**Федеральное государственное бюджетное образовательное учреждение**

**высшего образования**

**«Российский экономический университет имени Г.В. Плеханова»**

**Высшая школа кибертехнологий, математики и статистики**

**Базовая кафедра цифровой экономики института развития информационного общества**

**Индивидуальное задание №1**

по дисциплине «Компьютерная графика»

**Выполнила:**

студент группы 15.11Д-МО10/20б

очной формы обучения

Высшей школы кибертехнологий,

математики и статистики

Исаева Диана Фармановна

Москва - 2023 г.

# Задание

Требуется разработать программу, отображающую фигуру, в соответствии с вариантом работы (рис. 1). Фигура должна отображаться функциями, реализованными вами в соответствии с базовыми алгоритмами компьютерной графики. Например, если фигура состоит из линий, то должна использоваться самостоятельно написанная функция, использующая для построения алгоритм Брезенхэма, если фигура состоит из окружностей, то функция окружности, написанная с применением алгоритма Брезенхэма для построения окружностей.

Второй частью работы является задание движения данной фигуры, с использованием аффинных преобразований на плоскости, в соответствии с вариантом работы (рис. 1). Движение производится нажатием левой кнопки мыши. Переключение аффинных преобразований осуществляется нажатием правой кнопки мыши.

**ВАЖНОЕ ТРЕБОВАНИЕ!** При движении объекта новое изображение должно строиться в дополнительном буфере, а затем копироваться в экранную область для исключения эффекта мерцания (данная часть дорабатывается после написания программы).

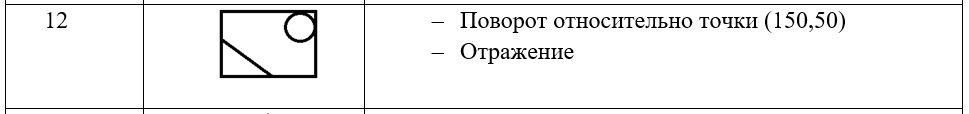


Рисунок 1 Вариант работы

# Описание работы алгоритма

# Текст программы и скрины работы алгоритма

// ИДЗ\_1.cpp : Определяет точку входа для приложения.

//

#include <math.h>

#include <windows.h>

#include "framework.h"

#include "ИДЗ\_1.h"

#define MAX\_LOADSTRING 100

// Глобальные переменные:

HINSTANCE hInst; // текущий экземпляр

WCHAR szTitle[MAX\_LOADSTRING]; // Текст строки заголовка

WCHAR szWindowClass[MAX\_LOADSTRING]; // имя класса главного окна

// Отправить объявления функций, включенных в этот модуль кода:

ATOM MyRegisterClass(HINSTANCE hInstance);

BOOL InitInstance(HINSTANCE, int);

LRESULT CALLBACK WndProc(HWND, UINT, WPARAM, LPARAM);

INT\_PTR CALLBACK About(HWND, UINT, WPARAM, LPARAM);

int APIENTRY wWinMain(\_In\_ HINSTANCE hInstance,

\_In\_opt\_ HINSTANCE hPrevInstance,

\_In\_ LPWSTR lpCmdLine,

\_In\_ int nCmdShow)

{

UNREFERENCED\_PARAMETER(hPrevInstance);

UNREFERENCED\_PARAMETER(lpCmdLine);

// TODO: Разместите код здесь.

// Инициализация глобальных строк

LoadStringW(hInstance, IDS\_APP\_TITLE, szTitle, MAX\_LOADSTRING);

LoadStringW(hInstance, IDC\_MY1, szWindowClass, MAX\_LOADSTRING);

MyRegisterClass(hInstance);

// Выполнить инициализацию приложения:

if (!InitInstance (hInstance, nCmdShow))

{

return FALSE;

}

HACCEL hAccelTable = LoadAccelerators(hInstance, MAKEINTRESOURCE(IDC\_MY1));

MSG msg;

// Цикл основного сообщения:

while (GetMessage(&msg, nullptr, 0, 0))

{

if (!TranslateAccelerator(msg.hwnd, hAccelTable, &msg))

{

TranslateMessage(&msg);

DispatchMessage(&msg);

}

}

return (int) msg.wParam;

}

//

// ФУНКЦИЯ: MyRegisterClass()

//

// ЦЕЛЬ: Регистрирует класс окна.

