Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Вятский государственный университет»

Колледж ВятГУ

**ОТЧЕТ**

**ПО ДОМАШНЕЙ КОНТРОЛЬНОЙ РАБОТЕ №1**

**«Исследование фракталов»**

**ПО МДК 05.02 «Анализ кода информационных систем»**

Выполнил: студент учебной группы

ИСПк-202-52-00

Карманова Мария Игоревна

Преподаватель:

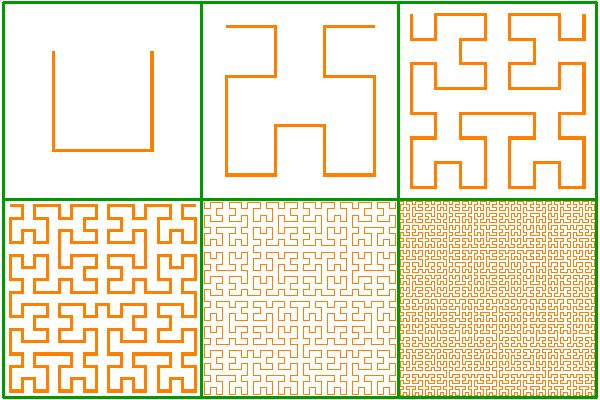
Сергеева Елизавета Григорьевна

Киров

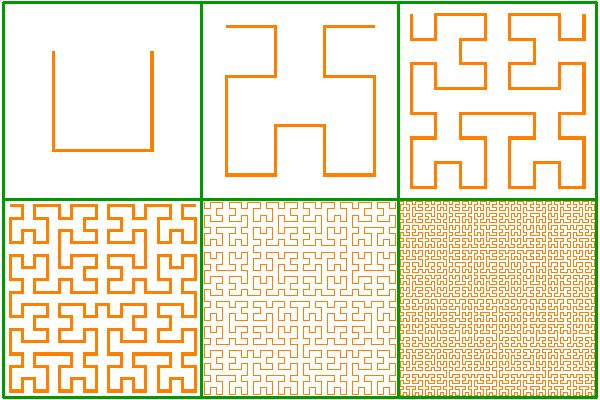
2024

1. Цель работы – получение навыков реализации алгоритмов с рекурсивными вычислениями, знакомство с фракталами.
2. Задание (вариант 10):
3. Написать программу для визуализации фрактала «кривая Гильберта».
4. Предусмотреть возможность масштабирования, изменения глубины прорисовки и перемещения полученной фигуры.
5. Построение множества ломанных, образующих фрактал должно осуществляться в отдельном модуле.
6. Описание алгоритма

Основным элементом кривой Гильберта является П-образный элемент.



