ВІДОКРЕМЛЕННИЙ СТРУКТУРНИЙ ПІДРОЗДІЛ

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

Циклова комісія спеціальності «Інженерія програмного забезпечення»

**З В І Т**

**З НАВЧАЛЬНОЇ ПРАКТИКИ**

спеціальність 121 Інженерія програмного забезпечення

освітньо-кваліфікаційний рівень «фаховий молодший бакалавр»

|  |  |  |
| --- | --- | --- |
| ПІДСУМКОВА ОЦІНКА  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (оцінка цифрою та прописом)  Керівники практики від коледжу:  \_\_\_\_\_\_\_\_\_\_\_\_\_\_ А. С. Нікітенко  (оцінка) (підпис)  \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Т. І. Лумпова  (оцінка) (підпис)  \_\_\_\_\_\_\_\_\_\_\_\_\_\_ М. С. Ходжаєв  (оцінка) (підпис) |  | Виконав: |
| студент 2-го курсу групи ІПЗ-21 |
| Коротков Даніел Сергійович  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) | |  | |
|  | | | | |
| (П.І.Б студента.) | | | | |
| Спеціальність | *121 Інженерія програмного забезпечення* | Група | | *ІПЗ-21* |

|  | **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-21

ОМФК.0121 НП 20

**ЗМІСТ**

[ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ І ТЕРМІНІВ 6](#_Toc167919156)

[ВСТУП 7](#_Toc167919157)

[1 БЕЗПЕКА ЖИТТЄДІЯЛЬНОСТІ ПРИ РОБОТІ З КОМП’ЮТЕРОМ 8](#_Toc167919158)

[2 РОЗДІЛ "ІНФОРМАТИКА" 9](#_Toc167919159)

[**2.1** **Розроблення засобами MS Word документа складної структури** 9](#_Toc167919160)

[**2.2 Розв‘язок задач за допомогою засобів табличного редактора MS Excel** 9](#_Toc167919161)

[**2.3 Виконання** **практичних завдань за допомогою засобів табличного редактора MS Access** 9](#_Toc167919162)

[3 РОЗДІЛ "ПРОГРАМУВАННЯ С++" 10](#_Toc167919163)

[**3.1 Теоретичні відомості** 10](#_Toc167919164)

[3.1.1 Загальні поняття. Елементи мови С++ - константи, змінні, операції, перетворення типів 10](#_Toc167919165)

[3.1.2 Арифметичні операції 10](#_Toc167919166)

[3.1.3 Поняття вхідного та вихідного потоку, найпростіші математичні функції 10](#_Toc167919167)

[3.1.4 Програмування лінійних та розгалужених обчислювальних процесів 10](#_Toc167919168)

[3.1.5 Програмування циклічних обчислювальних процесів 10](#_Toc167919169)

[**3.2 Практичне завдання** 11](#_Toc167919170)

[3.2.1 Завдання 11](#_Toc167919171)

[3.2.2 Постановка задачі 11](#_Toc167919172)

[3.2.3 Код програми 11](#_Toc167919173)

[3.2.4 Скріншот виконання програми 11](#_Toc167919174)

[**3.3 Виконання тестів** 11](#_Toc167919175)

[4 РОЗДІЛ " ПРАКТИЧНА РОБОТА З С++" 12](#_Toc167919176)

[**4.1 Тема 1. Введення та виведення у базових консольних програмах** 12](#_Toc167919177)

[**4.2 Тема 2. Структури даних для управління станом програми** 12](#_Toc167919178)

[**4.3 Тема 3. Робота з файлами: запис та читання** 12](#_Toc167919179)

[**4.4 Тема 4. Формати файлів та їх обробка** 12](#_Toc167919180)

[**4..5 Тема 5. Системи контролю версій (VCS)** 13](#_Toc167919181)

[**4.6 Тема 6. Мова Markdown для опису проекті** 13](#_Toc167919182)

[**4.7 Розроблення ігрової програми** 13](#_Toc167919183)

[4.7.1 Код програми 13](#_Toc167919184)

[4.7.2 Скріншоти виконання програми 13](#_Toc167919185)

[5 РОЗДІЛ ”ВЕБ-ДИЗАЙН ТА HTML” 14](#_Toc167919186)

[**5.1 Мова розмітки HTML та структура Веб-сторінки** 14](#_Toc167919187)

[**5.2 Розмітка тексту за допомогою HTML. Зображення та посилання** 14](#_Toc167919188)

[**5.3 Каскадні таблиці стилів CSS та їх структура** 14](#_Toc167919189)

[**5.4 Спадковості і каскадування за допомогою CSS** 14](#_Toc167919190)

[**5.5 Практична частина** 15](#_Toc167919191)

[5.5.1 Постановки задачі 15](#_Toc167919192)

[5.5.2 HTML документ 15](#_Toc167919193)

[ВИСНОВКИ 16](#_Toc167919194)

[ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ ІНФОРМАЦІЇ 17](#_Toc167919195)

[ДОДАТКИ 18](#_Toc167919196)

ОМФК.0121 НП 20

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

# ВСТУП

Автор: Коротков Даніел Сергійович

Дата: 12.06.2024

Мета: вивчення компьютерної мови С++ та HTML

Перелік вивчених тем:

1 БЕЗПЕКА ЖИТТЄДІЯЛЬНОСТІ ПРИ РОБОТІ З КОМП'ЮТЕРОМ.

2 РОЗДІЛ "ІНФОРМАТИКА"

3. Розроблення засобами MS Word документа складної структури.......

4. Розв'язок задач за допомогою засобів табличного редактора MS Exce

5. Виконання практичних завдань за допомогою засобів табличного редактора MS Access

3 РОЗДІЛ "ПРОГРАМУВАННЯ С++"

6. Теоретичні відомості.

7. Загальні поняття. Елементи мови С++ константи, змінні, операції, перетворення типів..

8. Арифметичні операції.

9. Поняття вхідного та вихідного потоку, найпростіші математичні функції.

10. Програмування лінійних та розгалужених обчислювальних процесів

11. Програмування циклічних обчислювальних процесів

12. Тема 1. Введення та виведення у базових консольних програмах

13. Тема 2. Структури даних для управління станом програми

14. Тема 3. Робота з файлами: запис та читання

15. Тема 4. Формати файлів та їх обробка

16. Тема 5. Системи контролю версій (VCS)

17. Тема 6. Мова Markdown для опису проекті

18. Розроблення ігрової програми

19. Код програми.

20. Скріншоти виконання програми.

