**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**Природничо-гуманітарний фаховий коледж**

**ДВНЗ «Ужгородський національний університет»**

***Реєстраційний номер \_\_\_\_\_\_\_\_\_\_\_***

***Дата реєстрації* \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Росохи Максима Валентиновича

студента ІІ курсу

групи КН-23

денної форми навчання

залікова книжка № 253005 МБД

**ЗВІТНА РОБОТА**

**з навчальної практики з програмування**

**на тему:**

**«Реалізація геометричних перетворень тривимірних об**’**єктів. Літак»**

* 1. **Допущено до захисту Науковий керівник:**

Дата «\_\_\_\_»  2020 р. викл. Пастор Н. Е.

Підпис **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Ужгород 2020

**Робота захищена «\_\_\_\_» 2020 р.**

**з оцінкою «\_\_\_\_\_\_\_\_\_/\_\_\_\_/\_\_\_»**

Підписи членів комісії **\_\_\_\_\_\_\_**

ЗМІСТ

ВСТУП 3

1 ТЕОРЕТИЧНА ЧАСТИНА 4

1.1 Основне поняття про літак 4

1.2 Опис предметної області та постановка задачі 4

1.3 Обґрунтування вибору методу розв’язку задачі 7

1.4 Огляд аналогів 7

1.5 Огляд засобів розробки 8

2 ПРАКТИЧНА ЧАСТИНА 11

2.1 Проектна частина 11

2.1.1 Інформаційна структура моделі 11

2.1.2 Функціональні частини програми 11

2.1.3 Фізична структура проекту 12

2.1.4 Опис алгоритму 12

2.1.5 Опис структури даних та використаних змінних 13

2.1.6 Тестування проекту 14

2.2 Експлуатаційна частина 22

2.2.1 Вимоги до технічних засобів 22

2.2.2 Склад інсталяційного пакету та інструкція 22

2.2.3 Опис інтерфейсу 23

2.2.4 Ієрархічна структура проекту 23

2.2.5 Керівництво користувача та приклади використання програмного продукту 24

ВИСНОВКИ 25

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ 26

Додаток 1. Лістинг програми 27

Додаток 2. Блок-схеми 35

Щоденник практики 44

Темою даного навчального проекту є «Реалізація геометричних перетворень тривимірних об’єктів. Літак».

Метою навчальної практики є поглиблення знань, отриманих під час вивчення дисципліни «Основи програмування та алгоритмічні мови», здобуття теоретичних та практичних знань у програмуванні.

Головним завданням даної навчальної практики є побудова програмного продукту, який буде містити в собі можливості, у вигляді геометричних перетворень тривимірного об’єкту — літак, такі як:

• переміщення по осі X та Y;

• зміна масштабу фігури (збільшення та зменшення);

• обертання по різних осях координатної площини (X, Y та Z).

А також закріпити навички, які були здобуті на протязі вивчення дисципліни «Основи програмування та алгоритмічного мовлення» та засвоїти навички самостійно писати програмний код.

**1 ТЕОРЕТИЧНА ЧАСТИНА**

**1.1 Основне поняття про літак**

Літак — літальний апарат важчий за повітря для польотів в атмосфері за допомогою двигуна та нерухомих крил (крила). Літак здатний переміщуватися з високою швидкістю (летіти), використовуючи підіймальну силу крила і тягу двигуна. Нерухоме крило відрізняє літак від махольота, а наявність двигуна — від планера (рисунок 1.1) [1].



Рисунок 1.1 Літак

**1.2 Опис предметної області та постановка задачі**

Для того, щоб легше ініціалізувати точки літака у програмі, потрібно змалювати математичну модель самого літака (рисунок 1.2).

Математичну модель літака здійснено за допомогою програм: “Microsoft Paint” і “Microsoft PowerPoint 2010”.

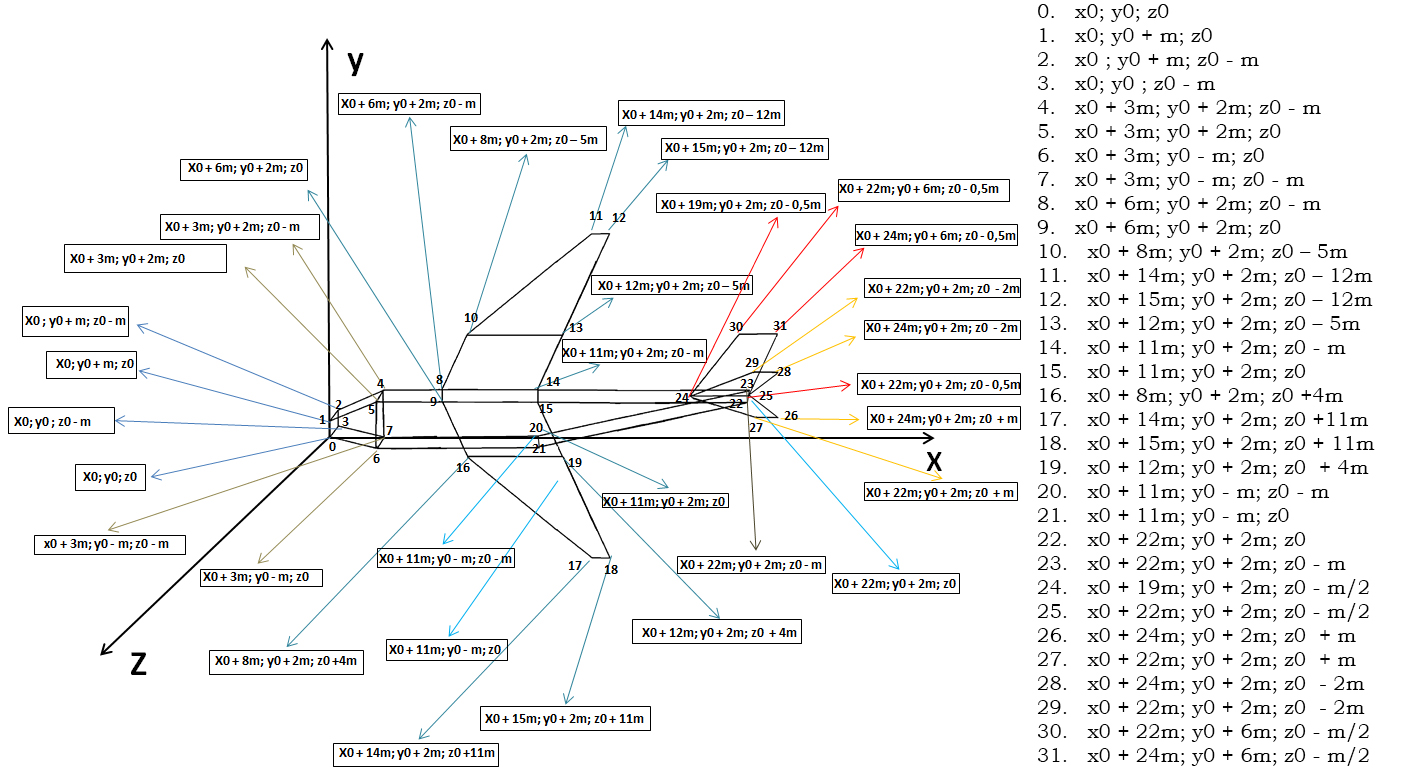


Рисунок 1.2 Точки та їх координати на математичній моделі літака

Оскільки координати визначені, то можна реалізовувати їх паралельне перенесення у просторі, завдяки зміненню даних x0, y0, z0.

Для того, щоб зобразити фігуру на екрані, необхідно перетворити просторові координати в екранні. Я для цього буду використовувати паралельну проекцію.

Отже, для того, щоб змалювати літак на екрані, потрібно задати всі його 32 точки, а також з’єднати певні з них так, щоб вийшло 48 ліній. Таким чином, буде змальовано літак.

Таким чином «літак» — суперпозиція таких графічних примітивів: 32 точки та 48 лінії.

Оскільки точка у просторі описується трьома координатами, то для того, щоб представити її у пам’яті комп’ютера, логічно і зручно буде використати наступну структуру даних (рисунок 1.3):

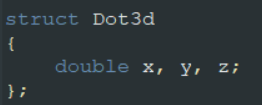


Рисунок 1.3 Структура, яка представляє точки в пам’яті ПК

Таким чином, щоб представити фігуру на екрані, потрібно створити масив точок (рисунок 1.4):

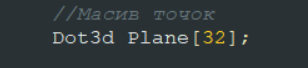


Рисунок 1.4 Масив з точками

Перенести координати точок з математичної моделі у програмне середовище Dev-С++ (рисунок 1.5):

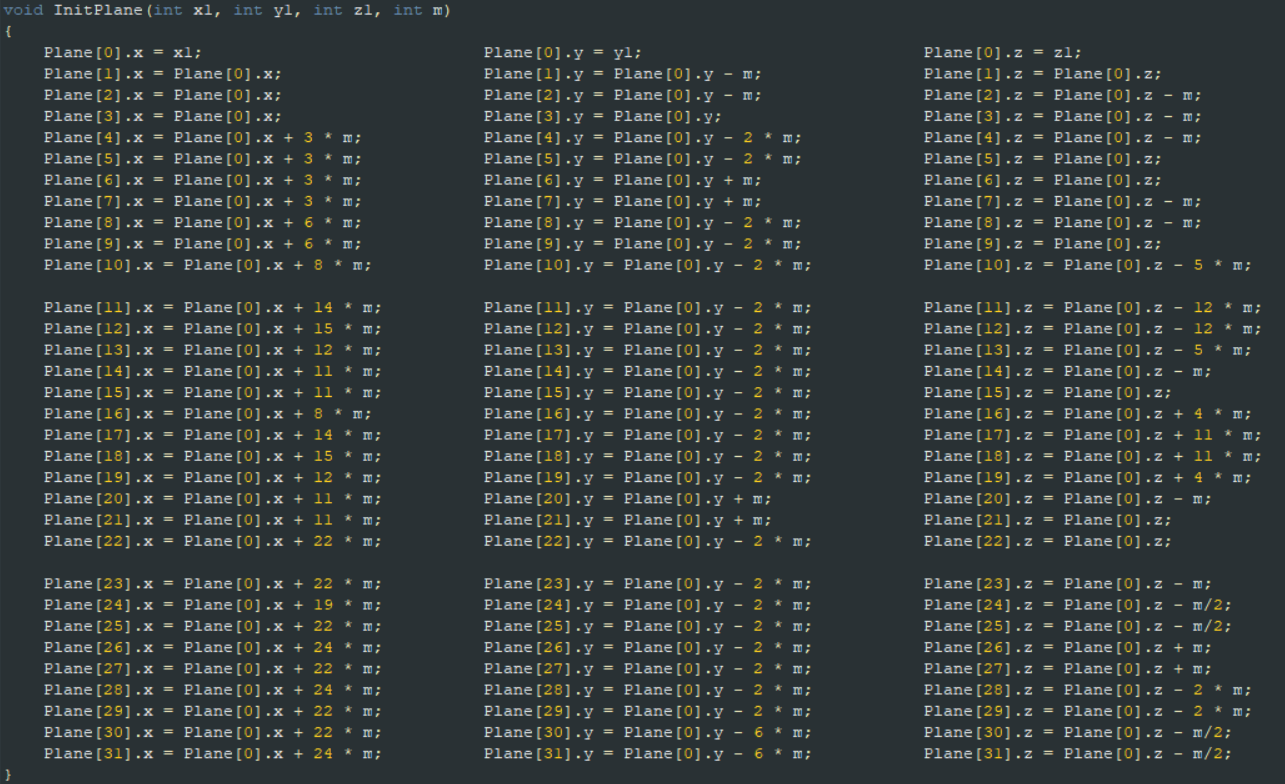


Рисунок 1.5 Масив точок у просторі

Завдяки введенню змінної m можна змінювати пропорцію розміру літака. Оскільки при паралельному перенесенні відстані між точками не змінюються, то можна задати координати точок-вершин відносно однієї точки (назвемо її ключовою точкою). Виберемо як ключову точку вершину, що збігається із початком координат і назвемо її O (x0; y0; z0). Завдяки введенню відносних координат можливо здійснювати паралельне перенесення об’єкта у просторі через зміну трійки значень (x0; y0; z0). Для зображення об’єкту на екрані необхідно здійснити перетворення просторових координат в екрані. Для цієї мети використовуються проекції.

Проекція — зображення тривимірної фігури на так званій картинній (проекційній) площині. У даній моделі використано паралельну проекцію.

Паралельна проекція — це проекція об’єкта в тривимірному просторі на фіксовану площину, відому як площину проекції або площину зображення, де промені, відомі як лінії огляду або лінії проекції, паралельні один одному.

Отже, для того щоб розробити літак у просторі достатньо задати координати точок вершин. Для відображення літака на екрані необхідно побудувати проекції точок (рисунок 1.5) та вказати пари вершин, які мають бути з’єднані лініями (рисунок 1.6).

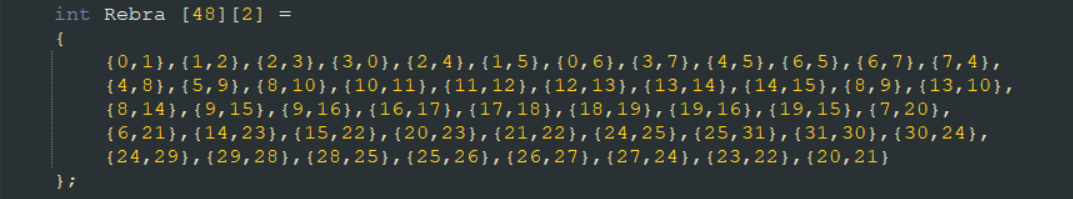


Рисунок 1.6 Масив з’єднаних точок

Ввесь код програми знаходиться у Додатку 1.

**1.3 Обґрунтування вибору методу розв’язку задачі**

Темою даної навчальної практики є «Реалізація геометричних перетворень тривимірних об’єктів. Літак». Оскільки програма повинна бути написана мовою програмування C++ — вибір середовища розробки припав у користь Dev-C++. Використовуючи графічну бібліотеку «winbgim.h».

**1.4 Огляд аналогів**

Як аналог було взято програми для побудови лінії, квадрата (рисунок 1.7) і куба (рисунок 1.8), тому що це основне, що потрібно для побудови моделі літака [3].

