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: |
| студента 2-го курсу групи ІПЗ-23 |
| Щур Д.В.  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 | |
|  | (навчальної, технологічної, виробничої) | |  | |
| *Щур Діани Володимирівни* | | | | |
| (П.І.Б студента.) | | | | |
| 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-22

ОМФК.0121 НП 20

Зміст

[ВСТУП 5](#_Toc169534291)

[1 БЕЗПЕКА ЖИТТЄДІЯЛЬНОСТІ ПРИ РОБОТІ З КОМП’ЮТЕРОМ 7](#_Toc169534292)

[2 РОЗДІЛ "ІНФОРМАТИКА" 9](#_Toc169534293)

[**2.1** **Розроблення засобами MS Word документа складної структури** 9](#_Toc169534294)

[**2.2 Розв‘язок задач за допомогою засобів табличного редактора MS Excel** 18](#_Toc169534295)

[**2.3 Виконання** **практичних завдань за допомогою засобів табличного редактора MS Access** 22](#_Toc169534296)

[3 РОЗДІЛ "ПРОГРАМУВАННЯ С++" 24](#_Toc169534297)

[**3.1 Теоретичні відомості** 24](#_Toc169534298)

[3.1.1 Загальні поняття. Елементи мови С++ - константи, змінні, операції, перетворення типів 24](#_Toc169534299)

[3.1.2 Арифметичні операції 25](#_Toc169534300)

[3.1.3 Поняття вхідного та вихідного потоку, найпростіші математичні функції 25](#_Toc169534301)

[3.1.4 Програмування лінійних та розгалужених обчислювальних процесів 26](#_Toc169534302)

[3.1.5 Програмування циклічних обчислювальних процесів 27](#_Toc169534303)

[**3.2 Практичне завдання** 28](#_Toc169534304)

[3.2.1 Завдання 28](#_Toc169534305)

[3.2.2 Постановка задачі 28](#_Toc169534306)

[3.2.3 Код програми 29](#_Toc169534307)

[3.2.4 Скріншот виконання програми 31](#_Toc169534308)

[**3.3 Виконання тестів** 32](#_Toc169534309)

[4 РОЗДІЛ " ПРАКТИЧНА РОБОТА З С++" 32](#_Toc169534310)

[**4.1 Тема 1. Введення та виведення у базових консольних програмах** 32](#_Toc169534311)

[**4.3 Тема 3. Робота з файлами: запис та читання** 36](#_Toc169534312)

[**4.4 Тема 4. Формати файлів та їх обробка** 37](#_Toc169534313)

[**4..5 Тема 5. Системи контролю версій (VCS)** 38](#_Toc169534314)

[**4.6 Тема 6. Мова Markdown для опису проекті** 39](#_Toc169534315)

[Зображення 41](#_Toc169534316)

[**4.7 Розроблення ігрової програми** 43](#_Toc169534317)

[4.7.1 Код програми 43](#_Toc169534318)

[4.7.2 Скріншоти виконання програми 47](#_Toc169534319)

[5 РОЗДІЛ ”ВЕБ-ДИЗАЙН ТА HTML” 49](#_Toc169534320)

[**5.1 Мова розмітки HTML та структура Веб-сторінки** 49](#_Toc169534321)

[**5.2 Розмітка тексту за допомогою HTML. Зображення та посилання** 51](#_Toc169534322)

[**5.3 Каскадні таблиці стилів CSS та їх структура** 55](#_Toc169534323)

[**5.4 Спадковості і каскадування за допомогою CSS** 60](#_Toc169534324)

[5.5 Практична частина 61](#_Toc169534325)

[5.5.1 Постановки задачі 61](#_Toc169534326)

[5.5.2 HTML документ 62](#_Toc169534327)

[ВИСНОВКИ 69](#_Toc169534328)

[ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ ІНФОРМАЦІЇ 70](#_Toc169534329)

ОМФК.0121 НП 20

# ВСТУП

Автор: Щур Діана Володимирівна

Дата: 17.05.2024

Мета:

Перелік вивчених тем:

1. Засобами MS Word розробити документ складної структури та зберегти у різних форматах
2. Розв’язування рівнянь
3. Розв’язування оптимізаційних задач
4. Основи статистичного аналізу даних. Ряди даних. Кореляційний аналіз даних
5. Обчислення основних статистичних характеристик вибірки
6. Візуалізація рядів і трендів даних
7. Розв‘язок задач за допомогою засобів табличного редактора MS Excel
8. Проектування бази даних та створення бази даних MS ACCESS
9. Основні поняття реляційних баз даних
10. Введення, коригування, вилучення інформації з бази даних MS ACCESS
11. Вибірка, пошук, фільтрація даних в MS ACCESS
12. Виконання практичних завдань за допомогою засобів табличного редактора MS Access
13. Розділ "Програмування С++"
14. Алгоритмізація обчислювальних процесів. Розробка постановки задачі, визначення вхідних та вихідних даних, оформлення блок-схем.
15. Типи даних в мові С/С++, внутрішнє представлення даних, операції перетворення типів даних.
16. Оператор присвоювання. Операції введення- виведення даних
17. Програмування лінійних та розгалужених обчислювальних процесів
18. Програмування циклічних обчислювальних процесів
19. Введення та виведення у базових консольних програмах
20. Структури даних для управління станом програми
21. Робота з файлами: запис та читання
22. Формати файлів та їх обробка
23. Системи контролю версій (VCS)
24. Мова Markdown для опису проекті
25. Розроблення ігрової програми
26. Мова розмітки HTML та структура Веб-сторінки
27. Розмітка тексту за допомогою HTML. Зображення та посилання
28. Каскадні таблиці стилів CSS та їх структура
29. Спадковості і каскадування за допомогою CSS
30. Систематизація матеріалів, оформлення звіту з навчальної практики

# 1 БЕЗПЕКА ЖИТТЄДІЯЛЬНОСТІ ПРИ РОБОТІ З КОМП’ЮТЕРОМ

Кабінет інформатики та інформаційно-комунікаційних технологій на­сичений різноманітною електронною технікою, яка живиться від елект­ричної мережі з напругою змінного струму 220 В. Це потребує дотриман­ня додаткових правил безпеки під час проведення навчальних занять і позаурочних заходів.

**До початку роботи:**

* з дозволу вчителя займіть своє робоче місце за комп’ютером, налаш­туйте висоту стола, стільця, підставки для ніг, кут нахилу монітора так, щоб (рис. 2.57):
* середина екрана монітора знаходилася трохи нижче горизонтальної лінії зору;
* відстань від очей до поверхні екрана становила 40-80 см, залежно від розмірів об’єктів на екрані;
* лінія від очей до центра екрана монітора була перпендикулярна до площини екрана;
* пальці рук вільно лежали на клавіатурі;
* руки утворювали в ліктьовому суглобі кут, близький до 90;
* клавіатура лежала на поверхні стола або на спеціальній полиці на відстані 10-30 см від краю і була нахилена під кутом 5-15;
* спина опиралася на спинку стільця;
* ноги опиралися на підлогу або на спеціальну підставку;
* наведіть лад на робочому столі, приберіть з нього предмети, які не потрібні для роботи;
* перевірте чистоту своїх рук, за потреби вимийте їх і витріть насухо;
* за потреби, з дозволу вчителя і тільки спеціальною серветкою протріть екран монітора, клавіатуру, килимок і мишу;
* перевірте відсутність зовнішніх пошкоджень комп’ютерного облад­нання;
* з дозволу вчителя ввімкніть ком­п’ютер.

**Під час роботи:**

* тримайте робоче місце охайним, не розміщуйте на ньому сторонніх ре­чей;
* витримуйте правильну поставу - не нахиляйтеся близько до поверхні ек­рана, не згинайтеся, тримайте руки без напруження;
* після 15-20 хв роботи або при від­чутті втоми виконайте комплекс вправ для очей і для зняття м’язово­го напруження;
* не намагайтеся самостійно усунути перебої в роботі комп’ютера, при їх виникненні негайно покличте учителя;
* не торкайтеся задніх стінок монітора та системного блока, не чіпай­те дротів живлення;
* не торкайтеся екрана монітора руками (це забруднює його і порушує антиблікове покриття);
* категорично забороняється знімати кришки корпусів пристроїв комп’ютера, самостійно без дозволу вчителя приєднувати і від’єдну­вати пристрої комп’ютера;
* акуратно вставляйте і виймайте змінні носії.

**Після закінчення роботи:**

* приберіть своє робоче місце;
* з дозволу вчителя вимкніть комп’ютер або закінчіть сеанс роботи.

# 2 РОЗДІЛ "ІНФОРМАТИКА"

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

Основні складові частини інтерфейсу програми: рядок заголовка, стрічка з командами, робоча область документа, рядок стану. Рядок заголовка вікна містить: • панель швидкого доступу з часто використовуваними командами. Панель може настроюватися: можна міняти її розташування (над або під стрічкою команд), додавати, видаляти, переміщувати кнопки. Налаштування панелі швидкого доступу виконується за допомогою списку команд, що розкривається при натисненні на кнопку; • ім’я файлу поточного документа; • кнопки відновити, згорнути, закрити вікно програми. Стрічка команд складається з вкладок. На кожній вкладці знаходяться елементи управління: кнопки, списки, прапорці тощо, які об’єднані в групи. Назву групи наведено знизу, наприклад Буфер обміну, Шрифт, Абзац. У групі не завжди можна побачити всі команди. У правому нижньому кутку деяких груп команд знаходиться кнопка групи – значок зі стрілкою. Клацання по значку відкриває відповідне цій групі діалогове вікно. На відміну від інших вкладка Файл відображає меню команд із збереження, відкриття, створення, друку документа, його захисту, налаштування параметрів програми Word тощо. Крім постійних існують контекстні вкладки, які з’являються автоматично при виділенні деяких об’єктів в документі (наприклад, таблиці, малюнка, діаграми). Ці вкладки містять набори команд для налаштування відповідного об’єкта.

*Форматування документа* – це комплекс операцій, за допомогою яких

налаштовується потрібний вигляд документу без зміни інформації, яка в ньому

знаходиться. У Microsoft Word форматування і розмітка документа

розділяється на три рівні:

* + Форматування сторінок і розділів.
  + Форматування символів.
  + Форматування абзаців

При виконані деяких операцій форматування і розмітки документу в нього заносяться службові символи, які задають структуру тексту, наприклад, символи, що вставляються при натисканні клавіш пробілу, Enter, Tab. Такі символи називаються прихованими символами форматування або недрукованими знаками. У таблиці наведені зображення прихованих символів форматування, їх опис та клавіші для вставки цих символів в документ.

За умовчанням вони не відображаються в документі на екрані й не виводяться при друку на принтері, але завжди зберігаються в файлі документа з усім текстом. Для відображення або приховування символів форматування використовується команда Основне → Абзац → Відобразити всі знаки (¶). Відображення недрукованих знаків допомагає зрозуміти структуру текстового документа.

До основних параметрів форматування сторінок і розділів відносяться установка розміру сторінки, ширини полів, книжкової або альбомної орієнтації, додавання колонтитулів і номерів сторінок, розбиття тексту на колонки. Розділ – це найбільша структурна частина документа, у межах якої основні налаштування редактора (наприклад, розмір і орієнтація сторінки) є сталими. Розділ завершується невидимим символом кінця розділу. Довільний документ містить щонайменше один розділ. Розділи представляють собою частини документа, які можна форматувати незалежно один від одного. Для різних розділів можуть використовуватися різні параметри сторінок. Параметри сторінки можна встановлювати як для всього документа, так і для окремих його розділів. Тільки за допомогою розділів можливо створити документ складної структури (наприклад змішана орієнтація сторінок в документі, різна кількість стовбців тощо).

Слід зазначити, що всі параметри форматування сторінки можуть встановлюватися для поточного розділу, від поточного розділу до кінця документа і до всього документа.

