зі структурами даних, обробку файлів, використання різних форматів файлів, систему контролю версій (VCS) та мову Markdown для опису проектів. Особлива увага приділяється розробці ігрової програми, що дозволяє застосувати вивчені концепції на практиці.

Розділ "Веб-дизайн та HTML" охоплює навчання мові розмітки HTML, структури веб-сторінок, використання CSS для оформлення, вставку зображень та посилань. Практична частина включає розробку HTML документа відповідно до задачі, що сприяє розвитку навичок веб-дизайну.

Цей звіт дозволяє застосувати теоретичні знання на практиці, розвинути навички роботи з сучасними інструментами програмного забезпечення та методологіями розробки, що є важливим для професійної діяльності у сфері інформаційних технологій.

**2 РОЗДІЛ "ІНФОРМАТИКА"**

## **2.1 Розроблення засобами MS Word документа складної структури**

Вміння створювати складні структури документів у MS Word є важливим для ефективної роботи з текстовими матеріалами. У цьому розділі розглянемо процес створення таких документів, включаючи застосування стилів форматування, таблиць, виносок, змісту та інших необхідних елементів.

1. ***Створення заголовків та підзаголовків***

Використання заголовків та підзаголовків допомагає структурувати документ і зробити його більш читабельним. MS Word пропонує вбудовані стилі для заголовків, які можна легко застосувати:

* + Заголовок 1
  + Заголовок 2
  + Заголовок 3

1. ***Форматування тексту***

Використання стилів тексту дозволяє підтримувати консистентність у форматуванні. Для цього можна застосовувати вбудовані стилі або створювати власні:

* + Стиль "Звичайний" для основного тексту
  + Стиль "Цитата" для виділення важливих висловів

1. ***Створення таблиць***

Таблиці є важливим елементом для подання даних у структурованому вигляді. У MS Word можна створювати та форматувати таблиці, використовуючи інструменти вкладки "Вставлення":

* + Додавання таблиці
  + Об'єднання та розділення комірок
  + Форматування меж та заливки

1. ***Додавання виносок і приміток***

Виноски та примітки використовуються для надання додаткової інформації або посилань. Це допомагає уникнути перевантаження основного тексту. У MS Word можна додавати виноски та примітки за допомогою вкладки "Посилання".

1. ***Створення змісту***

Зміст документа дозволяє швидко знайти потрібний розділ. MS Word автоматично генерує зміст на основі заголовків, застосованих у документі:

* + Вставка змісту
  + Оновлення змісту

Створення складної структури документа у MS Word включає застосування різноманітних інструментів і функцій для форматування тексту, створення таблиць, вставки виносок і приміток, а також автоматичне створення змісту. Ці вміння дозволяють створювати документи в професійному стилі, які зручно читати та використовувати.

## **2.2 Розв‘язок задач за допомогою засобів табличного редактора MS Excel**

За допомогою табличного редактора MS Excel можна ефективно розв'язувати різноманітні задачі, які включають обробку даних, виконання розрахунків та аналіз результатів. У цьому розділі ми розглянемо процес розв'язання різних задач за допомогою MS Excel та покажемо приклади використання.

1. ***Введення даних***Першим етапом є введення вхідних даних на відповідний аркуш MS Excel. Це можуть бути числові дані, тексти, дати та інше. Наприклад, для задачі обчислення середньої температури за місяць, ми вводимо значення температур для кожного дня.
2. ***Формули та функції***

MS Excel має великий набір вбудованих функцій та формул, які дозволяють виконувати різноманітні розрахунки. Наприклад, для обчислення середнього значення температури за місяць ми можемо скористатися функцією AVERAGE.

=AVERAGE(B2:B31)

Де B2:B31 - діапазон клітин, що містять значення температур за кожен день місяця.

1. ***Аналіз результатів***

Після застосування відповідних формул та функцій, можна проаналізувати отримані результати. Наприклад, визначити максимальне та мінімальне значення температури за місяць, ідентифікувати найбільш часто зустрічаючеся значення температури, а також розрахувати медіану.

1. ***Візуалізація даних***

MS Excel дозволяє створювати різноманітні графіки і діаграми для візуалізації даних. Наприклад, можна побудувати графік температурного режиму за два місяці, що допомагає краще розібратися в динаміці змін температури протягом цього періоду.

Інструменти табличного редактора MS Excel є надзвичайно корисними для вирішення різноманітних завдань, пов'язаних з обробкою даних, виконанням розрахунків та аналізом результатів. Вони дозволяють ефективно працювати з великим обсягом інформації і допомагають зробити висновки на основі отриманих даних.

## **2.3 Виконання** **практичних завдань за допомогою засобів табличного редактора MS Access**

MS Access є частиною пакету Microsoft Office і використовується для управління базами даних. Вона дозволяє створювати, керувати та аналізувати бази даних, надаючи широкий спектр інструментів для роботи з даними.

Програма має зручний інтерфейс для створення таблиць, в яких можна зберігати різноманітну інформацію, таку як дані про клієнтів, замовлення та товари. Використовуючи конструктор таблиць, можна визначати поля і їх типи даних, такі як текст, число або дата/час, що забезпечує правильне введення та зберігання інформації.

MS Access дозволяє створювати запити для отримання необхідних даних, наприклад, замовлення певного клієнта за певний період часу або товари, що перевищують певну цінову категорію. Це дозволяє ефективно аналізувати інформацію з великих обсягів даних.

Форми в MS Access надають зручний інтерфейс для введення і відображення даних, такий як форми для додавання нових клієнтів або редагування існуючих записів. Вони містять різноманітні елементи керування, такі як текстові поля і випадаючі списки, що полегшує взаємодію з базою даних і зменшує ризик помилок.

Звіти в MS Access використовуються для створення структурованих і наочних звітів на основі даних, зокрема звітів про продажі за місяць або стан запасів на складі. Вони можуть бути форматовані та містити графічні елементи і діаграми для полегшення візуального аналізу даних.

Узагальнюючи, MS Access забезпечує потужні інструменти для ефективного управління даними, аналізу та візуалізації інформації, що робить її корисним інструментом для професіоналів у різних галузях.

# 3 РОЗДІЛ "ПРОГРАМУВАННЯ С++"

## **3.1 Теоретичні відомості**

### 3.1.1 Загальні поняття. Елементи мови С++ - константи, змінні, операції, перетворення типів

Константи

Константи — це змінні, значення яких не можна змінювати після ініціалізації. Вони оголошуються за допомогою ключового слова const.

*const int MAX\_AGE = 100;*

*const double PI = 3.14159;*

Змінні

Змінні — це іменовані області пам'яті, що використовуються для зберігання даних. В C++ змінні повинні бути оголошені перед використанням. Кожна змінна має тип даних, який визначає тип значень, що можуть бути збережені в цій змінній.

*int age = 25;*

*double salary = 4500.50;*

*char grade = 'A';*

*bool isValid = true;*

Операції

Операції дозволяють виконувати дії над змінними та константами. В C++ існують арифметичні, логічні, порівняльні, бітові та інші операції.