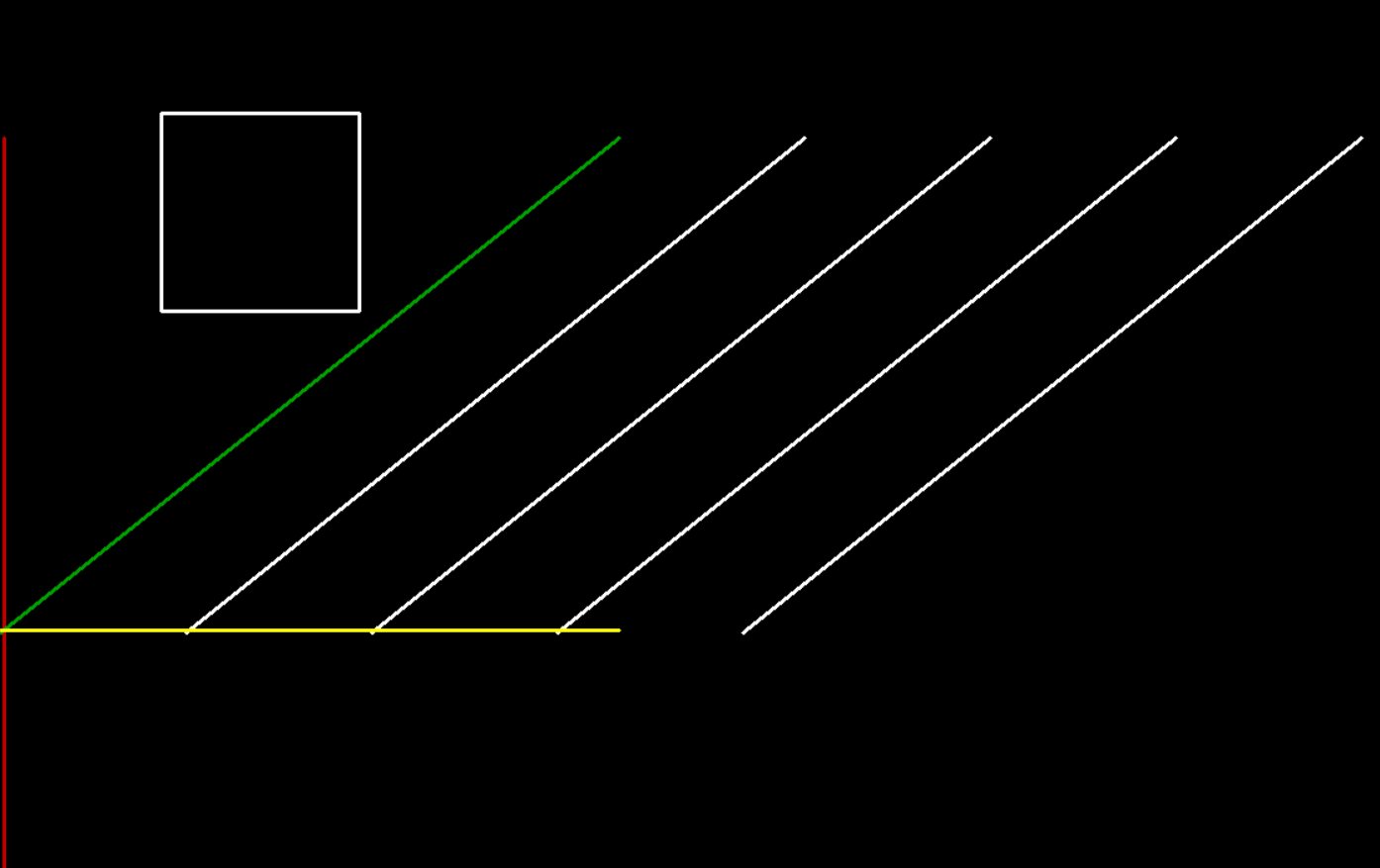


Рисунок 1.7 Програма побудови лінії і квадрата

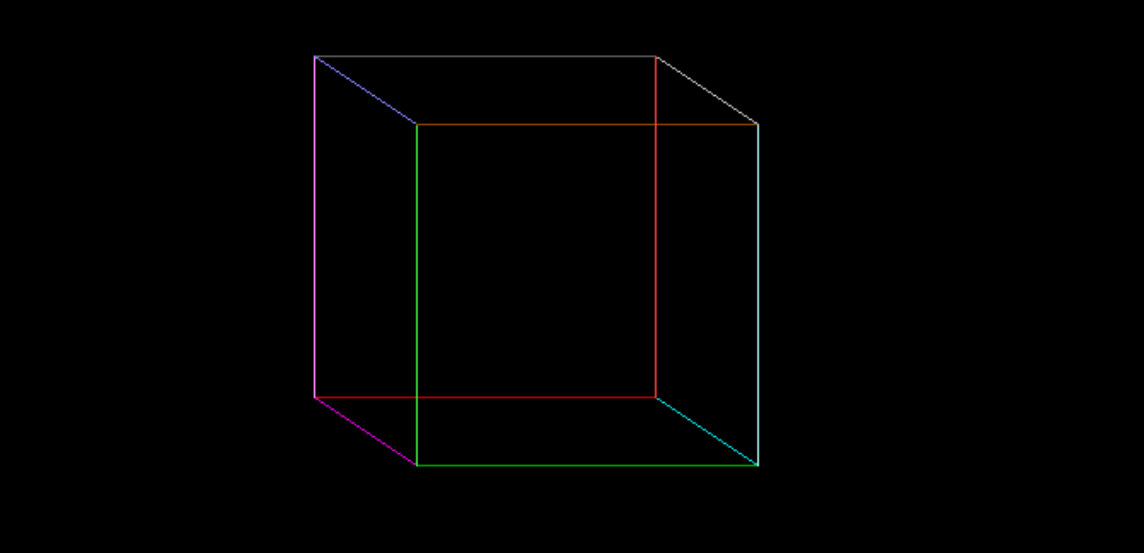


Рисунок 1.8 Програма побудови куба

**1.5 Огляд засобів розробки**

Dev-C++ — інтегроване середовище розробки для мов програмування С/С++. Сам Dev-C++ написаний на Delphi, який розповсюджується згідно з GPL. Для виконання поставленої задачі було вибрано середовище Dev-C++ 5.9.2. Вона повністю задовольняє всі умови, які були поставлені в постанові задачі.

Dev-C++ функціональне програмне середовище, воно володіє такими можливості:

* підвищена швидкість запуску;
* фіксований порядок вкладки випадкових фокусів у різних вікнах;
* легкий і зрозумілий інтерфейс;
* можливість вибрати зручний шрифт і оформлення програми;
* ярлики зберігаються незалежно від поточної мови;
* можливість виконувати програму покроково.

Переваги:

* нескладний синтаксис, що дозволяє дуже швидко освоїти мову;
* включає в себе багато компонентів;
* багато версій на ваш вибір;
* комфортна робота з графікою.

Недоліки:

* підтримка операційних систем тільки сімейства Windows;
* вимагає встановлених бібліотек з динамічним зв’язуванням DLL для роботи програми;
* новіші версії займають багато постійної пам’яті, але це компенсується широким функціоналом та прискіпливим підходом до налаштувань;
* новачкові буде дуже важко розібратися в усьому функціоналі.

Функціонал Dev-C++ дозволяє з перших же хвилин роботи почати розробку свого власного програмного продукту. Мова програмування, яка була вибрана — С++.

C++ — це мова програмування високого рівня, який широко використовується для розробки програмного забезпечення, будучи однією з найпопулярніших мов програмування. Сфера його застосування включає створення операційних систем, різноманітних застосовних програм, драйверів пристроїв, додатків для вбудовуваних систем [2] [4].

**2 ПРАКТИЧНА ЧАСТИНА**

**2.1 Проектна частина**

**2.1.1 Інформаційна структура моделі**

При відкритті програми, перед користувачем відкривається меню, де знаходяться варіанти вибору від 1 до 4. Якщо користувач вводить цифру один, то відкривається вікно де зображається сама фігура, і можливість виконувати дії над нею. При вводі цифри 2 відкривається вікно, де буде довідка у якій написано, що здійснює дана програма. Цифра 3 відповідає за відомості про автора. Цифра 4 відповідає за вихід з програми. Після вводу кожної цифри потрібно виходити повністю з програми й заходити знову (недосконалість).

**2.1.2 Функціональні частини програми**

Розроблено проект у середовищі Dev – C++ та мовою програмування C++.

Структура проекту подано на рисунку 2.1:

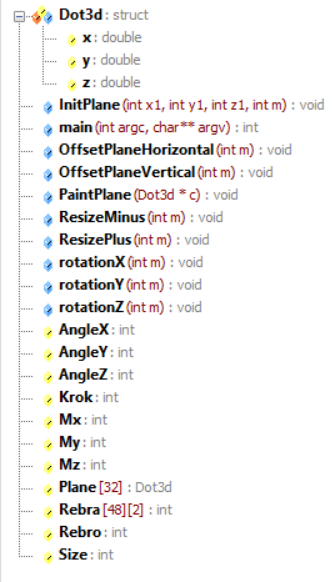


Рисунок 2.1 Внутрішня структура проекту

У програмі були використані такі бібліотеки (рисунок 2.2):

* «iostream» - бібліотека використовується для організації введення-виведення в мові програмування C++;
* «conio.h» - бібліотека створення текстового інтерфейсу;
* «math.h» - бібліотека для виконання простих математичних операцій;
* «winbgim.h» - графічна бібліотека.

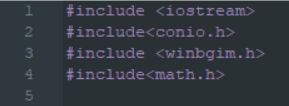


Рисунок 2.2 Бібліотеки програми

**2.1.3 Фізична структура проекту**

Для того, щоб програма працювала коректно, необхідно мати файли, які зображені на рисуноку 2.3:

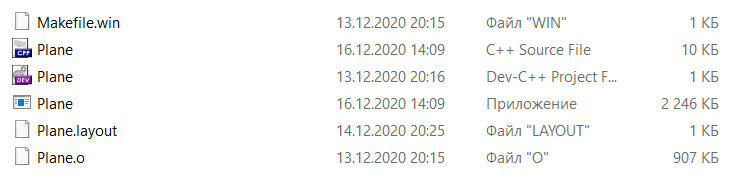


Рисунок 2.3 Вміст кореневої папки проекту

**2.1.4 Опис алгоритму**

Основою алгоритму програми є повний цикл реалізації запланованих

функцій, зокрема за прикладом:

1. Запуск програми, запуск меню користувача;
2. Відкриття пункту Демонстрація фігури;
3. Закриття пункту Демонстрація фігури;
4. Запуск програми, запуск меню користувача;
5. Відкриття пункту Довідка;
6. Закриття пункту Довідка;
7. Запуск програми, запуск меню користувача;
8. Відкриття пункту Про автора;
9. Закриття пункту Про автора;
10. Запуск програми, запуск меню користувача;
11. Вибір пункту Вихід;
12. Вихід із програми.

Конкретний вигляд блок-схем головної функції, меню, переміщення, масштабування та обертання можна переглянути у Додатку 2.

**2.1.5 Опис структури даних та використаних змінних**

Таблиця 2.1

|  |  |
| --- | --- |
| **Використані простори імен, бібліотеки, зміні, масиви та структури даних:** | **Опис** |
| iostream | організація введення-виведення |
| conio.h | створення текстового інтерфейсу |
| winbgim.h | графічна бібліотека |
| math.h | виконання простих математичних операцій |
| * int Krok * int Size * int AngleX * int AngleY * int AngleZ * int Rebro * int Mx, My * int m * char menu, action * Dot3d * масив Rebra * масив InitPlane | * використовується для позначення ступеню зміщення фігури; * використовується для позначення ступеня збільшення або зменшення фігури; * вказує кут оберту по осі X; * вказує кут оберту по осі Y; * вказує кут оберту по осі Z; * змінна довжини ребра; * початкове розміщення об’єкту;   Продовження таблиці 2.1   * відповідність; * визначення дії, яку вибере користувач; * для задавання точок на   координатній осі;   * масив індексів пар точок які будуть з’єднуватись лініями; * являє собою масив координат точок. |

**2.1.6 Тестування проекту**

План тестування:

1. Запуск програми;
2. Тестування меню програми;
3. Переміщення фігури;
4. Масштабування фігури;
5. Обертання фігури.

Тестування:

1. Для запуску програми потрібно відкрити файл «Plane.exe». При запуску програми на екран виводиться головне меню (рисунок 2.4), в якому є пункти: «Демонстрація фігури», «Довідка», «Про автора» та «Вихід».

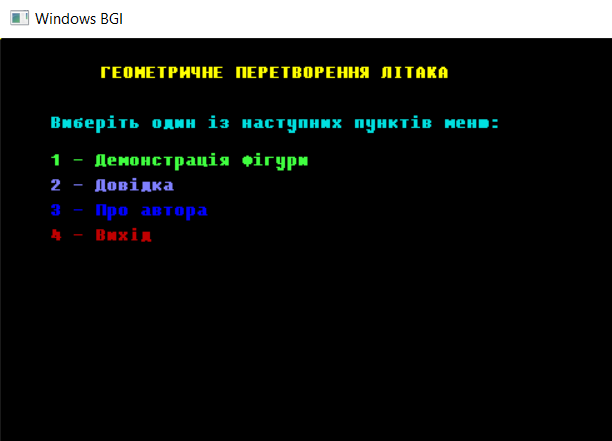


Рисунок 2.4 Головне меню

1. При виборі пункту «1» — відкривається графічне вікно в якому буде зображена фігура (рисунок 2.5).

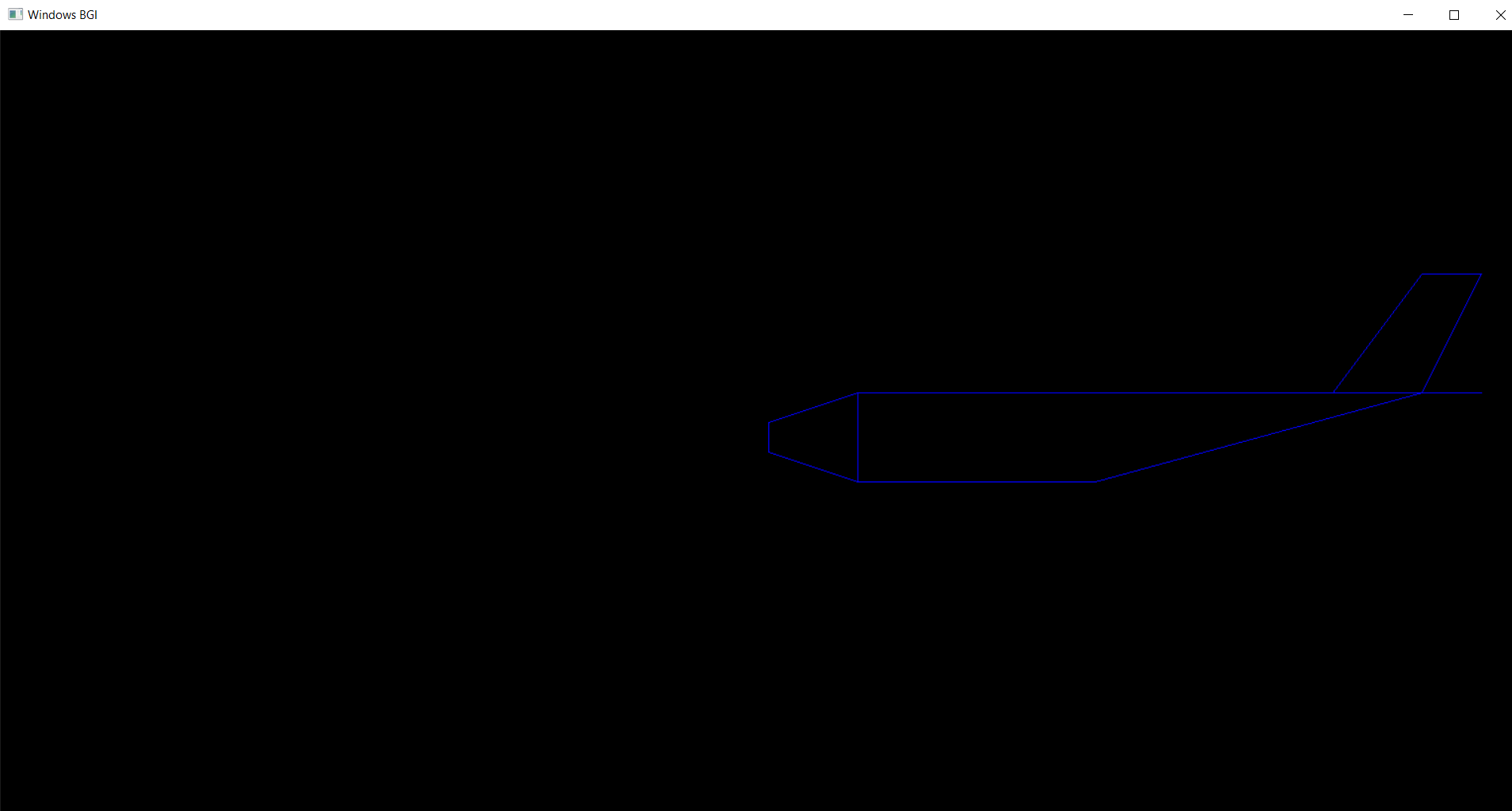


Рисунок 2.5 Демонстрація фігури

При виборі пункту «2» — відкривається графічне вікно в якому буде зображена довідка (рисунок 2.6).



