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: Возний Артем |
| студента 2-го курсу групи ІПЗ-21 |
| прізвище ім‘я по-батькові  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: Возний Артем Вікторович

Дата: 17.05.24 – 13.06.24

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. Алгоритмізація обчислювальних процесів. Розробка постановки задачі, визначення вхідних та вихідних даних, оформлення блок-схем.

14. Типи даних в мові С/С++, внутрішнє представлення даних, операції перетворення типів даних.

15. Оператор присвоювання. Операції введення- виведення даних

16. Програмування лінійних та розгалужених обчислювальних процесів

17. Програмування циклічних обчислювальних процесів

18. Складання програм

19. Виконання тестів

20. Введення та виведення у базових консольних програмах

21. Структури даних для управління станом програми

22. Робота з файлами: запис та читання

23. Формати файлів та їх обробка

24. Системи контролю версій (VCS)

25. Мова Markdown для опису проекті

26. Розроблення ігрової програми

27. Мова розмітки HTML та структура Веб-сторінки

28. Розмітка тексту за допомогою HTML. Зображення та посилання

29. Каскадні таблиці стилів CSS та їх структура

30. Спадковості і каскадування за допомогою CSS

31. Розробка та оформлення Постановки задачі

32. Розробка HTML документу

33. Систематизація матеріалів, оформлення звіту з навчальної практики

# 1 БЕЗПЕКА ЖИТТЄДІЯЛЬНОСТІ ПРИ РОБОТІ З КОМП’ЮТЕРОМ

# 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

### 

### 3.1.2 Арифметичні операції

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

### 3.1.4 Програмування лінійних та розгалужених обчислювальних процесів

### 3.1.5 Програмування циклічних обчислювальних процесів

## **3.2 Практичне завдання**

### 3.2.1 Завдання

### 3.2.2 Постановка задачі

Вхідні дані

Вихідні дані

Блок-схема алгоритму роботи програми представлена на рис. ххх

Блок-схема

Рисунок хх - Блок-схема алгоритму роботи програми

### 3.2.3 Код програми

### 3.2.4 Скріншот виконання програми

Скріншот виконання програми представлена на рис. ххх

Скріншот

Рисунок хх - Скріншот виконання програми

## **3.3 Виконання тестів**

# 4 РОЗДІЛ " ПРАКТИЧНА РОБОТА З С++"

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

### Оператори введення та виведення в програмуванні

В програмуванні під **оператором** розуміють інструкцію, що вказує комп’ютеру виконати певні дії. Оператори можуть бути **простими** (одна інструкція) або **складеними** (декілька інструкцій, об'єднаних в одну).

**Складені оператори** формуються за допомогою ключових слів begin..end в мові Pascal, або фігурних дужок {} в мовах Java та C/C++. Усі оператори, що містяться всередині цих блоків, сприймаються як одна інструкція.

Одними з найпоширеніших операторів в консольних програмах є оператори **введення** та **виведення**.

### Введення та виведення в C++

У мові C++ доступні різні методи введення і виведення даних. Одним із них є використання бібліотеки stdio, але частіше використовують **потокове введення та виведення** з бібліотеки iostream:

cpp

Копировать код

#include <iostream>

Ця бібліотека містить простір імен std, що включає в себе оператор cout для виведення на консоль. Синтаксис оператора cout виглядає наступним чином:

cpp

Копировать код

std::cout << operand1 << operand2 << operand3 << ...;

Кожен елемент для виведення записується через оператор << (два знаки менше). Наприклад:

cpp

Копировать код

std::cout << "Привіт " << 31;

std::cout << x << y << z;

Зверніть увагу, що за замовчуванням пробіли між елементами не вставляються, тому їх треба додавати вручну:

cpp

Копировать код

std::cout << x << " " << y << " " << z;

### Особливості виведення

Мови програмування, такі як Java та C/C++, підтримують **керуючі послідовності символів** у рядках, які починаються зі символа \ (backslash):

* \n – новий рядок
* \t – табуляція
* \\ – символ \
* \" – символ "
* \' – символ '
* \% – символ %
* \? – символ ?