* Арифметичні операції: +, -, \*, /, %

*int a = 5, b = 2;*

*int sum = a + b; // 7*

*int difference = a - b; // 3*

*int product = a \* b; // 10*

*int quotient = a / b; // 2*

*int remainder = a % b; // 1*

* Логічні операції: &&, ||, !

*bool x = true, y = false;*

*bool result = x && y; // false*

*result = x || y; // true*

*result = !x; // false*

* Порівняльні операції: ==, !=, <, >, <=, >=

*int a = 5, b = 2;*

*bool isEqual = (a == b); // false*

*bool isNotEqual = (a != b); // true*

*bool isLess = (a < b); // false*

*bool isGreater = (a > b); // true*

*bool isLessOrEqual = (a <= b); // false*

*bool isGreaterOrEqual = (a >= b); // true*

* Бітові операції: &, |, ^, ~, <<, >>

*int a = 5; // 0101 in binary*

*int b = 3; // 0011 in binary*

*int result = a & b; // 0001 (1 in decimal)*

*result = a | b; // 0111 (7 in decimal)*

*result = a ^ b; // 0110 (6 in decimal)*

*result = ~a; // 1010 (inverted bits)*

*result = a << 1; // 1010 (10 in decimal, left shift by 1)*

*result = a >> 1; // 0010 (2 in decimal, right shift by 1)*

Перетворення типів

Перетворення типів необхідне для зміни типу даних змінної на інший тип. Існує два види перетворень: неявні та явні.

* Неявне перетворення: Виконується автоматично компілятором, коли це можливо.

*int a = 10;*

*double b = a; // неявне перетворення з int до double*

* Явне перетворення: Виконується за допомогою оператора кастингу.

*double a = 9.7;*

*int b = (int)a; // явне перетворення з double до int*

### 3.1.2 Арифметичні операції

* Додавання (+) Використовується для додавання двох операндів.

*int a = 10;*

*int b = 5;*

*int sum = a + b; // sum буде дорівнювати 15*

* Віднімання (-) Використовується для віднімання одного операнда від іншого.

*int a = 10;*

*int b = 5;*

*int difference = a - b; // difference буде дорівнювати 5*

* Множення (\*) Використовується для множення двох операндів.

*int a = 10;*

*int b = 5;*

*int product = a \* b; // product буде дорівнювати 50*

* Ділення (/) Використовується для ділення одного операнда на інший.

*int a = 10;*

*int b = 3;*

*int quotient = a / b; // quotient буде дорівнювати 3*

*double result = (double)a / b; // result буде дорівнювати 3.33333*

* Залишок від ділення (%) Використовується для обчислення залишку від ділення одного операнда на інший.

*int a = 10;*

*int b = 3;*

*int remainder = a % b; // remainder буде дорівнювати 1*

### 3.1.3 Поняття вхідного та вихідного потоку, найпростіші математичні функції

Вхідний потік (input stream)

Вхідний потік використовується для читання даних. Найбільш поширеним прикладом вхідного потоку є std::cin, який використовується для читання даних з клавіатури.

*#include <iostream>*

*using namespace std;*

*int main() {*

*int number;*

*cout << "Enter a number: ";*

*cin >> number; // читання числа з клавіатури*

*cout << "You entered: " << number << endl;*

*return 0;*

*}*

Вихідний потік (output stream)

Вихідний потік використовується для запису даних. Найбільш поширеним прикладом вихідного потоку є std::cout, який використовується для виведення даних на екран.

*#include <iostream>*

*using namespace std;*

*int main() {*

*cout << "Hello, World!" << endl; // виведення тексту на екран*

*return 0;*

*}*

Основні математичні функції

* sqrt(x): Обчислює квадратний корінь з x.

*#include <iostream>*

*#include <cmath>*

*using namespace std;*

*int main() {*

*double number = 9.0;*

*double result = sqrt(number);*

*cout << "Square root of " << number << " is " << result << endl;*

*return 0;*

*}*

* pow(base, exponent): Підносить base до степеня exponent.

*#include <iostream>*

*#include <cmath>*

*using namespace std;*

*int main() {*

*double base = 2.0;*

*double exponent = 3.0;*

*double result = pow(base, exponent);*

*cout << base << " raised to the power of " << exponent << " is " << result << endl;*

*return 0;*

*}*

* abs(x): Обчислює абсолютне значення x.

*#include <iostream>*

*#include <cmath>*

*using namespace std;*

*int main() {*

*int number = -10;*

*int result = abs(number);*

*cout << "Absolute value of " << number << " is " << result << endl;*

*return 0;*

*}*

### 3.1.4 Програмування лінійних та розгалужених обчислювальних процесів

***Лінійні обчислювальні процеси***

Лінійні обчислювальні процеси виконуються послідовно, одна команда за іншою, без відхилень або умовних переходів. У цьому типі обчислювального процесу всі команди виконуються в тому порядку, в якому вони записані в коді.

***Розгалужені обчислювальні процеси***

Розгалужені обчислювальні процеси включають умовні оператори, які дозволяють виконувати різні блоки коду залежно від певних умов. Найбільш поширені умовні оператори в C++ — це if, else if, else та оператор вибору switch.

Оператор if

*#include <iostream>*

*using namespace std;*

*int main() {*

*int a = 5;*

*int b = 10;*

*if (a > b) {*

*cout << "a is greater than b" << endl;*

*} else if (a < b) {*

*cout << "a is less than b" << endl;*

*} else {*

*cout << "a is equal to b" << endl;*

*}*

*return 0;*

*}*

У цьому прикладі програма перевіряє умову a > b. Якщо умова виконується, виконується блок коду всередині if. Якщо ні, перевіряється наступна умова a < b. Якщо жодна з умов не виконується, виконується блок коду всередині else.

***Оператор switch***

Оператор switch використовується для вибору однієї з багатьох можливих гілок виконання залежно від значення виразу.

*#include <iostream>*

*using namespace std;*

*int main() {*

*int day = 3;*

*switch (day) {*

*case 1:*

*cout << "Monday" << endl;*

*break;*

*case 2:*

*cout << "Tuesday" << endl;*

*break;*

*case 3:*

*cout << "Wednesday" << endl;*

*break;*

*case 4:*

*cout << "Thursday" << endl;*

*break;*

*case 5:*

*cout << "Friday" << endl;*

*break;*

*case 6:*

*cout << "Saturday" << endl;*

*break;*

*case 7:*

*cout << "Sunday" << endl;*

*break;*

*default:*

*cout << "Invalid day" << endl;*

*break;*

*}*

*return 0;*

*}*

У цьому прикладі значення змінної day порівнюється з кожним значенням case. Якщо знаходиться відповідність, виконується відповідний блок коду. break використовується для виходу з блоку switch після виконання відповідного блоку коду, щоб запобігти виконанню наступних блоків.

## **3.2 Практичне завдання**

### 3.2.1 Завдання

Обчислити і вивести на екран у табличному вигляді значення функції f(x) на заданому інтервалі зміни значень аргументу х від xпоч до xкін з кроком h.

