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

Звіт

з лабораторної роботи № 2 з дисципліни

«Об’єктно-орієнтоване програмування»

Варіант 29

Виконав студент ІМ-11, Пономаренко Маргарита Альбертівна
(шифр, прізвище, ім'я, по батькові)

Перевірив Порєв В. М.
(прізвище, ім'я, по батькові)

Київ 2022

Мета роботи: отримати вміння та навички використовувати інкапсуляцію, абстракцію типів, успадкування та поліморфізм на основі класів C++, запрограмувавши простий графічний редактор в об'єктно-орієнтованому стилі.

Завдання:

1. Створити у середовищі Qt Creator проект типу Desktop Application з ім'ям Lab2.

2. Написати вихідний текст програми згідно варіанту завдання.

3. Скомпілювати вихідний текст і отримати виконуваний файл програми.

4. Перевірити роботу програми. Налагодити програму.

5. Проаналізувати та прокоментувати результати та вихідний текст програми.

6. Оформити звіт.

Варіант 29:

6. Масив вказівників для динамічних об'єктів типу Shape

- статичний масив Shape *pcshape[N];

7. "Гумовий" слід при вводі об'єктів

- суцільна лінія червоного кольору для студентів, у яких (Ж mod 4 = 1)

8. Чотири геометричні форми (крапка, лінія, прямокутник, еліпс) можуть мати наступні різновиди вводу та відображення.

8.1. Прямокутник

Ввід прямокутника:

- від центру до одного з кутів для (Ж mod 2 = 1)

Відображення прямокутника:

- чорний контур прямокутника без заповнення для (Ж mod 5 = 3 або 4)

8.2. Еліпс

Ввід еліпсу:

- по двом протилежним кутам охоплюючого прямокутника для (Ж mod 2 = 1)

Відображення еліпсу:

- чорний контур з кольоровим заповненням для (Ж mod 5 = 3 або 4)

Кольори заповнення еліпсу:

- помаранчевий для (Ж mod 6 = 5)

9. Позначка поточного типу об’єкту, що вводиться

- в заголовку вікна для (Ж mod 2 = 1)

Текст головного файлу (main.cpp)

```
#include "mainwindow.h"

#include <QApplication>

int main(int argc, char *argv[])
{
  QApplication a(argc, argv);
  MainWindow w;
  w.show();
  return a.exec();
}
```

Текст заголовку головного вікна (mainwindow.h)

```
#ifndef MAINWINDOW_H
#define MAINWINDOW_H

#include <QMainWindow>
#include "shapeobjectseditorview.h"

QT_BEGIN_NAMESPACE
namespace Ui { class MainWindow; }
QT_END_NAMESPACE

class MainWindow : public QMainWindow
{
  Q_OBJECT

public:
  MainWindow(QWidget *parent = nullptr);
  ~MainWindow();
```

```
private slots:
  void on_actionNew_triggered();

  void on_drawPoint_triggered();

  void on_drawLine_triggered();

  void on_drawRect_triggered();

  void on_drawElipse_triggered();

private:
  Ui::MainWindow *ui;
  ShapeObjectsEditorView *shapeEditorView;
};
#endif // MAINWINDOW_H
```

Текст реалізації головного вікна (mainwindow.cpp)

```
#include "mainwindow.h"
#include "ui_mainwindow.h"
#include "scenewindow.h"

MainWindow::MainWindow(QWidget *parent)
  : QMainWindow(parent)
  , ui(new Ui::MainWindow)
{
  ui->setupUi(this);

}

MainWindow::~MainWindow()
{
  delete ui;
}

void MainWindow::on_actionNew_triggered()
{
  shapeEditorView = createEditor(this);
}

void MainWindow::on_drawPoint_triggered()
{
  setWindowTitle("Режим малювання: крапка");
  shapeEditorView->drawPoint();
}


void MainWindow::on_drawLine_triggered()
{
```

```
  setWindowTitle("Режим малювання: лінія");
  shapeEditorView->drawLine();
}


void MainWindow::on_drawRect_triggered()
{
  setWindowTitle("Режим малювання: прямокутник");
  shapeEditorView->drawRect();
}


void MainWindow::on_drawElipse_triggered()
{
  setWindowTitle("Режим малювання: еліпс");
  shapeEditorView->drawElipse();
}
```

Текст заголовку файлу для створення сцени (scenewindow.h)

```
#ifndef SCENEWINDOW_H
#define SCENEWINDOW_H

#include <QWidget>
#include "mainwindow.h"
#include "shapeobjectseditorview.h"

ShapeObjectsEditorView * createEditor(QWidget *parent);  //метод створення сцени

#endif // SCENEWINDOW_H
```