*Колонтитул* – це інформація, яка повторюється на кожній сторінці документа. Під колонтитули відводиться спеціальне місце вгорі (верхній колонтитул) або внизу (нижній колонтитул) сторінки. В колонтитулі, як правило, містяться заголовок публікації, назви глав або розділів, ім’я автора, номера сторінок, дата створення. Робота з колонтитулом ведеться в режимі Розмітка сторінки. При редагуванні колонтитулів доступні всі основні параметри форматування тексту. Можна додавати в документ колонтитули з колекції стандартних зразків або створювати власні.

*Шрифт* – це графічний образ символів шрифту. Кожна гарнітура має свою назву, комплект шрифтів різних накреслень, насиченості, кегля і щільності, об’єднаних спільністю малюнка символів. MS Word надає для роботи кілька десятків гарнітур. Найбільш широкого поширення набули Times New Roman, Arial, Courier New. За замовчуванням використовується гарнітура Calibri. Стиль шрифту – це графічний різновид шрифту в межах однієї гарнітури. У Word використовується три накреслення: звичайне (пряме), курсивне і напівжирне. При цьому можна використовувати поєднання напівжирний курсив. Розмір (кегль) – це висота букви або символу. Кегль вимірюється в пунктах: 1 пт = 1/72 дюйма = 0,353 мм. 1 дюйм = 2,54 см. Мінімально можливий розмір шрифту дорівнює 1 пт, максимальний – 1638 пунктів. У редакторі Word розмір шрифту встановлюється з точністю до 0,5 пунктів. Зазвичай для оформлення основної частини тексту використовують шрифти розміром від 10 до 14 пунктів. *Підкреслення шрифту* – виділення тексту горизонтальними лініями (прямі, пунктирні, хвилясті і ін.). Зміна положення символів в верхній (x2 – надрядковий) або нижній (x2 – підрядковий) індекс. Зміна регістра – це зміна букв на рядкові та прописні. Інтервал між символами (щільність) – відстань між символами в тексті. За замовчуванням використовується звичайний інтервал. Його можна замінити на розряджений або ущільнений. Можлива установка зсуву символів 19 вгору або вниз відносно горизонтальної лінії рядка. Кернинг – зміна інтервалу між окремими парами символів

*Абзац* – це частина тексту, пов’язана смисловою єдністю і виділена виступом першого рядка. Під абзацом у Word розуміють будь-яку кількість тексту і графіки (можуть існувати порожні абзаци), яка закінчується символом кінця абзацу ¶. Абзац на сторінці створюється кожним натисканням клавіші Enter. До форматування абзацу відносяться такі параметри:

* + - вирівнювання абзацу щодо лівого і правого полів сторінки (по лівому краю, по центру, по правому краю, по ширині);
    - відступ або виступ першого рядка абзацу;
    - відступи абзацу від поля сторінки зліва і справа;
    - інтервал між рядками;
    - інтервал перед та після абзацу;
    - положення абзацу при переході на наступну сторінку;
    - положення табуляції;
    - нумеровані і маркіровані списки.

*Перевірка правопису*. Під час введення тексту Word за замовчуванням

автоматично перевіряє документ на наявність орфографічних і граматичних помилок. Налаштувати параметри перевірки правопису можна у вікні, що викликається командою Файл → Параметри → Правопис. При перевірці червоною хвилястою лінією підкреслюються слова, в яких є орфографічні помилки, або повторювані слова. Зеленою хвилястою лінією підкреслюються слова, розділові знаки, фрагменти речень і речення, в яких є граматичні та стилістичні помилки. Для виправлення орфографічної помилки необхідно відкрити контекстне меню підкресленого слова і вибрати правильний варіант написання. Варіанти написання слів пропонуються, якщо в словнику

редактора Word є схожі за написанням слова. Інакше в контекстному меню

буде вказано Немає варіантів, тоді необхідно внести виправлення самостійно.

*Скасування і повернення дій*. При роботі в редакторі Word операції,

що виконуються в документі, запам’ятовуються, що дає можливість скасувати

щойно зроблені виправлення, відновити скасовані дії або повторити будь-яку

операцію. Для скасування останньої виконаної дії використовується кнопка

Скасувати на панелі швидкого доступу або комбінація клавіш Ctrl + Z.

Можна скасувати відразу декілька дій. Для цього потрібно клацнути по стрілці кнопки Скасувати і вибрати дію для скасування, при цьому скасовуються всі операції, розташовані в списку вище обраної. Скасовані дії можна повернути. Для повернення останнього скасованої дії використовується кнопка Повернути на Панелі швидкого доступу або комбінація клавіш Ctrl + Y. Для повернення декількох дій потрібно кілька разів натиснути кнопку Повернути. Якщо все скасовані дії були повернуті, або дії не скасовувалися, кнопка Повернути замінюється на кнопку Повторити, щоб повторити останню операцію.

*Створення таблиці.* Створити таблицю в документі можна різними

способами. Перший:

* 1. Встановити курсор в місце вставки таблиці в документі.
  2. Виконати команду Вставка → Таблиці → Таблиця
  3. У макеті виділити потрібну кількість стовбців і рядків і натиснути ліву кнопку миші.

Другий спосіб:

1. Встановити курсор в місце вставки таблиці в документі.
2. Виконати команду Вставлення → Таблиця → Вставити таблицю.
3. У діалоговому вікні Вставка таблиці встановити потрібні параметри (число рядків, стовпців, спосіб підбору ширини стовпців) і натиснути кнопку OK.

*Стилем* називається іменований набір параметрів форматування, який

застосовується до окремих символів, абзаців, таблиць і списків. Стилі

дозволяють однією дією застосувати відразу всю групу параметрів

форматування до елементу, забезпечують однаковість в оформленні

документів, можливість швидкої зміни оформлення, дозволяють автоматично створити зміст документа.

Редактор MS Word пропонує широкий набір інструментів по додаванню

в документ і налаштуванню різних видів ілюстрацій: малюнків з графічних файлів, малюнків з колекції кліпів Microsoft Office, фігур і створених з їх допомогою малюнків, об’єктів SmartArt, художньо оформленого тексту WordArt тощо. Всі графічні об’єкти можна переміщувати по сторінці за допомогою клавіш зі стрілками на клавіатурі або за допомогою миші. Для зручності вирівнювання об’єктів на сторінці можна відобразити сітку за допомогою прапорця Подання → Відображення → Лінії сітки.

При управлінні розташуванням графічних об’єктів потрібно знати, що документ Word має чотири шари:

1. Передній графічний шар.
2. Шар тексту.
3. Задній графічний шар.
4. Шар колонтитула і підкладки.

Кожен верхній шар затуляє своїм вмістом попередній. На самому нижньому шарі розташовуються колонтитули і підкладка документа. За замовчуванням малюнок з графічного файлу, кліп з колекції MicrosoftOffice, нове полотно, малюнок SmartArt поміщаються на шар тексту в текст. Окремі фігури (лінії, прямокутники та інші команди групи Вставлення→ Ілюстрації → Фігури) і декоративний текст WordArt за замовчуванням поміщаються на передній шар, затуляючи собою текст. Змінити шар розміщення графічного об’єкта можна за допомогою команд контекстного меню цього об’єкта:

• На передній план → Помістити перед текстом або Обтікання текстом → Перед текстом для розміщення об’єкта на передній шар;

• На задній план → Перемістити за текстом або Обтікання текстом → За текстом для розміщення об’єкта на задній шар;

• Обтікання текстом → У тексті для розміщення об’єкта в шар тексту.

Ці ж команди є на контекстній вкладці Формат відповідного об’єкта.

Усередині кожного з графічних шарів об’єкти розташовуються один над одним в порядку їх додавання. Управляти їх розташуванням відносно один одного можна за допомогою команд контекстного меню На передній план → Перемістити вперед і На задній план → Перемістити назад або за допомогою цих же команд контекстної вкладки Формат фігури → Упорядкувати. Об’єкти всередині одного шару можна групувати, об’єднуючи кілька об’єктів в один (команда Групувати контекстного меню або вкладка Формат фігури → Упорядкувати →Групувати об’єкти), вирівнювати щодо сторінки по горизонталі і вертикалі (Формат фігури → Упорядкувати →Вирівняти) і повертати на заданий кут (Формат фігури → Упорядкувати →Повернути об’єкти). Усередині шару тексту можна задати варіант обтікання об’єкта текстом (рис. 4.5): в тексті, навколо рамки, по контуру, наскрізне, зверху і знизу (команда Обтікання текстом контекстного меню об’єкта).

При редагуванні документа кожен малюнок переміщується разом з тим абзацом, до якого він прив’язаний. Всі об’єкти, що знаходяться не в тексті, а обтічні ним, або розташовані на передньому чи задньому шарі, називаються «плаваючими». Коли виділено «плаваючий» графічний елемент, зліва від абзацу, до якого він прив’язаний, відображається якір (при відображенні прихованих символів форматування). Це значок прив’язки об’єкта до абзацу,

його можна пересунути мишею в інший абзац. Редактор Word відстежує, щоб

малюнок завжди знаходився на одній сторінці з абзацом, до якого він прив’язаний.

Вставка малюнка з графічного файлу. MS Word дозволяє вставляти в документ малюнки з файлів всіх основних графічних форматів: jpeg, bmp, gif,

tiff тощо. Для вставки малюнка потрібно:

1. Встановити курсор в місце вставки.
2. Виконати команду Вставлення→ Ілюстрації → Зображення.
3. У вікні Вставлення зображення з такого джерела малюнка перейти в папку з потрібним графічним файлом.
4. Двічі клацнути по значку файлу малюнка або виділити файл і натиснути кнопку Вставити.

Word дозволяє створювати малюнки з колекції автофігур: ліній, кривих, прямокутників, фігурних стрілок і інших. Для вставки окремої фігури в документ необхідно:

1. Виконати команду Вставлення→ Ілюстрації → Фігури.
2. У меню Фігури вибрати клацанням миші з колекції відповідно розтягнути її контур в місці вставки в документі.

Для додавання в середину фігури тексту слід вибрати в її контекстному меню команду Додати текст. Введений текст редагується і форматується за допомогою стандартних інструментів вкладки Основне. Задати або змінити параметри фігури можна за допомогою контекстної вкладки Формат фігури або команд контекстного меню.

Якщо передбачається робота відразу з кількома фігурами, наприклад при створенні блок-схеми, їх можна додати в спеціальну область малювання під назвою полотно. Для вставки полотна в документ використовується команда Вставлення→ Ілюстрації → Фігури → Створити полотно. Під час роботи з полотном навколо нього відображається межа у вигляді рамки.

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

Табличний процесор Excel – потужний програмний засіб дляефективного опрацювання даних, представлених у табличній формі: проведення різноманітних розрахунків, побудови графічнихьзалежностей, створення баз даних, моделювання, прогнозування та оптимізації процесів, знаходження коренів рівнянь тощо.

Структура вікна табличного процесора в багато чому схожа зі структурою вікон інших додатків ОС Windows. У вікні Excel відображаються рядок заголовка, рядок меню, за замовчуванням дві панелі інструментів Стандартная і Форматирование, рядок формул, адресний рядок, робоче поле, рядок стану, смуги прокручування. При завантаженні Excel автоматично створюється новий документ під назвою Книга 1, який може мати до 255 аркушів. Кожний аркуш робочої книги представлений у вигляді таблиці, що складається із 256 (28) стовпчиків і 65536 (216) рядків. Заголовки стовпчиків, зазвичай, позначені літерами латинського алфавіту, а заголовки рядків – числами. На перетині кожного стовпчика і рядка розташований прямокутник, який називається коміркою (клітинкою, чарункою). Активною вважається комірка, по якій клацнули лівою клавішею миші, вона виокремлюється жирною чорною рамкою. Кожна комірка має свою адресу (назву), яка визначається заголовком відповідного стовпчика та номером рядка (наприклад, комірка, що утворена перетином стовпчика F і рядка 2, має адресу F2). В Excel, окрім адрес окремих комірок, можуть задаватися адреси діапазонів комірок, що включають частину стовпчика (D2:D8), частину рядка (С3:С12) або блок комірок (В3:Е6). При цьому символом “:” відокремлюють адресу першої комірки (верхньої лівої) із діапазону і останньої (нижньої правої).

