

Міністерство освіти і науки України
Національний університет «Запорізька політехніка»

кафедра програмних засобів

ЗВІТ

з лабораторної роботи № 1

з дисципліни «Комп'ютерна графіка та обробка зображень» на тему:

«ВСТУП ДО OPENGL»

Виконав:

ст. гр. КНТ-113сп

Іван ЩЕДРОВСЬКИЙ

Прийняв:

Асистент

Артем ТУЛЕНКОВ

2025

1 Мета роботи

Отримати практичні навички роботи з бібліотекою OpenGL, використовуючи мову програмування C++. Навчитися малювати 3D об'єкти, виконувати базові маніпуляції з ними, створювати вікна, керувати камерою, працювати з освітленням, створювати прості шейдери та зрозуміти як працюють бібліотеки GLEW, GLFW, GLM.

2 Завдання до лабораторної роботи

- 2.1 Змініть колір та розмір вікна (як приклад можна змінити розмір під існуючий пристрій, наприклад планшет або телефон).
- 2.2 Наведіть коментарі до коду.
- 2.3 Внесіть індивідуальні зміни до коду.

3 Виконання лабораторної роботи

Для виконання лабораторної роботи було використано Visual Studio 17 2022

Лабораторна робота виконувалась, в тому числі, з взаємодією з книгою “Learn OpenGL – Graphics Programming” від Joey de Vries 2020 року, яку можна безкоштовно знайти на сайті learnopengl.com.

Першим етапом було встановлення бібліотеки GLFW. Для максимальної сумісності з системою та середовищем розробки замість використання готової версії було виконана побудова файлів бібліотеки з її вихідного коду. Це було виконано через використання CMake, щоб з вихідного коду створити Visual Studio 17 2022 проект, а далі, всередині IDE, була виконана компіляція бібліотеки. Використання CMake та вихідні файли показані на рисунках 1 та 2 відповідно.