Коефіцієнти a, b, c – дійсні числа. Значення a, b, c, xпоч, xкін, h вводити з клавіатури. Передбачити перевірку допустимості введених значень.

### 3.2.2 Постановка задачі



### 3.2.3 Код програми

#include <iostream>

#include <iomanip>

#include <cmath>

#include <string>

#include <stdexcept>

#include <windows.h>

enum Message { ZeroDivide = 1, SqrtNegative };

void handleError(int);

float calculateFunction(float x, float a, float b, float c);

int main() {

SetConsoleOutputCP(CP\_UTF8);

SetConsoleCP(CP\_UTF8);

float a, b, c;

float Beg, End, Step;

std::cout << " a= "; std::cin >> a;

std::cout << " b= "; std::cin >> b;

std::cout << " c= "; std::cin >> c;

do {

std::cout << " Beg= "; std::cin >> Beg;

std::cout << " End= "; std::cin >> End;

std::cout << " Step= "; std::cin >> Step;

} while (!(((Beg >= End && Step < 0) || (Beg <= End && Step > 0)) && Step != 0));

if (Beg >= End && Step < 0) {

for (float x = Beg; x >= End; x += Step) {

float Fx = calculateFunction(x, a, b, c);

std::cout << "x=" << x << " F(x)=" << Fx << std::endl;

}

}

else {

for (float x = Beg; x <= End; x += Step) {

float Fx = calculateFunction(x, a, b, c);

std::cout << "x=" << x << " F(x)=" << Fx << std::endl;

}

}

return 0;

}

float calculateFunction(float x, float a, float b, float c) {

float Fx;

if (x + 10 > 0 && c > 3) {

if (a \* x == 0) { handleError(ZeroDivide); return -1; }

Fx = (b + 1) / (a \* x) - (1.0 / 2) \* x;

}

else if (x + 10 < 0 && b - c < 0) {

Fx = x - (2 \* a) / (b \* b) + x;

}

else {

if (15 \* a \* b \* c == 0) { handleError(ZeroDivide); return -1; }

Fx = sqrt(x \* x + 1) / (15 \* a \* b \* c);

}

return Fx;

}

void handleError(int itext) {

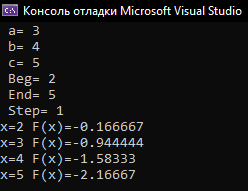
std::string MesText;

if (itext == ZeroDivide) MesText = "Zero Divide";

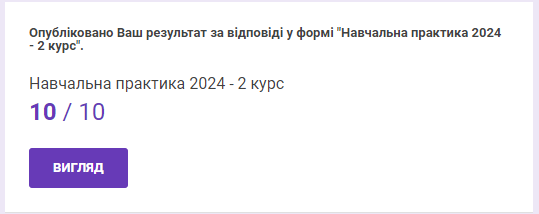
std::cout << MesText << std::endl;

}

### 3.2.4 Скріншот виконання програми



## **3.3 Виконання тестів**



# 4 РОЗДІЛ " ПРАКТИЧНА РОБОТА З С++"

## **4.1 Тема 1. Введення та виведення у базових консольних програмах**

Введення та виведення у базових консольних програмах є одним із фундаментальних аспектів програмування. Ці операції дозволяють користувачам спілкуватися з програмою та отримувати результати її роботи через консольний інтерфейс.

***1. Введення даних:*** У мові C++ для цього використовується об'єкт std::cin, який дозволяє отримати дані з клавіатури. Наприклад:

#include <iostream>

int main() {

int number;

std::cout << "Enter a number: ";

std::cin >> number;

std::cout << "You entered: " << number << std::endl;

return 0;

}

У цьому прикладі користувач вводить число з клавіатури, а програма виводить його назад у консоль.

***2. Виведення даних:*** Для виведення даних у консоль у мові C++ використовується об'єкт std::cout. Наприклад:

#include <iostream>

int main() {

std::cout << "Hello, world!" << std::endl;

return 0;

}

У цьому прикладі рядок "Hello, world!" виводиться у консоль за допомогою об'єкта std::cout.

Ці операції введення та виведення даних у консоль дозволяють створювати взаємодію між користувачем та програмою, а також надають можливість відображення результатів обчислень та роботи програми.

## **4.2 Тема 2. Структури даних для управління станом програми**

У програмуванні мовою C++, структури даних, що керують станом програм, відіграють ключову роль у організації та управлінні даними, необхідними для контролю різних аспектів програми, таких як стан об'єктів, параметри функцій та поточний режим роботи. Основними інструментами для цього є класи і структури. Класи дозволяють об'єднувати дані та методи в один об'єкт, який може мати різні рівні доступу до своїх елементів. Структури також використовуються для організації даних, але мають більш обмежені можливості порівняно з класами.

Для ефективного управління станом програм використовуються різні концепції об'єктно-орієнтованого програмування, такі як інкапсуляція, наслідування та поліморфізм. Ці концепції дозволяють створювати програми, які легко розширювати та змінювати з часом.

Наприклад, для управління станом графічного інтерфейсу можна створювати класи, що представляють окремі вікна, кнопки, тексти тощо. Кожен об'єкт цих класів зберігає дані про свій поточний стан (наприклад, текст кнопки або видимість вікна) і має методи для зміни цього стану (наприклад, зміну тексту кнопки або закриття вікна).

Ефективне управління станом програми є важливим аспектом розробки програмного забезпечення, і вміння правильно застосовувати структури даних для цієї мети значно спрощує процес програмування і покращує якість результуючого коду.

## **4.3 Тема 3. Робота з файлами: запис та читання**

JSON (JavaScript Object Notation) - це легкий формат обміну даними, який використовується для зберігання та передачі структурованих даних між різними системами. Він базується на синтаксисі JavaScript, але може бути використаний в багатьох інших мовах програмування.

Основні складові JSON - це пари "ключ-значення", які можуть бути вкладені одна в одну для представлення складних структур даних. JSON широко використовується для обміну даними між клієнтськими та серверними додатками, а також для зберігання налаштувань, конфігурацій та інших структурованих даних.

***Робота з файлами JSON:***

Переваги:

1) Простота читання та редагування: Формат JSON має зрозумілу структуру, яка полегшує як людям, так і комп'ютерам читання та редагування файлів JSON.

2) Підтримка багатьох мов програмування: Більшість сучасних мов мають інструменти для роботи з JSON, що робить його досить універсальним для використання.

3) Гнучкість: JSON дозволяє представляти різноманітні дані, включаючи тексти, числа, масиви та об'єкти.

4) Популярність: JSON є одним з найпоширеніших форматів обміну даними в програмуванні, що сприяє наявності багатьох інструментів та бібліотек для його використання.

Недоліки:

1)Великий обсяг даних: JSON може стати неефективним для обробки великих обсягів даних через багато зайвих символів для форматування.

2)Відсутність вбудованих засобів для роботи з великими обсягами даних: JSON не надає механізмів для індексації або пошуку, що може ускладнити роботу з великими файлами даних.

