Міністерство освіти і науки України
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра обчислювальної техніки

Лабораторна робота №2

з дисципліни:
«Об’єктно-орієнтоване програмування»

Виконав:
Баран Павло Ю.
студент групи ІП-03
Номер у списку групи: 1

Перевірив:
Порєв В. М.

2021

Розробка графічного редактора об’єктів на C++

Варіант:

1. Статичний масив Shape *pcshape[N];
2. "Гумовий" слід при вводі об’єктів: суцільна лінія червоного кольору.
3. Ввід прямокутника: від центру до одного з кутів.
4. Відображення прямокутника: чорний контур з світло-зеленим заповненням.
5. Ввід еліпсу: по двом протилежним кутам охоплюючого прямокутника.
6. Відображення еліпсу: чорний контур з білим заповненням.
7. Позначка поточного типу об’єкту, що вводиться, в заголовку вікна.

Код програми:

Lab2.cpp

```
#include "framework.h"
#include "Lab2.h"
#include "shape_editor.h"

#define MAX_LOADSTRING 100

// Глобальные переменные:
ShapeObjectsEditor Editor;
HINSTANCE hInst;                                // текущий экземпляр
WCHAR szTitle[MAX_LOADSTRING];                  // Текст строки заголовка
WCHAR szWindowClass[MAX_LOADSTRING];            // имя класса главного окна
```

.....

.....

```
LRESULT CALLBACK WndProc(HWND hWnd, UINT message, WPARAM wParam, LPARAM lParam)
{
    switch (message)
    {
    case WM_LBUTTONDOWN: //натиснуто ліву кнопку миші у клієнтській частині вікна
        Editor.OnLBdown(hWnd);
        break;
    case WM_LBUTTONUP: //відпущено ліву кнопку миші у клієнтській частині вікна
        Editor.OnLBup(hWnd);
```

```
        break;
    case WM_MOUSEMOVE: //пересунуто мишу у клієнтській частині вікна
        Editor.OnMouseMove(hWnd);
        break;
    case WM_PAINT: //потрібно оновлення зображення клієнтської частини вікна
        Editor.OnPaint(hWnd);
        break;
    case WM_INITMENUPOPUP: //позначка пунктів меню
        Editor.OnInitMenuPopup(hWnd);
        break;
    case WM_COMMAND:
        {
            int wmId = LOWORD(wParam);
            // Разобрать выбор в меню:
            switch (wmId)
            {
            case IDM_POINT:
                Editor.StartPointEditor(hWnd); //початок вводу точкових об'єктів
                break;
            case IDM_LINE:
                Editor.StartLineEditor(hWnd); //початок вводу об'єктів-ліній
                break;
            case IDM_RECT:
                Editor.StartRectEditor(hWnd); //початок вводу прямокутників
                break;
            case IDM_ELLIPSE:
                Editor.StartEllipseEditor(hWnd); //початок вводу еліпсів
                break;
            case IDM_ABOUT:
                DialogBox(hInst, MAKEINTRESOURCE(IDD_ABOUTBOX), hWnd, About);
                break;
            case IDM_EXIT:
                DestroyWindow(hWnd);
                break;
            default:
                return DefWindowProc(hWnd, message, wParam, lParam);
            }
        }
        break;
    case WM_DESTROY:
        PostQuitMessage(0);
        break;
    default:
        return DefWindowProc(hWnd, message, wParam, lParam);
    }
    return 0;
}
```

shape.h

```
#pragma once
#include "framework.h"

class Shape
{
	protected:
		long xs1, ys1, xs2, ys2;
	public:
		Shape() : xs1{0}, ys1{0}, xs2{0}, ys2{0} {}
		void Set(long x1, long y1, long x2, long y2);
		virtual void Show(HDC hdc) = 0;
};

class PointShape : public Shape
{
	void Show(HDC hdc);
};

class LineShape : public Shape
{
	void Show(HDC hdc);
};

class RectShape : public Shape
{
	void Show(HDC hdc);
};

class EllipseShape : public Shape
{
	void Show(HDC hdc);
};
```