21. РОЗДІЛ "ВЕБ-ДИЗАЙН ТА HTML"

22. Мова розмітки HTML та структура Веб-сторінки

23. Розмітка тексту за допомогою HTML. Зображення та посилання ...

24. Каскадні таблиці стилів CSS та їх структура

25. Спадковості і каскадування за допомогою CSS

26. Практична частина

27. Постановки задачі

28. HTML документ.

# 1 БЕЗПЕКА ЖИТТЄДІЯЛЬНОСТІ ПРИ РОБОТІ З КОМП’ЮТЕРОМ

1. Фізичне здоров'я

Робоче місце

-Ергономіка: Забезпечте правильну організацію робочого місця. Стілець повинен бути зручною, зі спинкою, що підтримує поперек. Висота столу повинна дозволяти тримати руки на рівні ліктів.

- Монітор: Розташовуйте монітор на рівні очей, на відстані витягнутої руки, щоб зменшити напруження очей та шиї.

- Освітлення: Забезпечте достатнє освітлення, щоб уникнути перенапруження очей. Уникайте відблисків на екрані.

Режим роботи

- Перерви: Робіть регулярні короткі перерви кожні 20-30 хвилин, щоб дати очам і тілу відпочинок.

- Вправи: Виконуйте прості вправи для очей, шиї, плечей та спини.

2. Психологічне здоров'я

- Стрес: Контролюйте рівень стресу, пов'язаного з роботою. Використовуйте техніки релаксації або медитації.

- Баланс роботи та відпочинку: Дотримуйтеся здорового балансу між роботою та особистим життям.

3. Інформаційна безпека

Захист даних

- Паролі: Використовуйте складні паролі та змінюйте їх регулярно.

- Шифрування: Шифруйте важливі файли та дані.

Мережева безпека

- Брандмауер: Використовуйте брандмауери для захисту від несанкціонованого доступу.

- VPN: Використовуйте віртуальні приватні мережі (VPN) при роботі з важливою інформацією через незахищені мережі.

4. Соціальна безпека

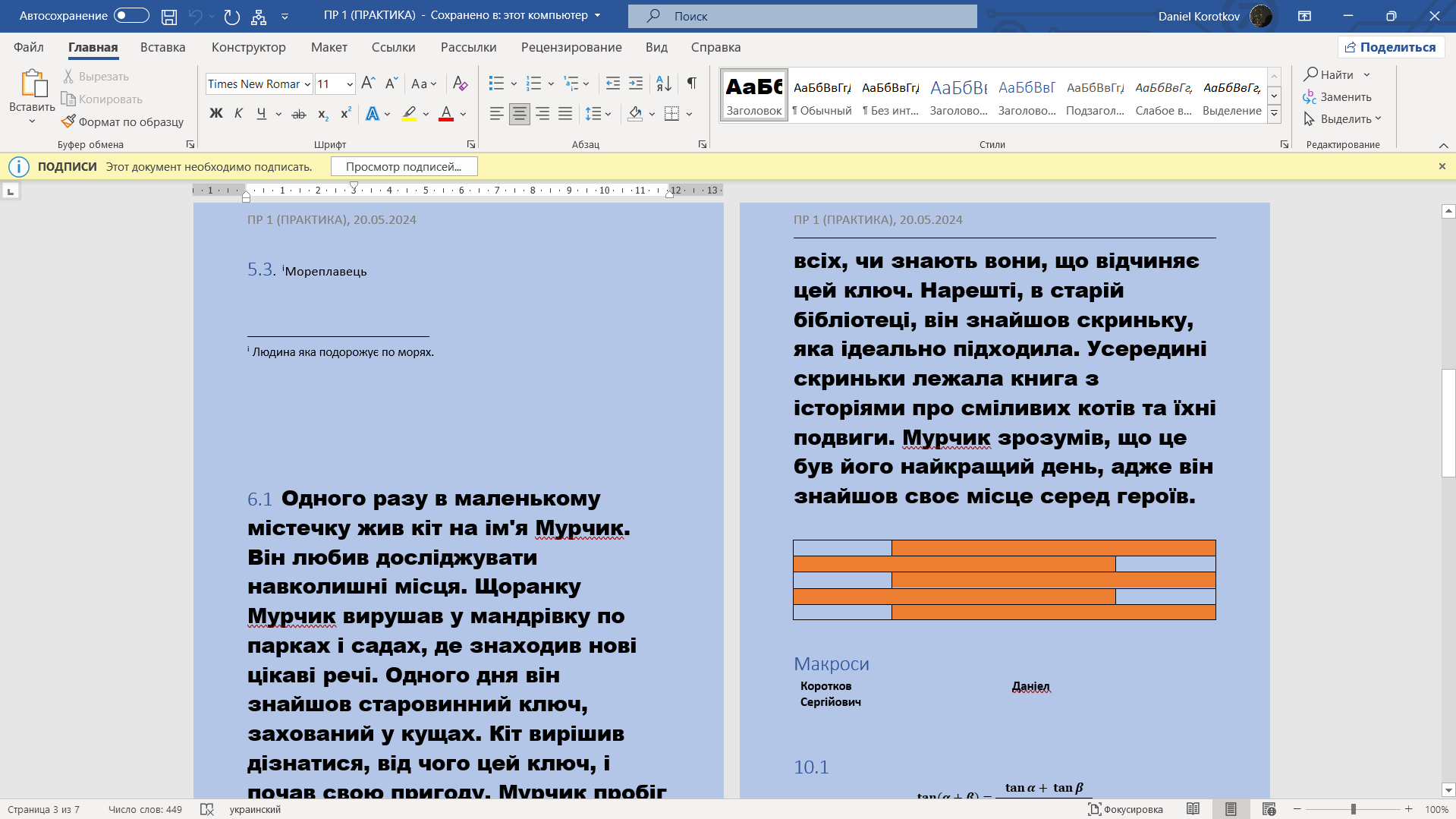
- Кібербулінг: Будьте обачними при спілкуванні в Інтернеті та реагуйте на будь-які прояви кібербулінгу.