//

ATOM MyRegisterClass(HINSTANCE hInstance)

{

WNDCLASSEXW wcex;

wcex.cbSize = sizeof(WNDCLASSEX);

wcex.style = CS\_HREDRAW | CS\_VREDRAW;

wcex.lpfnWndProc = WndProc;

wcex.cbClsExtra = 0;

wcex.cbWndExtra = 0;

wcex.hInstance = hInstance;

wcex.hIcon = LoadIcon(hInstance, MAKEINTRESOURCE(IDI\_MY1));

wcex.hCursor = LoadCursor(nullptr, IDC\_ARROW);

wcex.hbrBackground = (HBRUSH)(COLOR\_WINDOW+1);

wcex.lpszMenuName = MAKEINTRESOURCEW(IDC\_MY1);

wcex.lpszClassName = szWindowClass;

wcex.hIconSm = LoadIcon(wcex.hInstance, MAKEINTRESOURCE(IDI\_SMALL));

return RegisterClassExW(&wcex);

}

//

// ФУНКЦИЯ: InitInstance(HINSTANCE, int)

//

// ЦЕЛЬ: Сохраняет маркер экземпляра и создает главное окно

//

// КОММЕНТАРИИ:

//

// В этой функции маркер экземпляра сохраняется в глобальной переменной, а также

// создается и выводится главное окно программы.

//

BOOL InitInstance(HINSTANCE hInstance, int nCmdShow)

{

hInst = hInstance; // Сохранить маркер экземпляра в глобальной переменной

HWND hWnd = CreateWindowW(szWindowClass, szTitle, WS\_OVERLAPPEDWINDOW,

CW\_USEDEFAULT, 0, CW\_USEDEFAULT, 0, nullptr, nullptr, hInstance, nullptr);

if (!hWnd)

{

return FALSE;

}

ShowWindow(hWnd, nCmdShow);

UpdateWindow(hWnd);

return TRUE;

}

class Figure

{

public:

// Функция рисования прямоугольника с кругом и линией

void Draw(HWND hwnd)

{

PAINTSTRUCT ps;

HDC hdc = BeginPaint(hwnd, &ps);

// Рисование прямоугольника

Rectangle(hdc, 50, 100, 400, 300);

// Рисование круга в правом верхнем углу

DrawCircle(hdc, 350, 150, 50);

// Рисование линии в левом нижнем углу

DrawLine(hdc, 50, 200, 200, 300);

EndPaint(hwnd, &ps);

}

// Функция рисования окружности по алгоритму Брезенхэма

void DrawCircle(HDC hdc, int x0, int y0, int radius)

{

int x = 0, y = radius;

int d = 3 - 2 \* radius;

while (x <= y)

{

SetPixel(hdc, x0 + x, y0 + y, RGB(0, 0, 255));

SetPixel(hdc, x0 - x, y0 + y, RGB(0, 255, 0));

SetPixel(hdc, x0 + x, y0 - y, RGB(255, 0, 0));

SetPixel(hdc, x0 - x, y0 - y, RGB(255, 255, 255));

SetPixel(hdc, x0 + y, y0 + x, RGB(0, 255, 255));

SetPixel(hdc, x0 - y, y0 + x, RGB(255, 0, 255));

SetPixel(hdc, x0 + y, y0 - x, RGB(255, 255, 0));

SetPixel(hdc, x0 - y, y0 - x, RGB(192, 192, 192));

if (d < 0)

d += 4 \* x++ + 6;

else

d += 4 \* (x++ - y--) + 10;