shape.cpp

```
#include "shape.h"

void Shape::Set(long x1, long y1, long x2, long y2)
{
	xs1 = x1;
	ys1 = y1;
	xs2 = x2;
	ys2 = y2;
}

void PointShape::Show(HDC hdc)
```

```
{
	SetPixel(hdc, xs1, ys1, RGB(0,0,0));
}

void LineShape::Show(HDC hdc)
{
	HBRUSH hBrush, hBrushOld;
	hBrush = (HBRUSH)CreateSolidBrush(RGB(0, 0, 0)); //створюється пензль
	hBrushOld = (HBRUSH)SelectObject(hdc, hBrush);
	MoveToEx(hdc, xs1, ys1, NULL);
	LineTo(hdc, xs2, ys2);
	SelectObject(hdc, hBrushOld); //відновлюється пензль-попередник
	DeleteObject(hBrush);
}

void RectShape::Show(HDC hdc)
{
	HBRUSH hBrush, hBrushOld;
	hBrush = (HBRUSH)CreateSolidBrush(RGB(0, 255, 0)); //створюється пензль
	hBrushOld = (HBRUSH)SelectObject(hdc, hBrush);
	Rectangle(hdc, xs1, ys1, xs2, ys2);
	SelectObject(hdc, hBrushOld); //відновлюється пензль-попередник
	DeleteObject(hBrush);
}

void EllipseShape::Show(HDC hdc)
{
	HBRUSH hBrush, hBrushOld;
	hBrush = (HBRUSH)CreateSolidBrush(RGB(255, 255, 255)); //створюється пензль
	hBrushOld = (HBRUSH)SelectObject(hdc, hBrush);
	Ellipse(hdc, xs1, ys1, xs2, ys2);
	SelectObject(hdc, hBrushOld); //відновлюється пензль-попередник
	DeleteObject(hBrush);
}
```

editor.h

```
#pragma once

class Editor
{
public:
	virtual void OnLBdown(HWND hWnd) = 0;
	virtual void OnLBup(HWND hWnd) = 0;
	virtual void OnMouseMove(HWND hWnd) = 0;
	virtual void OnPaint(HWND hWnd) = 0;
};
```

shape_editor.h

```
#pragma once
#include "shape.h"
#include "editor.h"

class ShapeEditor : public Editor
{
protected:
	int x1, y1, x2, y2;
public:
	ShapeEditor() : x1{0}, y1{0}, x2{0}, y2{0} {}
	void OnLBdown(HWND);
	void OnLBup(HWND);
	void OnMouseMove(HWND);
	void OnPaint(HWND);
	virtual void OnInitMenuPopup(HWND); //додатковий інтерфейсний метод
};

class ShapeObjectsEditor
{
private:
	ShapeEditor *pse = NULL;
public:
	ShapeObjectsEditor();
	~ShapeObjectsEditor();
	void StartPointEditor(HWND);
	void StartLineEditor(HWND);
	void StartRectEditor(HWND);
	void StartEllipseEditor(HWND);
	void OnLBdown(HWND);
	void OnLBup(HWND);
	void OnMouseMove(HWND);
	void OnPaint(HWND);
	void OnInitMenuPopup(HWND);
};

class PointEditor : public ShapeEditor
{
public:
	void OnLBdown(HWND);
	void OnLBup(HWND);
	void OnInitMenuPopup(HWND);
};

class LineEditor : public ShapeEditor
{
public:
	void OnLBdown(HWND);
```

```
	void OnMouseMove(HWND);
	void OnLBup(HWND);
	void OnInitMenuPopup(HWND);
};

class RectEditor : public ShapeEditor
{
public:
	void OnLBdown(HWND);
	void OnMouseMove(HWND);
	void OnLBup(HWND);
	void OnInitMenuPopup(HWND);
};

class EllipseEditor : public ShapeEditor
{
public:
	void OnLBdown(HWND);
	void OnMouseMove(HWND);
	void OnLBup(HWND);
	void OnInitMenuPopup(HWND);
};
```