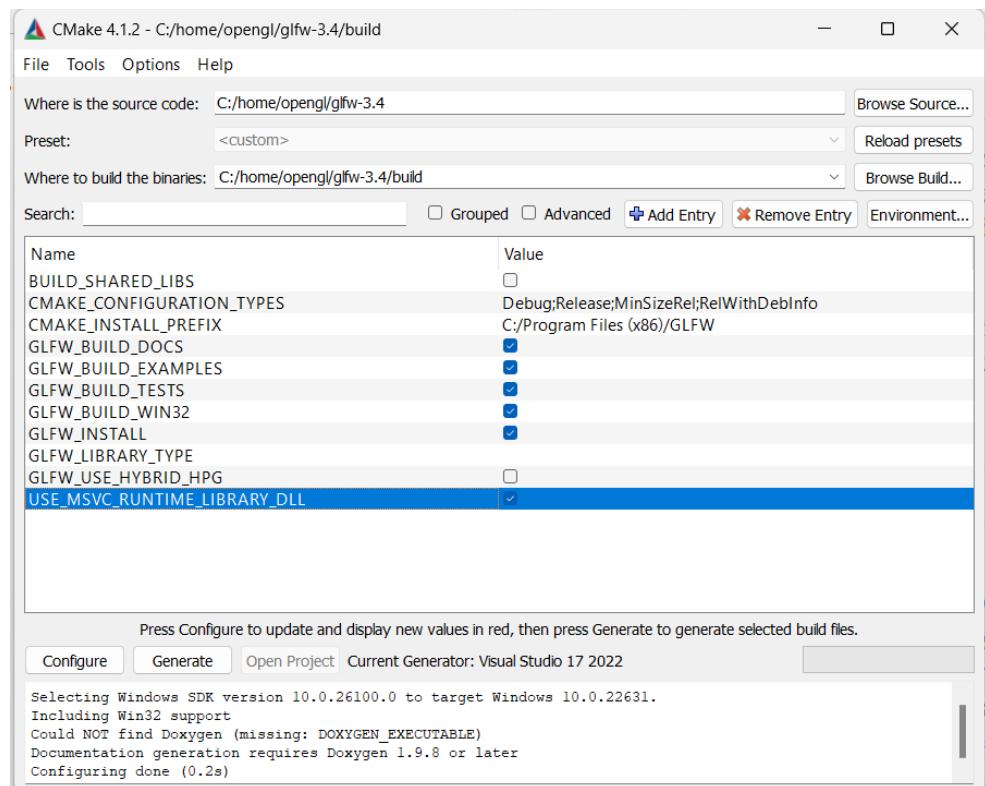


Рисунок 1 – Cmake

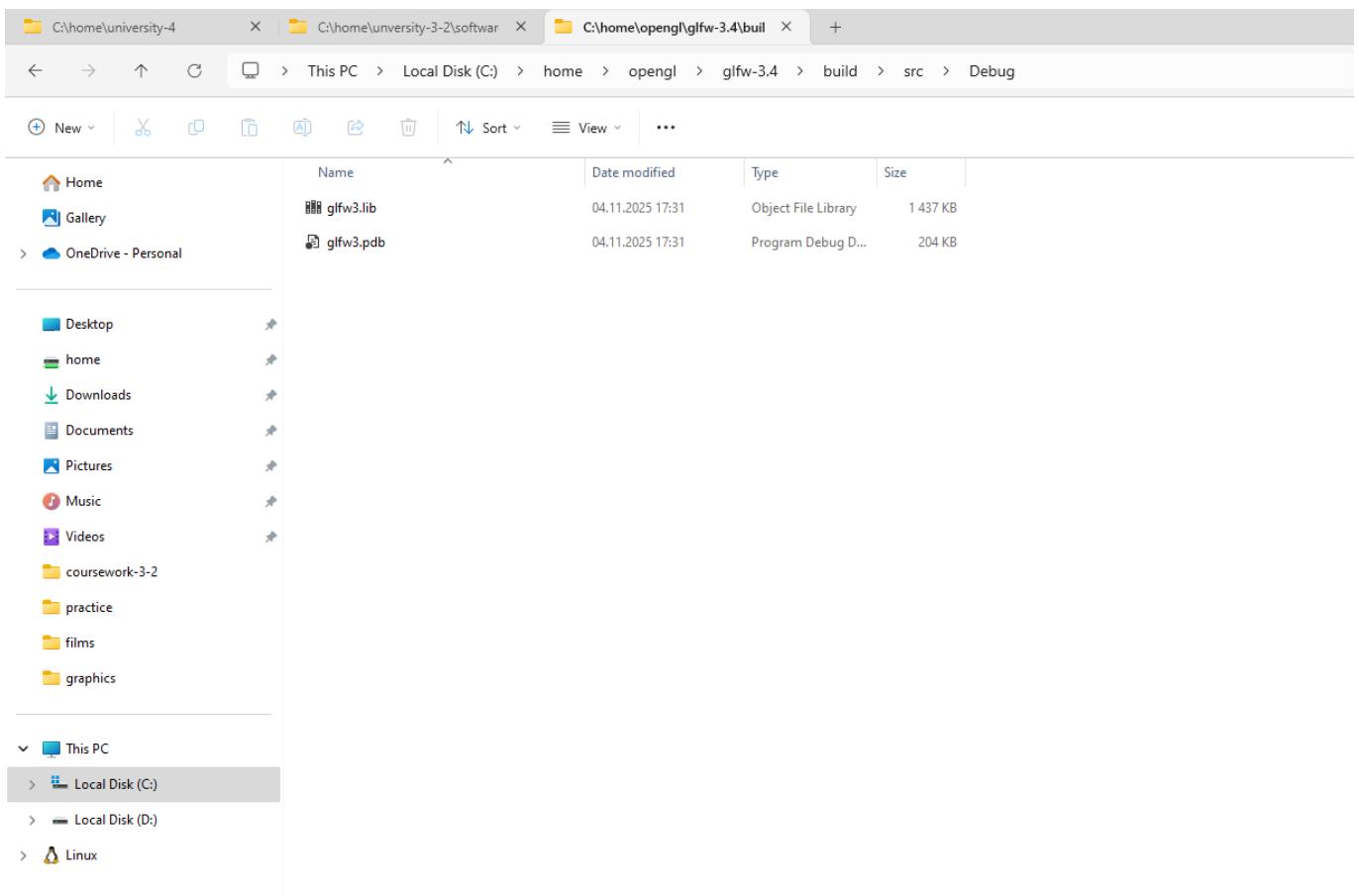


Рисунок 2 – Вихідні файли після компіляції

Наступним етапом було з'єднання IDE та файлів цієї бібліотеки. Для зручної взаємодії з бібліотеками в майбутньому було створено спеціальну директорію, яка зберігає всі include та lib файли. Дерево файлів цієї директорії показано на рисунку 3

```
C:\home\cpp-links>tree /f
Folder PATH listing
Volume serial number is D64A-66D0
C:.
    README.md

    includes
        glad
            glad.h

        GLFW
            glfw3.h
            glfw3native.h

        KHR
            khrplatform.h

    lib
        glfw3.lib
        glfw3.pdb
```

Рисунок 3 – Дерево файлів директорії посилань

Після цього, всередині Visual Studio було додано директорії include та lib до VC++ Directories. Це показано на рисунку 4