}

}

// Функция рисования линии по алгоритму Брезенхэма

void DrawLine(HDC hdc, int x1, int y1, int x2, int y2)

{

int dx = abs(x2 - x1);

int dy = abs(y2 - y1);

int sx = (x1 < x2) ? 1 : -1;

int sy = (y1 < y2) ? 1 : -1;

int err = dx - dy;

int e2;

while (true)

{

SetPixel(hdc, x1, y1, RGB(0, 0, 0));

if (x1 == x2 && y1 == y2)

break;

e2 = 2 \* err;

if (e2 > -dy)

{

err -= dy;

x1 += sx;

}

if (e2 < dx)

{

err += dx;

y1 += sy;

}

}

}

};

//

// ФУНКЦИЯ: WndProc(HWND, UINT, WPARAM, LPARAM)

//

// ЦЕЛЬ: Обрабатывает сообщения в главном окне.

//

// WM\_COMMAND - обработать меню приложения

// WM\_PAINT - Отрисовка главного окна

// WM\_DESTROY - отправить сообщение о выходе и вернуться

//

//

LRESULT CALLBACK WndProc(HWND hWnd, UINT message, WPARAM wParam, LPARAM lParam)

{

static int cxClient, cyClient;

static POINT ptOld;

static int iTransformation = 0; // начальное преобразование - поворот

static HDC hdc, hdcMem;

static HBITMAP hBitmap, hBitmapOld;

static int x = 150, y = 50; // координаты центра фигуры

static int r = 50; // радиус фигуры

static int angle = 30; // угол поворота фигуры

static int sign = 1; // знак отражения фигуры

switch (message)

{

case WM\_LBUTTONDOWN:

ptOld.x = LOWORD(lParam);

ptOld.y = HIWORD(lParam);

break;

case WM\_LBUTTONUP:

{

int dx = LOWORD(lParam) - ptOld.x;

int dy = HIWORD(lParam) - ptOld.y;

// вычисление новых координат центра фигуры

x += dx;

y += dy;

// отрисовка фигуры в буфере

Figure figure;

PAINTSTRUCT ps;

HDC hdc = BeginPaint(hWnd, &ps);

// Рисование прямоугольника

Rectangle(hdc, 50 - r, 100 - r, 400 + r, 300 + r);

// Рисование круга в правом верхнем углу

figure.DrawCircle(hdc, 350 - r, 150 - r, 50 + r);

// Рисование линии в левом нижнем углу

figure.DrawLine(hdc, 50 - r, 200 - r, 200 + r, 300 + r);

EndPaint(hWnd, &ps);

// перенос фигуры с буфера на экран

hdc = GetDC(hWnd);

BitBlt(hdc, 0, 0, cxClient, cyClient, hdcMem, 0, 0, SRCCOPY);

ReleaseDC(hWnd, hdc);

}

break;

case WM\_RBUTTONDOWN:

iTransformation = (iTransformation + 1) % 2; // переключение преобразования

break;

case WM\_MOUSEMOVE:

if (wParam & MK\_LBUTTON)

{

int dx = LOWORD(lParam) - ptOld.x;

int dy = HIWORD(lParam) - ptOld.y;

// вычисление угла поворота фигуры

angle = (int)(atan2(y - ptOld.y, ptOld.x - x) / 3.141592654 \* 180);

// отрисовка старой фигуры в буфере

PAINTSTRUCT ps;

HDC hdc = BeginPaint(hWnd, &ps);

Figure figure;

figure.Draw(hWnd);

// поворот фигуры в буфере

SelectObject(hdcMem, GetStockObject(BLACK\_BRUSH));

SetGraphicsMode(hdcMem, GM\_ADVANCED);

XFORM xform;

xform.eM11 = (FLOAT)cos((angle + 90) / 180.0 \* 3.141592654);

xform.eM12 = (FLOAT)sin((angle + 90) / 180.0 \* 3.141592654);

xform.eM21 = (FLOAT)-sin((angle + 90) / 180.0 \* 3.141592654);

xform.eM22 = (FLOAT)cos((angle + 90) / 180.0 \* 3.141592654);

xform.eDx = (FLOAT)ptOld.x;

xform.eDy = (FLOAT)ptOld.y;