Алгоритм построения



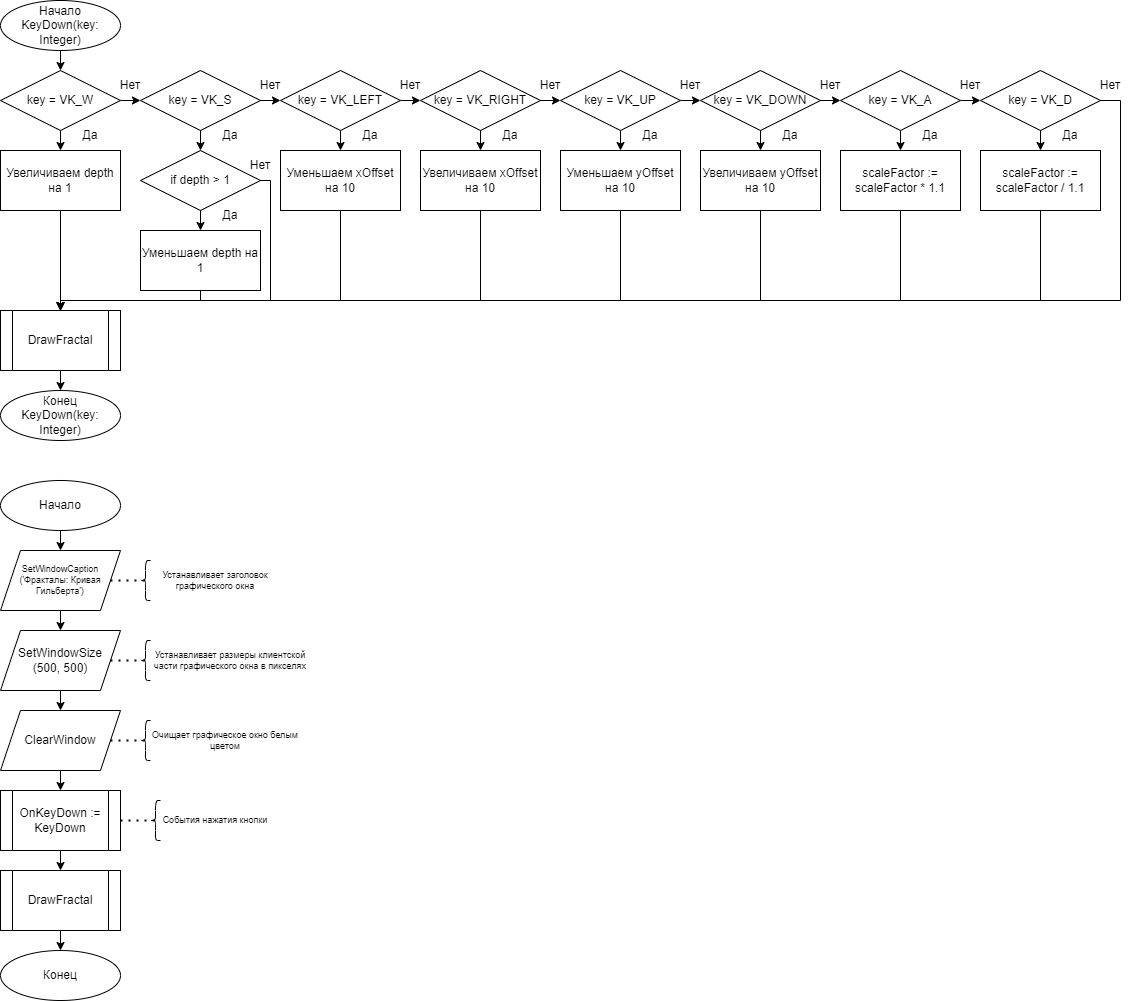
1. Начните с отрезка (или в другом некотором простом случае) и представьте его как кривую Гильберта нулевого порядка.

2. Поверните эту кривую на 90 градусов и отразите ее относительно осей координат.

3. Склейте полученную кривую с исходной, образовав кривую Гильберта первого порядка.

4. Повторите процесс поворота и отражения для кривой Гильберта первого порядка, чтобы получить кривую Гильберта второго порядка.