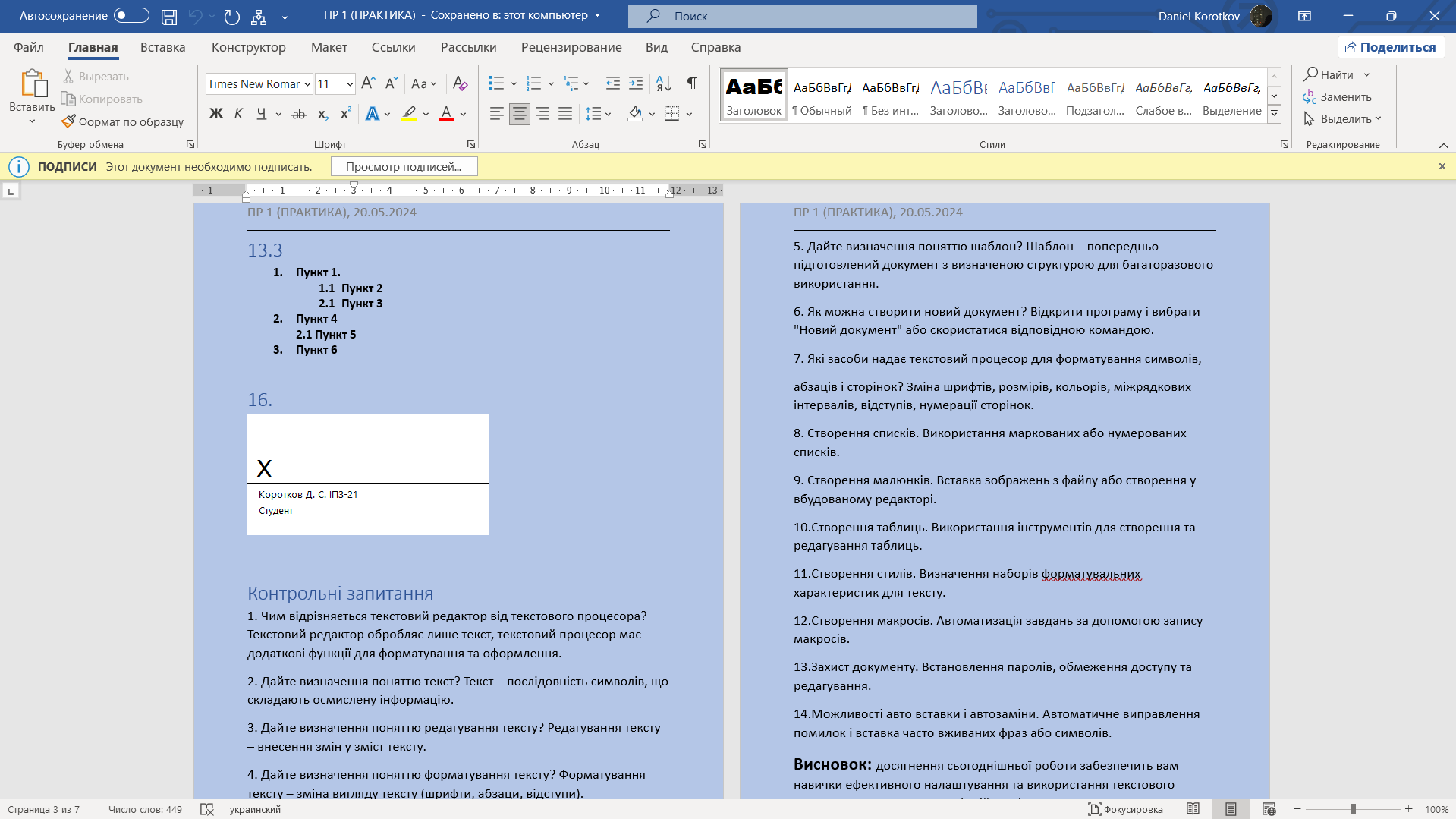
- Приватність: Обмежуйте кількість особистої інформації, що публікується в Інтернеті.

# 2 РОЗДІЛ "ІНФОРМАТИКА"

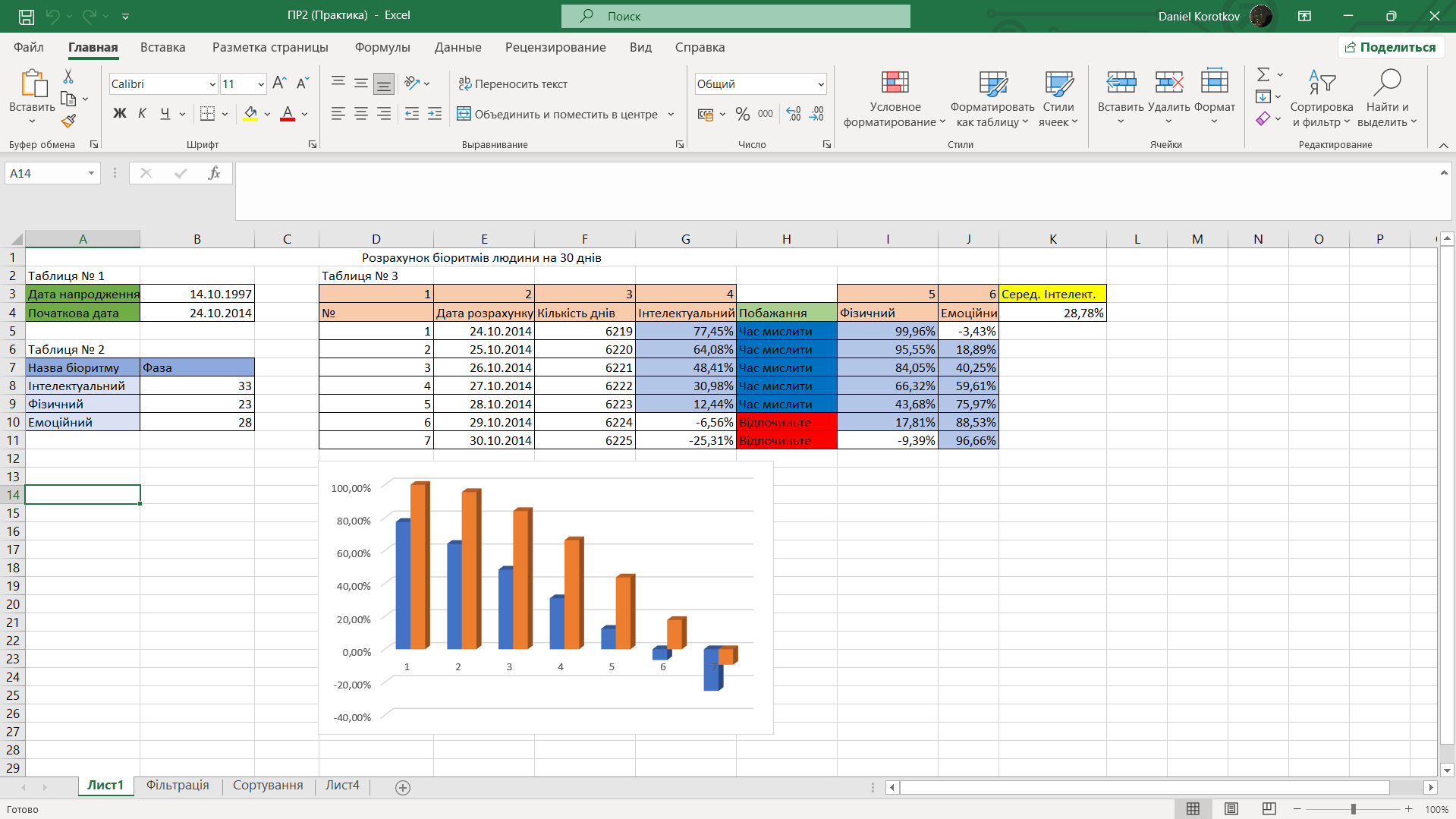
## **2.1 Розроблення засобами MS Word документа складної структури**