Запуск Microsoft Excel

* завантажують операційну систему Windows;
* виконують дії: Пуск Программы Microsoft Excel;
* або: клацають мишею по зображенню значка, або ярлика додатка, якщо вони розташовані на робочому столі;
* або: Пуск Создать документ Office Новая книга.

*Створення таблиці*

Робоче поле аркуша Excel має вигляд таблиці, що складається із однакових за шириною стовпчиків і однакових по висоті рядків, на перетині яких знаходиться комірка. Користувач, як правило, створює структуру таблиці залежно від особистих потреб: надає певного вигляду шапці таблиці шляхом об’єднання комірок, змінює ширину стовпчиків та висоту рядків.

*Об’єднання комірок*

Об’єднання комірок застосовують для формування структури таблиці, а не для збільшення ширини чи висоти комірок. Процедура об’єднання комірок передбачає створення із певної кількості комірок – однієї. З цією метою:

* виокремлюють комірки, які планується об’єднати (протягують курсор вздовж комірок при натиснутій лівій клавіші миші);
* клацають по кнопці Объединить и поместить в центре ( ) на панелі форматування;
* або виконують послідовність дій: Формат Ячейки Выравнивание Объединение ячеек;
* у разі помилкового об’єднання комірок виконують аналогічну послідовність дій і прибирають (вимикають) опцію Объединение ячеек.

Зміна висоти рядків та ширини стовпчиків. Висота рядків та ширина стовпчиків залежить від вмісту комірок, що утворені їх перетином, вимог до зовнішнього вигляду таблиці та особистих уподобань користувача. Найчастіше розміри комірки змінюють за допомогою миші. З цією метою:

* встановлюють курсор між назвами стовпчиків – для зміни ширини стовпчика, або між номерами рядків – для зміни висоти рядка, щоб він набув вигляду ◄┼►;
* при натиснутій лівій клавіші миші переміщують межі рядків та стовпчиків, встановлюючи потрібні висоту та ширину.

Зміна ширини та висоти групи комірок

Для надання групі комірок однакового розміру, відмінного від встановленого за замовчуванням, попередній спосіб мало ефективний і недостатньо точний. З цією метою:

* виокремлюють групу стовпчиків, ширину яких планується змінити на однакову величину;
* виконують дії: Формат Столбец Ширина; у діалоговому вікні, що відкриється, з клавіатури вводять значення ширини (одна одиниця відповідає ширині одного символу, розміру); натискують кнопку ОК;
* виокремлюють групу рядків, висоту яких планується змінити на однакову величину;
* виконують дії: Формат Строка Высота;
* у діалоговому вікні у відповідне поле з клавіатури вводять значення висоти рядка (число 12,75 відповідає висоті символу 10 розміру, 25 – відповідно 20 розміру); натискують кнопку ОК.

Вставка комірки (рядка, стовпчика)

* виокремлюють комірку, поруч з якою треба вставити порожню комірку (рядок, стовпчик);
* виконують дії: ВставкаЯчейки (Строки, Столбцы);
* при вставці порожньої комірки, в діалоговому вікні, що відкриється при виконанні зазначених дій, відмічають напрямок переміщення заповнених комірок; стовпчик автоматично вставляється зліва від активної комірки, рядок – зверху.

***Числа*** Будь-яке число вводиться в активізовану комірку та автоматично вирівнюється по правому нижньому краю. За потреби, місце розташування чисел у комірці змінюють. Числові дані можуть бути введені в різних формах та форматах: у вигляді цілого числа (124); десяткового дробу (14,426) із точністю до 30 знаків після коми; звичайного дробу (1 1 /5 ); грошового формату (1,20 грн., 5$). Якщо у числі більше, ніж 12 цифр, Excel автоматично переводить його у формат з плаваючою комою, тобто введене у комірку число 0,000000000137 після зміни активної комірки набуде вигляду 1,37Е - 10. При записі десяткового дробу ціла частина від десяткової відокремлюється комою! Для запису числа у певному форматі виконують послідовність дій: Формат⇒Ячейки⇒Число і вибирають потрібний формат. 34 Дата, час Однією з форм числової інформації в комірці Excel може бути дата або час. Дати найчастіше вводяться у формі 14.3, 14.03.05, 14 бер 05 тощо. Дні, місяці і роки при введенні дати відокремлюють крапкою, що автоматично переводить число у формат дати (часто це є причиною помилки при записі десяткового дробу). Аналогічно записується час. Вигляд запису дати і часу можна змінювати в широких межах, виконуючи дії: Формат⇒Ячейки⇒Число ⇒Дата. Автозаповнення комірок Excel у своєму активі має досить зручний інструмент автозаповнення, який значно прискорює введення числових та текстових даних, що змінюються в межах певного інтервалу або за певним законом. Прикладом таких даних можуть бути порядкові номери (1,2,3 ...), дати (1.02.05, 2.02.05 ...), дні тижня (понеділок, вівторок, ...), місяці року (січень, лютий ...) та інші списки. Заповнення суміжних комірок числовими даними з використанням функції автозаповнення можна здійснити різними способами.

Незалежно від типу інформації, введеної в комірки таблиці, до неї можуть бути застосовані засоби форматування в широких межах. З метою скорочення часу на проведення однотипних операцій з форматування (встановлення типу та розміру шрифту, формату чисел, вирівнювання в комірці тощо) доцільно подібні операції проводити з групою комірок.

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

Бази даних (БД) – це сукупність відомостей про реальні об'єкти,

процеси, події або явища, що відносяться до певної предметної

області, організована таким чином, щоб забезпечити зручне

представлення цієї сукупності як в цілому, так і будь-якої її частини.

Система управління базами даних – це комплекс програмних і

мовних засобів, необхідних для структуризації інформації, розміщення

її в таблицях, обробки баз даних і підтримки їх в актуальному стані.

Microsoft Access 2010 – одна з останніх розробок продукту

корпорації Microsoft, що являє собою на даний момент найбільш

розповсюджену у світі систему управління реляційними базами даних.

Microsoft Access є настільною СУБД реляційного типу, яка має всі

необхідні засоби для виконання перерахованих вище функцій.

Перевагою Microsoft Access є те, що вона має дуже простий графічний

інтерфейс, який дозволяє не тільки створювати власну базу даних, але і

розробляти прості і складні додатки.

У файлі бази даних Microsoft Access використовують такі типи

об’єктів:

таблиці для зберігання даних;

* запити для пошуку та виявлення необхідних даних;
* форми для перегляда, додавання і зміни даних в таблицях;
* звіти для аналізу і друку даних у певному форматі.

Microsoft Access 2010 зберігає всі об’єкти бази даних в одному

файлі (з розширенням .accdb), хоча і розподіляє їх по різних таблицях.

Одним з

основних нових властивостей є вкладка Файл і вікно представлення

Backstage, що пов’язано з нею. За допомогою панелі вікна

представлення Backstage можна відкривати нові файли, отримувати

доступ до файлів, що недавно використовувались, зберігати і виводити

на друк документи, закривати додатки, що виконуються та інше.

Для зміни налаштування стрічки достатньо перейти до вікна

представлення Backstage, відкрити вікно Параметры Access і обрати

вкладку Настройка ленты. У вікні, що з’явиться, можна створювати

нові вкладки і налаштовувати вже існуючі шляхом додавання і

перестановки кнопок у відповідних групах стрічки.

Завдяки темам Office візуальне оформлення бази даних в цілому та

її окремих компонентів значно спростилося. За допомогою різних тем

можна змінювати кольори і шрифти, а також включати різні графічні

елементи.

Попередньо розроблені шаблони баз даних, що доступні у вікні

представлення Backstage, дозволяють створювати бази даних, що

мають потрібні властивості.Завдяки навігаційним формам спрощується реорганізація та

відображення різних компонентів бази даних. Для відображення

окремих фрагментів бази даних достатньо скористатися стандартним

методом перетягування.

В Access 2010 використовуються події таблиць (тригери), що

дозволяють генерувати нові дані на основі інформації, що знаходиться

у таблицях.

Як бачимо, Access 2010 оснащено великою кількістю нових

можливостей і потужних інструментів, які використовуються для

створення таблиць, генерування звітів і форм, а також встановлення

зв’язків між таблицями.

# 3 РОЗДІЛ "ПРОГРАМУВАННЯ С++"

## **3.1 Теоретичні відомості**

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

Операції у виразі позначаються знаками (операторами), а значення – константами та іменами змінних.

**Алфавіт мови** – набір символів, які можна використовувати для записування програмного коду, це великі та малі літери латиниці, цифри, знаки операцій та спеціальні символи. Кожен зокрема чи в комбінації з іншими знаки

операцій і спеціальні символи дозволяють задавати лексеми, вирази і оператори, серед яких найбільш використовувані є такі:

**= –** присвоювання. Наприклад, якщо **x** і **y** – змінні, то запис **x** = **y** означає, що змінна х набуває значення **y**

**+, –, \*, /** – знаки операцій додавання, віднімання, множення та ділення;

**++, ––** – знаки операцій збільшення та зменшення на одиницю значення

операнди;

**;** – символ завершення команди;

**<** (менше), **<=** (менше чи дорівнює), **>, >=, ==, !=** – знаки операцій порівняння;

**+=, –=, \*=, /=, %=, <<=, >>=, ?=, ^=, |=** – знаки для записування операцій

присвоювання;

**%** – залишок від цілочисельного ділення двох чисел. Наприклад у виразі

**с = а % b** - значення залишку від ділення а на b присвоюється с;

&&, ||, ! – знаки логічних операцій;

**&, |, ^, <<, >>, ~** – знаки побітових логічних операцій;

**?:** – знаки умовної операції;

**\*, &** – знаки адресних операцій. Хоча такі символи вже зустрічались у

цьому переліку, однак їхнє специфічне застосовування у відповідних виразах

дозволяє виключити усіляку неоднозначність

**/\*, \*/, //** – сукупність знаків, використовуваних для записування коментарів;

**,** – знак, використовуваний для перелічування об’єктів;

**. , –>** – знаки використовувані при роботі зі структурованими типами і

класами;

**{ }** – операторні дужки, які поєднують блок команд;

**[ ]** – дужки для запису індексів масивів;

**( )** – дужки для встановлення черговості виконування операцій і записування аргументів функції;

**' '** – знаки для записування значення символьних констант;

**"** – знак для записування значення символьної інформації .

### 3.1.2 Арифметичні операції

****

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

При підключенні заголовку iostream, ми отримуємо доступ до всієї ієрархії класів бібліотеки iostream, які відповідають за функціонал вводу/виводу даних (включаючи клас, який називається iostream).

Клас istream використовується для роботи з вхідними потоками. Оператор вилучення >> використовується для вилучення значень з потоку. В цьому є сенс: коли користувач натискає на клавішу клавіатури, код цієї клавіші поміщається у вхідний потік. Потім програма витягує це значення з потоку і використовує його.

Клас ostream використовується для роботи з вихідними потоками. Оператор вставки << використовується для вставки значень в потік. В цьому також є сенс: ви вставляєте свої значення в потік, а потім споживач даних (наприклад, монітор) використовує їх.

Клас iostream може обробляти як ввід, так і вивід даних, що дозволяє йому здійснювати двонаправлений ввід/вивід.

Константа – величина, яка не змінюється упродовж виконування програми. Для оголошення константи у програмі використовується специфікатор const. При цьому зазначається тип константи і обов’язково надається значення, наприклад:

const double Pi = 3.14159;

У якості значення константи можна подавати константний вираз, який містить раніш оголошені константи та змінні. Наприклад:

const double Pi2 = 2 \* Pi, k = Pi /180;

Математичні функції широко використовуються для записування різних математичних виразів.