Текст реалізіції файлу для створення сцени (scenewindow.cpp)

```
#include <QGraphicsView>
#include "scenewindow.h"
#include "shapeobjectseditor.h"

ShapeObjectsEditorView *createEditor(QWidget *parent)
{
  const int topMargin = 25;
  int width = parent->width();
  int height = parent->height() - topMargin;

  ShapeObjectsEditor *editor = new ShapeObjectsEditor;  //створення сцени
  editor->setSceneRect(0, 0, width, height);

  ShapeObjectsEditorView * view = new ShapeObjectsEditorView(parent);  //створення view
(відображення сцени)
  view->setEditor(editor);
  view->setWindowTitle(QT_TRANSLATE_NOOP(QGraphicsView, "Режим малювання:"));
```

```
    view->setGeometry(0, topMargin, width + 2, height + 2);
    view->show();

    return view;
}
```

Текст заголовку класу ShapeObjectsEditor (shapeobjecteditor.h)

```
#ifndef SHAPEOBJECTSEDITOR_H
#define SHAPEOBJECTSEDITOR_H
#include <QGraphicsScene>

enum class DrawType{
    POINT = 0,
    LINE,
    RECT,
    ELIPSE,
};

class ShapeObjectsEditor: public QGraphicsScene
{
public:
    ShapeObjectsEditor();

    void drawPoint();
    void drawLine();
    void drawRect();
    void drawElipse();

private:
    DrawType drawType = DrawType::POINT;
    bool drawStatus = false;

    // QGraphicsScene interface
protected:
    void mousePressEvent(QGraphicsSceneMouseEvent *event) override;
    void mouseReleaseEvent(QGraphicsSceneMouseEvent *event) override;
    void mouseMoveEvent(QGraphicsSceneMouseEvent *event) override;
};

#endif // SHAPEOBJECTSEDITOR_H
```

Текст реалізації класу ShapeObjectsEditor (shapeobjecteditor.cpp)

```
#include "shapeobjectseditor.h"
#include "pointshape.h"
#include "lineshape.h"

#include <QGraphicsSceneMouseEvent>
```

```
ShapeObjectsEditor::ShapeObjectsEditor()
{

}

void ShapeObjectsEditor::drawPoint()
{
  drawType = DrawType::POINT;
}

void ShapeObjectsEditor::drawLine()
{
  drawType = DrawType::LINE;
}

void ShapeObjectsEditor::drawRect()
{
  drawType = DrawType::RECT;
}

void ShapeObjectsEditor::drawElipse()
{
  drawType = DrawType::ELIPSE;
}


void ShapeObjectsEditor::mousePressEvent(QGraphicsSceneMouseEvent *event)
{
  switch (drawType){
  case DrawType::POINT:
  { PointShape *point = new PointShape;
     int x = event->scenePos().x();  //ініціалізація позиції x та y курсора
     int y = event->scenePos().y();
     point->Set(x, y, x, y);
     this->addItem(point);  //додавання об'єкту крапка
     point->Show();
     break; }

  case DrawType::LINE:
  {
     int x1 = event->scenePos().x();
     int y1 = event->scenePos().y();
     if (event->buttons() & Qt::LeftButton)
     {
        if(!drawStatus){
           qDebug()<<"Set up start coordinates" + QString::number(drawStatus);
           x1 = event->scenePos().x();
           y1 = event->scenePos().y();
        }
        drawStatus = !drawStatus;
```

```
        qDebug()<<"Status:" + QString::number(drawStatus);
    }
    break; }

  case DrawType::RECT:
    break;
  case DrawType::ELIPSE:
    break;

  }
}

void ShapeObjectsEditor::mouseReleaseEvent(QGraphicsSceneMouseEvent *event)
{
  switch (drawType){
  case DrawType::LINE:
  { qDebug()<<"Set up end coordinates" + QString::number(drawStatus);
    LineShape *line = new LineShape;
    int x2 = event->scenePos().x();
    int y2 = event->scenePos().y();
    line->Set(x1, y1, x2, y2);
    qDebug()<<"Start coord:" + QString::number(x1) + "; ";
    this->addItem(line);
    line->Show();

    drawStatus = false;
    break; }
  }
}

void ShapeObjectsEditor::mouseMoveEvent(QGraphicsSceneMouseEvent *event)
{

}
```