1. Схема алгоритма с комментариями



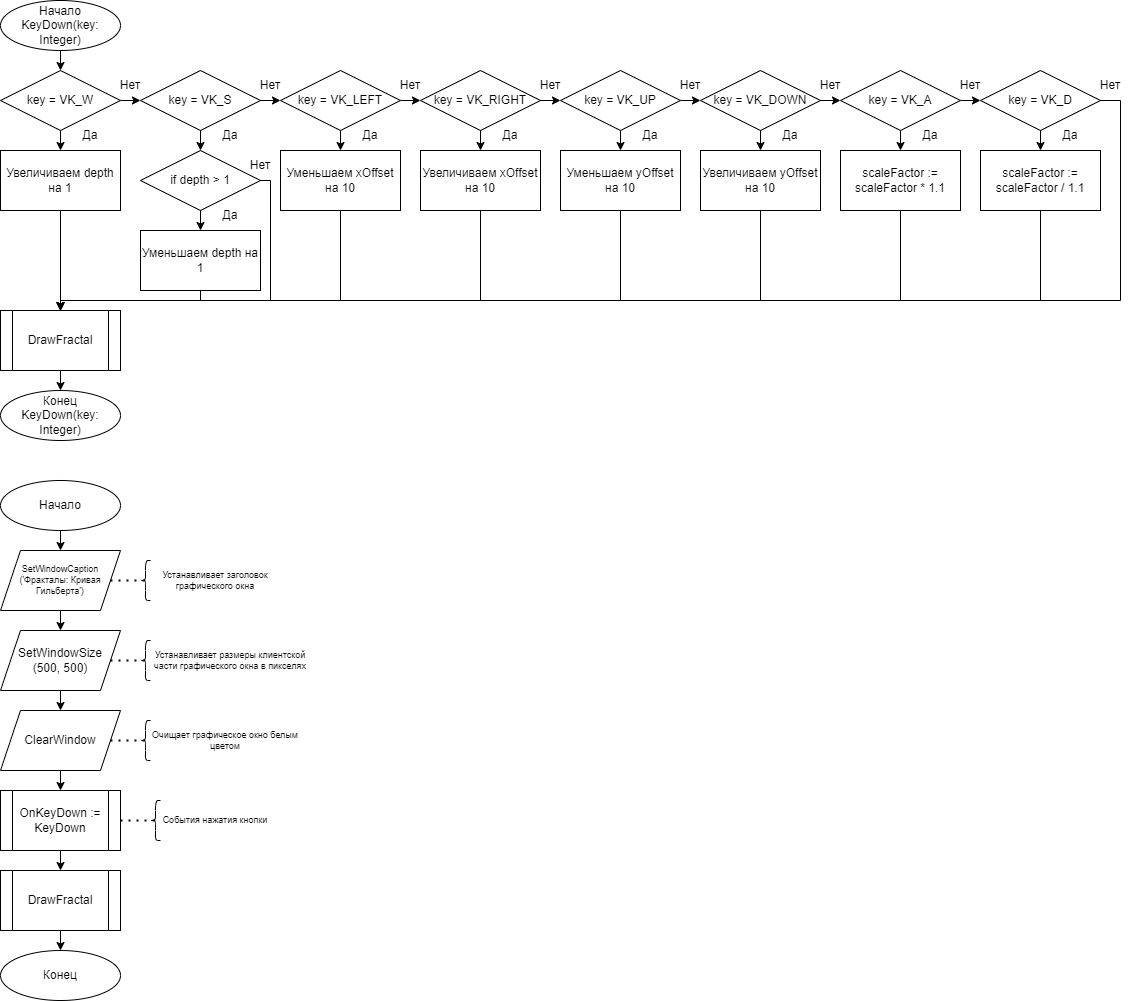


Рисунок 1 – алгоритм основной программы

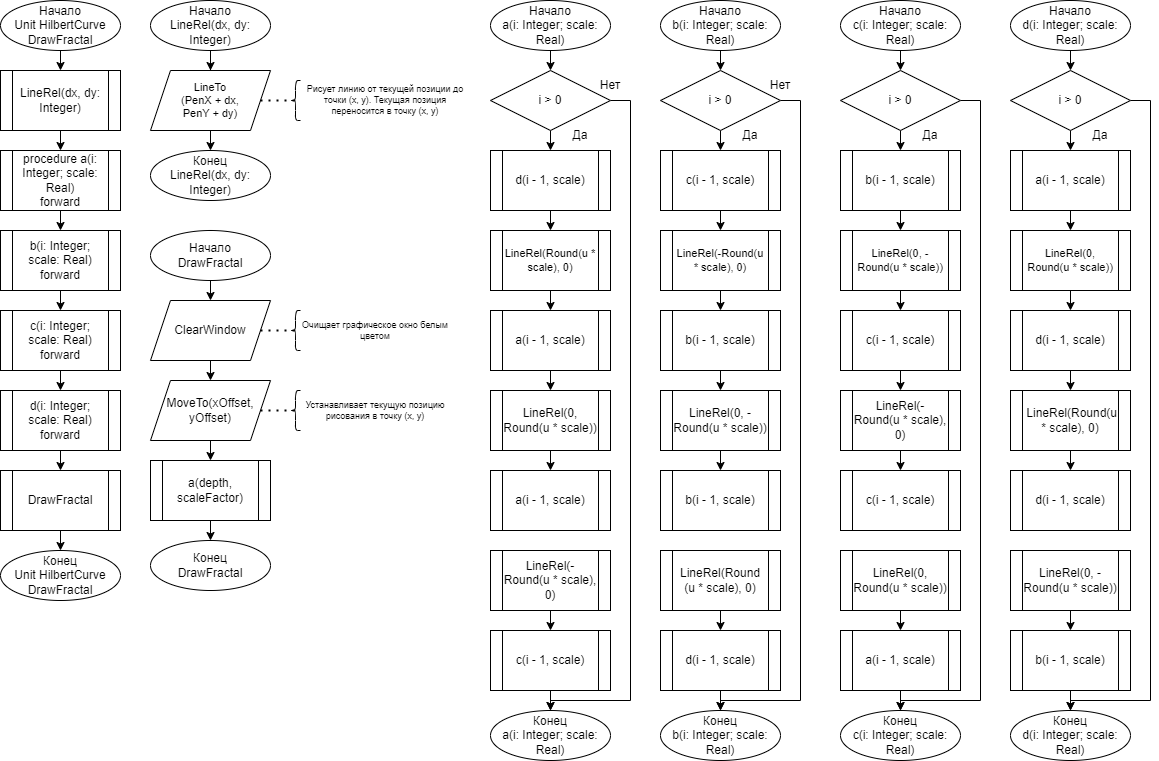


Рисунок 2 – алгоритм модуля фрактала

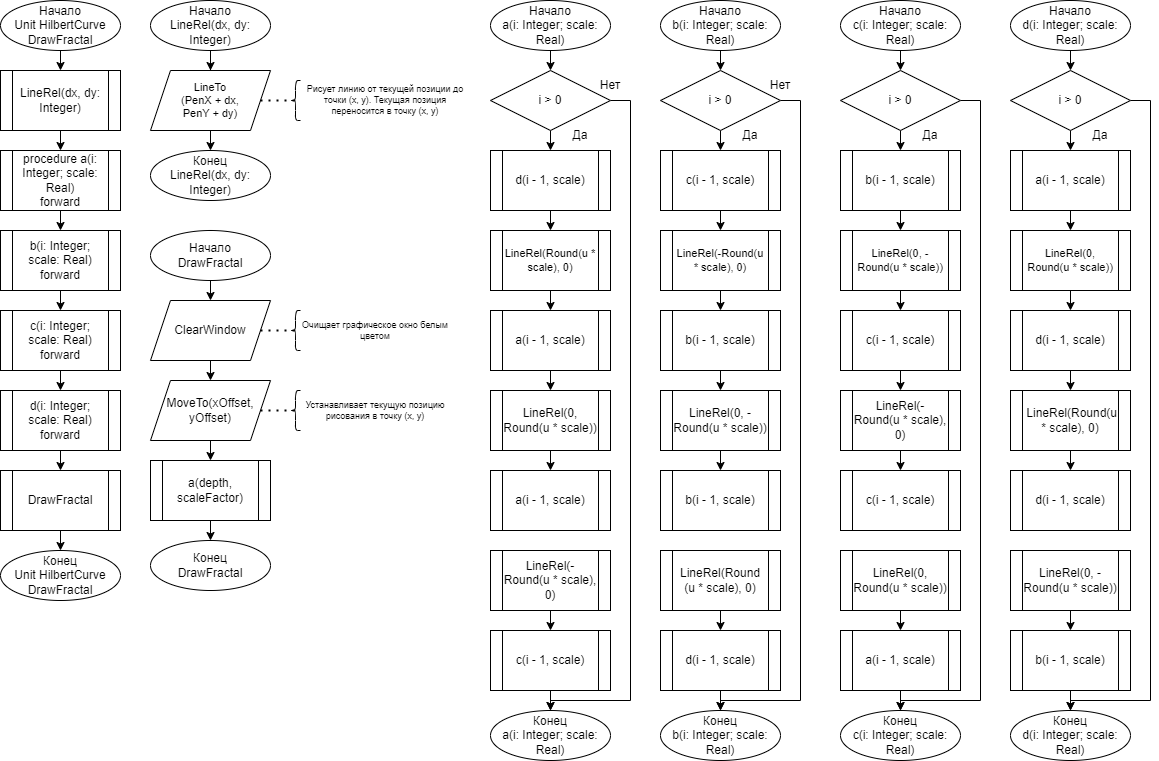


Рисунок 3 – алгоритм процедуры LineRel

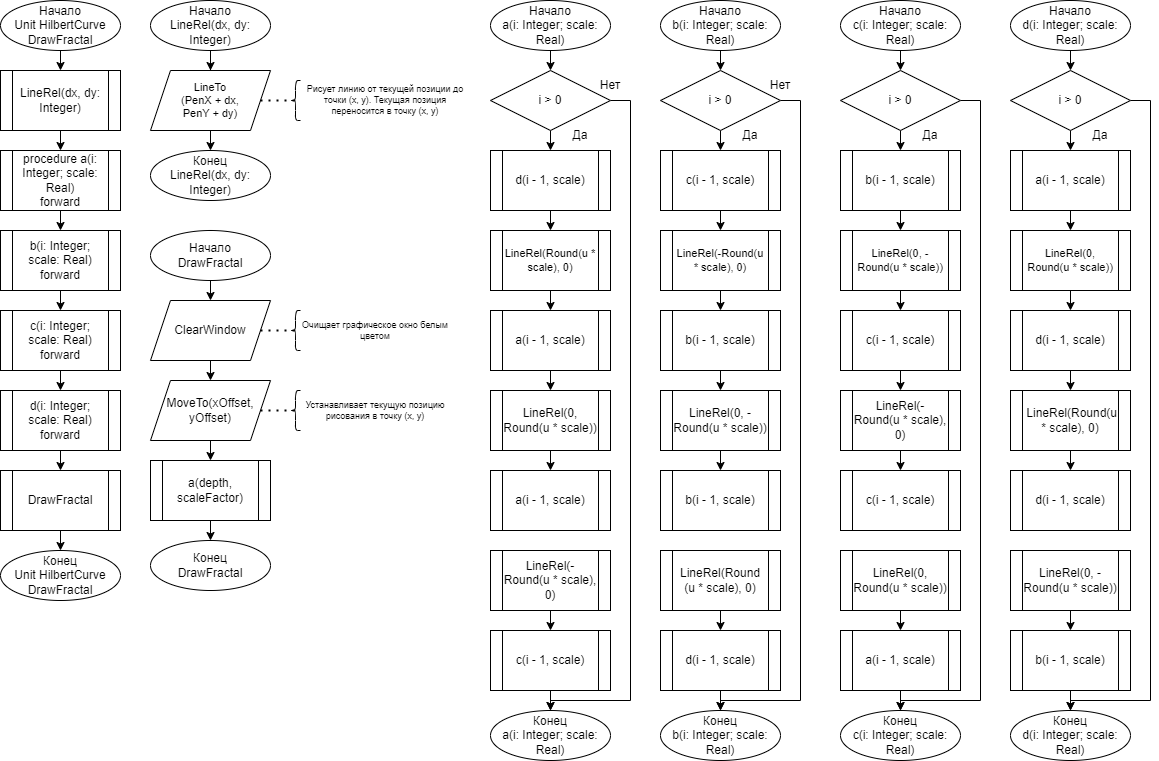


Рисунок 4 – алгоритм процедур a, b, c, d

1. Код программы

**uses** GraphABC, HilbertCurve;

**procedure** KeyDown(key: Integer);

**begin**

**case** key **of**

VK\_W: Inc(depth); //Кнопка W - увеличить глубину