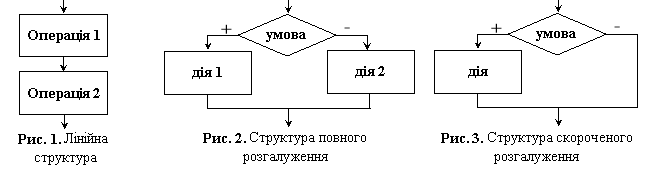
Стандартні математичні функції С++ розміщені у заголовному файлі (бібліотеці) math.h, який долучають директивою #include<math.h>.

### 3.1.4 Програмування лінійних та розгалужених обчислювальних процесів

Будь-який алгоритм можна уявити собі як деяку структуру, що складається з окремих базових елементів. Визначають три базових структурних елементи: лінійний, розгалужений, циклічний.

**Лінійним** елементом алгоритму називається така операція, яка визначає один елементарний крок обробки або відображення інформації.

До таких операцій відносяться дії зміни значення деяких величин, введення та виведення інформації тощо. Декілька лінійних елементів можуть об'єднуватися і утворювати складену лінійну структуру (див. рис. 1).

**Розгалуженим** елементом алгоритму називається така операція, за допомогою якої здійснюється вибір однієї з двох можливих дій в залежності від сформульованої умови.

**Циклічним** елементом алгоритму називається така операція, за допомогою якої здійснюється визначена кількість повторень однієї або декількох дій згідно сформульованої умови. Є два типи повторення: з передумовою та з після умовою.

### 3.1.5 Програмування циклічних обчислювальних процесів

Циклічним називається такий обчислювальний процес, який повторює одну і ту ж визначену послідовність операторів декілька разів. В Borland Pascal розрізняють три види циклів: оператор циклу з лічильником – FOR, оператор циклу з передумовою – WHILE, оператор циклу з післяумовою (постумовою) – REPEAT.

У випадках, коли заздалегідь відома кількість повторень, використовують оператор циклу з лічильником:

FOR <змінна>:=<початкове\_значення\_лічильника> TO

<кінцеве\_значення\_лічильника> DO

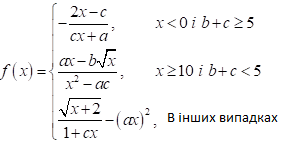
<оператор>

де змінна, якій спочатку присвоюється початкове значення, виконує роль лічильника, і при кожному повторенні збільшується на одиницю або зменшується на одиницю, якщо замість ключового слова TO записати інше ключове слово – DOWNTO (при цьому початкове значення повинно бути більше за кінцеве). Саме ця змінна і показує на кількість повторень.

Два інші оператори циклу використовують коли кількість повторень наперед невідома, ознакою закінчення циклу виступає виконання або невиконання деякої заданої умови.

## **3.2 Практичне завдання**

### 3.2.1 Завдання



### Постановка задачі

Метою розробки задачі є вивчення нових аспектів у роботі з мовою програмування С++, а саме ті, що використовуються при написанні програми для розв’язання прикладів, перевірки вхідних значень і повернення результату.

Вхідні дані

e value a: 5

Enter the value b: 5

Enter the value c: 5

Enter the starting value of x (xstart): 2

Enter the final value of x (xfinal): 8

Enter step h: 4

Вихідні дані

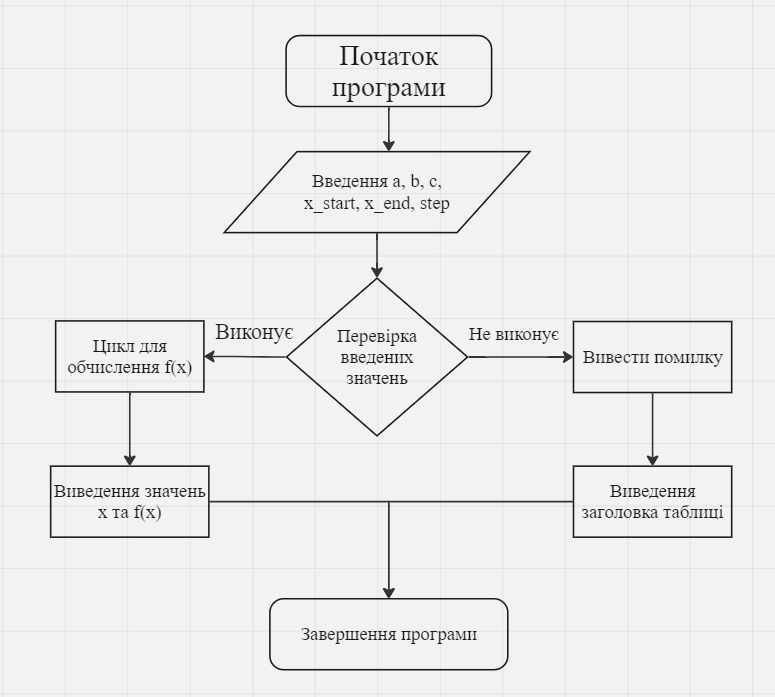
x f(x)

---------------------------

2 -99.8182

6 -899.909

Блок-схема алгоритму роботи програми представлена на рис. 1

 Рисунок 1 - Блок-схема алгоритму роботи програми

### 3.2.3 Код програми

#include <iostream>

#include <cmath>

#include <iomanip>

using namespace std;

double f(double x, double a, double b, double c) {

if (x < 0 && b + c >= 5) {

return (-2 \* x - c) / (c \* x + a);

}

else if (x >= 10 && b + c < 5) {

return (a \* x - b \* sqrt(x)) / (x \* x - a \* c);

}

else {

return (sqrt(x + 2)) / (1 + c \* x) - pow(a \* x, 2);

}

}

int main() {

double a, b, c;

double x\_start, x\_end, h;

cout << "Enter the value a: ";

cin >> a;

cout << "Enter the value b: ";

cin >> b;

cout << "Enter the value c: ";

cin >> c;

cout << "Enter the starting value of x (xstart): ";

cin >> x\_start;

cout << "Enter the final value of x (xfinal): ";

cin >> x\_end;

cout << "Enter step h: ";

cin >> h;

if (h <= 0) {

cerr << "Error: Step h must be positive." << endl;

return 1;

}

if (x\_start > x\_end) {

cerr << "Error: The initial value of x must be less than or equal to the final value of x." << endl;

return 1;

}

// Заголовок таблиці

cout << setw(10) << "x" << setw(20) << "f(x)" << endl;

cout << "-------------------------------------" << endl;

// Виведення значень f(x) для кожного x на інтервалі з кроком h

for (double x = x\_start; x <= x\_end; x += h) {

double fx = f(x, a, b, c);

cout << setw(10) << x << setw(20) << fx << endl;

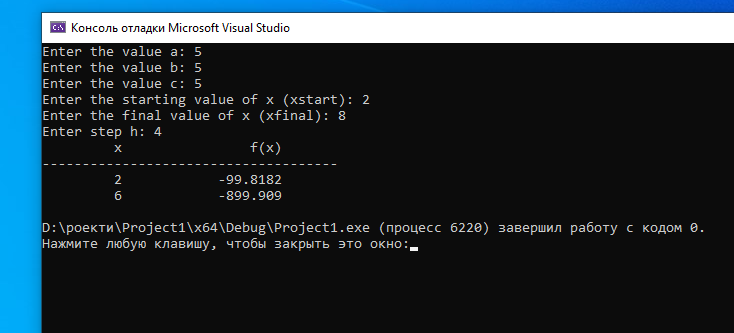
}

return 0;

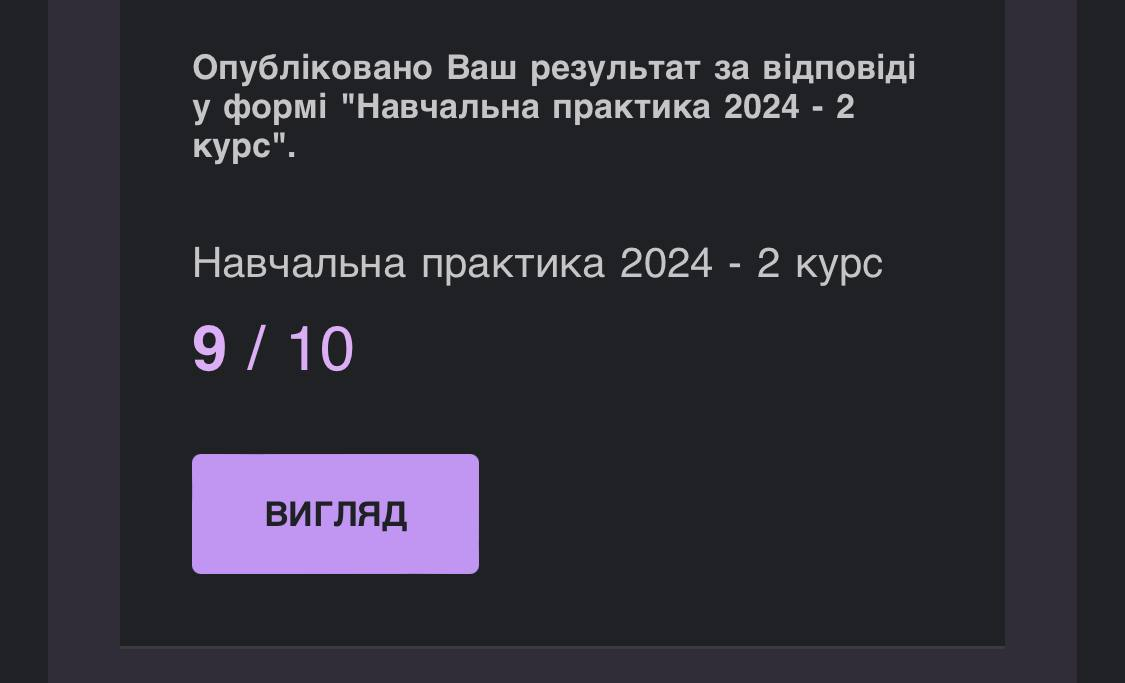
}

### 3.2.4 Скріншот виконання програми

Скріншот виконання програми представлена на рис. 2

 Рисунок 2 - Скріншот виконання програми

## **3.3 Виконання тестів**



# 4 РОЗДІЛ " ПРАКТИЧНА РОБОТА З С++"

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

У базових консольних програмах C++ об'єкт `cout` використовується для виведення даних на заданий пристрій. Синтаксис `cout << data` дозволяє виводити змінні, константи або вирази. Для форматування виведення даних використовуються різні маніпулятори, такі як `hex` та `oct`, які задають виведення чисел у 16-ковій та 8-ковій системах відповідно. Маніпулятори `setw()`, `setfill()`, і `setprecision()` змінюють стан об’єкта `cout`: `setw()` задає ширину поля для виведення, `setfill()` заповнює незаповнені позиції встановленим символом, а `setprecision(n)` визначає кількість знаків після коми.

Маніпулятори `setiosflags` і `resetiosflags` встановлюють або скидають певні глобальні прапори, які використовує клас `iostream` для визначення поведінки при введенні та виведенні. Наприклад, `setiosflags()` встановлює зазначені прапори, а `resetiosflags()` їх очищає.

У класичному C для виведення даних використовується функція `printf`, яка також доступна у C++. Синтаксис `printf(char \*format, ...)` дозволяє виводити дані з використанням специфікаторів формату. Форматний рядок складається з символів, що виводяться без змін, і спеціальних символів формату, які задають перетворення даних. Специфікатори включають `%d` або `%i` для десяткового цілого зі знаком, `%u` для десяткового цілого без знака, `%x` для шістнадцятирічного цілого без знака, `%f` для чисел з плаваючою точкою, `%e` для експоненційного запису чисел, `%c` для символів, `%s` для рядків, і `%p` для вказівників. Символи `L` або `l` використовуються для довгих типів даних, таких як `long double` або `long int`.

Для введення даних у C++ використовується об’єкт `cin`, синтаксис якого `cin >> змінна` дозволяє вводити змінні. При введенні рядків кожне слово вводиться окремо, а роздільником між ними є пробіл або Enter.