## 





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



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

# 3 РОЗДІЛ "ПРОГРАМУВАННЯ С++"

## **3.1 Теоретичні відомості**

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

***Змінні і типи даних***

У мові С++ передбачені вбудовані дані наступних типів:

– Цілий (int) – цілочисельні змінні (типу int, long, short) зберігають цілі значення, і можуть бути знаковими та беззнаковими. Знакові змінні можуть представляти як позитивні, так і негативні числа. Для цього в їх представленні один біт (найстарший) виділяється під знак. На відміну від них, беззнакові змінні містять тільки позитивні значення. Щоб вказати, що змінна буде беззнаковою, використовується ключове слово unsigned. За замовчуванням цілочисельні змінні вважаються знаковими (signed, найчастіше опускається; використовується при перетворенні типів даних).

– Дійсний (float та double) – для представлення чисел із плаваючою крапкою застосовують тип даних float. Цей тип, як правило, використовується для зберігання не дуже великих чисел. Якщо число може приймати більші значення, використовують змінні подвійної точності, тип double.

– Булевий (bool) – займає всього 1 байт і використовується, насамперед, у логічних операціях, тому що може приймати значення 0 (false, хибність) або відмінне від нуля (true, істина).

– Символьний (char) – (окремий випадок int) застосовується, коли змінна повинна нести інформацію про ASCII код. Цей тип даних часто використовується для побудови більш складних конструкцій, таких, як рядок, символьні масиви і т. п. Дані типу char також можуть бути знаковими й беззнаковими.

– Відсутність типу (void) – змінна типу void не має значення й служить для узгодження синтаксису. Наприклад, синтаксис вимагає, щоб функція повертала значення. Якщо не потрібно використовувати повернуте значення, перед іменем функції ставиться тип void.

– У стандарті С++11 для оголошення типу даних дозволено використовувати ключове слово auto, якщо компілятор може визначити тип змінної, виходячи із правої частини виразу. Приклад: auto i = 0; Оскільки праворуч від знаку присвоєння стоїть ціле число, компілятор може автоматично визначити тип змінної. Якщо замінити оператор на auto i; то буде видана помилка компіляції, бо неможливо

визначити тип змінної і.

***Змінні***

Правила побудови:

– Першим символом змінної C++ може бути тільки літера.

– Наступними символами ідентифікатора можуть бути літери, літери-цифри і літери-підкреслення.

– Довжина ідентифікатора необмежена (фактично довжина залежить від реалізації системи програмування).

Змінна – об'єкт програми, що займає в загальному випадку кілька байт пам'яті, покликаний зберігати дані. Щоб змінну можна було використовувати в програмі, вона повинна бути попередньо оголошена. При оголошенні змінної для неї резервується деяка область пам'яті, розмір якої залежить від конкретного типу змінної. Розмір того самого типу даних може відрізнятися на комп'ютерах різних платформ, а також може залежати від використовуваної операційної системи. Тому при оголошенні тієї або інший змінної потрібно чітко представляти, скільки байт вона буде займати в пам'яті, щоб уникнути проблем, пов'язаних з переповненням і неправильною інтерпретацією даних.

Приклад оголошення змінних:

int а, b = 0;

float age;

double ab = 55.4;

bool tipb = false;

char letter = 'Z';

***Константи***

Константи, так само як і змінні, являють собою область пам'яті для зберігання даних з тією лише відмінністю, що значення, присвоєне константі спочатку, не може бути змінене протягом виконання всієї програми.

Константи бувають літеральними і типізованими, причому літеральні константи діляться на: символьні, строкові, цілі і дійсні.

– Символьні константи представляються окремим символом, укладеним в одинарні лапки (апострофи): 'е', '@', '<'.

– Рядкові константи – це послідовність символів, укладена в подвійні лапки: "Це приклад не самої довгої строкової константи!".

– Цілі константи бувають наступних форматів: десяткові; восьмирічні; шістнадцятирічні.

– Десяткові можуть бути представлені як послідовність цифр (від 0 до 9), що починається не з нуля, наприклад: 123; 2384.

***Коментарі***

Існує два типи коментарів: однорядковий та багаторядковий.

Приклад:

//це однорядковий коментар до кінця поточного рядка

/\*це багаторядковий коментар, який може включати декілька

рядків тексту

або операторів\*/

### 

### 3.1.2 Арифметичні операції

**Додавання (+)**

Додає два числа.

int a = 5;

int b = 3;

int sum = a + b;

**Віднімання (-)**

Віднімає одне число від іншого.

int a = 5;

int b = 3;

int difference = a - b;

**Множення (\*)**

Множить два числа.

int a = 5;

int b = 3;

int product = a \* b;

**Ділення (/)**

Ділить одне число на інше. Важливо пам'ятати про типи даних, оскільки цілочислове ділення відкидає дробову частину.

int a = 10;

int b = 3;

int quotient = a / b;

double x = 10.0;

double y = 3.0;

double result = x / y;

### 

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

Вхідний потік std::cin використовується для введення даних з клавіатури. Оператором >> (stream extraction operator) ми можемо зчитувати значення різних типів.

#include <iostream>

int main() {

int number;

std::cout << "Введіть число: ";

std::cin >> number;

std::cout << "Ви ввели число: " << number << std::endl;

return 0;

}

Вихідний потік std::cout використовується для виведення даних на екран. Оператором << (stream insertion operator) ми можемо виводити значення різних типів.