VK\_S: **if** depth > 1 **then** Dec(depth); //Кнопка S - уменьшить глубину

VK\_LEFT: Dec(xOffset, 10);

VK\_RIGHT: Inc(xOffset, 10);

VK\_UP: Dec(yOffset, 10);

VK\_DOWN: Inc(yOffset, 10);

VK\_A: scaleFactor := scaleFactor \* 1.1; // Кнопка A - увеличиваем масштаб

VK\_D: scaleFactor := scaleFactor / 1.1; // Кнопка D - уменьшаем масштаб

**end**;

DrawFractal;

**end**;

**begin**

SetWindowCaption('Фракталы: Кривая Гильберта');

SetWindowSize(500, 500);

ClearWindow;

OnKeyDown := KeyDown;

DrawFractal;

**end**.

**Unit** HilbertCurve;

**uses** GraphABC;

**const**

u = 10;

p = 1;

**var**

scaleFactor: Real = 1.0;

depth: Integer = p;

xOffset: Integer = 100;

yOffset: Integer = 100;

**procedure** LineRel(dx, dy: Integer);

**begin**

LineTo(PenX + dx, PenY + dy);

**end**;

**procedure** a(i: Integer; scale: Real); **forward**;

**procedure** b(i: Integer; scale: Real); **forward**;

**procedure** c(i: Integer; scale: Real); **forward**;

**procedure** d(i: Integer; scale: Real); **forward**;

**procedure** a(i: Integer; scale: Real);

**begin**

**if** i > 0 **then**

**begin**

d(i - 1, scale);