Рисунок 2.6 Довідка

При виборі пункту «3» — відкривається графічне вікно в якому буде зображена інформація про автора (рисунок 2.7).

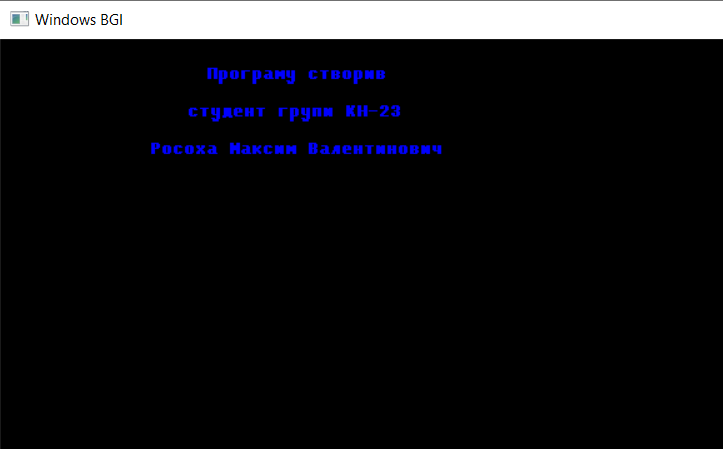


Рисунок 2.7 Про автора

При виборі пункту «4» — закривається графічне вікно (рисунок 2.8).

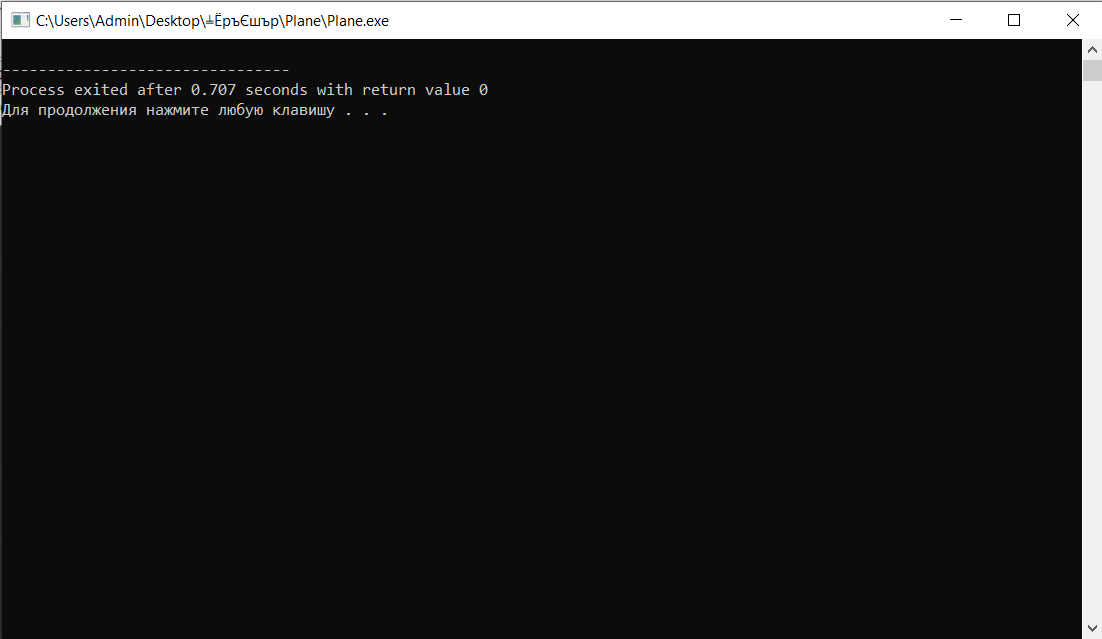


Рисунок 2.8 Вихід з програми

1. Для переміщення фігури використовуються стрілкові клавіші (рисунки 2.9, 2.10, 2.11, 2.12, 2.13):

