**ВІДОКРЕМЛЕНИЙ СТРУКТУРНИЙ ПІДРОЗДІЛ  
«ОПТИКО-МЕХАНІЧНИЙ ФАХОВИЙ КОЛЕДЖ КИЇВСЬКОГО  
НАЦІОНАЛЬНОГО УНІВЕРСИТЕТУ ІМЕНІ ТАРАСА ШЕВЧЕНКА»**Циклова комісія програмування та інформаційних технологій

**ЗВІТ**

**З НАВЧАЛЬНОЇ ПРАКТИКИ**

спеціальність *121 Інженерія програмного забезпечення*освітньо-кваліфікаційний рівень *«Фаховий молодший бакалавр»*

|  |  |
| --- | --- |
| ОЦІНКА  (оцінка цифрою та прописом)  Керівники практики від коледжу:  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (підпис) | Виконав:  студентка 2-го курсу групи ІПЗ-23  *Ющенко Михайло Олександрович*  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (підпис виконавця)  « \_» 20 року |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (підпис) |  |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (підпис) |  |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (підпис) |  |

« \_»\_ 2024 року

КИЇВ — 2024

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

на період *Навчальної* практики

Ющенка Михайла Олександровича

Спеціальність *121 Інженерія програмного забезпечення* Група *ІПЗ-23*

|  |  |  |
| --- | --- | --- |
| **4** | Розділ "Практична робота з С++" |  |
| 4.1 | Введення та виведення у базових консольних програмах |  |
| 4.2 | Робота з файлами: запис та читання |  |
| 4.3 | Формати файлів та їх обробка |  |
| 4.4 | Системи контролю версій (VCS) |  |
| 4.5 | Мова Markdown для опису проектів |  |
| 4.6 | Створення ігрової програми |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Голова циклової комісії |  |  |  |
|  | (підпис) | (ініціали, прізвище) | (дата) |
| Керівники практики від коледжу |  |  |  |
|  | (підпис) | (ініціали, прізвище) | (дата) |
|  |  |  |  |
|  | (підпис) | (ініціали, прізвище) | (дата) |
|  |  |  |  |
|  | (підпис) | (ініціали, прізвище) | (дата) |

**Зміст**

**ОСНОВНА ЧАСТИНА**

**Теоретичні відомості**

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

У мовi програмування С++ введення та виведення інформації здійснюються за допомогою бібліотеки **<iostream>**, яка містить основні об'єкти **cin** та **cout**. Об'єкт **cin** використовується для введення даних з клавіатури, а **cout** — для виведення даних на екран.

Приклад базової програми з введенням та виведенням:

#include <iostream>

using namespace std;

int main() {

int number;

cout << "Введіть число: ";

cin >> number;

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

return 0;

}

***Тема 2. Робота з файлами: запис та читання***

У С++ для роботи з файлами використовуються бібліотеки **<fstream>**, **<ifstream>** та **<ofstream>**. Вони дозволяють створювати, відкривати, читати, записувати та закривати файли.

Приклад запису та читання файлу:

#include <iostream>

#include <fstream>

using namespace std;

int main() {

// Запис у файл

ofstream outFile("example.txt");

outFile << "Це приклад запису у файл.\n";

outFile.close();

// Читання з файлу

ifstream inFile("example.txt");

string line;

if (inFile.is\_open()) {

while (getline(inFile, line)) {

cout << line << endl;

}

inFile.close();

}

return 0;

}

***Тема 3. Формати файлів та їх обробка***

Існує багато різних форматів файлів, які можуть використовуватися для зберігання даних: текстові файли (TXT), бінарні файли, файли конфігурацій (INI, JSON, XML) та інші. У С++ обробка текстових файлів зазвичай здійснюється за допомогою бібліотеки **<fstream>**.