XML (Extensible Markup Language) - це розширювана мова розмітки, яка використовується для представлення та обміну структурованими даними між комп'ютерними системами. XML базується на тезі, що документ має бути структуований та зрозумілий як комп'ютерам, так і людям.

В основі XML лежить синтаксис тегів, подібний до HTML, але він не має заздалегідь визначених тегів та може бути розширений для використання в будь-якому контексті. XML використовується для створення структурованих документів, таких як конфігураційні файли, налаштування програм, обміну даними між веб-серверами та клієнтами, а також для зберігання даних в базах даних.

***Робота з файлами XML:***

Переваги:

1)Структурованість: XML має чітку та строго визначену структуру, що робить його ідеальним для представлення складних структур даних та документів з вкладеними елементами.

2)Можливість валідації: XML легко піддається валідації за допомогою схем XSD (XML Schema Definition), що дозволяє перевіряти коректність структури та типів даних у файлі.

3)Підтримка стандартів: XML підтримується багатьма стандартами та протоколами, що робить його важливим для обміну даними між різними системами.

Недоліки:

1) Великий обсяг даних: XML-файли можуть стати досить великими через значну кількість тегів та атрибутів, що може призвести до збільшення розміру файлу та ускладнення його обробки.

2) Складність читання та редагування: Велика кількість тегів та вкладеність може зробити XML-файли складними для читання та редагування людьми, особливо в разі обробки великих обсягів даних

## **4.4 Тема 4. Формати файлів та їх обробка**

У мові програмування C++ існують два основних формати файлів для зберігання та обробки даних.

1) Текстові файли: Це файли, які зберігають дані у звичайному текстовому форматі. Вони можуть бути зручно читані та редаговані людиною за допомогою текстових редакторів. У C++ для роботи з текстовими файлами використовуються потоки введення/виведення (ifstream та ofstream).

Плюси: Легко зрозумілі та зручні для редагування. Ідеально підходять для зберігання конфігурацій, логів та інших текстових даних.

Мінуси: Менш ефективні для великих обсягів даних через потребу в більшому просторі для зберігання. Обробка може бути менш ефективною порівняно з бінарними файлами через конвертацію даних у текстовий формат.

Бінарні файли: Вони містять дані у вигляді двійкового коду, що дозволяє ефективніше зберігати великі обсяги інформації. У C++ також використовуються потоки введення/виведення, але операції читання та запису відбуваються в бінарному форматі.

Плюси: Ефективніші для зберігання великих обсягів даних, оскільки не потребують такого великого обсягу простору для зберігання. Обробка може бути швидшою, оскільки дані не потребують конвертації в текст.

Мінуси: Не підходять для читання та редагування людиною через представлення у вигляді двійкового коду. Менш зручні для відладки та ручних втручань

.

Ці два типи файлів мають свої переваги та обмеження, і вибір між ними залежить від конкретних вимог та характеристик програми.

**4.5 Тема 5. Системи контролю версій (VCS)**

Системи контролю версій (VCS) - це програмні засоби, які використовуються для відстеження змін у файлів і коду програм під час розробки програмного забезпечення.

Основна мета систем контролю версій (VCS) полягає в зберіганні історії змін програмного коду, сприянні одночасній роботі багатьох співробітників над одним проектом і можливості відновлення попередніх версій програми у випадку помилок або необхідності повернення до попередньої функціональності.

Основні переваги систем контролю версій (VCS) включають:

1) Історія змін: Всі модифікації коду або файлів зберігаються і можуть бути переглянуті, включаючи коментарі до змін та інформацію про автора та час змін.

2) Робота з різними версіями: Можливість відновлювати попередні версії файлів та порівнювати різні версії для аналізу змін.

3) Гілки розробки: Здатність створювати окремі гілки для паралельної розробки різних функціональностей або аспектів проекту.

4) Співпраця: Можливість одночасної роботи кількох розробників над одним проектом, автоматичне об'єднання їхніх змін.

Централізовані та розподілені системи контролю версій відрізняються тим, де зберігається головний репозиторій та як розробники отримують доступ до нього:

1) Централізовані системи (наприклад, SVN): Репозиторій зберігається на центральному сервері, доступ до нього здійснюється через мережу.

2) Розподілені системи (наприклад, Git): Кожен розробник має свою копію повного репозиторію, з якою вони можуть працювати незалежно.

Функції такі як відгалуження та злиття (Branching and Merging), командна робота (Collaboration), робота з галузями (Forking) та відстеження проблем (Issue Tracking) дозволяють розробникам ефективно співпрацювати над проектами, відстежувати прогрес та вносити зміни з мінімальними ризиками.

## **4.6 Тема 6. Мова Markdown для опису проекті**

Markdown - це проста у використанні мова розмітки, призначена для форматування тексту в веб-середовищах. Вона широко застосовується для створення різноманітних документів, включаючи описи проектів, README файли, блоги та форуми.

***Основні можливості Markdown***

1. ***Заголовки****:* Використовуються знаки # для створення заголовків різних рівнів. Кількість # визначає рівень заголовка (від 1 до 6).

markdown

# Заголовок 1

## Заголовок 2

### Заголовок 3

1. ***Форматування тексту****:*
   * ***Жирний текст***: Обгорніть текст подвійними зірочками або підкресленнями (\*\*текст\*\* або \_\_текст\_\_).
   * ***Курсив****:* Обгорніть текст однією зірочкою або підкресленням (\*текст\* або \_текст\_).
   * ***Закреслений текст***: Використовуйте подвійні тильди (~~текст~~).

markdown

Це \*\*жирний текст\*\*, це \*курсив\*, і це ~~закреслений текст~~.

1. ***Списки****:*
   * ***Нумеровані списки****:* Використовуйте числа з крапкою.
   * ***Марковані списки****:* Використовуйте зірочки, плюси або дефіси.

markdown

1. Пункт 1

2. Пункт 2

- Підпункт 1

- Підпункт 2

- Пункт 1

- Пункт 2

1. ***Код****:*
   * ***Інлайн код****:* Обгорніть текст зворотніми лапками (``).
   * ***Блок коду***: Використовуйте три зворотні лапки (```).