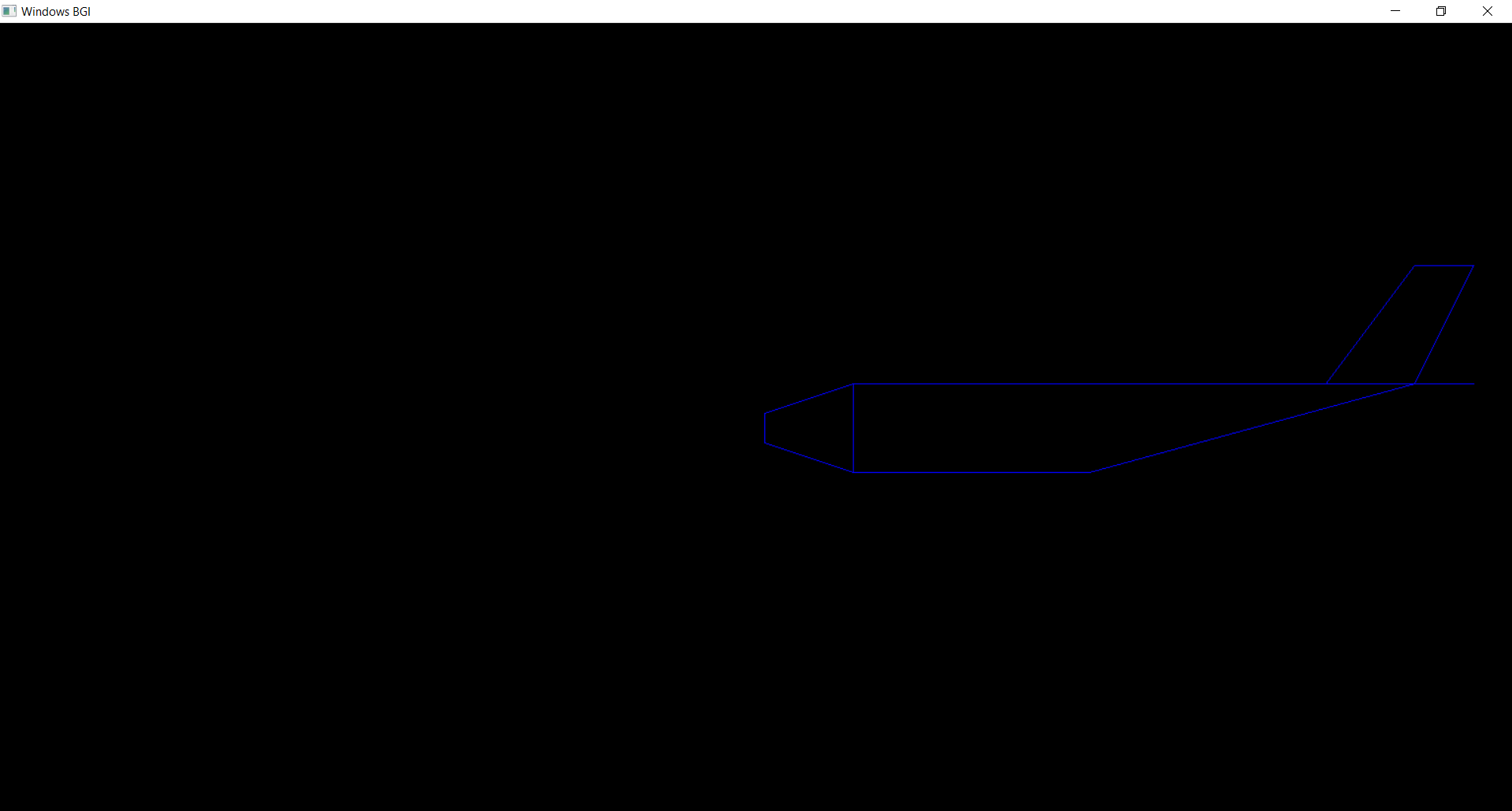


Рисунок 2.9 Початкове положення фігури

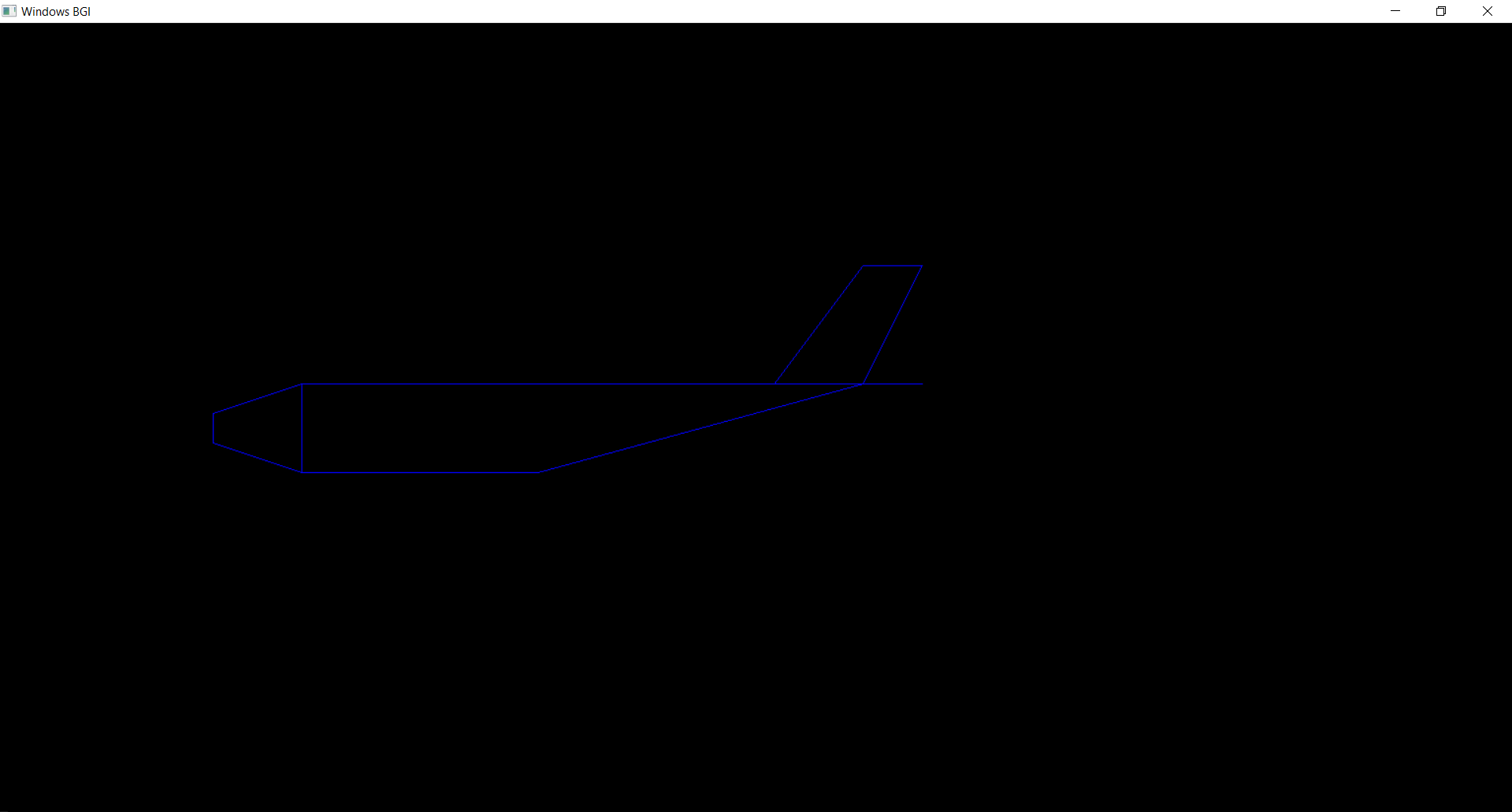


Рисунок 2.10 Рух фігури вправо



Рисунок 2.11 Рух фігури вгору

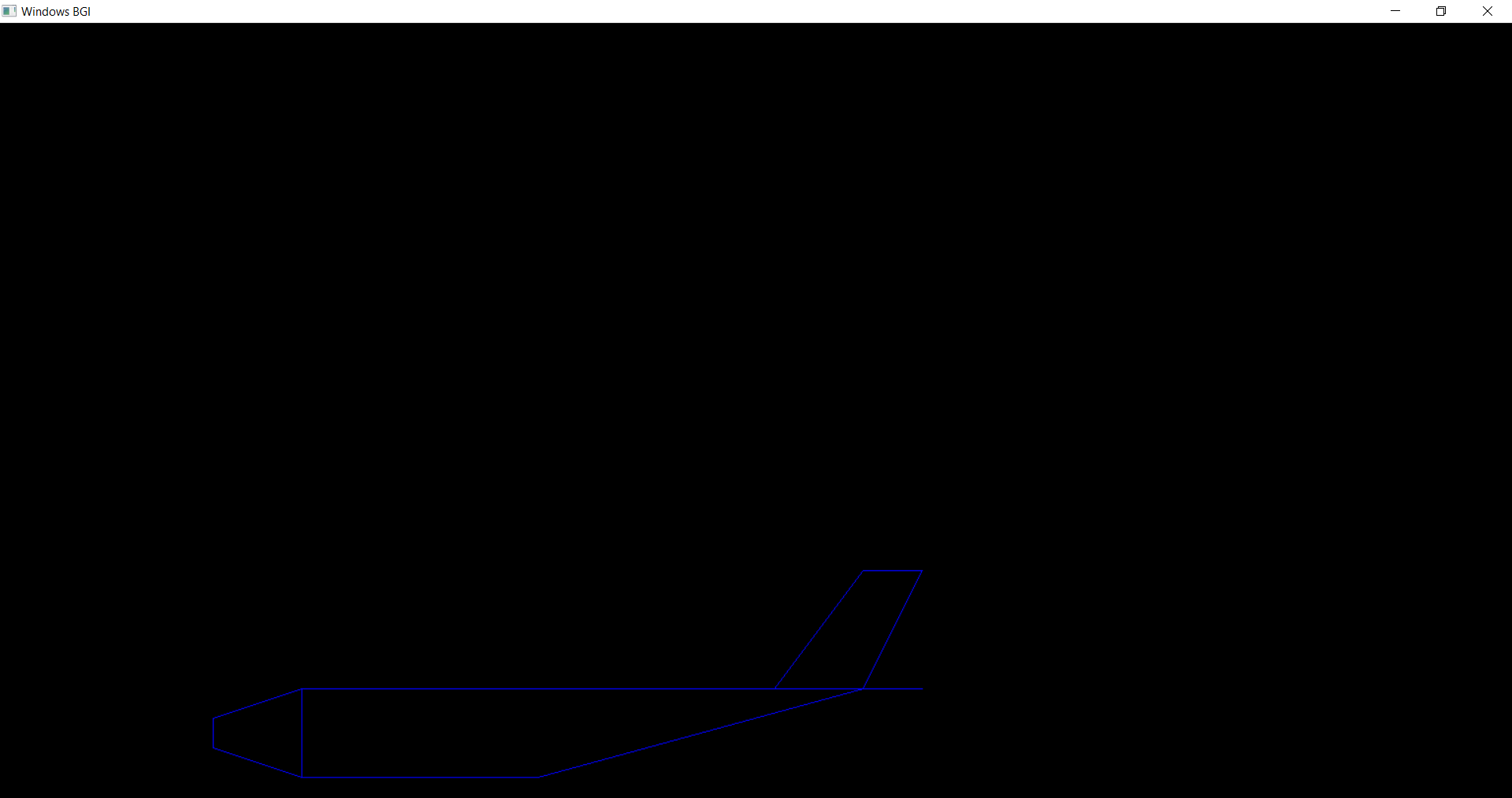


Рисунок 2.12 Рух фігури вниз

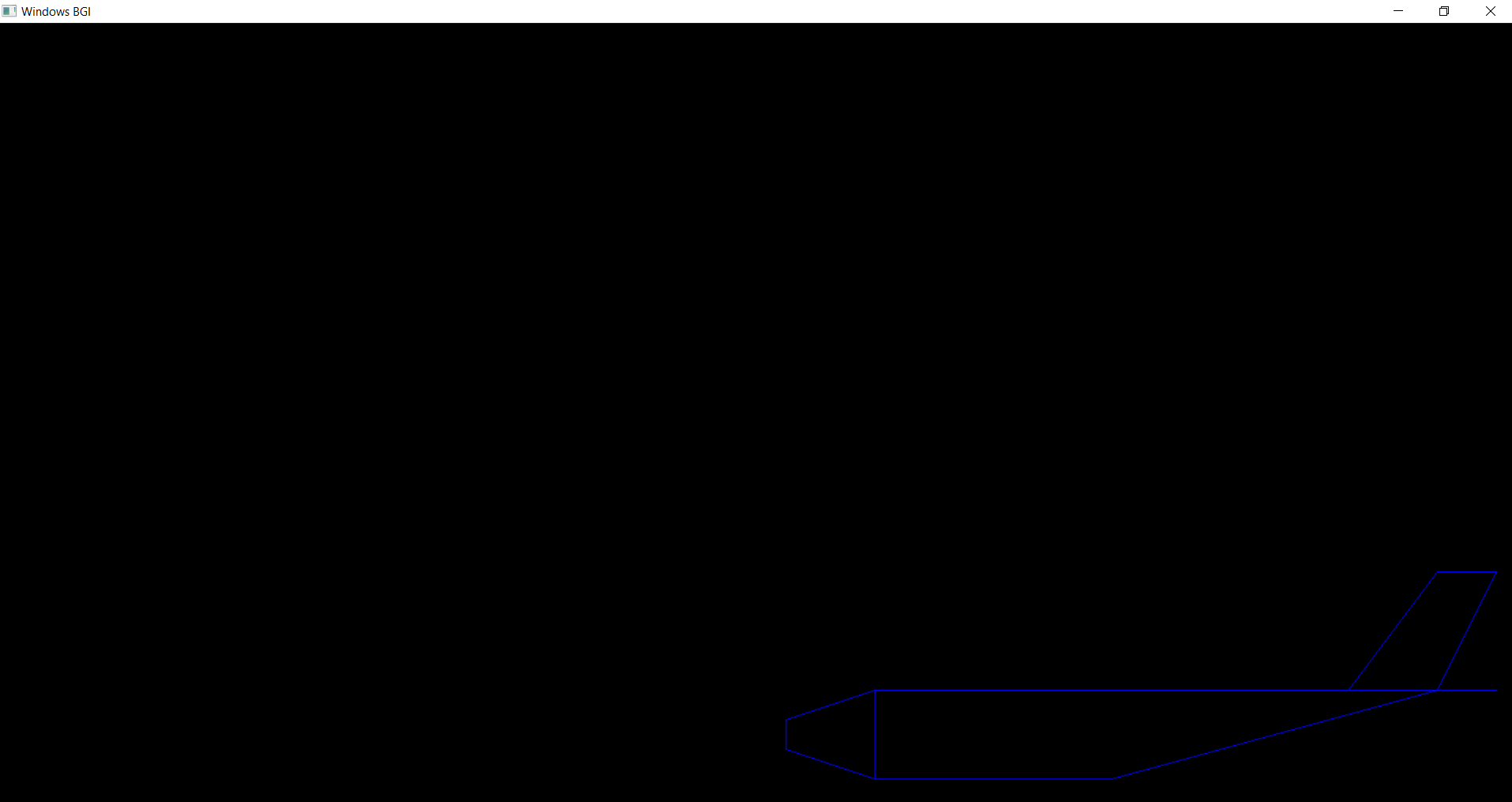


Рисунок 2.13 Рух фігури вліво

1. Для збільшення масштабу фігури використовується клавіша «х», а для зменшення клавіша «z» (рисунки 2.14 та 2.15):

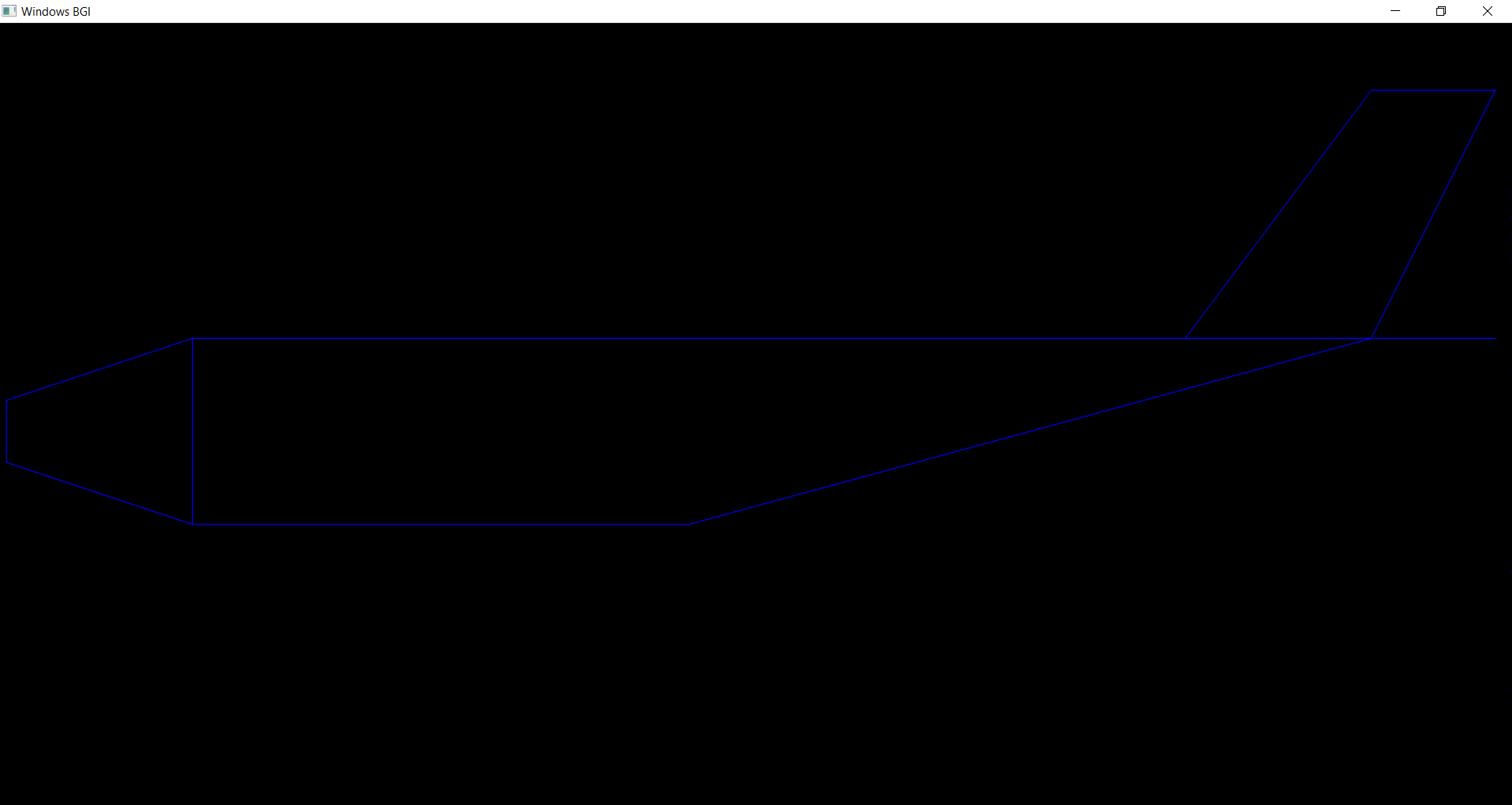


Рисунок 2.14 Збільшення маштабу фігури

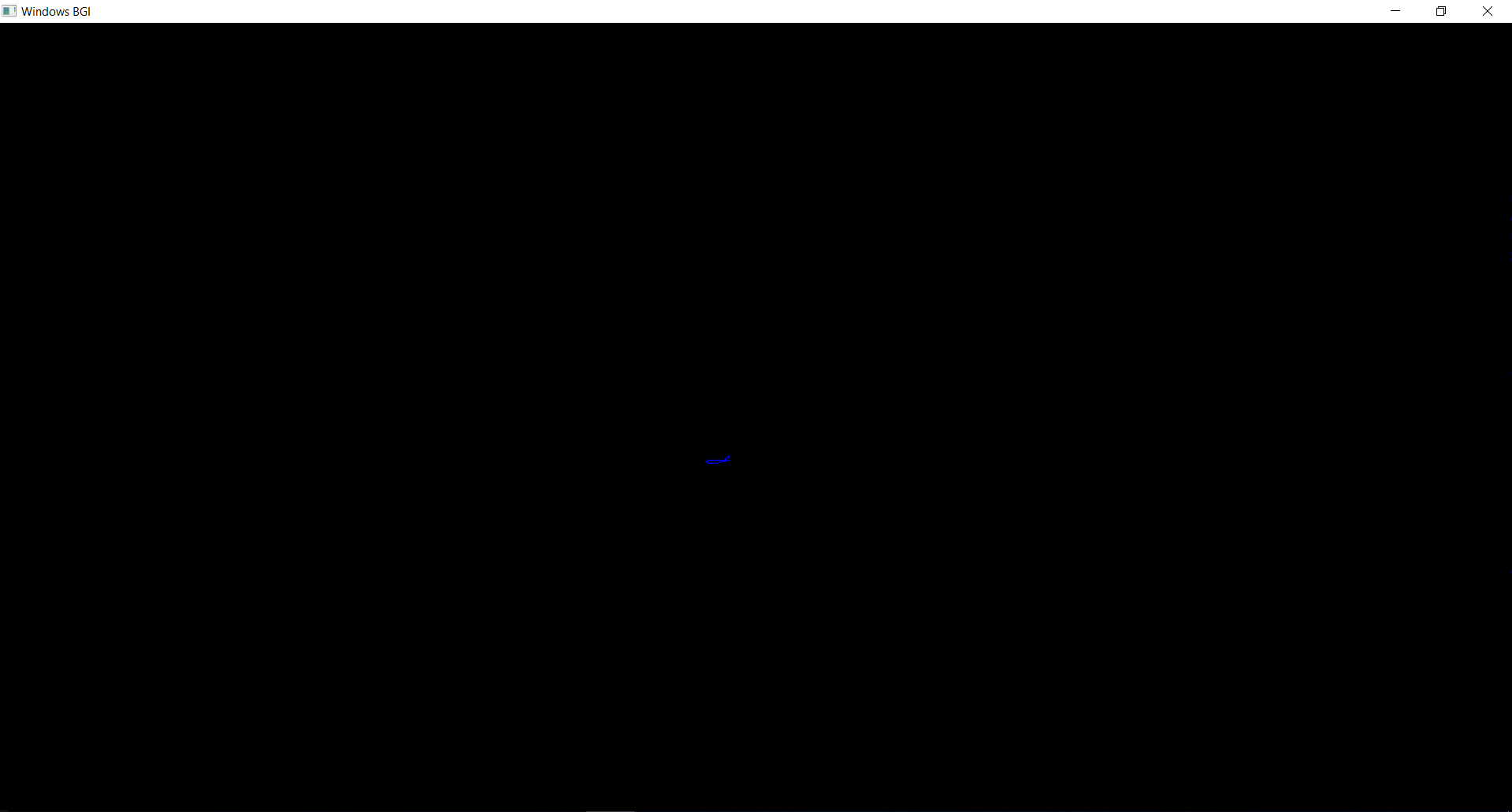


Рисунок 2.15 Зменшення масштабу фігури

1. Для обертання фігури використовуються клавіші (рисунки 2.16, 2.17, 2.18):

* «d» – для того, щоб обертати фігуру по осі X;
* «w» – для того, щоб обертати фігуру по осі Y;
* «a» – для того, щоб обертати фігуру по осі Z.