LineRel(Round(u \* scale), 0);

a(i - 1, scale);

LineRel(0, Round(u \* scale));

a(i - 1, scale);

LineRel(-Round(u \* scale), 0);

c(i - 1, scale);

**end**;

**end**;

**procedure** b(i: Integer; scale: Real);

**begin**

**if** i > 0 **then**

**begin**

c(i - 1, scale);

LineRel(-Round(u \* scale), 0);

b(i - 1, scale);

LineRel(0, -Round(u \* scale));

b(i - 1, scale);

LineRel(Round(u \* scale), 0);

d(i - 1, scale);

**end**;

**end**;

**procedure** c(i: Integer; scale: Real);

**begin**

**if** i > 0 **then**

**begin**

b(i - 1, scale);

LineRel(0, -Round(u \* scale));

c(i - 1, scale);

LineRel(-Round(u \* scale), 0);

c(i - 1, scale);

LineRel(0, Round(u \* scale));

a(i - 1, scale);

**end**;

**end**;

**procedure** d(i: Integer; scale: Real);

**begin**

**if** i > 0 **then**

**begin**

a(i - 1, scale);

LineRel(0, Round(u \* scale));

d(i - 1, scale);

LineRel(Round(u \* scale), 0);

d(i - 1, scale);

LineRel(0, -Round(u \* scale));

b(i - 1, scale);

**end**;

**end**;

**procedure** DrawFractal;

**begin**

ClearWindow;

MoveTo(xOffset, yOffset);

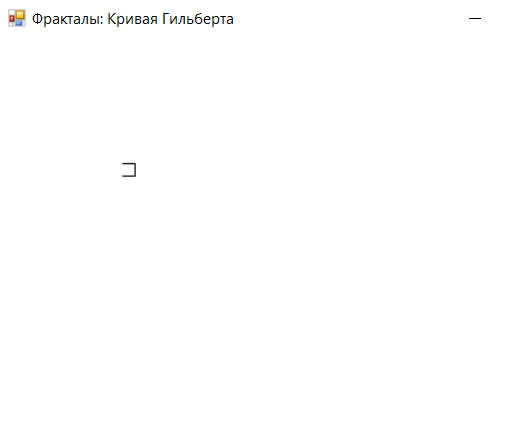
a(depth, scaleFactor);

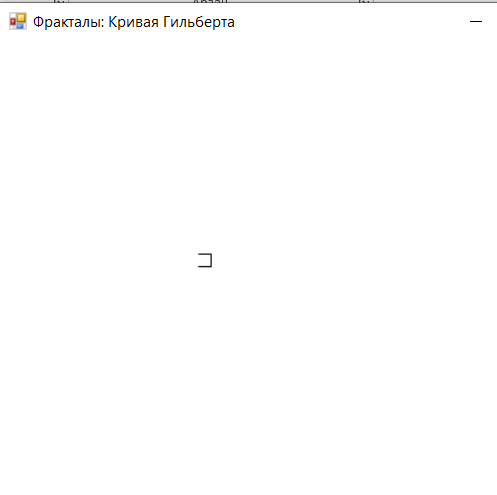
**end**;

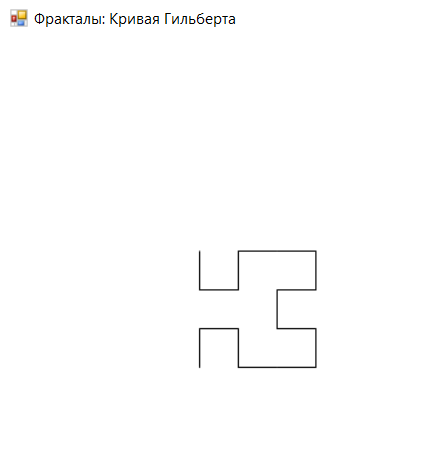
**begin**

**end**.

1. Результат выполнения программы







<Fractal.pas>

1. Вывод

В ходе данной работы были получены навыки реализации алгоритма с рекурсивными вычислениями, была проведена работа с фракталом «Кривая Гильберта». Был прописан отдельный модуль с множеством ломанных, образующих фрактал и реализована возможность масштабирования, изменения глубины прорисовки и перемещения.

В результате работы программа была успешно реализована.