Текст заголовку класу ShapeObjectsEditorView (shapeobjecteditorview.h)

```
#ifndef SHAPEOBJECTSEDITORVIEW_H
#define SHAPEOBJECTSEDITORVIEW_H
#include <QGraphicsView>
#include <QWidget>
#include "shapeobjectseditor.h"


class ShapeObjectsEditorView:public QGraphicsView
{
public:
  ShapeObjectsEditorView(QWidget *parent);
  void setEditor(ShapeObjectsEditor *editor);

  //action
```

```
    void drawPoint();
    void drawLine();
    void drawRect();
    void drawElipse();

private:
    ShapeObjectsEditor *editor;
};

#endif // SHAPEOBJECTSEDITORVIEW_H
```

Текст реалізації класу ShapeObjectsEditorView (shapeobjecteditorview.cpp)

```
#include "shapeobjectseditorview.h"


ShapeObjectsEditorView::ShapeObjectsEditorView(QWidget *parent):QGraphicsView(parent)
{
}

void ShapeObjectsEditorView::setEditor(ShapeObjectsEditor *editor)
{
    this->editor = editor;
    this->setScene(editor);
}

void ShapeObjectsEditorView::drawPoint()
{
    editor->drawPoint();
}

void ShapeObjectsEditorView::drawLine()
{
    editor->drawLine();
}

void ShapeObjectsEditorView::drawRect()
{
    editor->drawRect();
}

void ShapeObjectsEditorView::drawElipse()
{
    editor->drawElipse();
}
```

Текст заголовку класу Shape (shape.h)

```
#ifndef SHAPE_H
#define SHAPE_H
#include <QGraphicsItem>

class Shape:public QGraphicsItem
{
public:
  Shape();
  virtual ~Shape();
  void Set(int x1, int y1, int x2, int y2);
  virtual void Show() = 0;

protected:
  int xs1, ys1, xs2, ys2;
  Shape *objects [129];

};

#endif // SHAPE_H
```