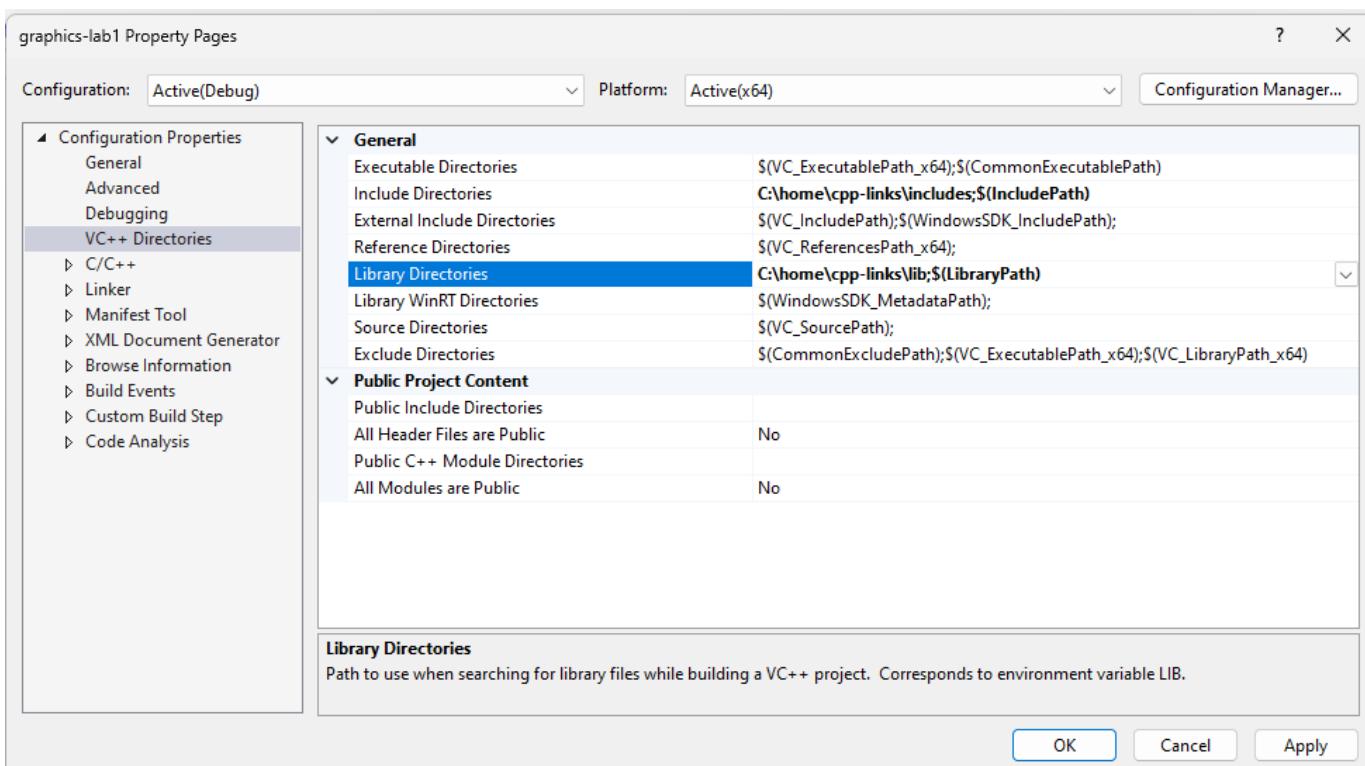


Рисунок 4 – Посилання на include та lib директорії

До Linker було додано додаткові залежності. Це показано на рисунку 5

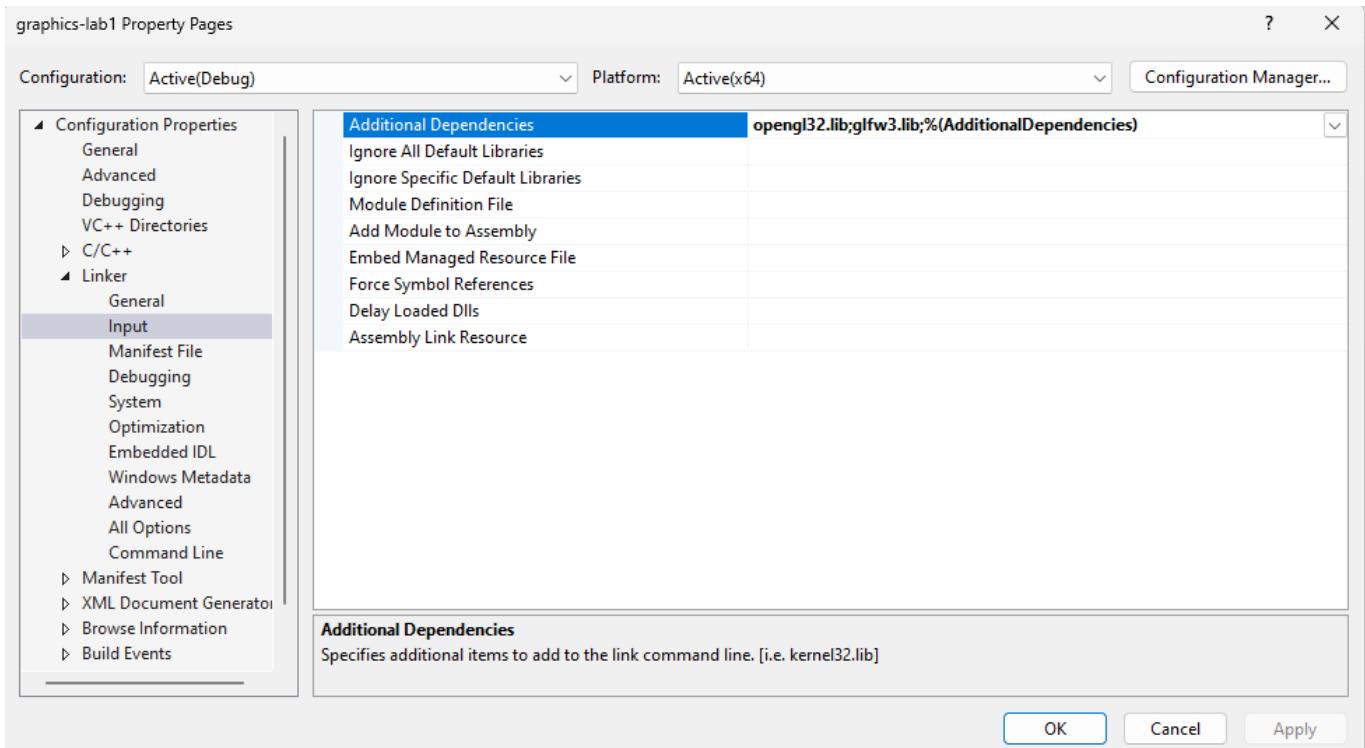


Рисунок 5 – Додаткові залежності в Linker

Оскільки одним з завдань було внести індивідуальні зміни до коду, то в якості такої зміни було вирішено встановити glad замість glew. Для цього було згенеровано конфігурацію GLAD на офіціальному сайті glad.dav1d.de для версії 4.6 в core профілі. Далі, include файли були додані до директорії зі всіма іншими include, а glad.c був доданий безпосередньо до вихідного коду застосунку.

Також, в якості зміни, до коду було додано закриття програми після кліку на ESC.

4 Тест розробленої програми

```
#include <glad/glad.h>
#include <glfw glfw3.h>
#include <iostream>

void framebuffer_size_callback(GLFWwindow* window, int width, int height);
void processInput(GLFWwindow* window);

int main() {
    if (!glfwInit()) {
        std::cout << "GLFW failed to start" << std::endl;
        glfwTerminate();
        return -1;
    }

    glfwWindowHint(GLFW_CONTEXT_VERSION_MAJOR, 3);
    glfwWindowHint(GLFW_CONTEXT_VERSION_MINOR, 3);
    glfwWindowHint(GLFW_OPENGL_PROFILE, GLFW_OPENGL_CORE_PROFILE);