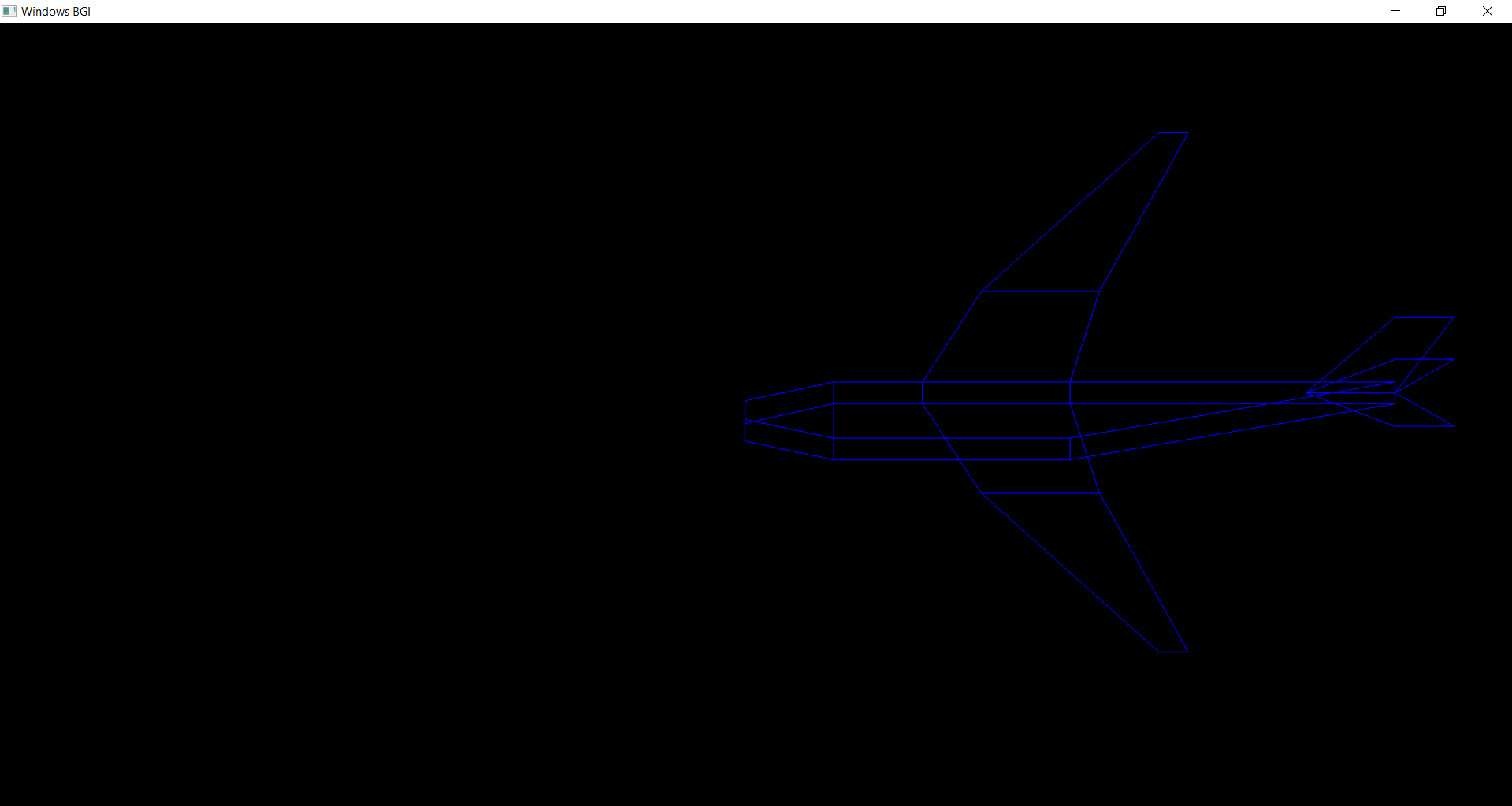


Рисунок 2.16 Обертання по осі X

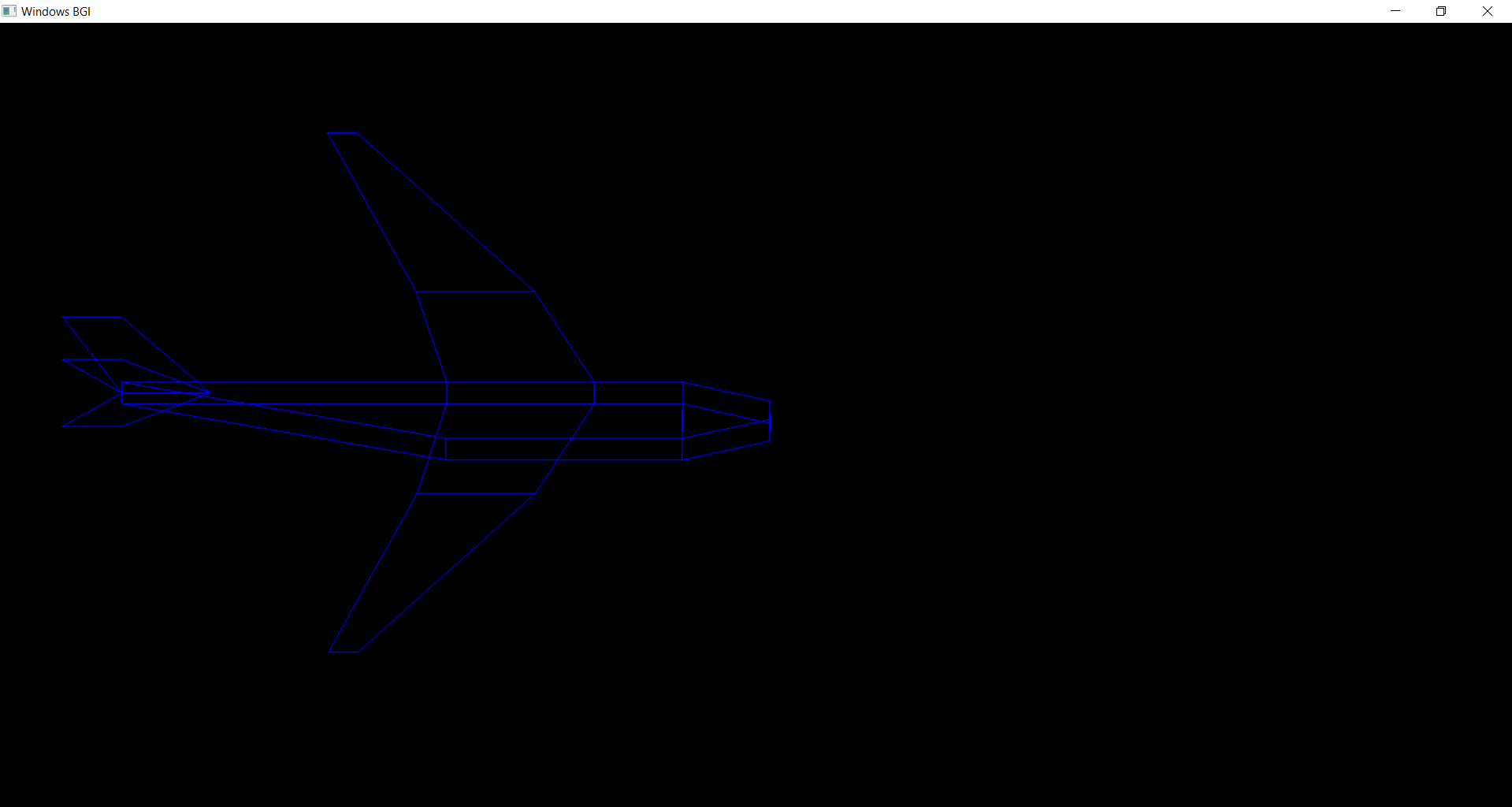


Рисунок 2.17 Обертання по осі Y

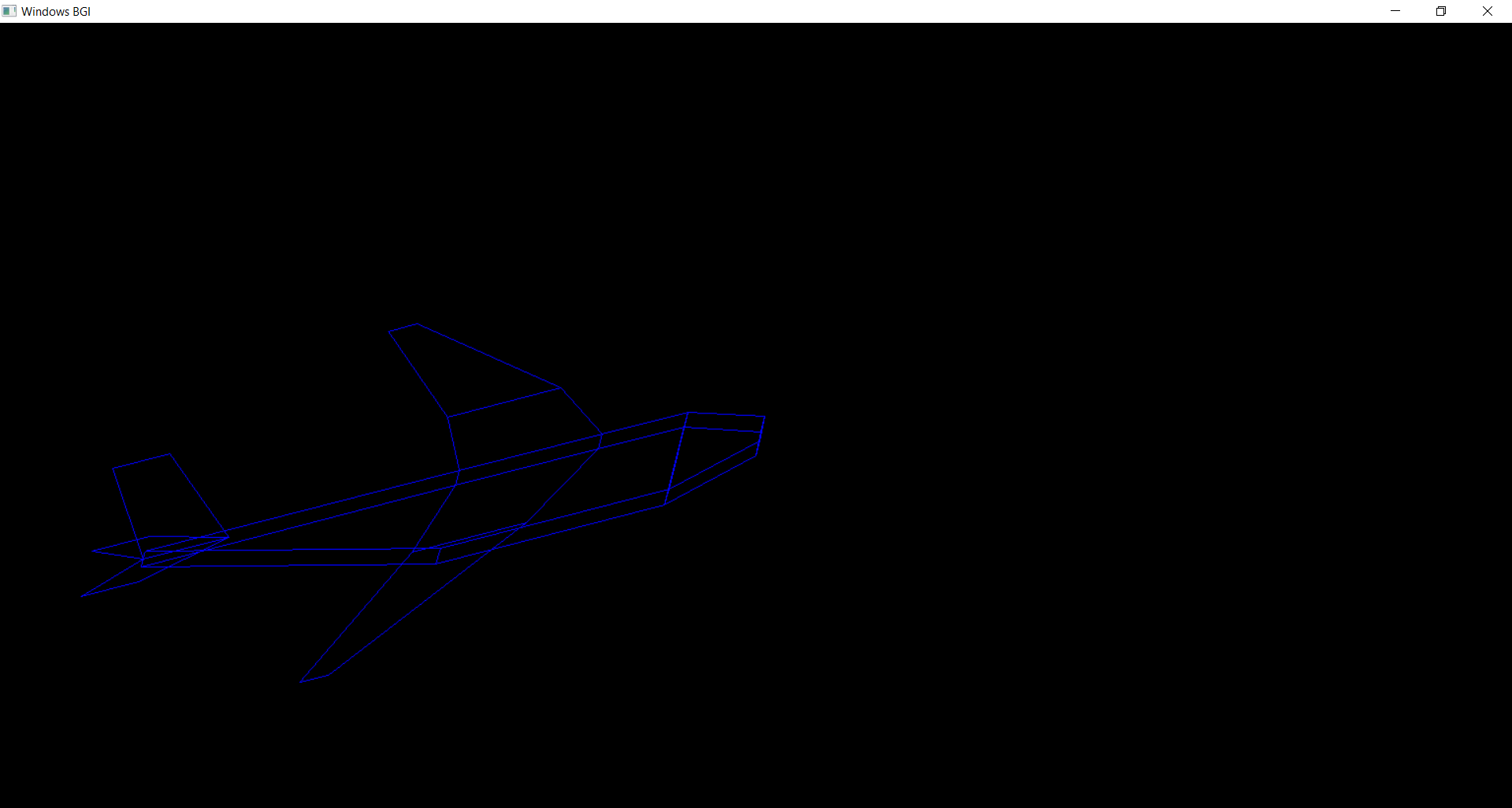


Рисунок 2.18 Обертання по осі Z

## 2.2 Експлуатаційна частина

### 2.2.1 Вимоги до технічних засобів

Для користування програмою «Plane.exe» користувач повинен мати базові навички користування комп’ютером, мати пристрої вводу (мишка, клавіатура), а також пристрої виводу (монітор з мінімальним розширенням 720p). Використовувати можна будь-яку операційну систему Windows починаючи з версії Windows XP.

**2.2.2 Склад інсталяційного пакету та інструкція**

Для того, щоб програма працювала правильно, потрібна папка із такими необхідними файлами (рисунок 2.19) і наявність графічної бібліотеки «winbgim.h» (рисунок 2.20):

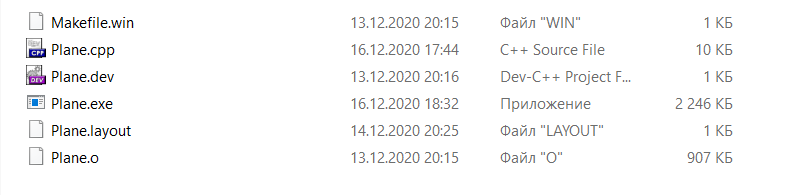


Рисунок 2.19 Вміст папки з програмою

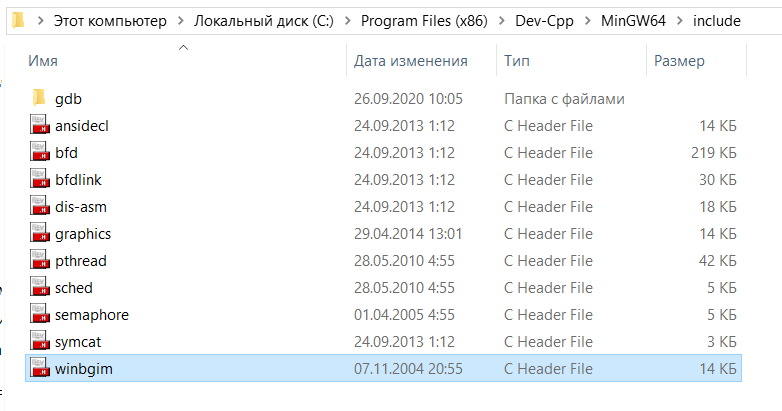


Рисунок 2.20 Місце розташування графічної бібліотеки «winbgim.h»

**2.2.3 Опис інтерфейсу**

При запуску програми, користувач побачить простий інтерфейс, який починається з головного меню, де знаходяться пункти які можна вибирати (рисунок 2.4).

Користувач може користуватися всіма кроками які описані в «2.1.2 Опис алгоритму» і детальну реалізацію проекту переглянути на рисунках 2.4 – 2.18.

**2.2.4 Ієрархічна структура проекту**

Навігація по проекту:

При виборі пункту «Демонстрація фігури», відкривається вікно в якому буде зображено саму фігуру і можливість дій над нею.

При виборі пункту «Довідка», у меню буде описано, що виконує ця програма і засоби керування, масштабування та обертання фігури.

При виборі пункту «Про автора», буде відкрито нове вікно де текстом буде виведено ім’я, прізвище та групу у якій навчається автор.

При виборі пункту «Вихід», буде здійснений вихід з програми.

Ілюстрації цих дій були здійсненні у пункті «Тестування».

**2.2.5 Керівництво користувача та приклади використання програмного продукту**

Для керування програмою використовується клавіатура. Запуск програми відбувається так само, як і у всіх інших програм на Windows, тобто подвійним клацанням лівою клавішею миші по ярлику програми «Plane.exe». Після того, як програму запущено, вона запропонує 4 пункти на вибір:

* Демонстрація фігури;
* Довідка(Інформація для користувача);
* Про автора(Інформація про творця програми);
* Вихід.

У вікні із самою фігурою, для того, щоб переміщати її використовуйте стрілкові клавіші. Щоб збільшувати та зменшувати — X та Z відповідно. Щоб обертати по осях X, Y та Z — D, W та A відповідно.

Під час написання даного навчального проекту виникли деякі труднощі, такі як те, що під час першого запуску програми фігура була перевернута по осі Y – виправив це я тим, що змінив всі «+» на «-» по осі Y і навпаки. Друга проблема була у тому, що хвіст літака будувався не по центру літака, а на «–z» назад, виправив я це тим, що змінив «Plane[0].z – 0,5 \* m;» на «Plane[0].z - m/2;».

На даний час у програмі є такі проблеми:

* некоректне обертання фігури по осі Z;
* немає обмеження щодо масштабування фігури;
* немає обмеження яке не дозволяє виходити фігурі за межі графічного вікна.

**ВИСНОВКИ**

Результатом виконання даної навчальної практики — є розробка програмного продукту який демонструє математичну модель “Літака”.

Під час написання даного навчального проекту, я закріпив навички, які були здобуті під час вивчення дисципліни «Основи програмування та алгоритмічні мови», ознайомився із бібліотекою для роботи із графікою у С++ — «winbgim.h», саме за допомогою цієї бібліотеки та створена дана програма. Засвоїв тему «масиви точок». Детальніше розібрався з темою «правильні геометричні фігури». Краще засвоїв синтаксис С++, використання циклів, операторів, умови та створення методів у різних ситуаціях.

**ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ**

1. Інтернет-ресурс «Літак»: <https://uk.wikipedia.org/wiki/%D0%9B%D1%96%D1%82%D0%B0%D0%BA> (дата звертання 07.12.2020).
2. Інтернет-ресурс «Dev-C++»: <https://soft.mydiv.net/win/download-DEV-C.html> (дата звертання 15.12.2020).
3. Інтернет-ресурс «graphics.h трехмерное изображение»: <https://www.cyberforum.ru/3dmodeling/thread571562.html> (дата звертання 08.12.2020).
4. Інтернет-ресурс «Dev-C++»: <https://uk.wikipedia.org/wiki/Dev-C%2B%2B> (дата звертання 15.12.2020).
5. Інтернет-ресурс «BGI Documentation for getch»: <https://www.cs.colorado.edu/~main/cs1300/doc/bgi/getch.html> (дата звертання 10.12.2020).
6. Інтернет-ресурс «Цвет строки в консоли»: <https://www.cyberforum.ru/win-api/thread892349.html> (дата звертання 09.12.2020).
7. Інтернет-ресурс «Тема 9 Графика Сpp»: <https://www.youtube.com/watch?v=Wmh8uBph6FM&t=1s&ab_channel=AzatKabdullin> (дата звертання 07.12.2020) .
8. Інтернет-ресурс «Консоль. Завершение цикла нажатием клавиши»: <https://www.cyberforum.ru/cpp-beginners/thread229346.html> (дата звертання 11.12.2020).
9. Інтернет-ресурс «Библиотека WINBGIM»: <https://www.cs.colorado.edu/~main/cs1300/doc/bgi/bgi.html> (дата звертання 09.12.2020).
10. Інтернет-ресурс «Параметры и аргументы функций»: <https://ravesli.com/urok-13-parametry-i-argumenty-funktsij/> (дата звертання 14.12.2020).

**Додаток 1. Лістинг програми**

// Бібліотеки

#include <iostream>

#include<conio.h>

#include <winbgim.h>

#include<math.h>

// Клавіші

#define ESC 27

#define LEFT 75

#define RIGHT 77

#define UP 72

#define DOWN 80

#define ROTX 100

#define ROTY 119

#define ROTZ 97

#define MAGNIFICATION 120

#define REDUCTION 122

using namespace std;

struct Dot3d // Структура, яка представляє точки в пам'яті ПК

{

double x, y, z;

};

Dot3d Plane[32];//Масив точок

int Rebro = 30;// Довжина ребра

int Rebra [48][2] = //Масив, який об'єднує певні точки між собою

{

{0,1},{1,2},{2,3},{3,0},{2,4},{1,5},{0,6},{3,7},{4,5},{6,5},{6,7},{7,4},

{4,8},{5,9},{8,10},{10,11},{11,12},{12,13},{13,14},{14,15},{8,9},{13,10},

{8,14},{9,15},{9,16},{16,17},{17,18},{18,19},{19,16},{19,15},{7,20},

{6,21},{14,23},{15,22},{20,23},{21,22},{24,25},{25,31},{31,30},{30,24},

{24,29},{29,28},{28,25},{25,26},{26,27},{27,24},{23,22},{20,21}

};

int Krok = 10;

int Size = 1;

int Mx, My;

int AngleX = 0;

int AngleY = 0;

int AngleZ = 0;

void InitPlane(int x1, int y1, int z1, int m);

void PaintPlane(Dot3d \* c);

void OffsetPlaneHorizontal(int);

void OffsetPlaneVertical(int);

void rotationX(int);

void rotationZ(int);

void rotationY(int);

void ResizePlus(int);

void ResizeMinus(int);

int main(int argc, char\*\* argv)

{

char menu;

//Відкриття графічного вікна

initwindow(1600, 1000);

//Початкове розміщення об'єкту

Mx = getmaxx()/2;;

My = getmaxy()/2;;

//Меню

setcolor (14);

outtextxy (80,20,"ГЕОМЕТРИЧНЕ ПЕРЕТВОРЕННЯ ЛIТАКА");

setcolor (3);

outtextxy (40,60,"Виберiть один iз наступних пунктiв меню:");

setcolor (10);

outtextxy (40,90,"1 - Демонстрацiя фiгури");

setcolor (9);

outtextxy (40,110,"2 - Довiдка");

setcolor (1);

outtextxy (40,130,"3 - Про автора");

setcolor (4);

outtextxy (40,150,"4 - Вихід");

//Пункти меню

while (true)

{

menu = getch();

switch (menu)

{

case '1':

{

cleardevice( );

InitPlane (Mx / 100, My / 100, 0 , Rebro);

PaintPlane (Plane);

while (true)

{

char action; // Дія

action = getch();

switch (action)

{

case ESC:

{

return 0;

break;

}

case LEFT:

{

OffsetPlaneHorizontal(-Krok);

break;

}

case RIGHT:

{

OffsetPlaneHorizontal(Krok);

break;

}

case UP:

{

OffsetPlaneVertical(-Krok);

break;

}

case DOWN:

{

OffsetPlaneVertical(Krok);

break;

}

case ROTX:

{

rotationX(5);

break;

}

case ROTY:

{

rotationY(5);

break;

}

case ROTZ:

{

rotationZ(5);

break;

}

case MAGNIFICATION:

{

ResizePlus(Size);

break;

}

case REDUCTION:

{

ResizeMinus(Size);

break;

}

}

}

}

case '2':

{

cleardevice( );

setcolor (3);

outtextxy (88,0,"Довідка:");

setcolor (10);

outtextxy (80,20,"Керування:");

setcolor (11);

outtextxy (10,50,"Стрілки - переміщення фігури;");

outtextxy (10,80,"z - зменшення фігури;");

outtextxy (10,100,"x - збільшення фігури;");

outtextxy (10,130,"a - обертання фігури по осі Z;");

outtextxy (10,150,"w - обертання фігури по осі Y;");

outtextxy (10,170,"d - обертання фігури по осі X;");

setcolor (4);

outtextxy (10,200,"ESC - вихід з програми;");

setcolor (8);

outtextxy (10,230,"Для коректної роботи в програмі мову потрібно змінити на англійську.");

getch ();

return 0;

break;

}

case '3':

{

cleardevice( );

setcolor (1);

outtextxy (165,20,"Програму створив");

outtextxy (150,50,"студент групи КН-23");

outtextxy (120,80,"Росоха Максим Валентинович");

getch ();

return 0;

break;

}

case '4':

{

return 0;

break;

}

}

}

return 0;

}

//Масив з координатими точок

void InitPlane(int x1, int y1, int z1, int m)

Plane[0].x = x1; Plane[0].y = y1; Plane[0].z = z1;

Plane[1].x = Plane[0].x; Plane[1].y = Plane[0].y - m; Plane[1].z = Plane[0].z;

Plane[2].x = Plane[0].x; Plane[2].y = Plane[0].y - m; Plane[2].z = Plane[0].z - m;

Plane[3].x = Plane[0].x; Plane[3].y = Plane[0].y; Plane[3].z = Plane[0].z - m;

Plane[4].x = Plane[0].x + 3 \* m; Plane[4].y = Plane[0].y - 2 \* m; Plane[4].z = Plane[0].z - m;

Plane[5].x = Plane[0].x + 3 \* m; Plane[5].y = Plane[0].y - 2 \* m; Plane[5].z = Plane[0].z;

Plane[6].x = Plane[0].x + 3 \* m; Plane[6].y = Plane[0].y + m; Plane[6].z = Plane[0].z;

Plane[7].x = Plane[0].x + 3 \* m; Plane[7].y = Plane[0].y + m; Plane[7].z = Plane[0].z - m;

Plane[8].x = Plane[0].x + 6 \* m; Plane[8].y = Plane[0].y - 2 \* m; Plane[8].z = Plane[0].z - m;

Plane[9].x = Plane[0].x + 6 \* m; Plane[9].y = Plane[0].y - 2 \* m; Plane[9].z = Plane[0].z;

Plane[10].x = Plane[0].x + 8 \* m; Plane[10].y = Plane[0].y - 2 \* m; Plane[10].z = Plane[0].z - 5 \* m;

Plane[11].x = Plane[0].x + 14 \* m; Plane[11].y = Plane[0].y - 2 \* m; Plane[11].z = Plane[0].z - 12 \* m;

Plane[12].x = Plane[0].x + 15 \* m; Plane[12].y = Plane[0].y - 2 \* m; Plane[12].z = Plane[0].z - 12 \* m;

Plane[13].x = Plane[0].x + 12 \* m; Plane[13].y = Plane[0].y - 2 \* m; Plane[13].z = Plane[0].z - 5 \* m;

Plane[14].x = Plane[0].x + 11 \* m; Plane[14].y = Plane[0].y - 2 \* m; Plane[14].z = Plane[0].z - m;

Plane[15].x = Plane[0].x + 11 \* m; Plane[15].y = Plane[0].y - 2 \* m; Plane[15].z = Plane[0].z;

Plane[16].x = Plane[0].x + 8 \* m; Plane[16].y = Plane[0].y - 2 \* m; Plane[16].z = Plane[0].z + 4 \* m;

Plane[17].x = Plane[0].x + 14 \* m; Plane[17].y = Plane[0].y - 2 \* m; Plane[17].z = Plane[0].z + 11 \* m;

Plane[18].x = Plane[0].x + 15 \* m; Plane[18].y = Plane[0].y - 2 \* m; Plane[18].z = Plane[0].z + 11 \* m;

Plane[19].x = Plane[0].x + 12 \* m; Plane[19].y = Plane[0].y - 2 \* m; Plane[19].z = Plane[0].z + 4 \* m;

Plane[20].x = Plane[0].x + 11 \* m; Plane[20].y = Plane[0].y + m; Plane[20].z = Plane[0].z - m;

Plane[21].x = Plane[0].x + 11 \* m; Plane[21].y = Plane[0].y + m; Plane[21].z = Plane[0].z;

Plane[22].x = Plane[0].x + 22 \* m; Plane[22].y = Plane[0].y - 2 \* m; Plane[22].z = Plane[0].z;

Plane[23].x = Plane[0].x + 22 \* m; Plane[23].y = Plane[0].y - 2 \* m; Plane[23].z = Plane[0].z - m;

Plane[24].x = Plane[0].x + 19 \* m; Plane[24].y = Plane[0].y - 2 \* m; Plane[24].z = Plane[0].z - m/2;

Plane[25].x = Plane[0].x + 22 \* m; Plane[25].y = Plane[0].y - 2 \* m; Plane[25].z = Plane[0].z - m/2;

Plane[26].x = Plane[0].x + 24 \* m; Plane[26].y = Plane[0].y - 2 \* m; Plane[26].z = Plane[0].z + m;

Plane[27].x = Plane[0].x + 22 \* m; Plane[27].y = Plane[0].y - 2 \* m; Plane[27].z = Plane[0].z + m;

Plane[28].x = Plane[0].x + 24 \* m; Plane[28].y = Plane[0].y - 2 \* m; Plane[28].z = Plane[0].z - 2 \* m;

Plane[29].x = Plane[0].x + 22 \* m; Plane[29].y = Plane[0].y - 2 \* m; Plane[29].z = Plane[0].z - 2 \* m;

Plane[30].x = Plane[0].x + 22 \* m; Plane[30].y = Plane[0].y - 6 \* m; Plane[30].z = Plane[0].z - m/2;

Plane[31].x = Plane[0].x + 24 \* m; Plane[31].y = Plane[0].y - 6 \* m; Plane[31].z = Plane[0].z - m/2;

}

void PaintPlane(Dot3d \* c)

{

int yt1, yt2, xt1, xt2, zt1, zt2;

double anX = (AngleX % 360 / 360.0) \* 3.14 \* 2;

double anY = (AngleY % 360 / 360.0) \* 3.14 \* 2;

double anZ = (AngleZ % 360 / 360.0) \* 3.14 \* 2;

// Малювання фігури

for (int i = 0; i < 48; i++)

{

xt1 = (c[Rebra[i][0]].x);

yt1 = (c[Rebra[i][0]].y) \* cos(anX) + c[Rebra[i][0]].z \* sin(anX);

zt1 = -(c[Rebra[i][0]].y) \* sin(anX) + c[Rebra[i][0]].z \* cos(anX);

xt2 = (c[Rebra[i][1]].x);

yt2 = (c[Rebra[i][1]].y) \* cos(anX) + c[Rebra[i][1]].z \* sin(anX);

zt2 = -(c[Rebra[i][1]].y) \* sin(anX) + c[Rebra[i][1]].z \* cos(anX);

xt1 = xt1 \* cos(anY) + zt1 \* sin(anY);

yt1 = yt1;

zt1 = -xt1 \* sin(anY) + zt1 \* cos(anY);

xt2 = xt2 \* cos(anY) + zt2 \* sin(anY);

yt2 = yt2;

zt2 = -xt2 \* sin(anY) + zt2 \* cos(anY);

xt1 = xt1 \* cos(anZ) - yt1 \* sin(anZ);

yt1 = -xt1 \* sin(anZ) + yt1 \* cos(anZ);

zt1 = zt1;

xt2 = xt2 \* cos(anZ) - yt2 \* sin(anZ);

yt2 = -xt2 \* sin(anZ) + yt2 \* cos(anZ);

zt2 = zt2;

// Колір фігури (синій)

setcolor(1);

// Малювання ліній

line (xt1 + Mx, yt1 + My, xt2 + Mx, yt2 + My);

}

}

// Переміщення фігури по горизонталі

void OffsetPlaneHorizontal(int m)

{

for(int i = 0; i < 32; i++)

{

Plane[i].x = Plane[i].x + m;

cleardevice();

PaintPlane(Plane);

}

}

// Переміщення фігури по вертикалі

void OffsetPlaneVertical(int m)

{

for(int i = 0; i < 32; i++)

{

Plane[i].y = Plane[i].y + m;

cleardevice();

PaintPlane(Plane);

}

}

// Обертання фігури по осі X

void rotationX(int m)

{

AngleX += m;

cleardevice();

InitPlane(Plane[0].x, Plane[0].y, Plane[0].z, Rebro);

PaintPlane(Plane);

}

// Обертання фігури по осі Y

void rotationY(int m)

{

AngleY += m;

cleardevice();

InitPlane(Plane[0].x, Plane[0].y, Plane[0].z, Rebro);

PaintPlane(Plane);

}

// Обертання фігури по осі Z

void rotationZ(int m)

{

AngleZ += m;

cleardevice();

InitPlane(Plane[0].x, Plane[0].y, Plane[0].z, Rebro);

PaintPlane(Plane);

}

// Збільшення фігури

void ResizePlus(int m)

{

Rebro += m;

cleardevice();

InitPlane(Plane[0].x, Plane[0].y, Plane[0].z, Rebro);

PaintPlane(Plane);

}

// Зменшення фігури

void ResizeMinus(int m)

{

Rebro -= m;

cleardevice();