#include <iostream>

int main() {

int number = 42;

std::cout << "Число: " << number << std::endl;

return 0;

}

**Найпростіші математичні функції в C++**

C++ підтримує багато математичних функцій через стандартну бібліотеку <cmath>. Ось деякі з найпоширеніших математичних функцій:

sqrt (обчислення квадратного кореня),

pow (піднесення до степеня), abs (абсолютне значення),

sin, cos, tan (тригонометричні функції)

log та log10 (логарифми).

### 3.1.4 Програмування лінійних та розгалужених обчислювальних процесів

Лінійний обчислювальний процес означає, що всі інструкції виконуються послідовно одна за одною.

#include <iostream>

int main() {

int a = 5;

int b = 3;

int sum = a + b;

std::cout << "Сума: " << sum << std::endl;

return 0;

}

Розгалужені обчислювальні процеси використовують умовні оператори для прийняття рішень. У мові C++ основними умовними операторами є if, else if, else та switch.

#include <iostream>

int main() {

int a = 5;

int b = 3;

if (a > b) {

std::cout << "a більше за b" << std::endl;

} else if (a < b) {

std::cout << "a менше за b" << std::endl;

} else {

std::cout << "a дорівнює b" << std::endl;

}

return 0;

}

### 

### 3.1.5 Програмування циклічних обчислювальних процесів

Цикл for використовується для виконання блоку коду фіксовану кількість разів. Він складається з трьох частин: ініціалізація, умова і оновлення.

#include <iostream>

int main() {

// Виведення чисел від 1 до 10

for (int i = 1; i <= 10; ++i) {

std::cout << i << " ";

}

std::cout << std::endl;

return 0;

}

Цикл while виконує блок коду, доки умова залишається істинною. Він може використовуватися, коли кількість ітерацій не визначена заздалегідь.

#include <iostream>

int main() {

int i = 1;

// Виведення чисел від 1 до 10

while (i <= 10) {

std::cout << i << " ";

++i;

}

std::cout << std::endl;

return 0;

}

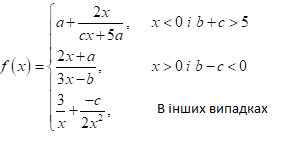
## **3.2 Практичне завдання**

### 3.2.1 Завдання

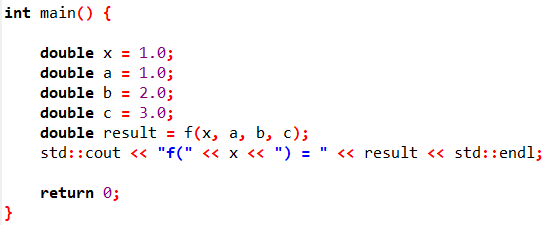
Обчислити і вивести на екран у табличному вигляді значення функції *f(x)* на заданому інтервалі зміни значень аргументу *х* від *xпоч* до *xкін* з кроком *h.*

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

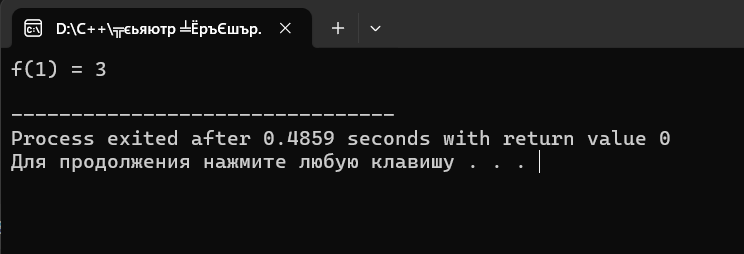
### 3.2.2 Постановка задачі



Вхідні дані



Вихідні дані



Блок-схема алгоритму роботи програми представлена на рис. ххх

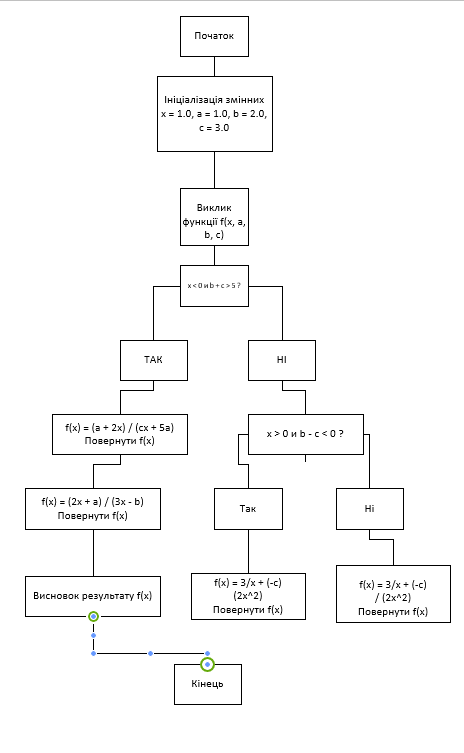


Рисунок хх - Блок-схема алгоритму роботи програми

### 3.2.3 Код програми

#include <iostream>

#include <cmath>

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

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

return (a + 2 \* x) / (c \* x + 5 \* a);

} else if (x > 0 && b - c < 0) {

return (2 \* x + a) / (3 \* x - b);

} else {

return 3 / x + (-c) / (2 \* x \* x);

}

}

int main() {

double x = 1.0;

double a = 1.0;

double b = 2.0;

double c = 3.0;

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

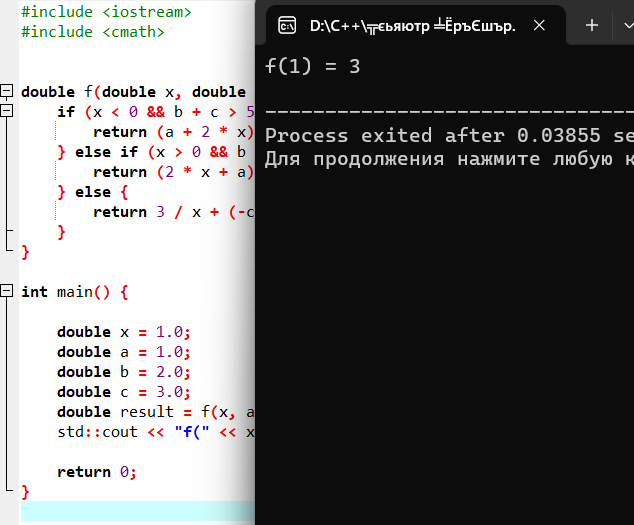
std::cout << "f(" << x << ") = " << result << std::endl;

return 0;

}

### 3.2.4 Скріншот виконання програми

Скріншот виконання програми представлена на рис. ххх

 Рисунок хх - Скріншот виконання програми

## **3.3 Виконання тестів**

Виконано

# 4 РОЗДІЛ " ПРАКТИЧНА РОБОТА З С++"

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

## Введення

Для введення даних з консолі використовують об'єкт std::cin.