shape_editor.cpp

```
#include "shape_editor.h"
#include <string>

Shape *pcshape[101];
static int arrayCounter = 0;
static int check = 0;

void ShapeEditor::OnPaint(HWND hWnd)
{
	PAINTSTRUCT ps;
	HDC hdc;
	hdc = BeginPaint(hWnd, &ps);
	for (int i = 0; i < arrayCounter; i++)
		if (pcshape[i])
			pcshape[i]->Show(hdc);
	EndPaint(hWnd, &ps);
}

void ShapeEditor::OnLBdown(HWND hWnd) {
	POINT pt;
	GetCursorPos(&pt);
	ScreenToClient(hWnd, &pt);
	x1 = x2 = pt.x; //кудись записуємо координати початкової точки
```

```
	y1 = y2 = pt.y;
	check = 1;
}

void ShapeEditor::OnLBup(HWND hWnd) {
	POINT pt;
	GetCursorPos(&pt);
	ScreenToClient(hWnd, &pt);
	x2 = pt.x; //кудись записуємо координати початкової точки
	y2 = pt.y;
	check = 0;
}

void ShapeEditor::OnMouseMove(HWND hWnd) {}

void ShapeEditor::OnInitMenuPopup(HWND hWnd) {}

ShapeObjectsEditor::ShapeObjectsEditor()
{
	pse = new PointEditor;
}

ShapeObjectsEditor::~ShapeObjectsEditor()
{
	for (int i = 0; i < arrayCounter; i++)
		delete pcshape[i];
}

void ShapeObjectsEditor::StartPointEditor(HWND hWnd)
{
	if (pse)
		delete pse;
	pse = new PointEditor;
	pse->OnInitMenuPopup(hWnd);
}

void ShapeObjectsEditor::StartLineEditor(HWND hWnd)
{
	if (pse)
		delete pse;
	pse = new LineEditor;
	pse->OnInitMenuPopup(hWnd);
}

void ShapeObjectsEditor::StartRectEditor(HWND hWnd)
{
	if (pse)
		delete pse;
	pse = new RectEditor;
```

```
	pse->OnInitMenuPopup(hWnd);
}

void ShapeObjectsEditor::StartEllipseEditor(HWND hWnd)
{
	if (pse)
		delete pse;
	pse = new EllipseEditor;
	pse->OnInitMenuPopup(hWnd);
}

void ShapeObjectsEditor::OnInitMenuPopup(HWND hWnd)
{
	if (pse)
		pse->OnInitMenuPopup(hWnd);
}

void ShapeObjectsEditor::OnLBdown(HWND hWnd)
{
	if (pse)
		pse->OnLBdown(hWnd);
}

void ShapeObjectsEditor::OnLBup(HWND hWnd)
{
	if (pse)
		pse->OnLBup(hWnd);
}

void ShapeObjectsEditor::OnMouseMove(HWND hWnd)
{
	if (pse && check)
		pse->OnMouseMove(hWnd);
}

void ShapeObjectsEditor::OnPaint(HWND hWnd)
{
	ShapeEditor *draw = new ShapeEditor;
	draw->OnPaint(hWnd);
}

void PointEditor::OnLBdown(HWND hWnd)
{
	__super::OnLBdown(hWnd);
}

void PointEditor::OnLBup(HWND hWnd)
{
	__super::OnLBup(hWnd);
```