Функція `scanf`, яка також доступна у C, дозволяє вводити дані за допомогою форматних рядків, аналогічних до `printf`. Перед іменами змінних у `scanf` потрібно ставити символ `&`, який означає "взяти адресу" змінної для її зчитування.

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

У темі "Структури даних для управління станом програми" основні поняття включають опис найпоширеніших структур даних у C++ та їхні реалізації.

Масив є інструментом для представлення однорідних даних. Він складається з елементів, кожен з яких має індекс. Властивості масиву включають тип елементів, назву, розмірність і діапазони індексів.

У C++ індекси починаються з 0. Звернення до елемента масиву здійснюється за допомогою індексів у квадратних дужках.

Одновимірний масив, або вектор, має один індекс для кожного елемента. Двовимірний масив, або матриця, має два індекси: номер рядка та номер стовпчика.

**Зв'язний список** — лінійно упорядкована структура (послідовність) даних, елементи якої — вузли — містить два види даних:

* фактичні дані довільного типу;
* вказівник (посилання) на наступний вузол послідовності.

Изображение выглядит как текст, снимок экрана, Шрифт, диаграмма

Автоматически созданное описание

Основні дії зі зв'язним списком:

* додавання елемента у список;
* вилучення елемента зі списку;
* пошук елемента (з певними властивостями) у списку.

Контейнер list (список) у мові С++ задає двонаправлений список. У ці списки можна швидко вставляти, а також видаляти елементи з них. Доступ до елементів списку (як і всіх наступних структур) здійснюють за допомогою вказівників (ітераторів).

**Стек** — лінійно упорядкована структура (послідовність) даних, у яку можна вставити елемент або з якої можна вилучити елемент лише з одного кінця.

Стек нагадує стос аркушів паперу у вузькій шухляді: щоб узяти певний аркуш, потрібно спочатку прибрати всі аркущі над ним. Кажуть, що для стеку діє принцип: «Останній зайшов, перший вийшов» (англій­ською LIFO: Last In First Out).

**Изображение выглядит как снимок экрана, линия, дизайн

Автоматически созданное описание**

Основні операції зі стеком:

* вставлення елемента в кінець стеку;
* вилучення елемента з кінця;
* відображення вмісту останнього долученого елемента.

Наявні вбудовані функції для роботи з векторами дозволяють втілити ідею стеку. Контейнер stack безпосередньо втілює принцип: «Останній зайшов, перший вийшов» (англій­ською LIFO: Last In First Out):

**Черга** — лінійно упорядкована структура (послідовність) даних, у яку можна вставити елемент з одного кінця або з якої можна вилучити елемент з іншого кінця

Изображение выглядит как Шрифт, снимок экрана, текст, диаграмма

Автоматически созданное описание

Прообразом цієї структури є чергу людей у магазині: того, що став першим, буде обслужено першим. Якщо розглядати чергу щодо доступу до даних, то вона реалізує принцип: «Першим зайшов, перший вийшов» (англій­ською First In First Out, FIFO). Інакше кажучи, після додавання нового елементу всі елементи, які було додано до цього, потрібно вилучити до того, як новий елемент буде вилучено.

Основні операції з чергою:

* вставлення елемента в кінець черги;
* вилучення елемента з початку черги.

Контейнер queue безпосередньо втілює принцип: «Першим зайшов, перший вийшов» (англій­ською First In First Out, FIFO):

**Асоціативний масив** (словник в інших мовах програмування) — це вбудована структура даних для збереження даних у форматі ключ-значення.

Цю структура даних легко уявити як шафу з підписаними шухлядами. Усі дані зберігають у шухлядах. За назвою можна легко знайти шухляду і взяти з неї те значення, яке там розташовано.

Изображение выглядит как текст, Шрифт, белый, дизайн

Автоматически созданное описание

На відміну від звичайних шаф, в асоціативний масив будь-коли можна додати нові «шухляди» або вилучити вже наявні.

**Set** — контейнер для збереження унікальних значень (кожне не більше одного разу), що втілює математичне поняття множини.

При додаванні нового елемента в даному контейнері буде проведено упорядкування елементів. Повторювані елементи буде вилучено.

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

Робота з файлами у мові програмування C++ дозволяє програмістам зберігати та отримувати дані з файлової системи комп'ютера. Це може бути корисно для збереження конфігураційних даних, роботи зі структурованими даними, збереження результатів обчислень та багато іншого.

Для роботи з файлами у C++ використовуються два основних класи: ifstream і ofstream. Перший використовується для читання з файлу, а другий - для запису у файл. Вони базуються на базовому класі fstream, який також можна використовувати як універсальний інструмент для роботи з файлами.

Щоб використовувати файли у програмі, спочатку треба їх відкрити. Це робиться за допомогою конструктора класу ifstream або ofstream, якому передається шлях до файлу. Якщо файл існує, його відкриють для роботи, а якщо ні, то він буде створений.

Після відкриття файлу, програма може зчитувати дані з нього, записувати дані у нього або виконувати будь-які інші операції, пов'язані з файлом. Після завершення роботи з файлом його слід закрити за допомогою методу close().

Приклади використання:

* Для читання з файлу:

ifstream inputFile("example.txt");

* Для запису в файл:

ofstream outputFile("example.txt");

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

Текстові файли:

Текстові файли містять дані у формі тексту, що може бути зрозумілим для людини. Це може бути рядок символів, числа або інші дані, які можна зчитати та редагувати звичайним текстовим редактором.

Дані у текстових файлах зазвичай розділяються символами пробіла, комою, крапкою з комою або іншими розділовими знаками.

Бінарні файли:

Бінарні файли містять дані у бінарному форматі, який не призначений для прямого читання людиною. Дані зазвичай зберігаються у вигляді послідовності бітів, що представляють числа, структури даних тощо.

Бінарні файли зазвичай потребують спеціальної обробки для читання та запису, оскільки дані можуть бути представлені у складних форматах.

У мові програмування C++ для читання та запису файлів використовуються класи ifstream для читання та ofstream для запису. Крім того, для обробки бінарних файлів можна використовувати функції fread та fwrite.

Іноді для обробки певних форматів файлів (наприклад, JSON, XML) використовують спеціалізовані бібліотеки, які надають зручний інтерфейс для читання та запису даних у цих форматах.

## **4..5 Тема 5. Системи контролю версій (VCS)**

Система контролю версій (VCS) - це програмний інструмент, який дозволяє розробникам зберігати історію змін у вихідному коді свого програмного забезпечення. Основна мета VCS полягає в тому, щоб забезпечити зручну та безпечну роботу з кодом для розробників та команд розробників.

Використання систем контролю версій дозволяє:

* Зберігати історію змін: Кожне збереження (commit) в системі контролю версій записує зміни, внесені у вихідний код. Це дозволяє розробникам переглядати, аналізувати та повертатися до попередніх версій коду у разі потреби.
* Спільно працювати над проектом: VCS дозволяє розробникам працювати над проектом одночасно та координувати свою роботу. Кожен розробник може вносити свої зміни, а система автоматично об'єднує їх.
* Експериментувати та розгалужувати розробку: Розгалуження розробки (branching) дозволяє розробникам експериментувати з новими функціями або розвивати окремі функціональність у відокремлених гілках, не впливаючи на основний код проекту.

Варіанти систем контролю версій включають:

* Локальні системи контролю версій: Розробники працюють локально на своїх комп'ютерах і зберігають історію змін лише на своєму пристрої. Це простий підхід, але він не підходить для спільної роботи в команді.
* Централізовані системи контролю версій: Всі зміни зберігаються на центральному сервері, і розробники працюють з віддаленими копіями коду. Цей підхід підходить для команд, але може стати однією точкою відмови.
* Розподілені системи контролю версій: Кожен розробник має повну копію історії проекту. Це дозволяє працювати навіть без підключення до мережі та має більшу стійкість до відмов серверів.

Важливо також відзначити, що історія розвитку систем контролю версій проходить три покоління, починаючи з простих локальних систем і закінчуючи розподіленими системами з розгалуженням розробки та можливістю злиття гілок.

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

**Markdown** — полегшена мова розмітки даних, яку створено з ухилом на прочитність та зручність у публікації з подальшим перетворенням її на structurally valid XHTML або HTML. Чимало ідей для мови позичено з існуючих домовленостей у розмітці тексту в електронних листах. Першу реалізацію Markdown написано Грубером на Perl, але з часом з'явились багато реалізацій від сторонніх розробників. Реалізація на Perl розповсюджується за ліцензією BSD.

Такі сайти, як GitHub, Reddit та Stack Overflow використовують Markdown для полегшення обговорень між користувачами.

Ось декілька основних функцій мови Markdown, які були використані у оформлені мого репозиторію:

* Заголовки

Для того що б додати заголовок потрібно додати від одного до шести #-символів перед тестом заголовка. Кількість #-символів, яка використовується, визначатиме рівень ієрархії та розмір гарнітури заголовка. Наприклад:

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описание

Коли використовуєтсья два або більше заголовків, GitHub автоматично створює зміст, до якого можна отримати доступ, клацнувши в заголовку файлу. Кожна назва заголовка перерахована у змісті, і користувач може клацнути назву, щоб перейти до вибраного розділу.

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

* Стилізація тексту

|  |  |  |  |
| --- | --- | --- | --- |
| **Стиль** | **Синтаксис** | **Приклад** | **Вихідний результат** |
| Жирний | \*\* \*\* або \_\_ \_\_ | \*\*This is bold text\*\* | **This is bold text** |
| Курсивний | \* \* або \_ \_ | \_This text is italicized\_ | This text is italicized |
| Закреслення | ~~ ~~ | ~~This was mistaken text~~ | ~~This was mistaken text~~ |
| Жирний і вкладений курсив | \*\* \*\* and \_ \_ | \*\*This text is \_extremely\_ important\*\* | **This text is extremely**  **important** |
| Усі жирні та курсивні | \*\*\* \*\*\* | \*\*\*All this text is important\*\*\* | ***All this text is important*** |

* Посилання

Користувач може створити вбудоване посилання, помістивши текст посилання в квадратні дужки [ ], а потім увімкнувши URL-адресу в дужки ( ), або скористатися комбінацією клавіш Command+K, щоб створити посилання. Вибравши текст, можна вставити URL-адресу з буфера обміну, щоб автоматично створити посилання з виділеного тексту.

Також можна створити гіперпосилання Markdown, виділивши текст і скориставшись комбінацією клавіш Command+V. Якщо користувач хоче замінити текст посиланням, слід скористатися комбінацією клавіш Command+Shift+V.

Изображение выглядит как текст, Шрифт, снимок экрана, белый

Автоматически созданное описание

## Зображення

Користувач може відобразити зображення, додавши ! і альтернативний текст в [ ]. Альтернативний текст – це короткий текстовий еквівалент інформації на зображенні. Потім слід помістити посилання на зображення в дужки ().

Изображение выглядит как текст, мультфильм, графическая вставка, Анимация

Автоматически созданное описание

* Список

Невпорядкований список можна створити, поставивши перед одним або декількома рядками тексту -, \* або +.

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Щоб упорядкувати список, перед кожним рядком слід додати номер.