#include <iostream>

int main() {

int number;

std::cout << "Enter an integer: ";

std::cin >> number; // Введення цілого числа з консолі

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

return 0;

}

У цьому прикладі:

* std::cout використовується для виведення рядка "Enter an integer: ".
* std::cin чекає введення користувачем цілого числа, яке зберігається у змінну number.
* std::cout використовується для виведення введеного числа.

Виведення

Для виведення даних на консоль використовують об'єкт std::cout.

#include <iostream>

int main() {

std::cout << "Hello, world!" << std::endl; // Виведення рядка на консоль

return 0;

}

У цьому прикладі:

* std::cout використовується для виведення рядка "Hello, world!".
* std::endl використовується для переведення курсора на новий рядок.

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

В мові програмування С++ існує безліч структур даних, які допомагають ефективно управляти станом програми. Ось деякі з найбільш поширених структур даних, які використовуються для цього:

Масиви - це базова структура даних, яка зберігає фіксовану кількість елементів одного типу.

#include <iostream>

int main() {

int arr[5] = {1, 2, 3, 4, 5};

for (int i = 0; i < 5; ++i) {

std::cout << arr[i] << " ";

}

return 0;

}

Вектори - це динамічні масиви, які можуть змінювати свій розмір під час виконання програми. Вектори є частиною стандартної бібліотеки шаблонів (STL).

#include <iostream>

#include <vector>

int main() {

std::vector<int> vec = {1, 2, 3, 4, 5};

vec.push\_back(6); // Додає елемент до кінця вектора

for (int i = 0; i < vec.size(); ++i) {

std::cout << vec[i] << " ";

}

return 0;

}

Списки - це зв'язані списки, які можуть бути як однозв'язаними, так і двозв'язаними. Вони також є частиною STL.

#include <iostream>

#include <list>

int main() {

std::list<int> lst = {1, 2, 3, 4, 5};

lst.push\_back(6); // Додає елемент до кінця списку

for (auto it = lst.begin(); it != lst.end(); ++it) {

std::cout << \*it << " ";

}

return 0;

}

Черги - це структури даних, які працюють за принципом FIFO (First In, First Out). Вони є частиною STL.

#include <iostream>

#include <queue>

int main() {

std::queue<int> q;

q.push(1);

q.push(2);

q.push(3);

while (!q.empty()) {

std::cout << q.front() << " ";

q.pop();

}

return 0;

}

Стеки - це структури даних, які працюють за принципом LIFO (Last In, First Out). Вони також є частиною STL.

#include <iostream>

#include <stack>

int main() {

std::stack<int> s;

s.push(1);

s.push(2);

s.push(3);

while (!s.empty()) {

std::cout << s.top() << " ";

s.pop();

}

return 0;

}

Множини - це структури даних, які зберігають унікальні елементи в певному порядку. Вони є частиною STL.

#include <iostream>

#include <set>

int main() {

std::set<int> s = {3, 1, 4, 1, 5, 9};

for (auto it = s.begin(); it != s.end(); ++it) {

std::cout << \*it << " ";

}

return 0;

}

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

## Читання з файлу

Для читання з файлу використовується клас ifstream.

#include <iostream>

#include <fstream>

#include <string>

int main() {

std::ifstream inputFile("example.txt"); // Відкриття файлу для читання

if (!inputFile.is\_open()) {

std::cerr << "Could not open the file!" << std::endl;

return 1;

}

std::string line;

while (std::getline(inputFile, line)) { // Читання файлу по рядках

std::cout << line << std::endl;

}

inputFile.close(); // Закриття файлу

return 0;

}

Запис у файл

Для запису у файл використовується клас ofstream.

#include <iostream>

#include <fstream>

int main() {

std::ofstream outputFile("output.txt");

if (!outputFile.is\_open()) {

std::cerr << "Could not open the file!" << std::endl;

return 1;

}

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

outputFile << "This is a line of text." << std::endl;

outputFile.close();

return 0;

}

Комбіноване читання та запис

Для одночасного читання і запису використовується клас fstream.

#include <iostream>

#include <fstream>

#include <string>

int main() {

std::fstream file("combined.txt", std::ios::in | std::ios::out | std::ios::app);

if (!file.is\_open()) {

std::cerr << "Could not open the file!" << std::endl;

return 1;

}

file << "Appending a new line to the file." << std::endl;

file.seekg(0, std::ios::beg);

std::string line;

while (std::getline(file, line)) {

std::cout << line << std::endl;

}

file.close(); // Закриття файлу

return 0;

}

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

TXT файли зберігають текстову інформацію, кожен рядок відокремлюється символом нового рядка.

#include <iostream>

#include <fstream>

#include <string>

int main() {

std::ofstream outFile("example.txt");

if (outFile.is\_open()) {

outFile << "This is a line of text.\n";

outFile << "This is another line of text.\n";

outFile.close();

} else {

std::cerr << "Unable to open file for writing.\n";

}

std::ifstream inFile("example.txt");

if (inFile.is\_open()) {

std::string line;

while (std::getline(inFile, line)) {

std::cout << line << '\n';

}

inFile.close();

} else {

std::cerr << "Unable to open file for reading.\n";

}

return 0;

}

CSV файли зберігають табличні дані, де кожен рядок відповідає одному запису, а значення відокремлюються комами.

#include <iostream>

#include <fstream>

#include <sstream>

#include <vector>

#include <string>

int main() {

std::ofstream outFile("example.csv");

if (outFile.is\_open()) {

outFile << "Name,Age,Occupation\n";

outFile << "Alice,30,Engineer\n";

outFile << "Bob,25,Designer\n";

outFile.close();

} else {

std::cerr << "Unable to open file for writing.\n";

}

std::ifstream inFile("example.csv");

if (inFile.is\_open()) {

std::string line;

while (std::getline(inFile, line)) {

std::stringstream ss(line);

std::string item;

std::vector<std::string> row;

while (std::getline(ss, item, ',')) {

row.push\_back(item);

}

for (const auto& col : row) {

std::cout << col << " ";

}

std::cout << '\n';

}

inFile.close();

} else {

std::cerr << "Unable to open file for reading.\n";

}

return 0;

}

Для роботи з JSON файлами в C++ зазвичай використовують сторонні бібліотеки, такі як [nlohmann/json](https://github.com/nlohmann/json).