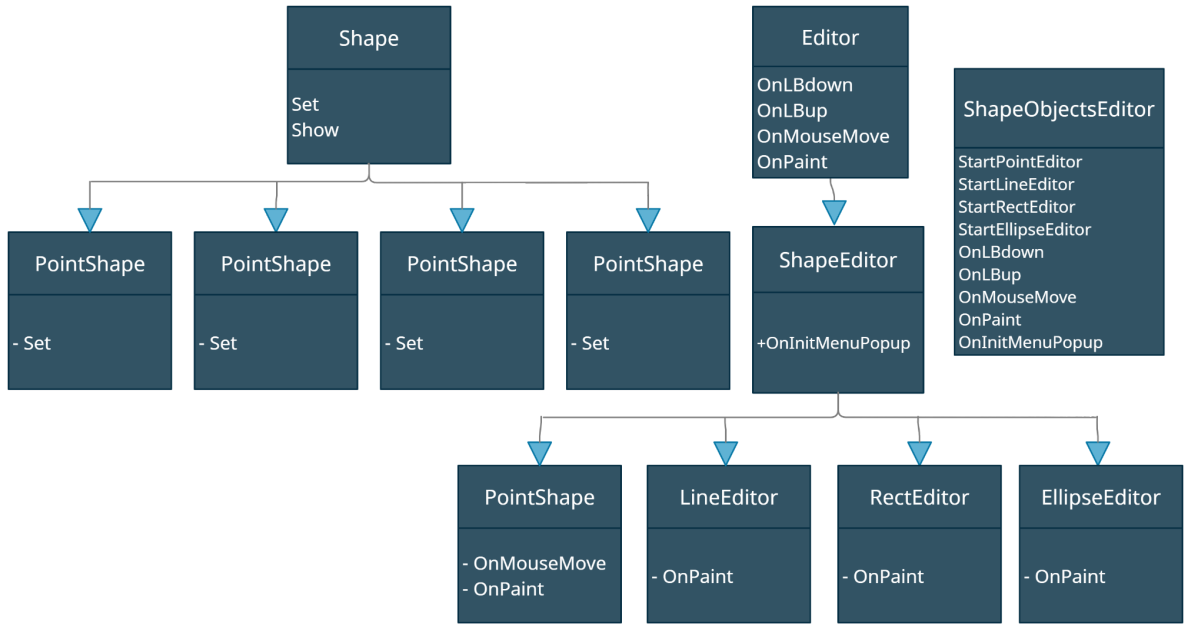
```
	PointShape* Point = new PointShape;
	Point->Set(x1, y1, x2, y2);
	pcshape[arrayCounter] = Point;
	arrayCounter++;
	InvalidateRect(hWnd, NULL, TRUE);
}

void PointEditor::OnInitMenuPopup(HWND hWnd)
{
	SetWindowText(hWnd, L"Режим вводу крапок");
}

void LineEditor::OnLBdown(HWND hWnd)
{
	__super::OnLBdown(hWnd);
}

void LineEditor::OnLBup(HWND hWnd)
{
	__super::OnLBup(hWnd);
	LineShape* Line = new LineShape;
	Line->Set(x1, y1, x2, y2);
	pcshape[arrayCounter] = Line;
	arrayCounter++;
	InvalidateRect(hWnd, NULL, TRUE);
}

void LineEditor::OnMouseMove(HWND hWnd) {
	POINT pt;
	HPEN hPen, hPenOld;
	HDC hdc = GetDC(hWnd);
	SetROP2(hdc, R2_NOTXORPEN);
	hPen = CreatePen(PS_SOLID, 1, RGB(255, 0, 0));
	hPenOld = (HPEN)SelectObject(hdc, hPen);
	MoveToEx(hdc, x1, y1, NULL);
	LineTo(hdc, x2, y2);
	GetCursorPos(&pt);
	ScreenToClient(hWnd, &pt);
	x2 = pt.x;
	y2 = pt.y;
	MoveToEx(hdc, x1, y1, NULL);
	LineTo(hdc, x2, y2);
	SelectObject(hdc, hPenOld);
	DeleteObject(hPen);
	ReleaseDC(hWnd, hdc);
}

void LineEditor::OnInitMenuPopup(HWND hWnd)
{
```

```
	std::wstring wstr = L"Режим вводу ліній";
	LPCWSTR lp = wstr.c_str();
	SetWindowText(hWnd, lp);
}

void RectEditor::OnLBdown(HWND hWnd)
{
	__super::OnLBdown(hWnd);
}

void RectEditor::OnLBup(HWND hWnd)
{
	__super::OnLBup(hWnd);
	RectShape* Rect = new RectShape;
	int x3 = 2 * x1 - x2;
	int y3 = 2 * y1 - y2;
	Rect->Set(x3, y3, x2, y2);
	pcshape[arrayCounter] = Rect;
	arrayCounter++;
	InvalidateRect(hWnd, NULL, TRUE);
}

void RectEditor::OnMouseMove(HWND hWnd)
{
	POINT pt;
	HPEN hPen, hPenOld;
	HDC hdc = GetDC(hWnd);
	SetROP2(hdc, R2_NOTXORPEN);
	hPen = CreatePen(PS_SOLID, 1, RGB(255, 0, 0));
	hPenOld = (HPEN)SelectObject(hdc, hPen);
	int x3 = 2 * x1 - x2;
	int y3 = 2 * y1 - y2;
	MoveToEx(hdc, x2, y2, NULL);
	LineTo(hdc, x2, y3);
	LineTo(hdc, x3, y3);
	LineTo(hdc, x3, y2);
	LineTo(hdc, x2, y2);
	GetCursorPos(&pt);
	ScreenToClient(hWnd, &pt);
	x2 = pt.x;
	y2 = pt.y;
	x3 = 2 * x1 - x2;
	y3 = 2 * y1 - y2;
	MoveToEx(hdc, x2, y2, NULL);
	LineTo(hdc, x2, y3);
	LineTo(hdc, x3, y3);
	LineTo(hdc, x3, y2);
	LineTo(hdc, x2, y2);
	SelectObject(hdc, hPenOld);
```