    GLFWwindow* window = glfwCreateWindow(800, 600, "LearnOpenGL", NULL, NULL);

    if (window == NULL) {
        std::cout << "Failed to create GLFW window" << std::endl;
        glfwTerminate();
        return -1;
    }

    glfwMakeContextCurrent(window);

    if (!gladLoadGLLoader((GLADloadproc)glfwGetProcAddress)) {
        std::cout << "Failed to initialize GLAD" << std::endl;
        return -1;
    }

    glfwSetFramebufferSizeCallback(window, framebuffer_size_callback);

    while (!glfwWindowShouldClose(window)) {
        processInput(window);

        glClearColor(0.2f, 0.3f, 0.3f, 1.0f);
        glClear(GL_COLOR_BUFFER_BIT);

        glfwSwapBuffers(window);
    }
}
```

```
        glfwPollEvents();
    }

    glfwTerminate();
    return 0;
}

/*
 * Process interaction with user
 */
void processInput(GLFWwindow* window) {
    if (glfwGetKey(window, GLFW_KEY_ESCAPE) == GLFW_PRESS) {
        glfwSetWindowShouldClose(window, true);
    }
}

/*
 * Auto update OpenGL viewport on resize
 */
void framebuffer_size_callback(GLFWwindow* window, int width, int height) {
    glViewport(0, 0, width, height);
}
```

5 Висновки

Було отримано практичні навички роботи з бібліотекою OpenGL, використовуючи мову програмування C++