### Введення даних

Коли програмі необхідно отримати дані від користувача, наприклад, з клавіатури, використовуються стандартні оператори введення. У C++ є можливість використання форматного введення з допомогою функції scanf, але частіше використовують потокове введення через оператор cin з бібліотеки iostream:

cpp

Копировать код

std::cin >> змінна1 >> змінна2 >> змінна3;

Синтаксис cin виглядає так: кожен наступний елемент введення записується через оператор >> (два знаки більше). Імена змінних вводяться без символа &.

### Приклад використання

cpp

Копировать код

#include <iostream>

int main() {

std::string прізвище;

int вік;

std::cout << "Введіть ваше прізвище: ";

std::cin >> прізвище;

std::cout << "Введіть ваш вік: ";

std::cin >> вік;

std::cout << "Пане " << прізвище << ", вам " << вік << " років." << std::endl;

std::cout << "Введіть x, y, z: ";

float x, y, z;

std::cin >> x >> y >> z;

return 0;

}

У цьому прикладі використовується cin для введення і cout для виведення даних на екран.

### Джерела інформації:

**1. cplusplus.com - Basic Input and Output**

http://www.cplusplus.com/doc/tutorial/basic\_io/

**2. Learn C++ - Basic Input/Output**

https://www.learncpp.com/cpp-tutorial/basic-inputoutput/

**3. W3Schools - C++ Input/Output**

https://www.w3schools.com/cpp/cpp\_input\_output.asp

**4. GeeksforGeeks - Input and Output Operators in C++**

https://www.geeksforgeeks.org/input-output-operators-in-c/

**5. Javatpoint - C++ Input/Output**

https://www.javatpoint.com/cpp-input-output

**6. Programiz - C++ Input and Output**

https://www.programiz.com/cpp-programming/input-output

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

Масив — це набір величин, що зберігаються в пам'яті під однією назвою (ідентифікатором) і мають різні порядкові номери (індекси). У програмуванні масиви часто відповідають математичним поняттям послідовності або матриці.

**Елемент масиву** — це окрема величина в масиві. Цей термін у програмуванні відповідає математичному поняттю елемента послідовності або матриці.

**Індекс масиву** — це значення, яке зазвичай є цілим числом, що вказує на конкретний елемент масиву. Це поняття відповідає математичному номеру елемента послідовності або номеру рядка/стовпця в матриці. У мові програмування C++ індексація починається з 0.

**Основні властивості масиву:**

* **Тип (типи) елементів** — тип даних, які зберігаються в масиві.
* **Назва (ідентифікатор)** — спільна для всіх елементів масиву.
* **Розмірність** — кількість індексів, необхідних для визначення розташування елемента в масиві.
* **Діапазони зміни індексів** — межі значень індексів, що використовуються для доступу до елементів масиву.

У мові C++ типи і розміри масиву не можуть змінюватися під час виконання програми. У деяких інших мовах програмування це можливо.

### Контейнер list у C++

Контейнер **list** у мові програмування C++ реалізує двонаправлений список. Цей тип списку дозволяє швидко додавати і видаляти елементи. Доступ до елементів здійснюється за допомогою вказівників (ітераторів).

* **#include <list>** — підключення необхідної бібліотеки.
* **list <тип\_даних> назва\_списку;** — оголошення списку.
* **list <тип\_даних> :: iterator назва\_ітератора;** — оголошення ітератора (вказівника).
* **l.begin()** — вказівник на початок списку l.
* **l.end()** — вказівник на кінець списку l.
* **l.rbegin()** — реверсивний вказівник на кінець списку l.
* **l.rend()** — реверсивний вказівник на початок списку l.
* **l.empty()** — перевірка, чи порожній список l.
* **l.push\_back(e)** — додавання елемента e в кінець списку l.
* **l.pop\_back()** — видалення останнього елемента зі списку l.
* **l.front()** — доступ до першого елемента списку l.
* **l.back()** — доступ до останнього елемента списку l.
* **l.insert(i, e)** — вставка елемента e на позицію i в списку l.
* **l.erase(j, k)** — видалення елементів від позиції j до позиції k у списку l.

Для отримання додаткової інформації відвідайте наш сайт: <https://chatgptonline.tech/ru/>.