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описание

## **4.7 Розроблення ігрової програми**

### 4.7.1 Код програми

#include <iostream>

using namespace std;

const int SIZE = 3;

char board[SIZE][SIZE] = {{'1', '2', '3'}, {'4', '5', '6'}, {'7', '8', '9'}};

int player1\_wins = 0;

int player2\_wins = 0;

char currentPlayer = 'X';

void displayBoard() {

cout << "Current Board:" << endl;

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

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

cout << board[i][j] << " ";

}

cout << endl;

}

}

bool checkWin(char mark) {

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

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

return true;

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

return true;

}

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

return true;

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

return true;

return false;

}

bool checkDraw() {

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

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

if (board[i][j] != 'X' && board[i][j] != 'O')

return false;

}

}

return true;

}

void switchPlayer() {

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

}

void updateScore(char winner) {

if (winner == 'X')

player1\_wins++;

else if (winner == 'O')

player2\_wins++;

}

void resetBoard() {

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

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

board[i][j] = '1' + i \* SIZE + j;

}

}

}

void displayScore() {

cout << "Player 1 Wins: " << player1\_wins << endl;

cout << "Player 2 Wins: " << player2\_wins << endl;

}

int main() {

int choice;

char winner;

do {

displayBoard();

cout << "Player " << currentPlayer << ", enter your move (1-9): ";

cin >> choice;

int row = (choice - 1) / SIZE;

int col = (choice - 1) % SIZE;

if (board[row][col] == 'X' || board[row][col] == 'O') {

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

continue;

}

board[row][col] = currentPlayer;

if (checkWin(currentPlayer)) {

cout << "Player " << currentPlayer << " wins!" << endl;

updateScore(currentPlayer);

displayScore();

cout << "Do you want to play again? (1 for Yes, 0 for No): ";

cin >> choice;

if (choice == 1) {

resetBoard();

continue;

} else {

break;

}

}

if (checkDraw()) {

cout << "It's a draw!" << endl;

cout << "Do you want to play again? (1 for Yes, 0 for No): ";

cin >> choice;

if (choice == 1) {

resetBoard();

continue;

} else {

break;

}

}

switchPlayer();

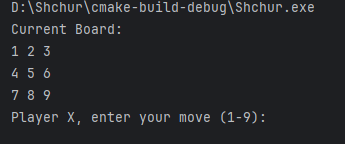
} while (true);

return 0;

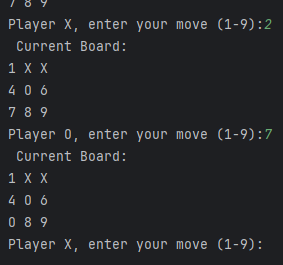
}

### 4.7.2 Скріншоти виконання програми

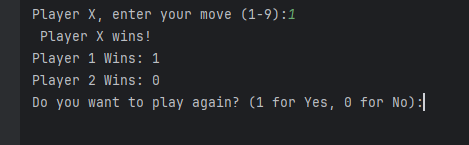
Початок програми:



Середина програми:



Кінець програми:



# 5 РОЗДІЛ ”ВЕБ-ДИЗАЙН ТА HTML”

HTML (від англ. HyperText Markup Language – «мова гіпертекстової розмітки») – стандартна мова розмітки документів в мережі Інтернет. Біль- шість веб-сторінок містять опис розмітки на мові HTML (або XHTML). Мова HTML інтерпретується браузерами і відображається у вигляді документа в зручній для людини формі.

Мова HTML є додатком SGML (стандартної узагальненої мови роз- мітки) і відповідає міжнародному стандарту ISO 8879.

У всесвітній павутині HTML-сторінки, як правило, передаються брау- зерам від сервера по протоколах HTTP або HTTPS, у вигляді простого тексту або з використанням шифрування.

Мова HTML була розробленою британським вченим Тімом Бернерс-Лі приблизно в 1986-1991 роках в стінах Європейського Центру ядерних дослі- джень в Женеві (Швейцарія). Елементами є структурні і семантичні елементи – дескриптори. Дескриптори також часто називають «тегами». Також, в HTML внесена підтримка гіпертексту.

Таким чином, веб-сторінки створюються за допомогою мови розмітки гіпертексту HTML. Опис веб-сторінок міститься в HTML-програмі, який збе- рігається у звичайному текстовому файлі з розширенням htm чи html. Про- грами мовою HTML містять інструкції (коди), що називаються тегами. Всі теги мови HTML виділяються символами-обмежувачами < і >, між якими за- писується ідентифікатор (ім’я) тега і, можливо, його параметри. Більшість те- гів HTML використовується попарно, тобто для окремого тега (відкриваючо- го) в документі є відповідний закриваючий тег, який записується так само, як і відкриваючий, але з символом / перед іменем тега (закриваючі теги не вико- ристовують параметри).

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

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>. У даному випадку ми говоримо браузеру, що HTML відповідає міжнародній специфікації версії 5.

Не треба намагатися запам'ятати цей рядок. Головне - необхідно знати, що він необхідний. Адже саме він допомагає зробити сайт, який буде однаково виглядати в усіх браузерах (як правило це Mozilla Firefox, Google Chrome, Opera, Internet Explorer).

Далі позначається початок і кінець документа тегами <html> і </html>, відповідно. Всередині цих тегів повинні знаходитися теги голови (<head> </head>) і тіла документу (<body> </body>).

Кодування HTML-сторінки потрібно вказувати для того, щоб веб-браузер міг правильно відображати текст на сторінці. Якщо браузер неправильно «вгадає» кодування, то замість тексту будуть відображатися ієрогліфи. Щоб повідомити браузеру кодування HTML-сторінки, необхідно всередині тега <head> використовувати тег:

<meta charset = "ім'я кодування">

Найпоширеніше сучасне кодування - utf-8. Використовуйте його в усіх своїх проектах. Для кирилиці в Windows charset часто задавали як windows-1251. Але зараз це вважається поганою практикою.

Є ціле сімейство службових тегів <meta>, які так і називають - **"мета-теги"**. Їх використовують всередині тега <head>. Мета-теги розрізняються набором атрибутів і їх значень. Ось деякі з атрибутів: content, http-equiv, name і scheme.

Мета-теги зберігають корисну для браузерів і пошукових систем інформацію. Один з таких тегів − це опис ключових слів сторінки. Задається він наступним чином:

<meta name = "keywords" content = "різні, ключові, слова">

В атрибуті content через кому перераховують найважливіші слова зі змісту сторінки. Раніше цей тег був дуже важливий для пошукових систем. Який стан справ зараз - велика таємниця пошукових систем.

Ще один корисний для пошукових систем мета-тег − короткий опис сторінки. Він задається так:

<meta name = "description" content = "короткий опис">

В атрибуті content має бути короткий зміст або анотація сторінки, яка часто використовується пошуковими системами при відображенні результатів пошуку.

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

Основними елементами форматування тексту в html є теги:

* P − використовується для розбивання тексту на параграфи;
* h1, h2, ..... h6 - застосовуються для створення заголовків 1,2 ... 6 рівнів, відповідно;
* br - використовується для переносу рядка;
* div, span - використовуються для виділення частини документа певним способом.

Наведені теги - це далеко не повний перелік елементів, за допомогою яких форматується тест веб-сторінки. Але їх може бути достатньо для її створення.

Списки часто використовуються в різних документах. Іноді, щоб зробити список, користувач просто нумерує рядки тексту. Такий підхід не є хорошим, оскільки в документі відсутня логічна сутність «список». У HTML існує сімейство тегів для створення списків: невпорядкованих, упорядкованих і списків визначень.

*Невпорядковані (або марковані) списки* створюються за допомогою тега <ul> (unordered list), який може містити всередині себе теги <li> (list item), 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. Наприклад:

<ul> - відкриваємо маркований список

<li>Перший пункт списку</li> -додаємо пункти списку

<li>Другий пункт списку</li> -додаємо пункти списку

<li>Наступний пункт списку</li> -додаємо пункти списку

</ul> - закриваємо маркований список

*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. Нумеровані або впорядковані списки призначені для елементів, які слідують в певному порядку, наприклад покрокові інструкції. Вони функціонують так само, як і описані раніше марковані списки, за винятком того, що визначені за допомогою тега <ol>, та замість маркерів браузер автоматично вказує номера перед елементами списку, тому вам не потрібно проставляти їх у вихідному документі. Це дозволяє легко міняти місцями елементи списку без повторної нумерації.

Для впорядкованого списку можна задати атрибут start, який змінює початок нумерації. Наприклад:

<ol start = "3">

<li> перший пункт списку </ li>

<li> другий пункт списку </ li>

</ ol>

*Вставка гіперпосилань*

Посилання є основою гіпертекстових документів. Вони дають змогу переходити з однієї веб-сторінки на іншу. Особливість посилань полягає в тому, що вони можуть вести не тільки на HTML-файли, а й на файл будь-якого типу, причому цей файл може розміщуватися зовсім на іншому сайті. Головне, щоб до документа, на який робиться посилання, був доступ. Іншими словами, якщо шлях до файлу можна вказати в адресному рядку браузера, і файл при цьому буде відкритий, то на нього можна зробити посилання.

Для створення посилання необхідно повідомити браузеру, який елемент є посиланням, а також вказати адресу документа, на який слід зробити посилання. Обидва дії виконуються за допомогою тега <a>. Загальний синтаксис створення посилань наступний:

<a href="URL"> текст посилання </a>

Атрибут href визначає URL (*Universal Resource Locator*, універсальний покажчик ресурсу), тобто адресу документа, на який слід перейти, а вміст контейнера <a> є посиланням. Текст, розташований між тегами <a> і </a>, за замовчуванням стає синього кольору і підкреслюється.

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

При створенні відносних посилань треба розуміти, яке значення для атрибута href слід вказувати, оскільки воно залежить від початкового розташування файлів.

*Файли розміщуються в одній папці*.

Необхідно зробити посилання з початкового файлу на файл за посиланням. У такому випадку код буде наступним

<a href="Файл за посиланням.html"> Посилання </a>

*Файли розміщуються в різних папках.*

Розглянемо декілька випадків. Наприклад, початковий файл знаходится в папці Folder, а файл за посиланням у корені сайту. Файли розміщуються в різних папках

У такому випадку перед ім'ям файлу в адресі посилання слід поставити дві крапки і слеш (/), як показано нижче.

<a href="../Файл за посиланням.html">Посилання</a>

*Види посилань*

Будь-яке посилання на веб-сторінці може знаходитися в одному з наступних станів.

Посилання, яке не відвідувалось (посилання, за якими перехід ще не відбувався). За замовчуванням невідвідані текстові посилання мають синій колір і підкреслення.

Активне посилання (посилання у момент відкриття, коли користувач здійснює на ньому клацання мишкою). Оскільки час між натисканням на посилання і початком завантаження нового документа досить малий, подібний стан посилання вельми короткочасний. Активним посилання стає також, при його виділенні за допомогою клавіатури. Колір такого посилання за замовчуванням червоний.

Відвідуване посилання. Як тільки користувач відкриває документ, на який вказує посилання, воно позначається як відвідане і змінює свій колір на фіолетовий, встановлений за замовчуванням.

*Атрибути гіперпосилань*

Основний атрибут href тега <a> ми вже освоїли, розглянемо ще декілька корисних, але необов'язкових атрибутів цього тега.

**download**

За замовчуванням цей атрибут вимкнено. Якщо його додати до тегу, то браузер не буде переходити за посиланням, а запропонує користувачу завантажити файл, вказаний у адресі посилання.

**target**

За замовчуванням, при переході за посиланням документ відкривається в поточному вікні або фреймі. При необхідності, ця умова може бути змінена атрибутом target тега <a>. Синтаксис наступний:

<a href="#" target="імя вікна"> ... </a>

Як значення використовується ім'я вікна або фрейму, заданий атрибутом name. Якщо встановлено неіснуюче ім'я, то буде відкрито нове вікно. У якості зарезервованих імен застосовуються такі:

* \_blank - завантажує сторінку в нове вікно браузера.
* \_self - завантажує сторінку в поточне вікно (це значення задається за замовчуванням).

**title**

Додає пояснювальний текст до посилання у вигляді підказки. Така підказка відображається, коли курсор миші затримується на посиланні, після чого підказка через деякий час зникає. Синтаксис наступний:

<a href="#" title="текст"> ... </a>

Як значення вказується будь-який текстовий рядок, який беруть у подвійні або одинарні лапки.

*Посилання на адресу електронної пошти.*

Створення посилання на адресу електронної пошти робиться майже так само як і посилання на веб-сторінку. Тільки в атрибуті href замість URL вказується [**mailto:**example@gmail.com](mailto:example@gmail.com).