markdown

Це `інлайн код`.

```cpp

#include <iostream>

using namespace std;

int main() {

cout << "Hello, World!" << endl;

return 0;

}

1. ***Посилання та зображення****:*
   * ***Посилання****:* Обгорніть текст в квадратні дужки, а URL - в круглі дужки.
   * ***Зображення*:** Схоже на посилання, але з префіксом !.

markdown

[Google](https://www.google.com)

## **4.7 Розроблення ігрової програми**

Мій код є реалізацією гри "хрестики нолики" на мові програмування C++. В головній функції виконується основний цикл гри. Гравець вводить координати для свого ходу, після кожного ходу перевіряється наявність переможця або нічии, та виводиться відповідне повідомлення. Після закінчення гри поляризація оновлюється для нової гри. drawOneRow(int rowNumber) - виводить один рядок ігрового поля на екран, а drawBoard() - використовує drawOneRow для виведення усього ігрового поля на екран. validateInput(int row, int col) - перевіряє, чи введені координати (рядок і стовпець) знаходяться в межах поля і чи вони не зайняті, setBoardCellValue(int row, int col) - встановлює значення (символ "X" або "O") на відповідному місці на полі, checkWinner() - перевіряє, чи є переможець в поточному стані поля, endGame(const string& winner) - оголошує переможця, оновлює лічильники перемог та очищає поле для наступної гри.

Ця програма створює просте текстове представлення гри "Хрестики-нолики" і дозволяє гравцям змагатися один з одним.

### 4.7.1 Код програми

#include <iostream>

using namespace std;

const int SIZE = 3;

char board[SIZE][SIZE] = {

{' ', ' ', ' '},

{' ', ' ', ' '},

{' ', ' ', ' '}

};

char currentSymbol = 'X';

int winsX = 0;

int winsO = 0;

void drawOneRow(int rowNumber) {

for (int i = 0; i < SIZE; ++i) {

cout << board[rowNumber][i];

if (i < SIZE - 1) cout << " | ";

}

cout << endl;

}

void drawBoard() {

for (int i = 0; i < SIZE; ++i) {

drawOneRow(i);

if (i < SIZE - 1) cout << "---------" << endl;

}

}

bool validateInput(int row, int col) {

return (row >= 0 && row < SIZE && col >= 0 && col < SIZE && board[row][col] == ' ');

}

void setBoardCellValue(int row, int col) {

board[row][col] = currentSymbol;

}

string checkWinner() {

for (int i = 0; i < SIZE; ++i) {

// Check rows

if (board[i][0] == currentSymbol && board[i][1] == currentSymbol && board[i][2] == currentSymbol)

return string(1, currentSymbol);

// Check columns

if (board[0][i] == currentSymbol && board[1][i] == currentSymbol && board[2][i] == currentSymbol)

return string(1, currentSymbol);

}

// Check diagonals

if (board[0][0] == currentSymbol && board[1][1] == currentSymbol && board[2][2] == currentSymbol)

return string(1, currentSymbol);

if (board[0][2] == currentSymbol && board[1][1] == currentSymbol && board[2][0] == currentSymbol)

return string(1, currentSymbol);

return "";

}

void updateCurrentSymbol() {

currentSymbol = (currentSymbol == 'X') ? 'O' : 'X';

}

void endGame(const string& winner) {

if (winner == "X") winsX++;

else if (winner == "O") winsO++;

cout << "Winner is " << winner << "!" << endl;

cout << "Score - X: " << winsX << ", O: " << winsO << endl;

for (int i = 0; i < SIZE; ++i)

for (int j = 0; j < SIZE; ++j)

board[i][j] = ' ';

currentSymbol = 'X';

}

int main() {

int row, col;

while (true) {

drawBoard();

cout << "Current symbol: " << currentSymbol << endl;

cout << "Enter row (0-2) and column (0-2): ";

cin >> row >> col;

if (validateInput(row, col)) {

setBoardCellValue(row, col);

string winner = checkWinner();

if (!winner.empty()) {

drawBoard();

endGame(winner);

}

else {

updateCurrentSymbol();

}

}

else {

cout << "Invalid move. Try again." << endl;

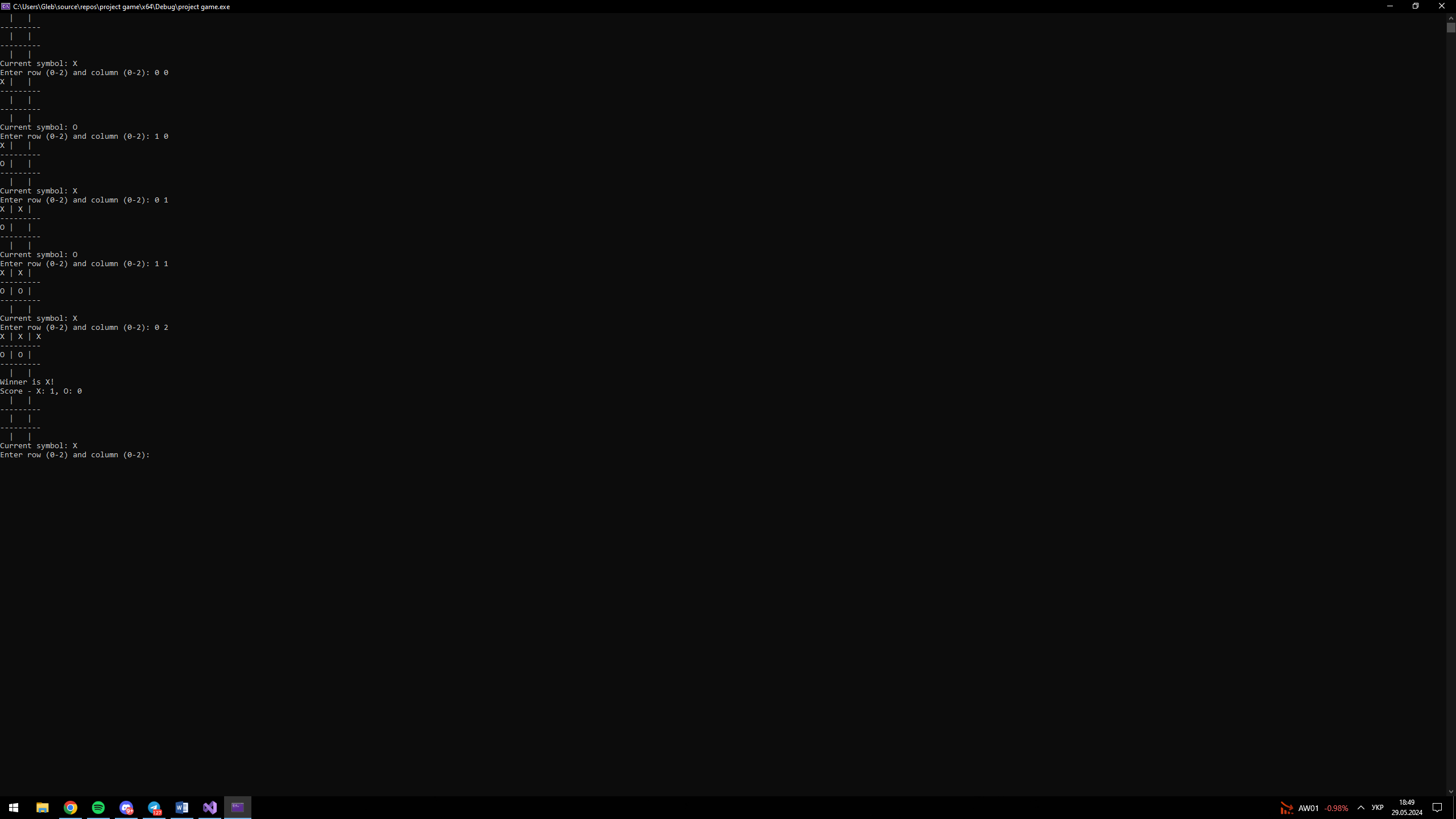
}

}

return 0;

}

### 4.7.2 Скріншоти виконання програми



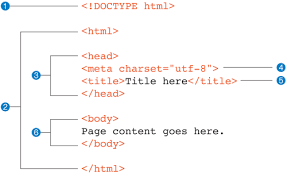
# 5 РОЗДІЛ ”ВЕБ-ДИЗАЙН ТА HTML”

## **5.1 Мова розмітки HTML та структура Веб-сторінки**

HTML — це мова розмітки , яка визначає структуру вмісту. HTML складається з ряду елементів , які використовують для включення або обгортання різних частин вмісту, щоб він виглядав певним чином або діяв певним чином. Об’ємні теги можуть створювати гіперпосилання на слово чи зображення в іншому місці, виділяти слова курсивом, збільшувати або зменшувати шрифт тощо.

Кожен HTML-документ, який відповідає специфікації HTML будь-якої версії, повинен починатися з рядка оголошення версії HTML <!DOCTYPE ...>, який зазвичай виглядає приблизно так: <!DOCTYPE html>.

* <html></html>— <html>елемент. Цей елемент огортає весь вміст усієї сторінки та іноді відомий як кореневий елемент. Він також містить langатрибут, що встановлює основну мову документа.
* <head></head>— <head>елемент. Цей елемент діє як контейнер для всього того, що ви хочете включити на сторінку HTML, але не є вмістом, який ви показуєте глядачам вашої сторінки. Це включає такі речі, як ключові слова та опис сторінки, які ви хочете відображати в результатах пошуку, CSS для стилізації нашого вмісту, оголошення набору символів тощо.