Сподіваємося, ця інформація була для вас корисною! Якщо шукаєте цікаві ідеї для татуювань, відвідайте [Tattoo Ideas GPT](https://chatgpt.com/g/g-t11nuTlg3-tattoo-ideas-gpt).

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

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

Формати файлів і способи їх обробки є важливими елементами роботи з інформацією в різних галузях. Нижче розглянемо кілька популярних форматів файлів та методи їх обробки.

**Текстові файли**

1. **TXT (Plain Text File)**: Простий текстовий файл, який не містить форматування. Використовується для зберігання сирого тексту.
   * **Обробка**: Можна редагувати за допомогою будь-якого текстового редактора (наприклад, Notepad, Sublime Text). Також підтримується програмна обробка мовами програмування, такими як Python, Java, C++ та інші.
2. **CSV (Comma-Separated Values)**: Формат для зберігання табличних даних, де значення розділені комами.
   * **Обробка**: Відкривається у програмах для роботи з електронними таблицями (наприклад, Excel, Google Sheets). Підтримується також програмна обробка мовами Python (з використанням бібліотеки pandas), R тощо.

mathematica

Копировать код

Date,Level,Message

2023-06-07,INFO,Application started

2023-06-07,ERROR,Failed to connect to database

2023-06-08,INFO,User logged in

1. **JSON (JavaScript Object Notation)**: Легкий формат обміну даними, зручний для читання і написання як людиною, так і програмами.

json

Копировать код

{

"team": "Warriors",

"wins": 50,

"losses": 20,

"players": [

{

"name": "Stephen Curry",

"position": "Point Guard",

"points\_per\_game": 29.1

},

{

"name": "Klay Thompson",

"position": "Shooting Guard",

"points\_per\_game": 22.3

}

]

}

* + **Обробка**: Підтримується програмами на JavaScript, Python (з використанням бібліотеки json), Java та іншими мовами програмування.

1. **XML (eXtensible Markup Language)**: Формат для зберігання та передачі структурованих даних.

xml

Копировать код

<team>

<name>Warriors</name>

<wins>50</wins>

<losses>20</losses>

<players>

<player>

<name>Stephen Curry</name>

<position>Point Guard</position>

<points\_per\_game>29.1</points\_per\_game>

</player>

<player>

<name>Klay Thompson</name>

<position>Shooting Guard</position>

<points\_per\_game>22.3</points\_per\_game>

</player>

</players>

</team>

* + **Обробка**: Використання парсерів, таких як lxml у Python або DOM у Java, редагування за допомогою спеціалізованих редакторів (наприклад, XMLSpy).

**Використання JSON**

Уявімо, що ми розробляємо веб-додаток для відстеження статистики спортивних команд, наприклад, команд NBA. У додатку необхідно зберігати інформацію про команди, їхні перемоги, поразки та гравців з їхніми індивідуальними статистиками. JSON є ідеальним форматом для цієї задачі. Він дозволяє зберігати вкладені структури даних, що зручно для представлення складних об'єктів. Крім того, JSON легко обробляється мовами програмування, особливо JavaScript, який широко використовується на клієнтській стороні веб-додатків.

**Використання CSV**

Розглянемо інший випадок: адміністратор системи потребує зберігати логи роботи додатку. Ці логи включають дату і час події, рівень важливості (наприклад, інформація або помилка) та повідомлення. CSV є найкращим вибором для цього завдання. Лог-файли зазвичай складаються з багатьох рядків, кожен з яких містить однаковий набір полів. CSV файли легко читати і записувати, вони займають мінімум місця і зручно імпортуються в електронні таблиці (наприклад, Excel або Google Sheets) для подальшого аналізу.

**Робота з файлами у C++**

У мові програмування C++ для роботи з файлами формату XML, CSV та JSON використовуються різні бібліотеки.

* **Читання/запис JSON файлів**: Використовується бібліотека nlohmann/json.
* **Читання/запис CSV файлів**: Використовується бібліотека rapidcsv.
* **Читання/запис XML файлів**: Використовується бібліотека pugixml.

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

Системи контролю версій (VCS) — це програмні інструменти, які допомагають відстежувати зміни у вихідному коді або інших типах файлів під час розробки програмного забезпечення. Основні функції VCS включають відстеження змін, створення резервних копій, злиття змін, а також полегшення спільної роботи над проектами.