#include <iostream>

#include <fstream>

#include <nlohmann/json.hpp>

using json = nlohmann::json;

int main() {

json j;

j["name"] = "John";

j["age"] = 30;

j["city"] = "New York";

std::ofstream outFile("example.json");

if (outFile.is\_open()) {

outFile << j.dump(4);

outFile.close();

} else {

std::cerr << "Unable to open file for writing.\n";

}

std::ifstream inFile("example.json");

if (inFile.is\_open()) {

json j\_in;

inFile >> j\_in;

std::cout << j\_in.dump(4) << '\n';

inFile.close();

} else {

std::cerr << "Unable to open file for reading.\n";

}

return 0;

}

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

Системи контролю версій (VCS) є важливим інструментом для керування змінами у коді, особливо у великих проєктах на C++. Вони дозволяють відслідковувати зміни, працювати спільно з іншими розробниками, відновлювати попередні версії коду і управляти гілками розробки. Найбільш популярною системою контролю версій є Git, але також існують інші системи, такі як Subversion (SVN), Mercurial і Perforce.

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

Мова Markdown є легковажною мовою розмітки, яка використовується для написання форматованого тексту за допомогою звичайного текстового редактора. Вона широко використовується для створення документації, опису проєктів та інших текстових документів завдяки простоті використання та зручності.

**Основи Markdown**

Ось основні елементи Markdown, які можна використовувати для опису проєктів:

**Заголовки**

Заголовки створюються за допомогою символу #. Кількість # визначає рівень заголовка (від 1 до 6).

**Абзаци**

Просто введіть текст в новому рядку для створення абзацу. Між абзацами повинні бути порожні рядки.

**Списки**

**Ненумеровані списки**

Для створення ненумерованого списку використовуються -, + або \*.

**Нумеровані списки**

Для створення нумерованого списку використовуйте числа з крапками.

**Посилання**

Посилання створюються за допомогою квадратних дужок для тексту посилання і круглих дужок для URL.

**Зображення**

Зображення додаються так само, як і посилання, але перед квадратними дужками ставиться знак оклику.

**Коди та блоки коду**

Для виділення коду в рядку використовуються зворотні апострофи `.

## **4.7 Розроблення ігрової програми**

### 4.7.1 Код програми

#include <iostream>

using namespace std;

char h[9] = {'-', '-', '-', '-', '-', '-', '-', '-', '-'};

void cleanboard()

{

for (int i = 0; i < 50; i++)

{

cout << "\n";

}

}

void board1()

{

cout << "\t\t\t\*\*\*\*\*\*\*\*\*\*\*" <<endl;

cout << "\t\t\t-7-|-8-|-9-"<<endl;

cout << "\t\t\t-4-|-5-|-6-"<<endl;

cout << "\t\t\t-1-|-2-|-3-"<<endl;

cout << "\t\t\t\*\*\*\*\*\*\*\*\*\*\*" <<endl;

}

void board()

{

cout << "\n";

cout << "\t\t\t\*\*\*\*\*\*\*\*\*\*\*" <<endl;

cout << "\t\t\t-" << h[6] << "-|-"<< h[7] << "-|-" << h[8] << "-"<<endl;

cout << "\t\t\t-" << h[3] << "-|-"<< h[4] << "-|-" << h[5] << "-"<<endl;

cout << "\t\t\t-" << h[0] << "-|-"<< h[1] << "-|-" << h[2] << "-"<<endl;

cout << "\t\t\t\*\*\*\*\*\*\*\*\*\*\*" <<endl;

}

int get\_move()

{

int move;

cin>>move;

while (move > 9 || move < 1 || h[move - 1] != '-')

{

cout << "ERROR!!!!" <<endl;

cin >> move;

}

return move;

}

int main()

{

board1();

for (int i = 0; i < 9; i++)

{

cleanboard();

board1();

board();

int move = get\_move();

cout << "Hod: "<< move <<endl;

if (i%2 == 0)

{

h [move - 1] = 'X';

}

else h[move - 1] = 'O';

}

return 0;

}

### 4.7.2 Скріншоти виконання програми



# 5 РОЗДІЛ ”ВЕБ-ДИЗАЙН ТА HTML”

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

<html> - позначає початок і кінець HTML-документа.

<head> - містить метадані веб-сторінки.

<title> - задає заголовок веб-сторінки, який відображається в заголовку браузера.

<body> - визначає основний вміст веб-сторінки.

<a> - створює посилання на іншу веб-сторінку або документ.

<main> - визначає головний вміст документа.

<h1> до <h6> - визначає заголовки різного рівня на веб-сторінці .

<thead> - задає заголовок таблиці.

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

## **Зображення**

Для вставки зображення на веб-сторінку використовується тег <img>.

 src: вказує шлях до зображення. Це може бути URL або відносний шлях на вашому сервері.

 alt: це текстовий опис зображення, який з'являється, коли зображення не може бути відображене або коли користувач використовує програму читання екрана.

**Посилання**

Для створення посилання використовується тег <a>.

 href: це атрибут, що вказує URL або шлях до сторінки або ресурсу, на який веде посилання.

 Текст між відкриваючим і закриваючим тегами <a> (Текст посилання в прикладі) відображається як посилання на сторінці.

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

Каскадні таблиці стилів CSS (Cascading Style Sheets) дозволяють задавати вигляд і відображення елементів HTML на веб-сторінці.

**Структура каскадних таблиць стилів CSS**

**Селектори**: Визначають, які елементи HTML будуть стилізовані. Селектори можуть бути елементами HTML, класами, ідентифікаторами або псевдокласами.

body {

font-family: Arial, sans-serif;

background-color: #f0f0f0;

}

.header {

background-color: #333;

color: #fff;

padding: 10px;

}

#main-content {

width: 80%;

margin: 0 auto;

}

a:hover {

text-decoration: underline;

}

**Принципи каскадування**

Каскадні таблиці стилів CSS дозволяють каскадування стилів, що означає, що стилі можуть бути успадковані або перевизначені відповідно до специфічності селекторів. Основні принципи каскадування включають:

* **Порядок визначення**: Стилі, визначені пізніше у коді, перевизначають стилі, визначені раніше, якщо вони використовують однакові селектори.
* **Специфичність селекторів**: Деякі селектори мають більшу специфічність, ніж інші. Наприклад, ідентифікатори мають більшу вагу, ніж класи або елементи.
* **Вага властивостей**: Якщо є конфлікт між стилями (наприклад, один і той самий елемент має різні значення для однієї властивості), використовується стиль з більшою вагою або специфічністю.