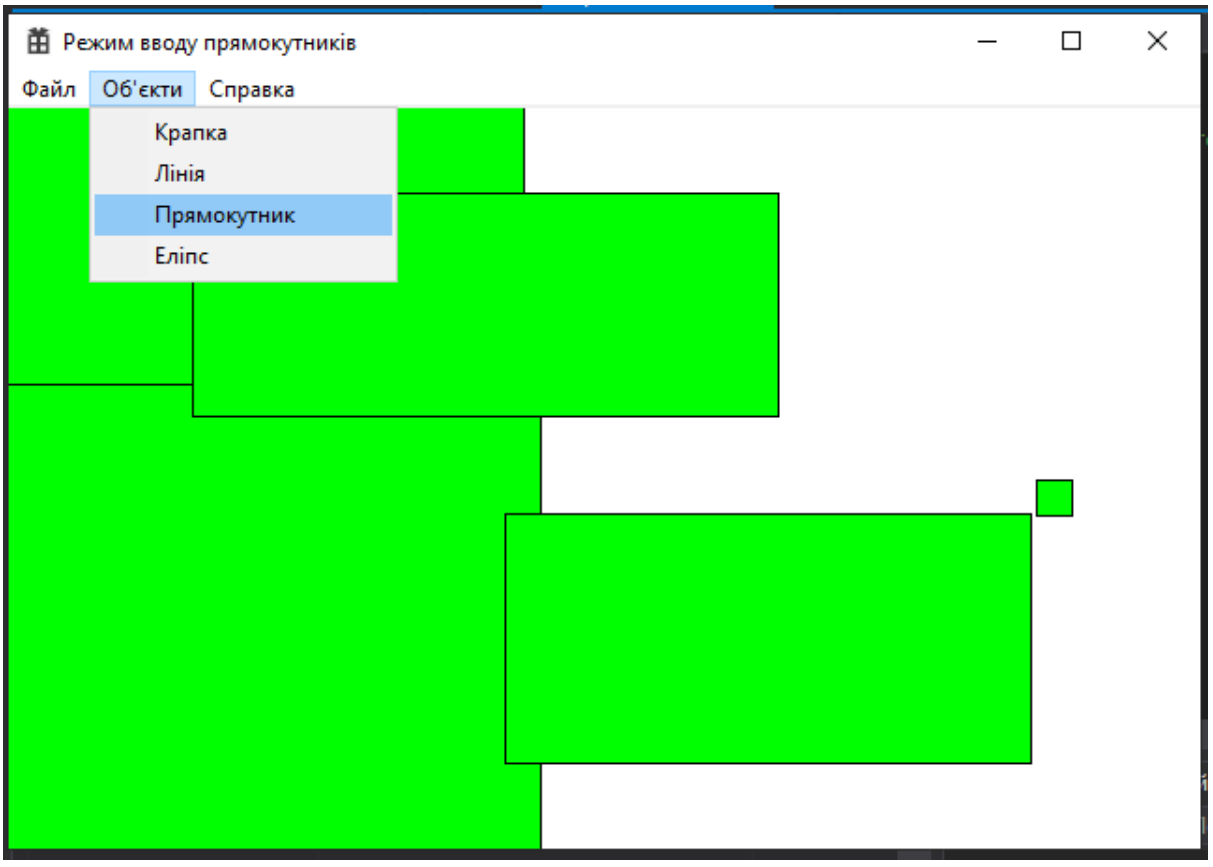
```
	DeleteObject(hPen);
	ReleaseDC(hWnd, hdc);
}

void RectEditor::OnInitMenuPopup(HWND hWnd)
{
	std::wstring wstr = L"Режим вводу прямокутників";
	LPCWSTR lp = wstr.c_str();
	SetWindowText(hWnd, lp);
}

void EllipseEditor::OnLBdown(HWND hWnd)
{
	__super::OnLBdown(hWnd);
}

void EllipseEditor::OnLBup(HWND hWnd)
{
	__super::OnLBup(hWnd);
	EllipseShape* Ellipse = new EllipseShape;
	Ellipse->Set(x1, y1, x2, y2);
	pcshape[arrayCounter] = Ellipse;
	arrayCounter++;
	InvalidateRect(hWnd, NULL, TRUE);
}

void EllipseEditor::OnMouseMove(HWND hWnd)
{
	POINT pt;
	HPEN hPen, hPenOld;
	HDC hdc = GetDC(hWnd);
	SetROP2(hdc, R2_NOTXORPEN);
	hPen = CreatePen(PS_SOLID, 1, RGB(255, 0, 0));
	hPenOld = (HPEN)SelectObject(hdc, hPen);
	Arc(hdc, x1, y1, x2, y2, 0, 0, 0, 0);
	GetCursorPos(&pt);
	ScreenToClient(hWnd, &pt);
	x2 = pt.x;
	y2 = pt.y;
	Arc(hdc, x1, y1, x2, y2, 0, 0, 0, 0);
	SelectObject(hdc, hPenOld);
	DeleteObject(hPen);
	ReleaseDC(hWnd, hdc);
}

void EllipseEditor::OnInitMenuPopup(HWND hWnd)
{
	std::wstring wstr = L"Режим вводу еліпсів";
	LPCWSTR lp = wstr.c_str();
```