#### Основні типи VCS

1. **Локальні VCS**:
   * **Опис**: Зберігають версії файлів на локальному комп'ютері.
   * **Особливості**: Простота використання, обмежена можливість командної роботи.
   * **Приклад**: RCS (Revision Control System).
2. **Централізовані VCS (CVCS)**:
   * **Опис**: Використовують центральний сервер для зберігання всіх версій файлів.
   * **Особливості**: Підтримка спільної роботи над проектами, залежність від центрального сервера.
   * **Приклад**: SVN (Subversion), CVS (Concurrent Versions System).
3. **Розподілені VCS (DVCS)**:
   * **Опис**: Кожен розробник має повну копію історії змін проекту на своєму комп'ютері.
   * **Особливості**: Гнучкість, стійкість до збоїв центрального сервера.
   * **Приклад**: Git, Mercurial.

#### Переваги використання VCS

* **Історія змін**: Відстеження всіх змін у файлах з можливістю перегляду попередніх версій.
* **Відновлення**: Можливість повернення до будь-якої попередньої версії файлу.
* **Спільна робота**: Легке управління змінами від різних розробників.
* **Відгалуження та злиття**: Створення окремих гілок для нових функцій або виправлення помилок та їх подальше злиття з основною гілкою.

#### Гілки (Branches) у Git

**Гілка** у Git — це окрема версія проекту, яка існує незалежно від інших версій. Гілки дозволяють розробникам працювати над різними завданнями одночасно, не заважаючи роботі один одного. Основна гілка зазвичай називається main або master. Коли створюється нова гілка, вона є копією основної гілки або іншої гілки, від якої вона була створена.

**Коміт (Commit)** у Git — це зафіксований знімок змін у проекті. Коміт створюється командою git commit, що зберігає поточний стан змін у репозиторії. Кожен коміт має унікальний ідентифікатор (хеш), метадані (автора, дату) та повідомлення, яке описує зміни.

### Git Flow

Git Flow — це методологія управління розробкою програмного забезпечення з використанням Git. Вона допомагає організувати роботу над проектом, розділяючи його на різні гілки, які мають свої завдання та життєві цикли. Цей підхід передбачає використання кількох основних і допоміжних гілок для легкого управління новими функціями, релізами і виправленнями багів.

#### Основні гілки Git Flow

* **Main (або Master)**: Гілка, яка завжди містить стабільну версію коду, готову до випуску. Всі релізи проекту йдуть з цієї гілки.
* **Develop**: Гілка, де ведеться активна розробка. Всі нові функції, вдосконалення і виправлення помилок спочатку додаються до цієї гілки. Після завершення і тестування зміни потрапляють до main.

#### Допоміжні гілки Git Flow

* **Feature гілки**: Використовуються для розробки нових функцій. Кожна нова функція розробляється в окремій гілці, що створюється від develop. Після завершення роботи над функцією, ця гілка зливається з develop.
* **Release гілки**: Використовуються для підготовки нового релізу. Коли гілка develop готова до випуску, створюється гілка release. У цій гілці проводиться остаточне тестування і виправлення незначних помилок. Після завершення release зливається з main (створюючи новий реліз) та з develop.
* **Hotfix гілки**: Використовуються для термінового виправлення критичних багів. Створюються від main, і після виправлення змін зливаються з main і develop.

Git Flow забезпечує структурований підхід до розробки, що дозволяє командам ефективно співпрацювати, впроваджувати нові функції, готувати релізи та швидко реагувати на проблеми, зберігаючи стабільність і якість коду.

### Основні команди Git

Git є найбільш популярною розподіленою системою контролю версій і надає потужний набір команд для управління версіями.

* **Ініціалізація репозиторію**:

bash

Копировать код

git init

* **Клонування існуючого репозиторію**:

bash

Копировать код

git clone https://github.com/user/repository.git

* **Перегляд статусу файлів**:

bash

Копировать код

git status

* **Додавання файлів до індексу**:

bash

Копировать код

git add filename

git add .