Подібне посилання нічим не відрізняється від посилання на веб-сторінку, але при натисканні на нього запускається поштова програма, встановлена за замовчуванням. Тому в назві посилання слід вказувати, що воно має відношення до електронної пошти, щоб користувачі розуміли, до чого призведе натискання на нього.

Можна також автоматично додати тему повідомлення, приєднавши до адреси електронної пошти через символ "знак питання" (?) параметр subject=тема повідомлення

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

Таблиці стилів дають змогу спростити процес створення сторінок і поліпшити їх зовнішній вигляд. Концепція стилів подібна до ідеї стилів, яку реалізовано в сучасних текстових редакторах — текст спочатку вводять, а потім форматують, користуючись стилями. Застосування стилів дає змогу вводити на сторінку потрібні тексти та інші елементи, не задумуючись над їхнім зовнішнім виглядом і розташуванням.

Таблиці стилів зазвичай створють окремо від html-файлу. Під час створення html-файлу увагу концентрують на змісті сторінки, а не на її зовнішньому вигляді, а під час створення таблиці стилів — навпаки. Отже, стилі дають змогу розділити етапи створення html-файлу й удосконалення зовнішнього вигляду сторінки.

Cпособи взаємодії таблиці стилів з html-файлом:

* зв'язування;
* імпортування;
* вбудовування (використання стильових класів).

*Зв'язування*

Таблицю стилів створюють і зберігають в окремому файлі з розширенням css. Таку таблицю називають зовнішньою. Щоб зв'язати основний файл з такою таблицею стилів, у середині тега <head> застосовують одинарний тег <link> з інформацією про таблицю.

*<head>*

*<link href="адреса таблиці стилів"*

*type="text/css"*

*rel="stylesheet">*

*</head>*

Тут

href="адреса таблиці стилів" — адреса розташування файлу,

type="text/css" — опис типу файлу,

rel="stylesheet" — вказівка на відношення основного файлу з файлом таблиці стилю.

*Імпорт*

Це те саме, що і зв'язування, але взаємодію файлів забезпечують засобами тегу <style> і такої вказівки:

@import URL(адреса таблиці стилів)

*Вбудовування (використання стильових класів)*

Якщо таблицю створено лише для деякого конкретного html-файлу, то її розташовують у цьому файлі в контейнері <style>. Таку таблицю стилів називають внутрішньою або вбудованою. Також стиль можна вбудувати безпосередньо в html-тег за допомогою атрибуту <style>. Наприклад, контейнер

<span style="color: blue">Blue </span>

створює блок з текстом синього кольору: Blue.

Таблиця стилів складаєтся з правил, а правило — з назви тега (селектора) чи списку назв тегів і описів стилів, які діятимуть у межах деякого html-файлу.

**Опис стилю** — це послідовність пар властивість: значення, які записують через крапку з комою у фігурних дужках.

Наприклад, для виведення усіх абзаців червоним кольором застосовують таке правило:

*p { color: red;}*

Проаналізуємо приклад складнішого правила для списку тегів.

*h3, li{ color: green;*

*font-family: pragmatica;*

*font-size: 16pt;*

*text-align: left;*

*border-style: ridge;*

*border-width: 2mm;}*

Браузер виведе всі заголовки третього рівня h3 і елементи списків <li> зеленим кольором, гарнітурою pragmatica розміру 16 пунктів, з вирівнюванням за лівим краєм вікна, рамкою товщиною 2 мм з видавленим контуром (ridge).

Правила можуть починатися або з назви тега без кутових дужок, або з деякого слова користувача, перед яким є крапка. Це слово стає назвою типу, який можна застосувати в контейнері body до іншого текстового блока, фрагменту тексту чи деякого елемента. Наприклад,

*.mystyle{color: red;*

*font-size: 40pt;*

*margin-top: 30px;}*

Внутрішні таблиці стилів описують у головному файлі в контейнері head за допомогою контейнера style.

Виокремлення текстового блоку, до якого буде застосовано стиль, здійснюють за допомогою контейнера

*<div class=назва стилю>…</div>*

де значення параметра class задає конкретний стиль.

Виокремлення у блоці фрагмента, для якого буде переозначено стиль, здійснюють за допомогою контейнера

*<span>…</span>*

**Жирність шрифту [font-weight]** описує, наскільки жирним (важким) має бути шрифт: normal або bold. Деякі браузери підтримують навіть числові значення 100..900 (у сотнях) для опису жирності шрифту.

**Розмір шрифту [font-size]** можна встановити, використавши одну з таких одиниць вимірювання:

**px** — фіксована величина піксель (pixels), екранна точка;

**pt** — фіксована величина (points), 1/72 дюйма;

% — за 100% беруть поточний розмір шрифта;

**em** — поточний розмір шрифта.

px і pt дають абсолютне значення розміру шрифту, а % і em — відносні. Багато користувачів не можуть читати дрібний текст. Щоб зробити документ доступним для всіх, бажано використовувати саме ті, що задають відносні розміри — % або em.

Скорочений запис [font] використовують, щоб вказати всі властивості шрифту в одному стильовому правилі. Наприклад, чотири рядки опису властивостей шрифту для абзацу:

*p{*

*font-style: italic;*

*font-weight: bold;*

*font-size: 30px;*

*font-family: arial, sans-serif;}*

можна замінити таким записом:

*p{font: italic bold 30px arial, sans-serif;}*

Порядок властивостей font такий:

font-style | font-variant | font-weight | font-size | font-family.

Відступ першого рядка задають значенням властивості text-indent (про одиниці вимірювання див. вище).

Вирівнювання тексту задають значенням властивості text-align:

* left — вирівняти за лівим краєм;
* right — вирівняти за правим краєм;
* center — центрувати;
* justify — вирівняти по ширині.

Декоративні ефекти задають значенням властивості text-decoration:

* underline — підкреслити текст;
* overline — провести лінію під текстом;
* line-through — перекреслити текст.

Для гіперпосилань можна використовувати зміну шрифту, кольору, підкреслення. Новим є те, що в CSS ці властивості можна визначати по-різному в залежності від того, відвідали вже посилання, чи воно активне, чи знаходиться вказівник над посиланням.

**Клас** — одна із форм селекторів html, який можна використати для визначення декількох елементів на відміну від ідентифікатора (id), які можуть визначати один элемент. Клас описують таким чином:

*.назва класу {опис властивостей}*

**Псевдоклас** — це властивість, яка дозволяє змінювати стиль елемента у залежності від дій користувача, а також розташування елемента (тега) у загальному потоці документа, що додає у дизайн сторінки деяку динаміку і логіку.

Класичним прикладом застосування псевдокласу є посилання, що змінює свій колір при наведенні на нього вказівника. Посилання в коді html задають контейнером a. У CSS можна використати цей тег як селектор:

*a{color: blue;}*

Псевдокласи можна використати для встановлення різних стилів для відвіданих і невідвіданих посилань відповідно. Перелічимо ці псевдокласи посилання:

*a:visited — відвідані;*

*a:link — невідвідані;*

*a:active — активні;*

*a:hover — під вказівником*

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

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. Розглянемо найпростішу сторінку:

*<html>*

*<head> </ head>*

*<body>*

*<p> Текст документа </ p>*

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

*</ body>*

*</ html>*

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.

До спадкоємною властивостей відносяться в першу чергу властивості, що визначають параметри відображення тексту:

font-size, font-family, font-style, font-weight, color, text-align, text-transform, text-indent, line-height, letter-spacing, word-spacing, white-space, direction і т. д.

Ці властивості можна і потрібно задавати через предків, слідуючи семантиці документа. Наприклад, параметри тексту часто не змінюються в межах окремих блоків сторінки: меню, основного змісту, інформаційних панелей. Тому загальні параметри тексту (колір, розмір, гарнітура) зазвичай вказують в стилях самих блоків.

Всі інші відносяться до ненаслідуваних. Це параметри позиціонування, розмірів, відступів, фону, рамок і т. д. А саме: background, border, padding, margin, width, height, position і ін.

Наприклад: якщо для будь-якого блоку встановлений внутрішній відступ, автоматично виставляти такий же відступ кожному вкладеному елементу немає ніякої потреби. Ці параметри найчастіше унікальні для кожного окремого блоку.

### 5.5 Практична частина

### 5.5.1 Постановки задачі

Основною задачею було створити веб-сайт у вигляді резюме. Основною 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. Мій проєкт наповнений ненумерованими списками, відформатованим текстом, а саме текстом зі зміненим кольором, розміром шрифту, стилю і т.д. Також у моїй роботі є такий контент як фото, активні посилання та елементи оформлення об’єктів для кращого сприйняття їх персоною, яка буде переглядати мій веб-сайт.

### 5.5.2 HTML документ

**НTML**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<script src="https://kit.fontawesome.com/aa02aa6dcf.js" crossorigin="anonymous"></script>

<title>Resume</title>

<style>

.container{ border: 5px solid silver;

text-indent: 10px;

width: 80%;

height: 100%;

background-color: #e2e4f1;

margin: auto;}

.photo{width: 163px;

float:right;

padding: 10px;

margin: 10px 14px auto 6px;}

.info>p{ margin: 0;

width: 250px;

display: block;

padding: 10px;

text-indent: 50px;}

.a{ border: 5px solid silver;

background-color: #ffffff;

margin: 5px;

padding: 5px;

text-decoration: none;

text-align: center;

width: 500px;

height: 30px;

display: block;}

.b{ border: 5px solid silver;

background-color: #ffffff;

margin: 5px;

padding: 5px;

text-decoration: none;

text-align: center;

width: 500px;

height: 30px;

display: block;}

.c{ border: 5px solid silver;

background-color: #ffffff;

margin: 5px;

padding: 5px;

text-decoration: none;

text-align: center;

width: 500px;

height: 30px;

display: block;}

.content{/\*height: 50px; \*/

padding: 5px;

margin: 0 auto;

box-sizing: border-box;

text-align: center;

line-height: 45px;

display: flex; }

.a:hover{background-color: slategrey;}

.b:hover{background-color: slategrey;}

.c:hover{background-color: slategrey;}

</style></head>

<body>

<div class="container">

<div class="photo">

<img src="https://scontent-iev1-1.xx.fbcdn.net/v/t39.30808-6/334569413\_200837725904013\_515704625506658431\_n.jpg?\_nc\_cat=109&ccb=1-7&\_nc\_sid=5f2048&\_nc\_ohc=NU-taW3Ivw4Q7kNvgFQGADH&\_nc\_ht=scontent-iev1-1.xx&oh=00\_AYCnwXjgi8cxaJxlv1NBA-9m0pCzo0yZWmEySRu\_fxXN5A&oe=66537273"

alt="my" style="width: 150px;" >

</div>

<div class="info">

<p> <p1> <h2>Особисті дані </h2> <ul>

<li>Дата народження: 3 серпня 2006 р.н. </li>

<li>Телефон: +380974340549 </li>

<li>Адреса: м. Київ, с. Троєщина, вул. Шевченка 8</li> </ul> </p1>

<p2> <h3>Освіта</h3>

<hr><h4>Базова середня освіта (2022) </h4>

<ul><li>Школа І-ІІІ ступенів №321 Деснянського району міста Києва</li></ul> <h4>Фаховий молодший бакалавр (2 курс)</h4><ul>

<li>Відокремлений структурний підрозділ «Оптико-механічний фаховий коледж Київського національного університету імені Тараса Шевченка»</li> </ul> </p2>

<p3> <h3>Особисті якості</h3>

<hr><ul>

<li>Уміння працювати в режимі багатозадачності в умовах тимчасового обмеження.</li>

<li>Стресостійкість, комунікабельність, вміння працювати в команді, вміння приймати рішення.</li>

<li>Вміння домовлятися, доводити розпочаті проєкти до кінця за будь-яких обставин, наполегливість, здатність ґенерувати ідеї;</li>

<li>Націленість на результат, уважність до деталей, вміння працювати з великими обсягами інформації, ініціативність, порядність..</li>