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

Спадковість і каскадування є ключовими концепціями у CSS, які дозволяють

ефективно стилізувати і контролювати вигляд елементів на веб-сторінці. Давайте розглянемо кожну з цих концепцій детальніше:

**Спадковість**

Спадковість в CSS означає, що деякі властивості стилів елемента можуть успадковуватись від його батьківського елемента. Це спрощує створення та зміну стилів, оскільки не потрібно вказувати однакові властивості для кожного дочірнього елемента окремо.

**Приклади успадкування:**

1. **\*\*Текстові властивості\*\***: Наприклад, властивості `font-family`, `font-size`, `font-weight`, `font-style`, `color` успадковуються від батьківського елемента до його дочірніх елементів.

2. **\*\*Властивості вирівнювання тексту\*\***: Наприклад, `text-align`, `line-height`, `text-indent` також успадковуються.

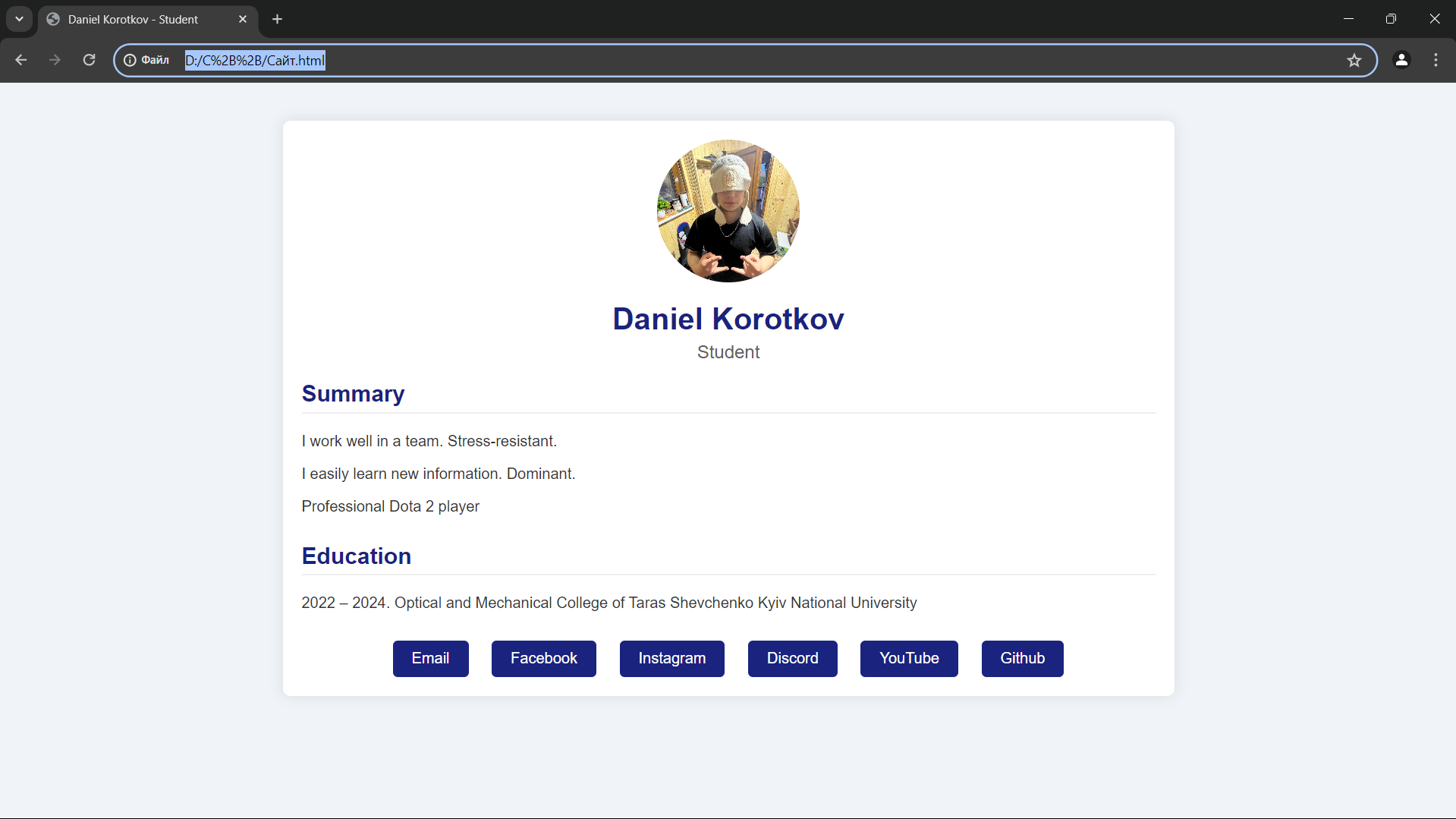
3. **\*\*Певні бордери та поля\*\***: Наприклад, `border-collapse`, `border-spacing`, `caption-side`, `empty-cells`, `table-layout`, `white-space`, `vertical-align`

## **5.5 Практична частина**

### 5.5.1 Постановки задачі

Наше завдання було створення веб-сайту у формі резюме. Мета роботи – вивчення мови HTML.

### 5.5.2 HTML документ



# ВИСНОВКИ

Вивчення HTML, CSS та C++ стало важливим етапом у моєму розвитку як програміста. Ці технології надали мені фундаментальні знання, необхідні для створення та стилізації веб-сторінок, а також для розробки програмних додатків. HTML допоміг мені зрозуміти, як структурувати веб-контент, CSS – як ефективно стилізувати і покращувати зовнішній вигляд веб-сторінок, а C++ – як розробляти ефективні та потужні програми. Вивчення цих тем значно розширило мої навички і дало міцну основу для подальшого вдосконалення у сфері програмування та веб-розробки.

# 

# ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ ІНФОРМАЦІЇ

1. ГОСТ 2.105-95 ЕСКД Общие требования к текстовым документам.

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

3. Cplusplus.com

4. GeeksforGeeks.com

5. Tutorialspoint.com

6. CppReference.com

7. MDN Web Docs.com

8. W3Schools.com

9. CSS-Tricks.com

10. [**Stack Overflow**](https://stackoverflow.com/)**.com**

# ДОДАТКИ