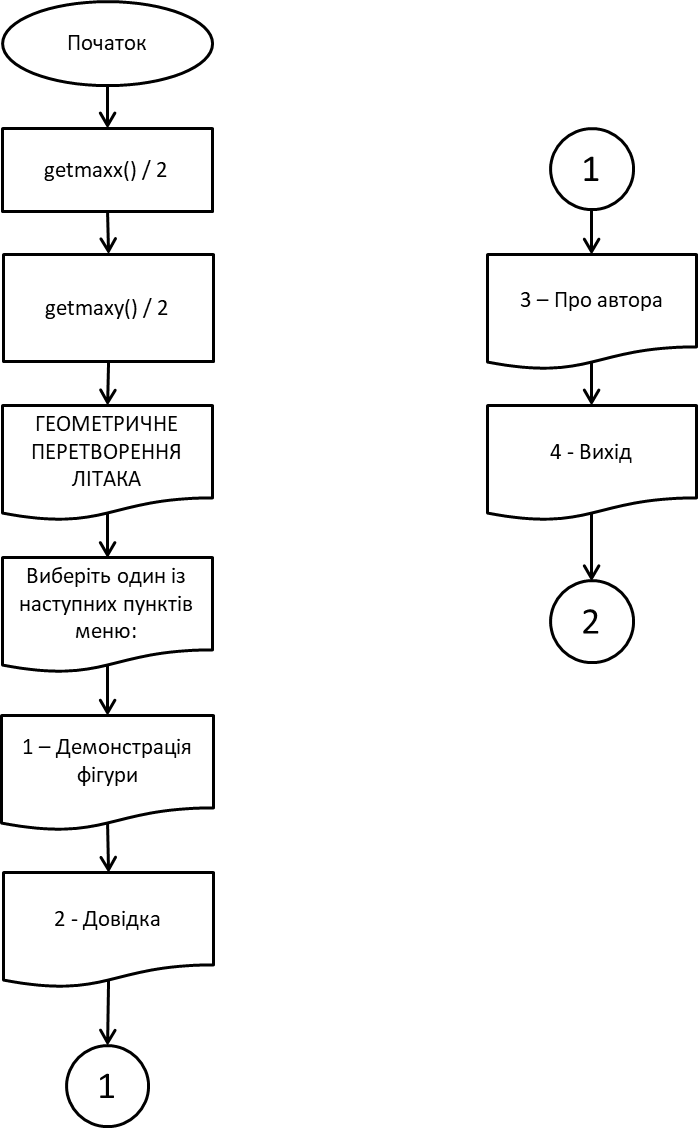
InitPlane(Plane[0].x, Plane[0].y, Plane[0].z, Rebro);

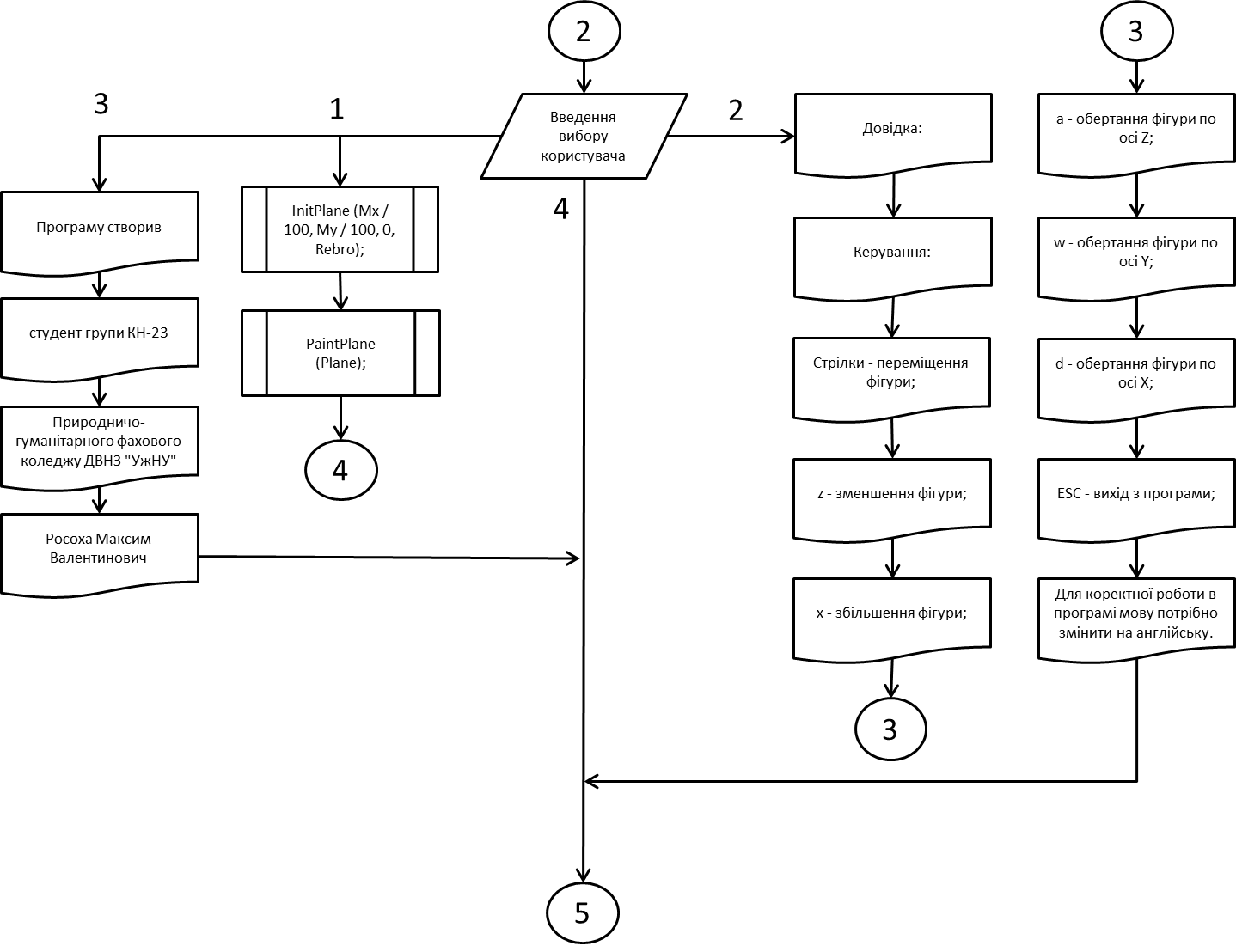
PaintPlane(Plane);