* **Фіксація змін**:

bash

Копировать код

git commit -m "Your commit message"

* **Перегляд історії комітів**:

bash

Копировать код

git log

* **Створення нової гілки**:

bash

Копировать код

git branch new-branch

* **Перемикання між гілками**:

bash

Копировать код

git checkout new-branch

* **Злиття гілок**:

bash

Копировать код

git merge branch-name

* **Оновлення локального репозиторію з віддаленого**:

bash

Копировать код

git pull

* **Відправка локальних змін до віддаленого репозиторію**:

bash

Копировать код

git push

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

Markdown для опису проекту:

### Заголовки

# Заголовок 1 рівня

## Заголовок 2 рівня

### Заголовок 3 рівня

### Виділення тексту

\*Курсив\* або \_Курсив\_

\*\*Жирний\*\* або \_\_Жирний\_\_

\*\*\*Жирний курсив\*\*\* або \_\_\_Жирний курсив\_\_\_

### Списки

\*\*Нумеровані:\*\*

1. Пункт 1

2. Пункт 2

3. Пункт 3

\*\*Марковані:\*\*

- Пункт 1

- Пункт 2

- Пункт 3

### Посилання та зображення

[Текст посилання](https://example.com)

![Альтернативний текст](https://example.com/image.jpg)

### Код

\*\*Інлайн-код:\*\*

`інлайн-код`

С++

#include <iostream>

int main() {

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

return 0;

}

### Таблиці

| Заголовок 1 | Заголовок 2 |

| ----------- | ----------- |

| Рядок 1 | Дані 1 |

| Рядок 2 | Дані 2 |

### Цитати та горизонтальні лінії

\*\*Цитати:\*\*

> Це цитата

\*\*Горизонтальна лінія:\*\*

---

## **4.7 Розроблення ігрової програми**

### 4.7.1 Код програми

#include <iostream>

using namespace std;

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

char currentPlayer = 'X';

void drawBoard() {

cout << " Tic-Tac-Toe" << endl;

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

for (int i = 0; i < 3; i++) {

for (int j = 0; j < 3; j++) {

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

}

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

}

}

bool isMoveValid(int choice) {

if (choice < 1, choice > 9) {

cout << "Invalid move! Choose a number between 1 and 9." << endl;

return false;

}

int row = (choice - 1) / 3;

int col = (choice - 1) % 3;

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

cout << "Cell already occupied! Choose another cell." << endl;

return false;

}

return true;

}

bool checkWin() {

for (int i = 0; i < 3; i++) {

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

return true;

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

return true;

}

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

return true;

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

return true;

return false;

}

void switchPlayer() {

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

}

int main() {

int choice;

bool gameOver = false;

while (!gameOver) {

drawBoard();

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

cin >> choice;

if (isMoveValid(choice)) {

int row = (choice - 1) / 3;

int col = (choice - 1) % 3;

board[row][col] = currentPlayer;

if (checkWin()) {

drawBoard();

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

gameOver = true;

}

else {

switchPlayer();

}

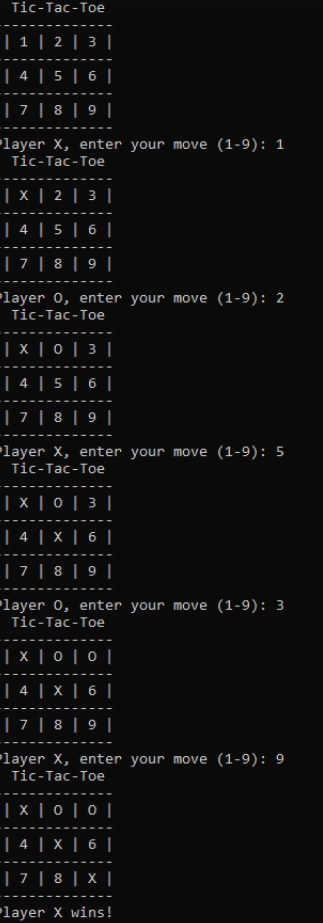
}

}

return 0;

}

### 4.7.2 Скріншоти виконання 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 Практична частина**

### 5.5.1 Постановки задачі

### 5.5.2 HTML документ

# ВИСНОВКИ

# 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.

# ДОДАТКИ