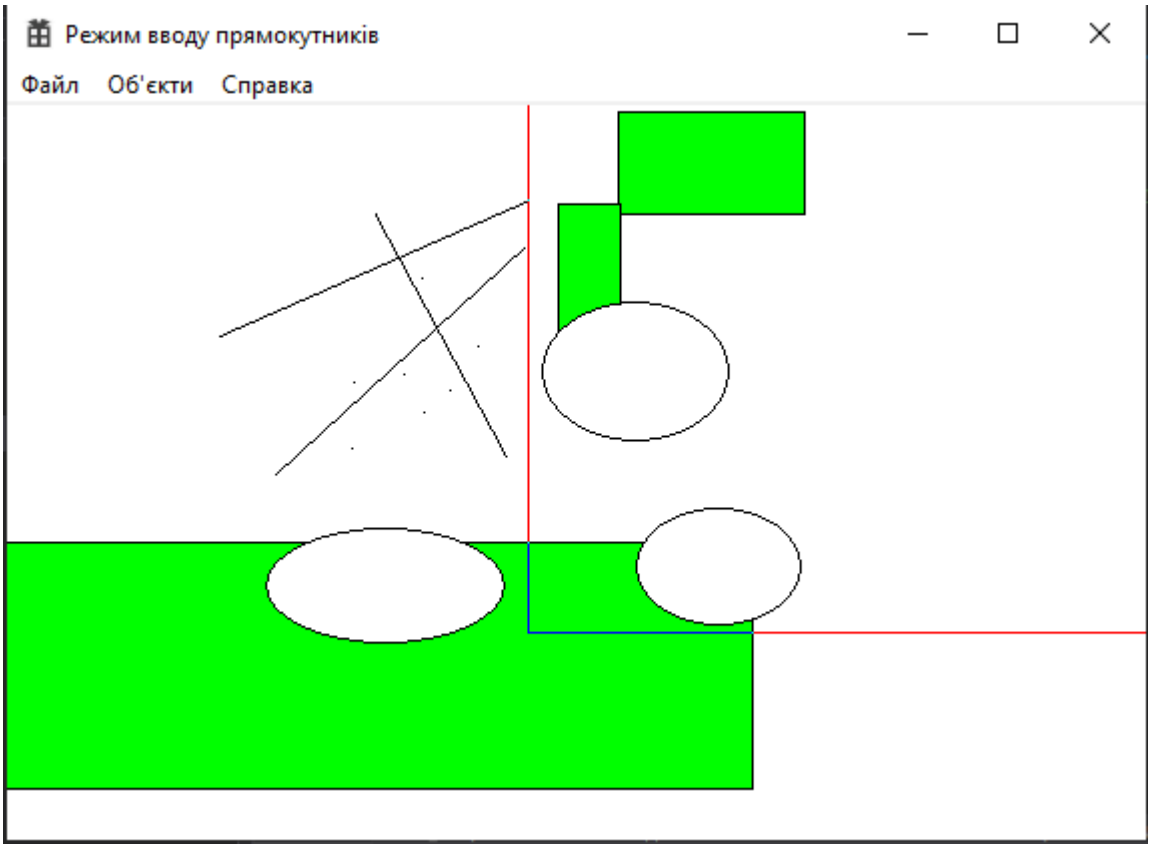
```
        SetWindowText(hWnd, lp);
}
```

Діаграма класів:



Ілюстрації роботи програми:





Висновки:

Виконуючи другу лабораторну роботу я почав розробку графічного редактора та познайомився з класами на C++.