* <meta charset="utf-8">— Цей елемент встановлює набір символів у вашому документі на UTF-8, який включає більшість символів переважної більшості письмових мов.
* <title></title>— <title>елемент. Це встановлює назву вашої сторінки, яка відображається на вкладці веб-переглядача, у яку завантажується сторінка.
* <body></body>— <body>елемент. Тут міститься весь вміст, який ви хочете показати веб-користувачам, коли вони відвідують вашу сторінку, будь то текст, зображення, відео, ігри, звукові доріжки для відтворення чи будь-що інше.

 За своїм призначенням та дією елементи HTML можна розділити на три категорії: *структурні, презентаційні* та *елементи для інтерактивної взаємодії з користувачем.*

*Структурні.* Елементи структурної розмітки застосовуються для опису семантики тексту та задання структури документу. В свою чергу, в залежності від змісту документа, структурні елементи поділяються на: текст, списки, таблиці, гіперпосилання та додаткові об'єкти (зображення, аплети та ін). Вони не зазначають ніякого спеціального (візуального) відтворення тексту, проте більшість браузерів мають стандартні стилі форматування для кожного елемента.

*Презентаційні*. Елементи презентаційної розмітки застосовуються задля опису зовнішнього вигляду документа та не зазначають при цьому функцій тексту, до якого вони застосовуються. Більшість з цих елементів є такими, що не рекомендується застосовувати у розмітці. В залежності від свого змісту, елементи презентаційної розмітки поділяють на: елементи додавання каскадних таблиць стилів, візуальні елементи тексту (шрифти, кольори, насиченість та нахил щрифту, лінії та інші).

*Елементи інтерактивної взаємодії.* Елементи інтерактивної взаємодії дозволяють користувачам впливати на вміст або вигляд документу. Вони поділяються на два види: форми для введення данних користувачами та скрипти для активних документів.

## **5.2 Розмітка тексту за допомогою HTML. Зображення та посилання**

Щоб реалізувати семантичну розмітку, HTML надає спеціальні теги, які можна використовувати для представлення таких розділів, наприклад:

* заголовок: <header> .
* панель навігації: <nav> .
* основний вміст:з різноманітними підрозділами вмісту, <main>представленими елементами , і .<article><section><div>
* бічна панель: <aside> ; часто розміщують всередині <main>
* колонтитул: <footer>

Абзаци можна утворити завдяки тегу <p>. За замовчуванням абзаци починаються з нового рядка і мають вертикальні відступи, якими можна управляти за допомогою стилів. Але використовувати їх можна тільки всередині тега <body>.

Для заголовків в мові HTML для виділення заголовків передбачено ціле сімейство тегів: від <h1> до <h6>. Тег <h1> позначає найважливіший заголовок (заголовок верхнього рівня), а тег <h6> позначає підзаголовок самого нижнього рівня.

Основні елементи для переносу та розділювання рядка: <br>, <hr>,

<br> створює розрив рядка в абзаці; це єдиний спосіб створити жорстку структуру в ситуації, коли вам потрібна серія фіксованих коротких рядків, наприклад, у поштовій адресі чи вірші. Без <br>елементів абзац просто відображатиметься в одному довгому рядку.

<hr>елементи створюють горизонтальну лінію в документі, яка позначає тематичну зміну в тексті (наприклад, зміну теми чи сцени). Візуально це просто виглядає як горизонтальна лінія.

Впорядкований список створюється за допомогою тега <ol> (ordered list), який може містити всередині себе теги <li>. Якщо елементи неупорядкованого списку за замовчуванням позначаються маркерами, то елементи впорядкованого списку - нумеруються.

Невпорядковані (або марковані) списки створюються за допомогою тега <ul>, який може містити всередині себе теги <li>, що позначають «елемент списку». Єдине, що дозволено в межах розмітки маркованого списку (тобто між відкриваючим і закриває тегами ul), - це один або кілька елементів списку. Ви не можете вставити туди інші елементи, і там не може бути нерозміченого тексту.

Посилання створюються за допомогою тегу <a>

Щоб перетворити виділений текст на посилання, помістіть його між відкриваючим і закриваючим тегами <а> та </а> і за допомогою атрибуту href вкажіть URL-адресу сторінки, на яку буде здійснюватися перехід. Вміст елемента якоря стає гіпертекстовим посиланням. Тег <a> без адреси не використовується, оскільки він описує посилання яке нікуди не веде. Також Посилання можна робити не лише за допомогою тексту, але і за допомогою зображень. Для цього потрібно обернути тег <img> в тег <a>.

Зображення додаються за допомогою тегу <img>

Щоб додати на сторінку зображення, потрібно використовувати одиночний тег <img> (image) з атрибутом src, в якому вказана адреса картинки. Щоб управляти шириною або висотою зображення, потрібно використовувати атрибути width і height. Найпоширенішими форматами зображень у мережі є JPEG і PNG.

## **5.3 Каскадні таблиці стилів CSS та їх структура**

Каскадні таблиці стилів — це набір параметрів форматування, який застосовується до елементів веб-сторінки для управління їх виглядом і положенням.

CSS можна додати до елементів HTML трьома способами:

* Inline (вбудований або рядковий) - використовуючи атрибут style в HTML елементах