Приклад обробки текстового файлу:

#include <iostream>

#include <fstream>

using namespace std;

int main() {

ifstream inFile("data.txt");

string line;

if (inFile.is\_open()) {

while (getline(inFile, line)) {

cout << line << endl;

}

inFile.close();

}

return 0;

}

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

Системи контролю версій (VCS), такі як Git, дозволяють розробникам відслідковувати зміни у коді, зберігати різні версії проекту та співпрацювати з іншими розробниками. Git є розподіленою системою контролю версій, яка дозволяє створювати локальні репозиторії, синхронізувати їх з віддаленими репозиторіями та зберігати історію змін.

Основні команди Git:

**git init** — ініціалізація нового репозиторію.

**git clone** — клонування існуючого репозиторію.

**git add** — додавання файлів до індексу (стейджинг).

**git commit** — фіксація змін у репозиторії.

**git push** — відправка змін на віддалений репозиторій.

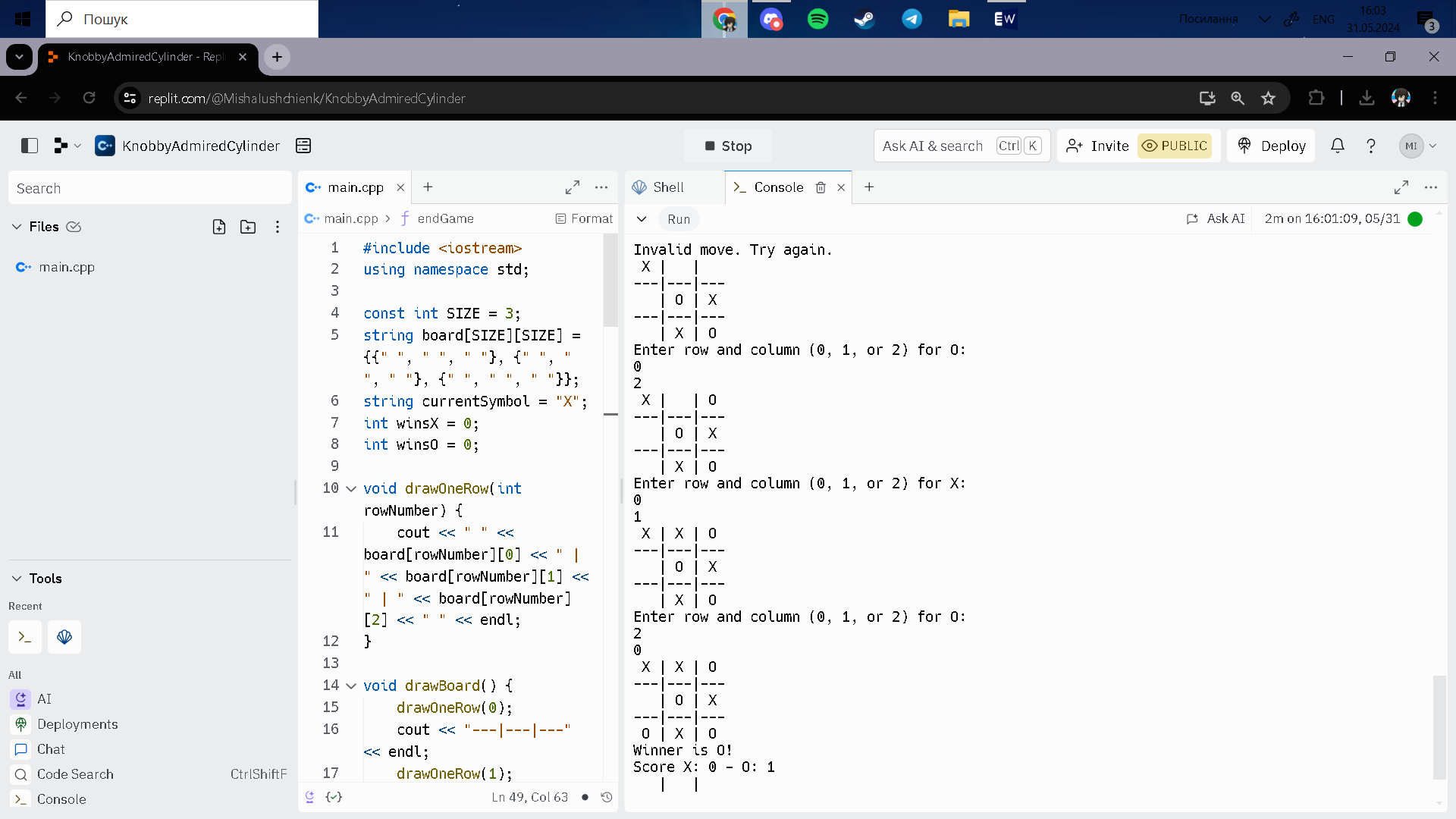
**git pull** — отримання змін з віддаленого репозиторію.