SetWorldTransform(hdcMem, &xform);

// отражение фигуры в буфере

if (sign == -1)

{

SelectObject(hdcMem, GetStockObject(WHITE\_BRUSH));

// Рисование прямоугольника

Rectangle(hdc, 50 - r, 100 - r, 400 + r, 300 + r);

// Рисование круга в правом верхнем углу

figure.DrawCircle(hdc, 350 - r, 150 - r, 50 + r);

// Рисование линии в левом нижнем углу

figure.DrawLine(hdc, 50 - r, 200 - r, 200 + r, 300 + r);

EndPaint(hWnd, &ps);

}

SelectObject(hdcMem, GetStockObject(BLACK\_BRUSH));

SetGraphicsMode(hdcMem, GM\_ADVANCED);

xform.eM11 = (FLOAT)sign;

xform.eM12 = 0;

xform.eM21 = 0;

xform.eM22 = (FLOAT)sign;

xform.eDx = (FLOAT)(2 \* x - ptOld.x);

xform.eDy = 0;

SetWorldTransform(hdcMem, &xform);

// перенос фигуры с буфера на экран

hdc = GetDC(hWnd);

BitBlt(hdc, 0, 0, cxClient, cyClient, hdcMem, 0, 0, SRCCOPY);

ReleaseDC(hWnd, hdc);

// сохранение старой позиции мыши

ptOld.x = LOWORD(lParam);

ptOld.y = HIWORD(lParam);

}

break;

case WM\_KEYUP:

if (wParam == VK\_ESCAPE)

{

// очистка экрана и выход из программы

SelectObject(hdcMem, GetStockObject(WHITE\_BRUSH));

SelectObject(hdcMem, GetStockObject(WHITE\_PEN));

Rectangle(hdcMem, 0, 0, cxClient, cyClient);

hdc = GetDC(hWnd);

BitBlt(hdc, 0, 0, cxClient, cyClient, hdcMem, 0, 0, SRCCOPY);

ReleaseDC(hWnd, hdc);

SendMessage(hWnd, WM\_DESTROY, 0, 0);

}

else if (wParam == VK\_RETURN)

{

// изменение знака отражения фигуры

sign = -sign;

}

break;

case WM\_PAINT:

{

Figure figure;

figure.Draw(hWnd);

}

break;

case WM\_DESTROY:

PostQuitMessage(0);

break;

default:

return DefWindowProc(hWnd, message, wParam, lParam);

}

return 0;

}

// Обработчик сообщений для окна "О программе".

INT\_PTR CALLBACK About(HWND hDlg, UINT message, WPARAM wParam, LPARAM lParam)

{

UNREFERENCED\_PARAMETER(lParam);

switch (message)

{

case WM\_INITDIALOG:

return (INT\_PTR)TRUE;

case WM\_COMMAND:

if (LOWORD(wParam) == IDOK || LOWORD(wParam) == IDCANCEL)

{

EndDialog(hDlg, LOWORD(wParam));

return (INT\_PTR)TRUE;

}

break;

}

return (INT\_PTR)FALSE;

}

Изображение выглядит как снимок экрана, текст, линия, График

Автоматически созданное описание

Рисунок 2 Результат отрисовки фигуры алгоритмом Брезенхэма

Изображение выглядит как текст, снимок экрана, диаграмма, линия

Автоматически созданное описание

Рисунок 3 Результат работы алгоритма отражения фигуры

Изображение выглядит как текст, диаграмма, снимок экрана, линия

Автоматически созданное описание

Рисунок 4 Результат поворота фигуры на 30 градусов относительно точки (150; 50)

# Список литературы

1. Преобразования на плоскости и в пространстве[Электронный ресурс] – Ресурс доступа: <https://studfile.net/preview/6010005/page:12/>