* Internal (внутрішній) - використовуючи елемент <style> в разділі <head>
* External (зовнішній) - за допомогою зовнішнього файлу CSS

Властивість CSS backgraund є скороченням ряду фонових довгописних властивостей. А властивість background-color визначає колір фону для будь-якого елемента в CSS. Вона приймає будь-які дійсні <color>. A background-color розширюється під полем вмісту та заповнення елемента. Також можна додавати фонові зображення за допомогою властивість background-image яка дозволяє відображати зображення на фоні елемента.

Властивість background-repeat використовується для керування поведінкою мозаїки зображень. Доступні значення:

* no-repeat— повністю припинити повторення фону.
* repeat-x— повторити по горизонталі.
* repeat-y— повторити вертикально.
* repeat— за замовчуванням; повторити в обох напрямках.
* space— повторіть стільки разів, скільки можливо, додаючи пробіл між зображеннями, якщо є додатковий простір.
* round— схожий на space, але розтягує зображення, щоб заповнити додатковий простір

## **5.4 Спадковості і каскадування за допомогою CSS**

Ієрархічне дерево

HTML-документ являє собою ієрархічне дерево. Це означає, що у кожного елемента (крім кореневого) є тільки один батько, тобто елемент, усередині якого він розташовується. У кореневого розділу батько відсутній.

<html>

<head> </ head>

<body>

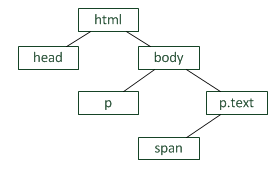
<p> Текст документа </ p>

         <p class = "text"> Виділена <span> рядок </ span> </ p>

</ body>

</ html>

Для цієї сторінки можна намалювати таке ієрархічне дерево:



Воно схематично відображає структуру вкладеності елементів. В даному прикладі видно, що в елемента span батьком є p.text, а у p.text батько - Body.

Ієрархічна структура документа визначає основи концепції наслідування.

Наслідування в CSS - механізм, за допомогою якого значення властивостей елемента-батька передаються його елементам-нащадкам. Стилі, присвоєні деякого елементу, успадковуються всіма нащадками (вкладеними елементами), якщо вони не перевизначені (перекриті) явно. Наприклад, розмір шрифту і його колір досить часто застосувати до body, щоб всі елементи всередині мали ті ж властивості.

Каскадність якраз і визначає, які саме властивості з цих джерел застосуються до даного абзацу. Є три основні концепції, керуючі порядком, в якому застосовуються CSS-властивості:

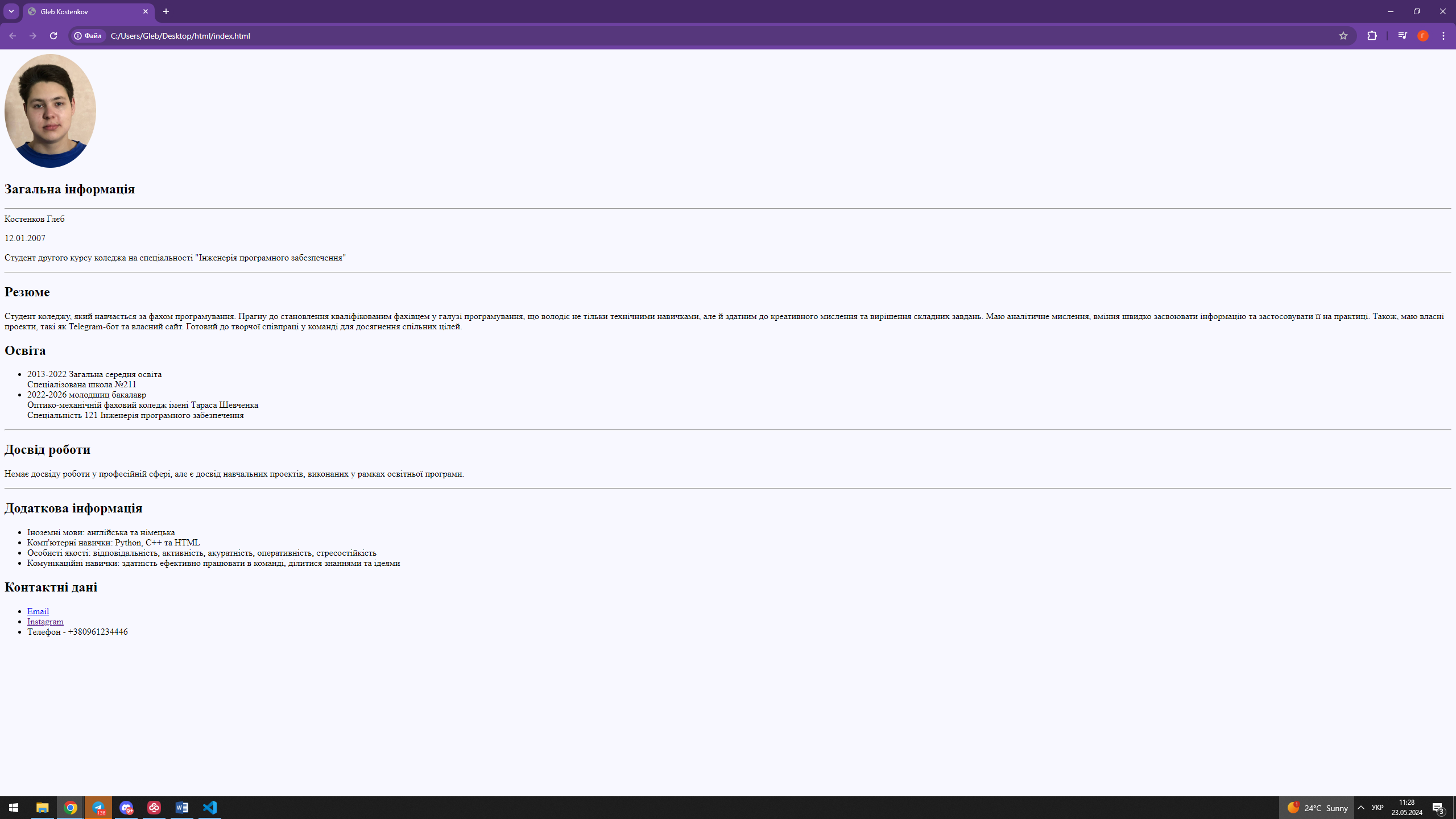
1. важливість;
2. специфічність;
3. порядок вихідного коду.

## **5.5 Практична частина**

### 5.5.1 Постановки задачі

Основною задачею було створити веб-сайт у вигляді резюме. Метою цього завдання було вивчення та розбір мови розмітки HTML та структури Веб-сторінки, знайомство з CSS та каскадуванням стилів, а також вивчення та застосування їх основних функцій.

### 5.5.2 HTML документ



**Розроблена веб-сторінка**<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Gleb Kostenkov</title>

<style>

body {

background-color: ghostwhite;

}

</style>

</head>

<body>

<img src="images/Gleb Kostenkov.jpg" alt="" height="200" style="border-radius: 50%;" />

<h2>Загальна інформація</h2>

<hr>

Костенков Глєб

<p>12.01.2007</p>

<p> Студент другого курсу коледжа на спеціальності "Інженерія програмного забезпечення"</p>

<hr>

<h2>Резюме</h2>

<p>Студент коледжу, який навчається за фахом програмування. Прагну до становлення кваліфікованим фахівцем у галузі програмування, що володіє не тільки технічними навичками, але й здатним до креативного мислення та вирішення складних завдань.

Маю аналітичне мислення, вміння швидко засвоювати інформацію та застосовувати її на практиці. Також, маю власні проекти, такі як Telegram-бот та власний сайт. Готовий до творчої співпраці у команді для досягнення спільних цілей.

</p>

<h2>Освіта</h2>

<ul>

<li style="">2013-2022 Загальна середня освіта <br> Спеціалізована школа №211 </li>

<li>2022-2026 молодшиц бакалавр <br> Оптико-механічній фаховий коледж імені Тараса Шевченка <br> Спеціальність 121 Інженерія програмного забезпечення </li>

</ul>

<hr>

<h2>Досвід роботи</h2>

<p>Немає досвіду роботи у професійній сфері, але є досвід навчальних проектів, виконаних у рамках освітньої програми.</p>

<hr>

<h2>Додаткова інформація</h2>

<ul>

<li> Іноземні мови: англійська та німецька</li>

<li> Комп'ютерні навички: Python, C++ та HTML </li>

<li> Особисті якості: відповідальність, активність, акуратність, оперативність, стресостійкість</li>

<li> Комунікаційні навички: здатність ефективно працювати в команді, ділитися знаннями та ідеями</li>

</ul>

<h2>Контактні дані</h2>

<ul>

<li> <a href="mailto:kostenkov3003@gmail.com" target="\_blank">Email</a></li>

<li><a href="https://www.instagram.com/untiledex/" target="\_blank">Instagram</a></li>

<li> Телефон - +380961234446 </li>

</ul>

</body>

</html>

# ВИСНОВКИ

***2 тема:***

Під час навчальної практики в розділі «Інформатика» було охоплено широкий спектр тем, починаючи від розроблення складних документів у MS Word і збереження їх у різних форматах, до вирішення практичних завдань з використанням MS Excel для аналізу даних і бази даних MS Access для проектування та управління інформацією.