</ul></p3>

<p4> <h3>Додаткові відомості </h3> <hr><ul>

<li>Українська — рідна; англійська – базовий рівень. </li>

<li>ПК – досвідчений користувач (повний пакет програм MS Office, навігація в Internet, довідково-інформаційні програми).</li>

<li>Формування особових справ аспірантів/докторантів (на волонтерських засадах)</li>

</ul> </p4> </p> </div>

<header> <div class="content">

<div class="a">

<i class="fa-solid fa-square-envelope"></i>

<a href="mailto:dianashchur4@gmail.com" target="\_blank">E-mail</a></div>

<div class="b">

<i class="fa-brands fa-instagram"></i>

<a href="https://www.instagram.com/sh.di.vl?igsh=MWpjNHpyaGswMGlieg==" target="\_blank">Instagram</a></div>

<div class="c">

<i class="fa-brands fa-facebook"></i>

<a href="https://www.facebook.com/diana.shchur/" target="\_blank">Facebook</a></div>

</div></header></div> </body> </html>

**CSS**

<style>

.container{ border: 5px solid silver;

text-indent: 10px;

width: 80%;

height: 100%;

background-color: #e2e4f1;

margin: auto;}

.photo{width: 163px;

float:right;

padding: 10px;

margin: 10px 14px auto 6px;}

.info>p{ margin: 0;

width: 250px;

display: block;

padding: 10px;

text-indent: 50px;}

.a{ border: 5px solid silver;

background-color: #ffffff;

margin: 5px;

padding: 5px;

text-decoration: none;

text-align: center;

width: 500px;

height: 30px;

display: block;}

.b{ border: 5px solid silver;

background-color: #ffffff;

margin: 5px;

padding: 5px;

text-decoration: none;

text-align: center;

width: 500px;

height: 30px;

display: block;}

.c{ border: 5px solid silver;

background-color: #ffffff;

margin: 5px;

padding: 5px;

text-decoration: none;

text-align: center;

width: 500px;

height: 30px;

display: block;}

.content{/\*height: 50px; \*/

padding: 5px;

margin: 0 auto;

box-sizing: border-box;

text-align: center;

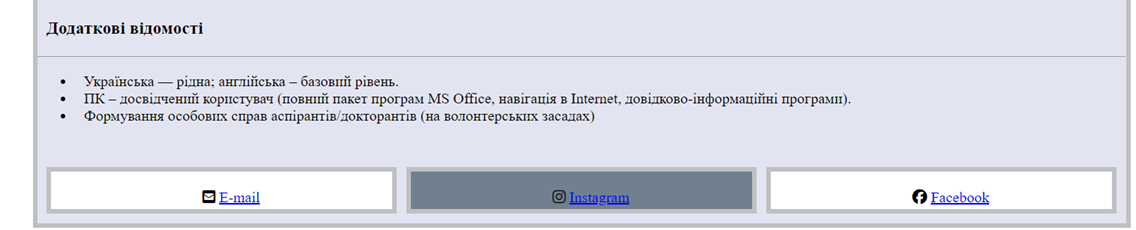
line-height: 45px;

display: flex; }

.a:hover{background-color: slategrey;}

.b:hover{background-color: slategrey;}

.c:hover{background-color: slategrey;}

****</style>

# ВИСНОВКИ

Аналіз роботи

В ході навчальної практики було виконано завдання з різних розділів, зокрема з інформатики, програмування на C++, веб-дизайну та HTML. Кожен розділ вимагав використання відповідного програмного інструментарію та середовища розробки, що дозволило всебічно ознайомитися з сучасними технологіями і методами розробки програмного забезпечення та аналізу даних.

Обраний програмний інструментарій

1. MS Word та MS Excel:
   * MS Word: Використовувався для створення документів складної структури та їх збереження у різних форматах.
   * MS Excel: Застосовувався для розв’язування задач, візуалізації даних, виконання статистичних аналізів та обчислення основних статистичних характеристик.
   * MS Access: Використовувався для проектування баз даних, введення, коригування та вилучення інформації, а також для пошуку, фільтрації та вибірки даних.
2. C++:
   * Редактор коду та компілятор: Використовувався Visual Studio Code або будь-який інший редактор коду, з підтримкою компілятора g++ для написання, компіляції та налагодження програм на мові C++.
   * Системи контролю версій (VCS): Git для контролю версій коду.
3. HTML та CSS:

* Редактор коду: Використовувався редактор коду (Visual Studio Code) для створення HTML-документів та стилів CSS.
* Браузер: Для перевірки відображення та функціональності створених веб-сторінок.

Середовище програмування

1. Для роботи з документами:

* MS Office (Word, Excel, Access): Робота виконувалась у середовищі офісного пакету Microsoft Office.

1. Для програмування на C++:

* IDE: Використовувалась інтегроване середовище розробки, Visual Studio
* Командний рядок: Для компіляції та запуску програм через командний рядок з використанням компілятора g++.

1. Для веб-розробки:

* Редактор коду: Visual Studio Code для написання коду HTML та CSS.
* Браузер: Google

Підсумок

Завдяки використанню зазначених інструментів та середовищ програмування, вдалося реалізувати всі заплановані завдання, що сприяло поглибленню знань та навичок у галузях інформатики, програмування, статистичного аналізу та веб-дизайну. Виконані практичні завдання допомогли закріпити теоретичні знання та отримати цінний досвід роботи з сучасними програмними інструментами.

# ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ ІНФОРМАЦІЇ

1. ДСТУ 1.5-93 Загальні вимоги до побудови, викладу, оформлення та змісту.
2. Мова програмування С++: конспект лекцій / О. В. Галкін, М. М. Верес. — К.: ДП “Вид. дім “Персонал”, 2017. — 260 с. — Біблі- огр.: с. 249.: <https://maup.com.ua/assets/files/lib/book/c_plisplus.pdf>
3. «ОСНОВИ ПРОГРАМУВАННЯ на C++» Зеленський О.С., Лисенко В.С. – Кривий Ріг: Державний університет економіки і технологій, 2023.-269 с.: <https://dspace.duet.edu.ua/jspui/bitstream/123456789/831/1/НП%20Osnovy_C%2B%2B.pdf>
4. Формальні та фактичні параметри, хеш-таблиця (словник), масив, список, стек, черга мовою C++: <https://www.google.com/url?Sa=i&url=https%3A%2F%2Fwww.kievoit.ippo.kubg.edu.ua%2Fkievoit%2F2016%2F65_C%2B%2B%2Findex.html&psig=aovvaw0e_izvl6jk9vdsho-jqp2y&ust=1717139073705000&source=images&cd=vfe&opi=89978449&ved=0cacqrpomahckewiah9ui7lsgaxuaaaaahqaaaaaqba>
5. Міністерство освіти і науки україни Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського» «Управління версіями програмних засобів проекту»: <https://ipze.kpi.ua/wp-content/uploads/2024/02/04-Upravlinnia_versiiamy_prohramnykh_zasobiv_proektu.pdf>
6. Матеріал з Вікіпедії — вільної енциклопедії. «Markdown»: <https://uk.wikipedia.org/wiki/Markdown>
7. MDN Web Docs - an open-source, collaborative project documenting Web platform technologies : <https://developer.mozilla.org/en-US/>
8. Міністерство освіти і науки україни державний вищий навчальний заклад «Ужгородський національний університет» математичний факультет, кафедра кібернетики і прикладної математики «Основи HTML та CSS»: <https://dspace.uzhnu.edu.ua/jspui/bitstream/lib/40974/1/Основи%20HTML.pdf>
9. Міністерство освіти і науки України державний вищий навчальний заклад “Ужгородський національний університет” математичний факультет, кафедра системного аналізу і теорії оптимізації «Вступ до web-програмування. Основи html»: <https://dspace.uzhnu.edu.ua/jspui/bitstream/lib/32871/1/Вступ%20до%20WEB-програмування.%20Основи%20HTML.pdf>
10. Е-helper.com.ua «Лекція 2. Розмітка тексту. Зображення. Посилання.»: <http://e-helper.com.ua/node/949>
11. «CASCADING STYLE SHEETS (CSS) - КАСКАДНІ ТАБЛИЦІ СТИЛІВ»: <https://www.google.com/url?sa=t&source=web&rct=j&opi=89978449&url=http://vv-steshyn.edu.kpi.ua/katalog/files/Lekciya-3.-CSS-kaskadni-tablici-stiliv-68.pdf&ved=2ahUKEwi5utTekqKGAxXD4AIHHaW3DtcQFnoECBIQAQ&usg=AOvVaw3_uxBU96i5-43dUKH8RfG4>
12. Е-helper.com.ua «Лекція 5. Наслідування та каскадування. Оформлення тексту за допомогою CSS»: <http://e-helper.com.ua/node/952>
13. А. Каплун, Ю. В. Баришев, А. В. Остапенко «ТЕХНОЛОГІЯ ПРОГРАМУВАННЯ ЛАБОРАТОРНИЙ ПРАКТИКУМ»: <https://web.posibnyky.vntu.edu.ua/fitki/14baryshev_tehnologiya_programuvannya/08.html>
14. МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ ДОНЕЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ВАСИЛЯ СТУСА ЕКОНОМІЧНИЙ ФАКУЛЬТЕТ КАФЕДРА ЕКОНОМІЧНОЇ ТА УПРАВЛІНСЬКОЇ АНАЛІТИКИ І. Г. ГЕВЛИЧ СТВОРЕННЯ ДОКУМЕНТІВ СКЛАДНОЇ СТРУКТУРИ В MS WORD. Методичні вказівки до виконання лабораторних робіт та завдань з самостійної роботи студентів за курсом «Інформаційні технології в економіці і управлінні»

<https://r.donnu.edu.ua/bitstream/123456789/2465/1/Гевлич_word.pdf>

1. Міністерство освіти і науки України Національний університет біоресурсів і природокористування України Кафедра комп’ютерних систем і мереж ОСИПОВА Тетяна Юріївна САВИЦЬКА Яна Артурівна ПРАКТИКУМ З ОБЧИСЛЮВАЛЬНОЇ МАТЕМАТИКИ ТА ПРОГРАМУВАННЯ

<https://nubip.edu.ua/sites/default/files/u34/posibnik_praktikum_z_obchislyuvalnoyi_matematiki.pdf>

1. Навчально–науковий інститут економіки та управління Обліково–фінансовий факультет Кафедра інформаційних систем і технологій ТЕХНОЛОГІЯ ПРОЕКТУВАННЯ ТА АДМІНІСТРУВАННЯ БАЗ ДАНИХ І СИСТЕМ ДАНИХ

<https://dspace.mnau.edu.ua/jspui/bitstream/123456789/2471/1/Tekhnolohiya_proektuvannya_ta_administruvannya_baz_danykh_i_system_danykh.pdf>

1. Основи програмування з використанням мови C++ <https://greengofactory.jimdofree.com/it-education/розробка-програмного-забезпечення/основи-програмування-з-використанням-мови-c-1-елементи-c/>
2. Арифметичні оператори <https://acode.com.ua/urok-42-aryfmetychni-operatory/>
3. Потоки вводу і виводу <https://acode.com.ua/urok-215-potoky-vvodu-i-vyvodu/#toc-2>
4. МАТЕМАТИЧНІ ФУНКЦІЇ ТА ЇХ ЗАСТОСУВАННЯ В С++ <https://pm11blog.blogspot.com/2018/05/blog-post_60.html>
5. Практична робота "Побудова лінійних та розгалужених алгоритмічних процесів" <https://naurok.com.ua/praktichna-robota-pobudova-liniynih-ta-rozgaluzhenih-algoritmichnih-procesiv-226377.html>
6. ПРОГРАМУВАННЯ ЦИКЛІЧНИХ ОБЧИСЛЮВАЛЬНИХ ПРОЦЕСІВ <https://library.nusta.edu.ua/depository/Електроннi%20книжки/Основи%20програмування%20на%20алгоритмічній%20мові%20Pascal/05.pdf>
7. .
8. .