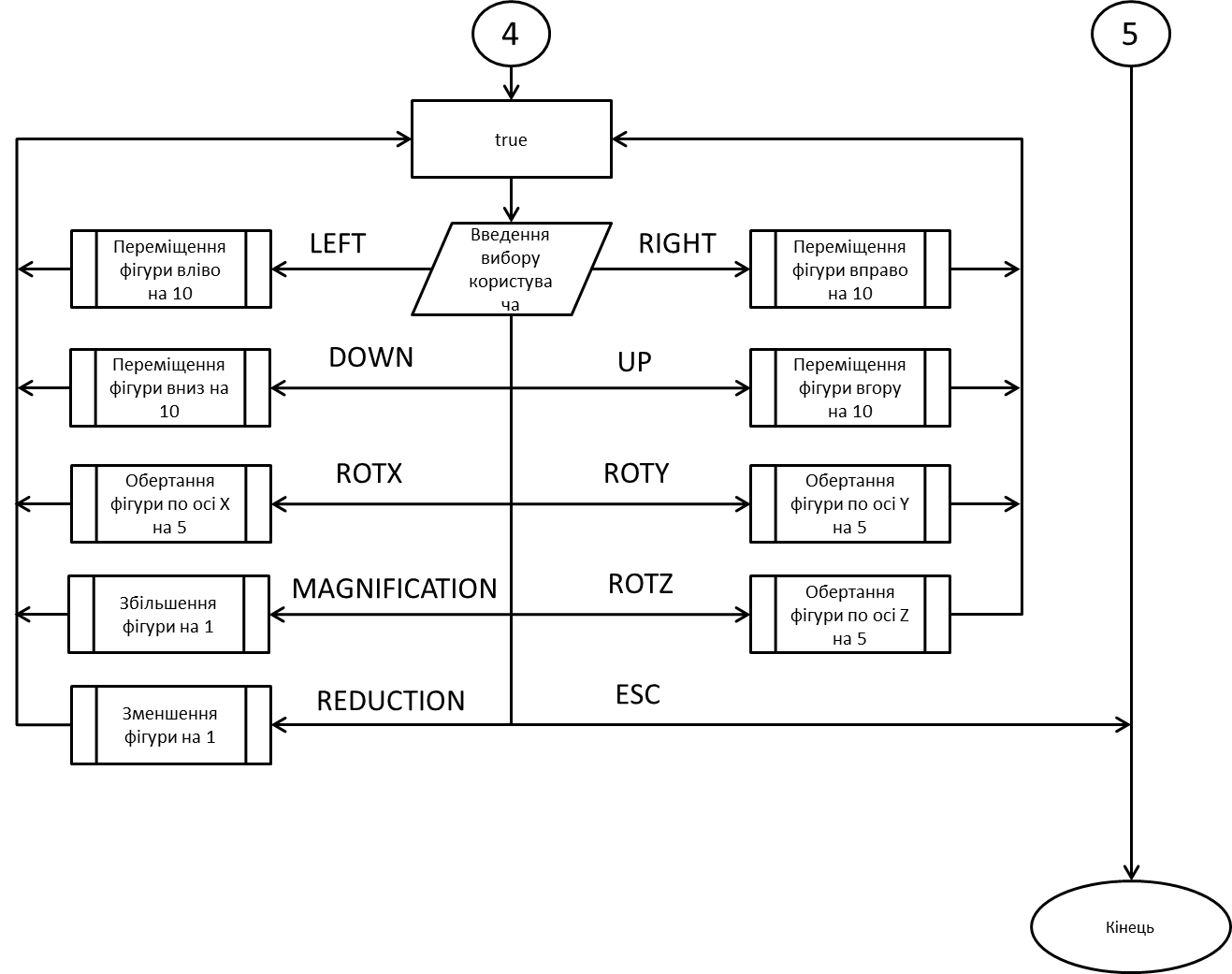
}

**Додаток 2. Блок-схеми**

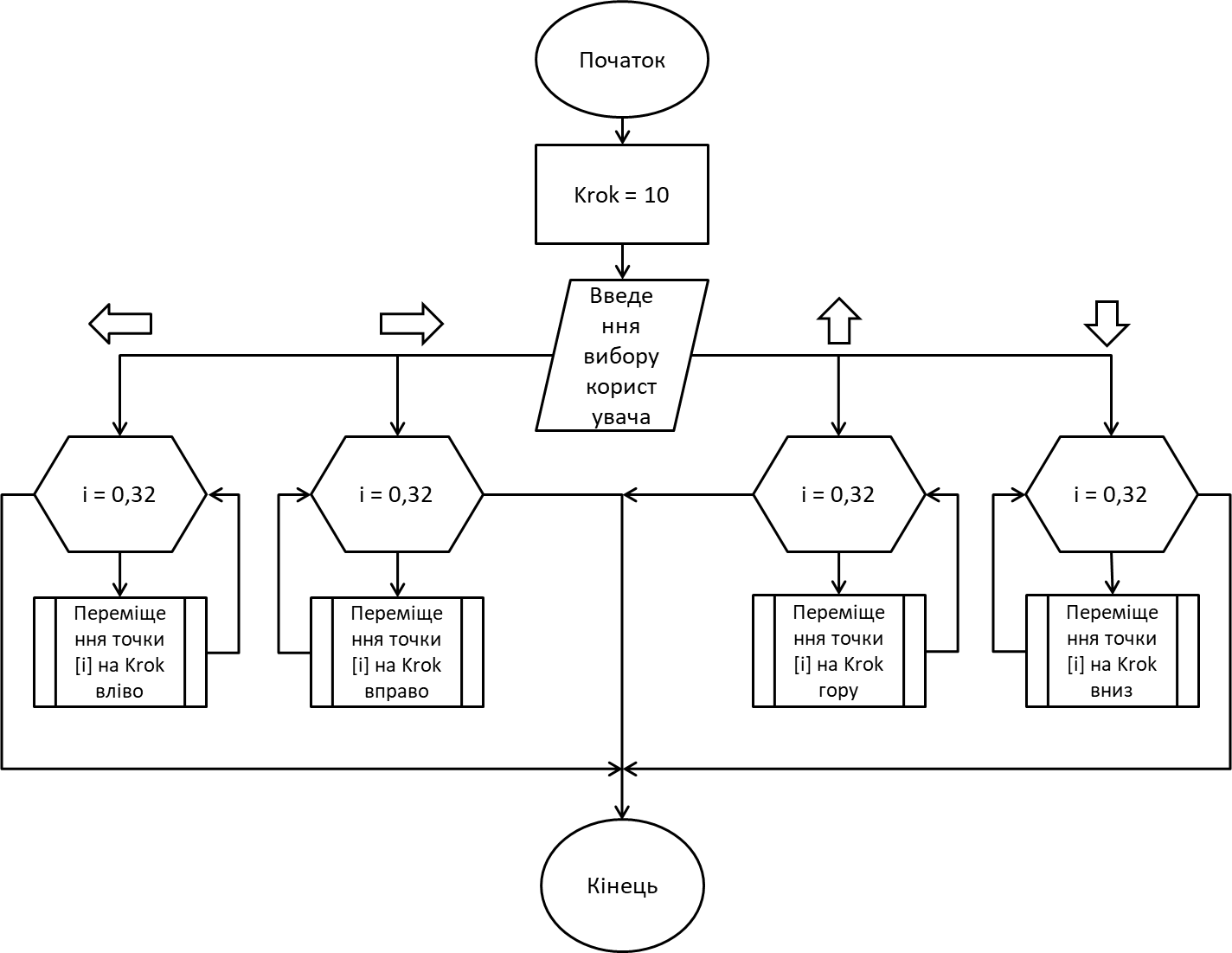
1. Головна функція



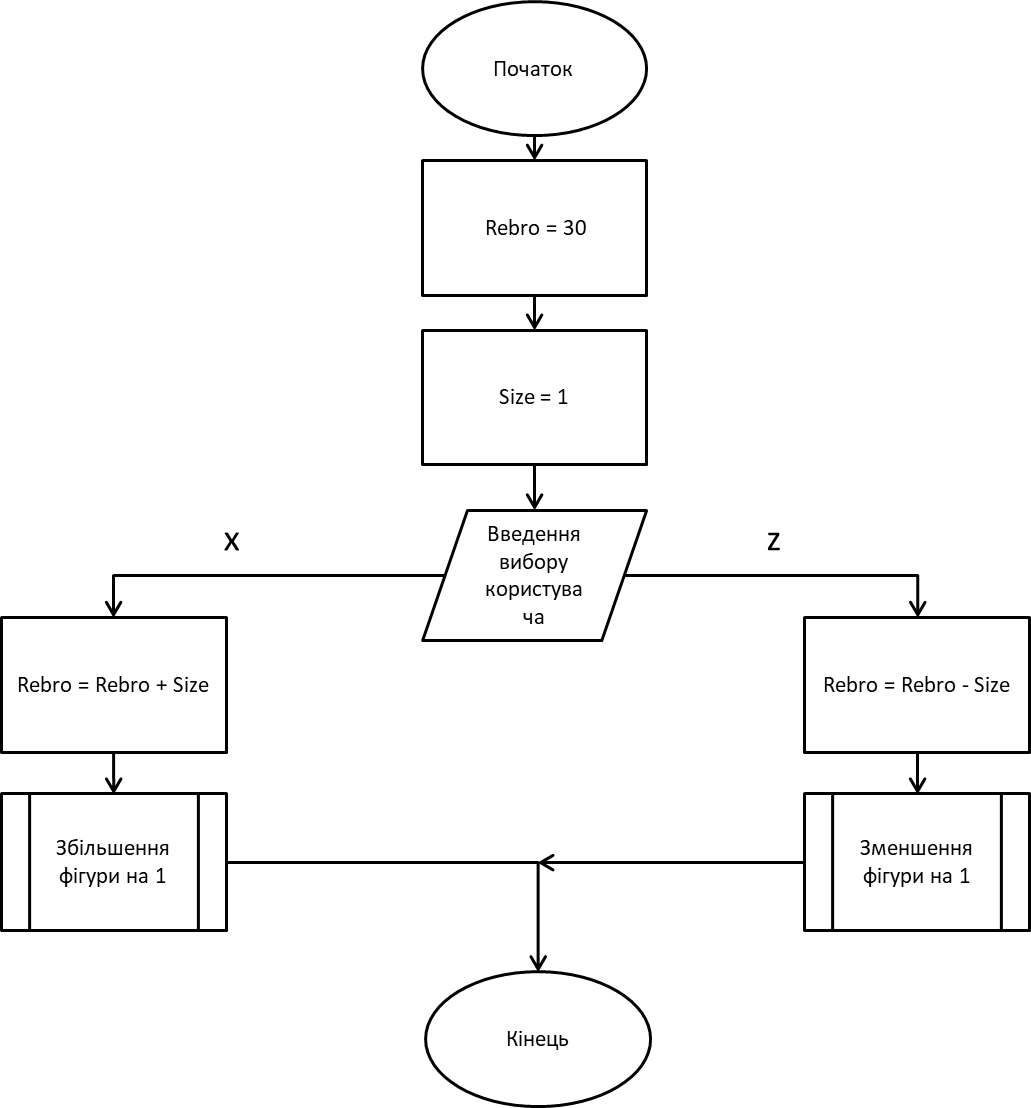




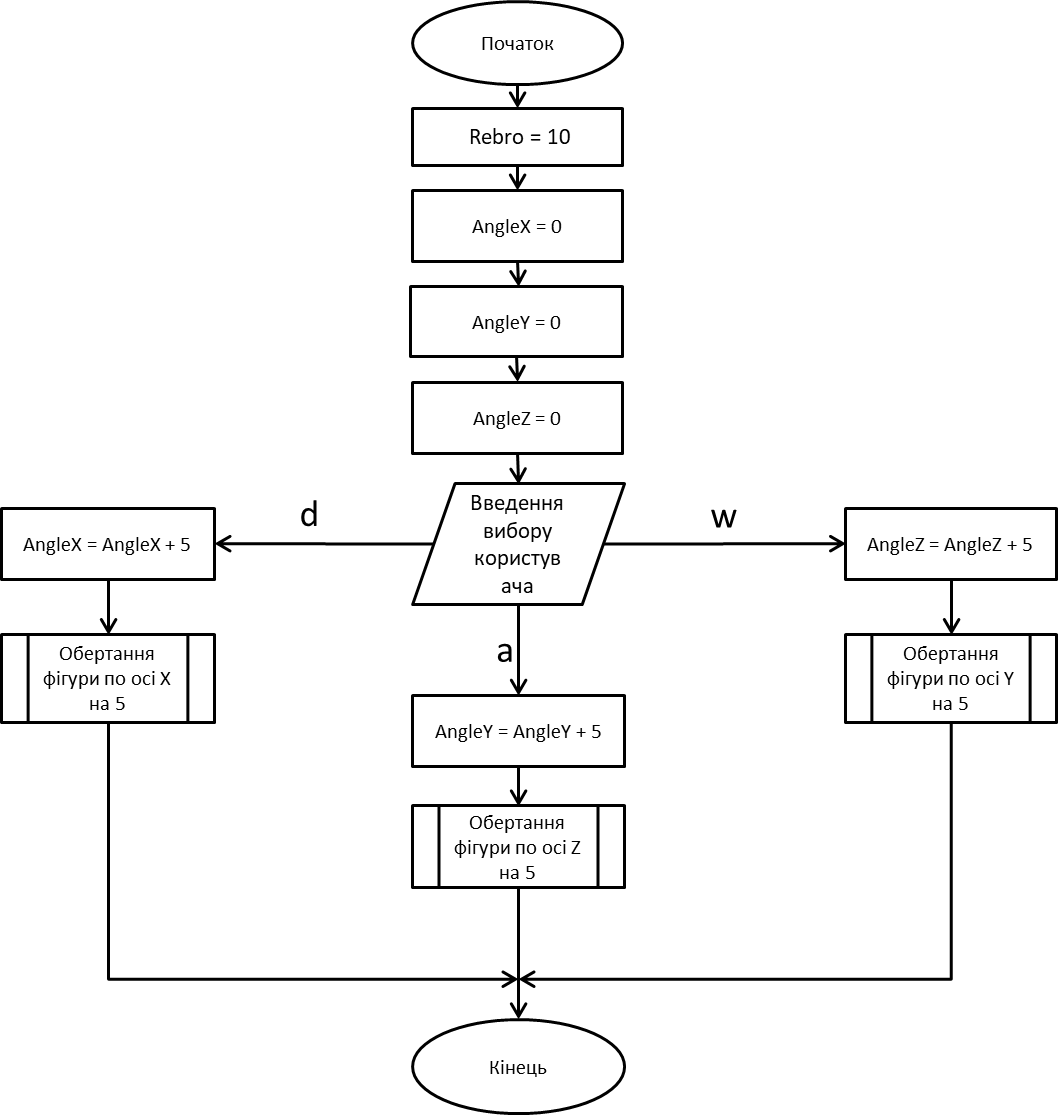
1. Переміщення



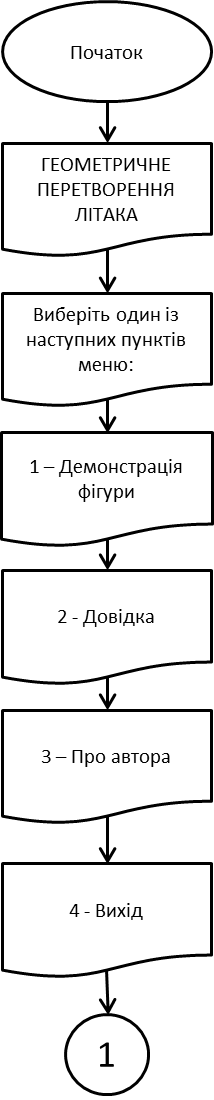
1. Маштабування

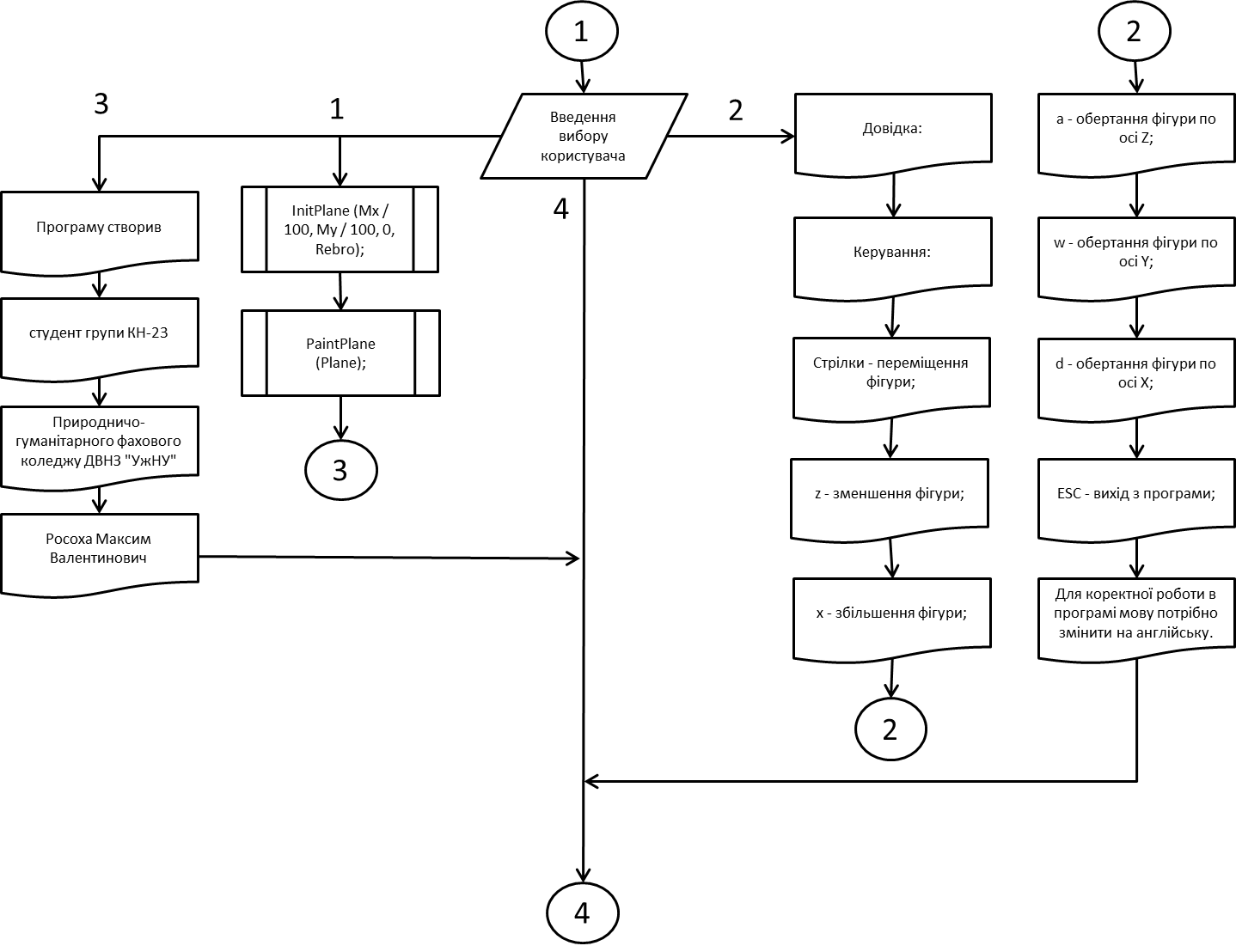


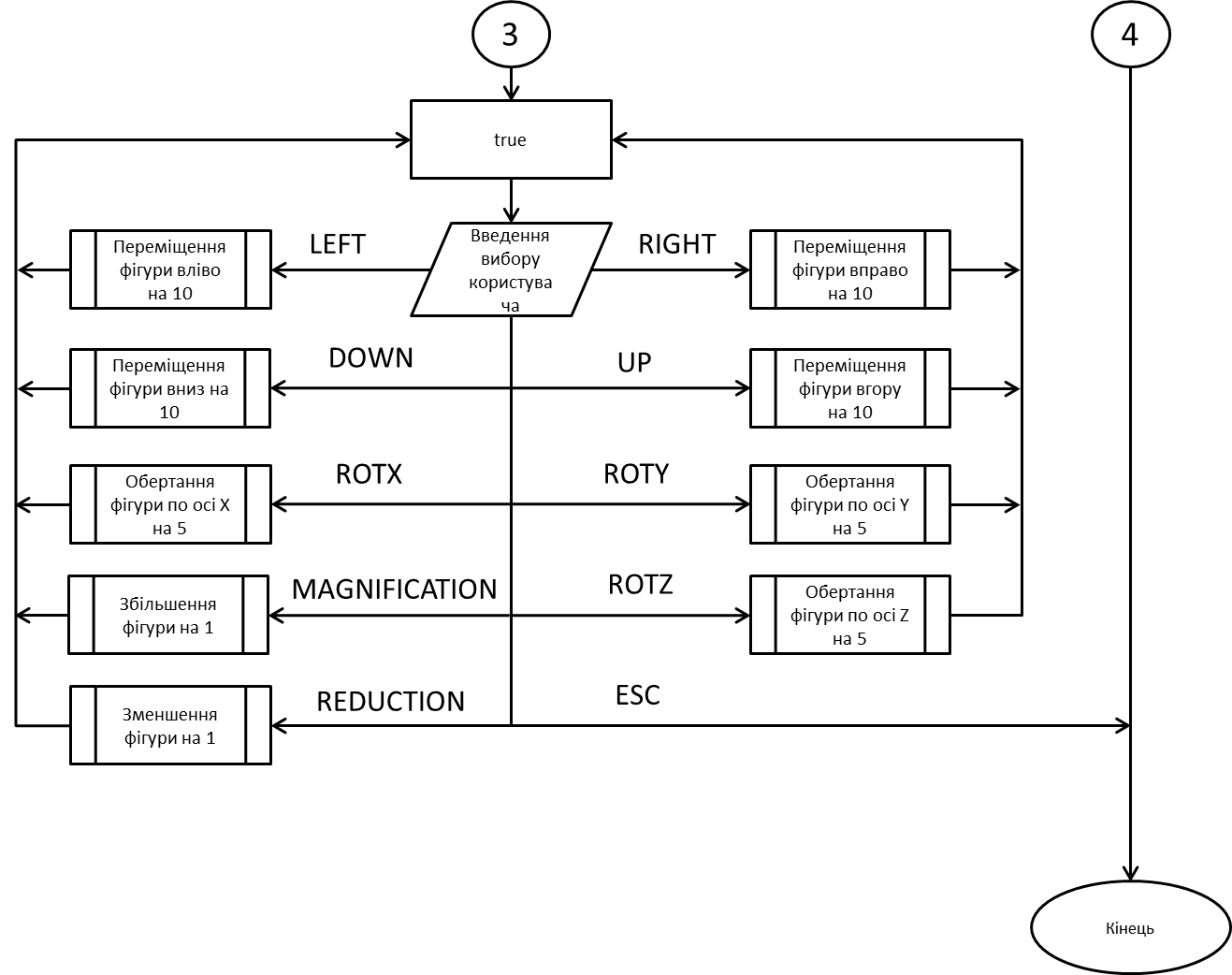
1. Обертання



1. Меню







**Щоденник практики**

|  |  |
| --- | --- |
| Дата | Зміст завдання |
| 07.12.2020 | Первинний інструктаж з техніки безпеки. |
| 07.12.2020 | Ознайомлення з практичним завданням. Підготовка інформаційного середовища до поставленої задачі. |
| 08.12.2020 | Побудова математичної моделі об’єкту. Визначення способу збереження математичної моделі в пам’яті ЕОМ для подальшої обробки. Поняття проекції |
| 09.12.2020 | Побудова алгоритму вирішення поставленої задачі (рівень абстракції — функції) |
| 09.12.2020 | Побудова алгоритму вирішення підзадач, теоретичне обґрунтування |
| 10.12.2020 | Побудова графу сценаріїв роботи програми та розробка меню користувача |
| 11.12.2020 | Розробка меню користувача. |
| 11.12.2020 | Розробка структури програми та тестування меню |
| 11.12.2020 | Реалізація функцій переміщення об’єкту на екрані (горизонтальні та вертикальні зсуви). Тестування |
| 11.12.2020 | Реалізація функцій масштабування об’єкту (збільшення, зменшення). Тестування |
| 15.12.2020 | Реалізація функцій повороту об’єкту. Тестування |
| 15.12.2020 | Тестування повного програмного продукту |
| 17.12.2020 | Підготовка супроводжувальної документації |
| 17.12.2020 | Підготовка керівництва користувача. |
| 15-18.12.2020 | Оформлення звіту з навчальної практики. |
| 18.12.2020 | Захист |