Вивчення та застосування методів для розв’язання рівнянь і оптимізаційних задач дозволило здійснити пошук оптимальних рішень у різноманітних ситуаціях. Освоєння основ статистичного аналізу даних сприяло збору, обробці та аналізу інформації для виявлення закономірностей і трендів.

Проектування баз даних і робота з ними у MS Access дозволили засвоїти основні концепції реляційних баз даних і виконати різноманітні завдання, від введення та коригування даних до їх вибірки, пошуку і фільтрації.

Ця практична робота з інформатики дала важливі навички і знання, які можна успішно застосовувати в різних аспектах життя і професійній діяльності.

***3 тема:***

Під час навчальної практики з розділу «Програмування С++» я розширив свої знання в галузі алгоритмізації обчислювальних процесів, типів даних, операцій введення та виведення, а також програмування лінійних, розгалужених і циклічних обчислювальних процесів.

У рамках практичного завдання я створив програму, яка приймає значення коефіцієнтів (a), (b), (c), початкове значення (Beg), кінцеве значення (End) та крок (Step), і виконує обчислення функції в циклі, перевіряючи коректність введених даних. Програма демонструє використання різних типів даних, таких як float і int, операторів присвоювання, операцій введення-виведення, а також лінійних, розгалужених і циклічних обчислень.

На початку програма запитує введення коефіцієнтів і перевіряє їх відповідність умовам за допомогою циклу do-while, який забезпечує коректність значень початкового і кінцевого значень та кроку.

Потім програма обчислює значення функції у циклі, перевіряючи умови зміни значень змінної (x). Функція calculateFunction виконує обчислення за трьома різними формулами залежно від умов, заданих у задачі.

***4 тема:***

Під час виконання навчальної практики у розділі "Практична робота з С++" я набув важливі практичні навички, які будуть сприяти подальшому розвитку у сфері програмування.  
Основні аспекти які я вивчив під час цієї практики:  
  
1) Введення та виведення у консольних програмах:

Навчився працювати з введенням та виведенням даних через консоль за допомогою стандартних бібліотек C++. Це базова навичка , яка дозволяє створювати зрозумілі інтерфейси в консольних додатках.

2) Робота з файлами: запис та читання:

Оволодів навичками роботи з файлами, зокрема збереження та обробку даних у зовнішніх файлах. Це корисно для зберігання даних між сеансами програми.

3) Робота з форматами файлів:

Ознайомився з форматами JSON та XML, що дозволяє структуровано зберігати та обмінюватися даними між різними системами.

4) Системи контролю версій (VCS):

Вивчив принципи управління версіями коду та співпрацю в команді через систему контролю версій Git.

5) Markdown для документації:

Дізнався про мову Markdown, яка дозволяє створювати зрозумілу та структуровану документацію для проектів, полегшуючи комунікацію в команді та з користувачами.

6) Створення ігрової програми:

Створив гру "хрестики нолики" на мові C++, що дозволило застосувати отримані знання у практичній роботі.

***5 тема:***

Під час моєї навчальної практики у веб-дизайні та HTML я вивчив основні принципи створення сучасних та адаптивних веб-сторінок, відповідних стандартам індустрії. Я отримав глибокі знання з HTML, CSS та веб-дизайну, які дозволяють мені ефективно створювати інтерфейси, залучаючі та зручні для користувачів. Практичні завдання допомогли закріпити теоретичні знання та забезпечили мене необхідними навичками для успішної реалізації веб-проектів. Цей досвід значно підвищив мої професійні здібності, і я впевнено планую продовжувати вдосконалювати свої навички у сфері веб-дизайну та розробки.

# ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ ІНФОРМАЦІЇ

ГОСТ 2.105-95 ЕСКД Общие требования к текстовым документам.

ДСТУ 1.5-93 Загальні вимоги до побудови, викладу, оформлення та змісту.

1. <https://cpto.dp.ua/public_html/posibnyky/basic_html/urok1.html>
2. <https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web>
3. <https://e-tk.lntu.edu.ua/mod/page/view.php?id=2894>
4. <https://e-helper.com.ua/node/949>
5. <https://acode.com.ua/uroki-po-cpp/>
6. <https://cherto4ka.xyz/2021/11/03/%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B8-%D0%BA%D0%B5%D1%80%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F-%D0%B2%D0%B5%D1%80%D1%81%D1%96%D1%8F%D0%BC%D0%B8/>
7. <https://highload.today/uk/mova-rozmitki-markdown-povnij-gajd-iz-sintaksisu-ta-formatuvannya-tekstu/>
8. <https://web.posibnyky.vntu.edu.ua/fitki/14baryshev_tehnologiya_programuvannya/08.html>
9. <https://support.microsoft.com/en-us/word>
10. <https://www.microsoft.com/en-us/learning/microsoft-word.aspx>
11. <https://www.wikihow.com/Use-Microsoft-Word>
12. <https://support.microsoft.com/en-us/excel>
13. <https://www.excel-easy.com/>
14. <https://support.microsoft.com/en-us/access>
15. <https://www.youtube.com/watch?v=WYnh4YXbiuo>
16. <https://www.linkedin.com/learning/topics/microsoft-access-2019>