***Тема 5. Мова Markdown для опису проектів***

Markdown — це легка мова розмітки, яка дозволяє створювати текст з форматуванням за допомогою простих синтаксичних конструкцій. Markdown часто використовується для написання документації, README-файлів, статей та повідомлень на форумах.

**Створення ігрової програми**

Для створення гри "Хрестики-нулики" на С++ ми використовуємо всі вищеописані концепції. Гра є консольною програмою, де користувачі можуть вводити свої ходи через командний рядок, а програма виводить поточний стан гри.



**Висновок**

У цьому звіті ми розглянули основні теми, пов'язані з програмуванням на C++ та створенням ігрової програми, а саме: введення та виведення у базових консольних програмах, роботу з файлами, формати файлів та їх обробку, системи контролю версій (VCS) та мову розмітки Markdown для опису проектів.

1. **Введення та виведення у базових консольних програмах**: Ми ознайомилися з використанням бібліотеки **<iostream>** для введення даних з клавіатури за допомогою **cin** та виведення даних на екран за допомогою **cout**. Ці знання дозволяють створювати базові програми для взаємодії з користувачем.
2. **Робота з файлами: запис та читання**: Ми розглянули, як можна використовувати бібліотеки **<fstream>**, **<ifstream>** та **<ofstream>** для запису та читання даних з файлів. Це дозволяє зберігати дані програми для подальшого використання, що є корисним для створення більш складних програм, які потребують збереження стану або історії.
3. **Формати файлів та їх обробка**: Ми вивчили текстові файли як один із найпростіших форматів для зберігання даних. Обробка текстових файлів у C++ здійснюється за допомогою стандартних бібліотек, що дозволяє легко зчитувати та записувати дані.
4. **Системи контролю версій (VCS)**: Ми ознайомилися з основними командами Git, що дозволяють відслідковувати зміни у коді, зберігати різні версії проекту та співпрацювати з іншими розробниками. Використання систем контролю версій є важливим для організації ефективного процесу розробки.
5. **Мова Markdown для опису проектів**: Ми розглянули основи мови Markdown, яка дозволяє створювати текст з форматуванням за допомогою простих синтаксичних конструкцій. Markdown є дуже корисним для написання документації та опису проектів, що полегшує зрозуміння і підтримку коду.

**Створення ігрової програми: гра "Хрестики-нулики"**

Ми застосували усі вивчені концепції для створення консольної гри "Хрестики-нулики" на C++. Гра дозволяє двом користувачам по черзі вводити свої ходи, відображати поточний стан гри на екрані, перевіряти умови перемоги та завершення гри.

Розробка цієї гри продемонструвала практичне застосування введення та виведення даних, роботи з файлами, обробки файлів та використання систем контролю версій. Крім того, ми підготували документацію для проекту за допомогою мови розмітки Markdown.

Загалом, цей проект допоміг глибше зрозуміти основні аспекти програмування на C++ та розробки програмного забезпечення, а також показав, як застосовувати ці знання на практиці для створення реальних програм.