Текст реалізації класу Shape (shape.cpp)

```
#include "shape.h"

Shape::Shape()
{

}

Shape::~Shape()
{

}

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

Результати тестування програми

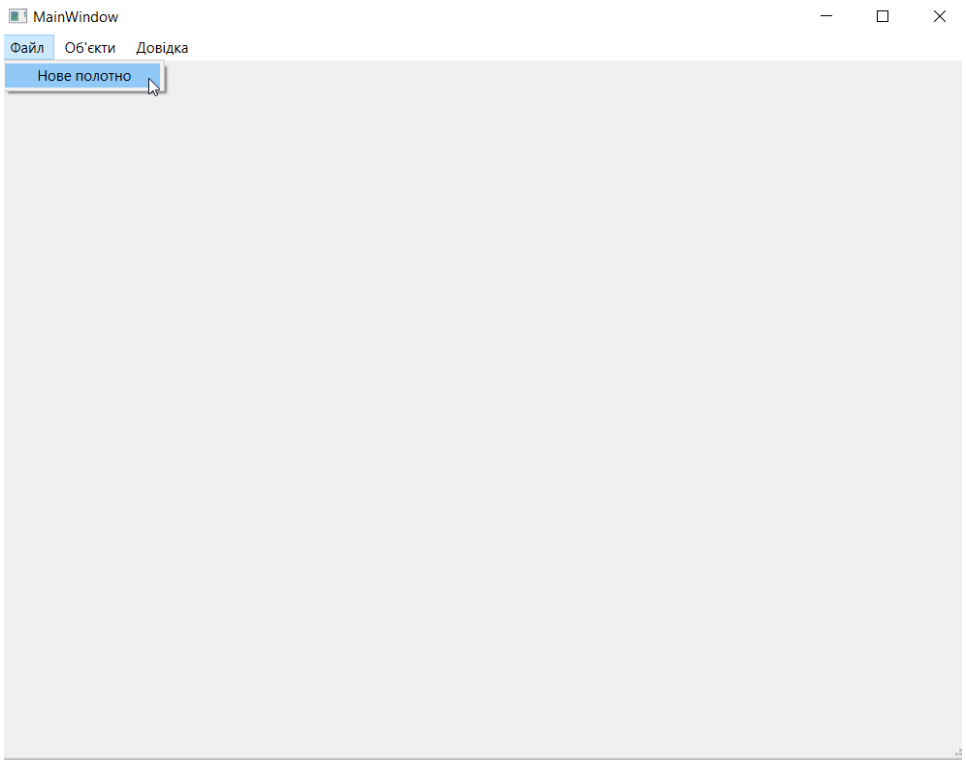


Рис 1.1: Демонстрація меню “Файл - Нове полотно” (створює полотно для малювання)

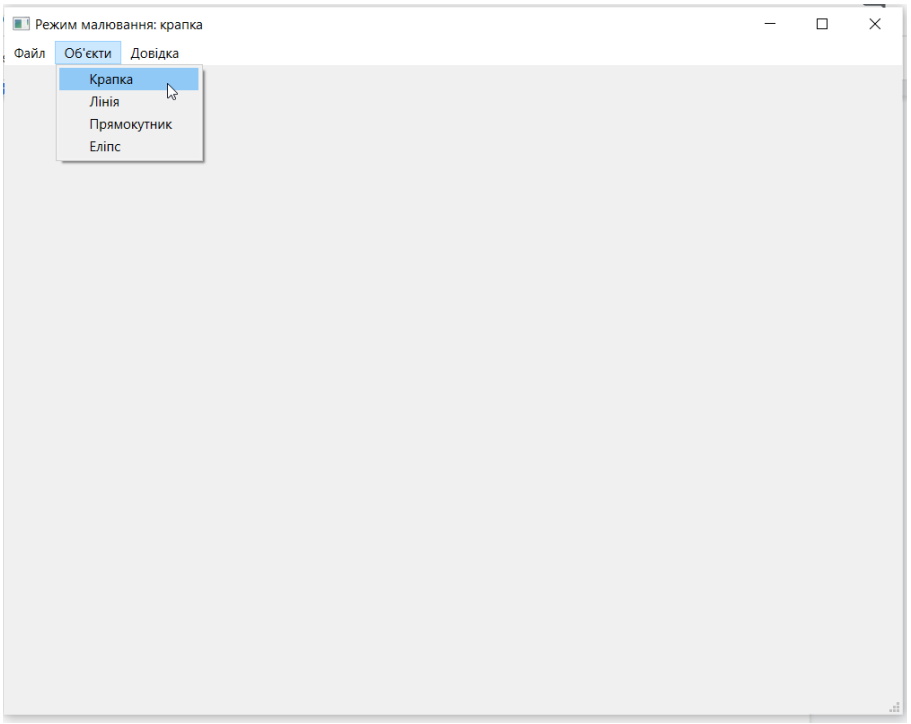


Рис 1.2: Демонстрація меню “Об’єкти - Крапка” (змінює заголовок вікна на “Режим малювання: крапка” та встановлює вибраний режим)

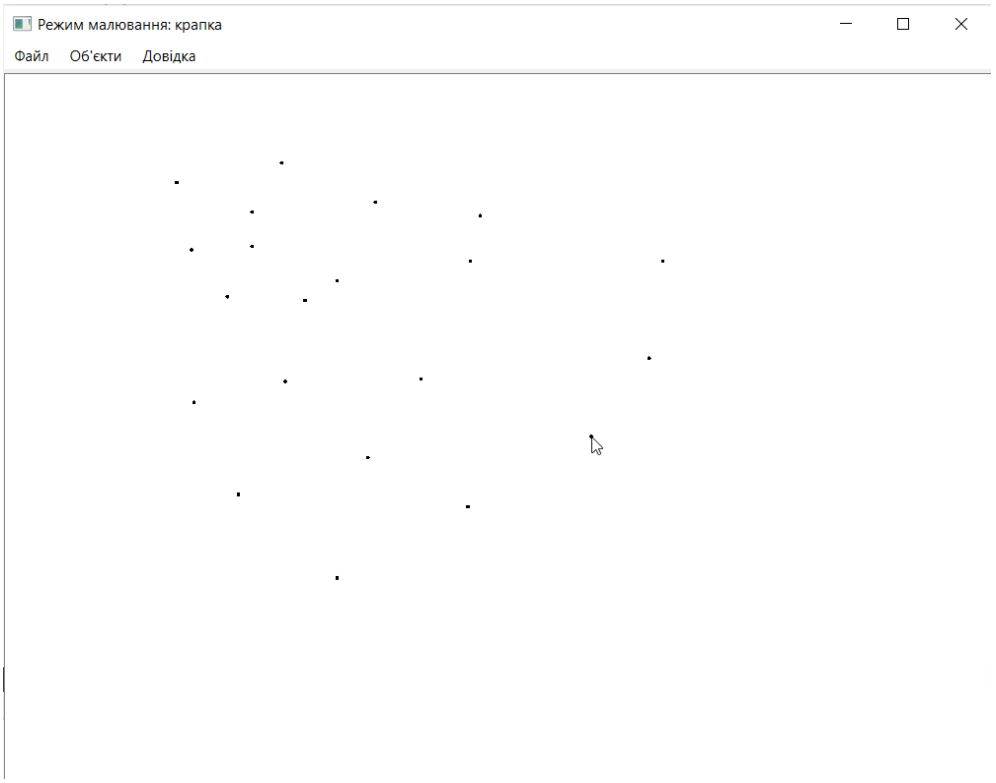


Рис 1.3: Демонстрація роботи режиму малювання крапки

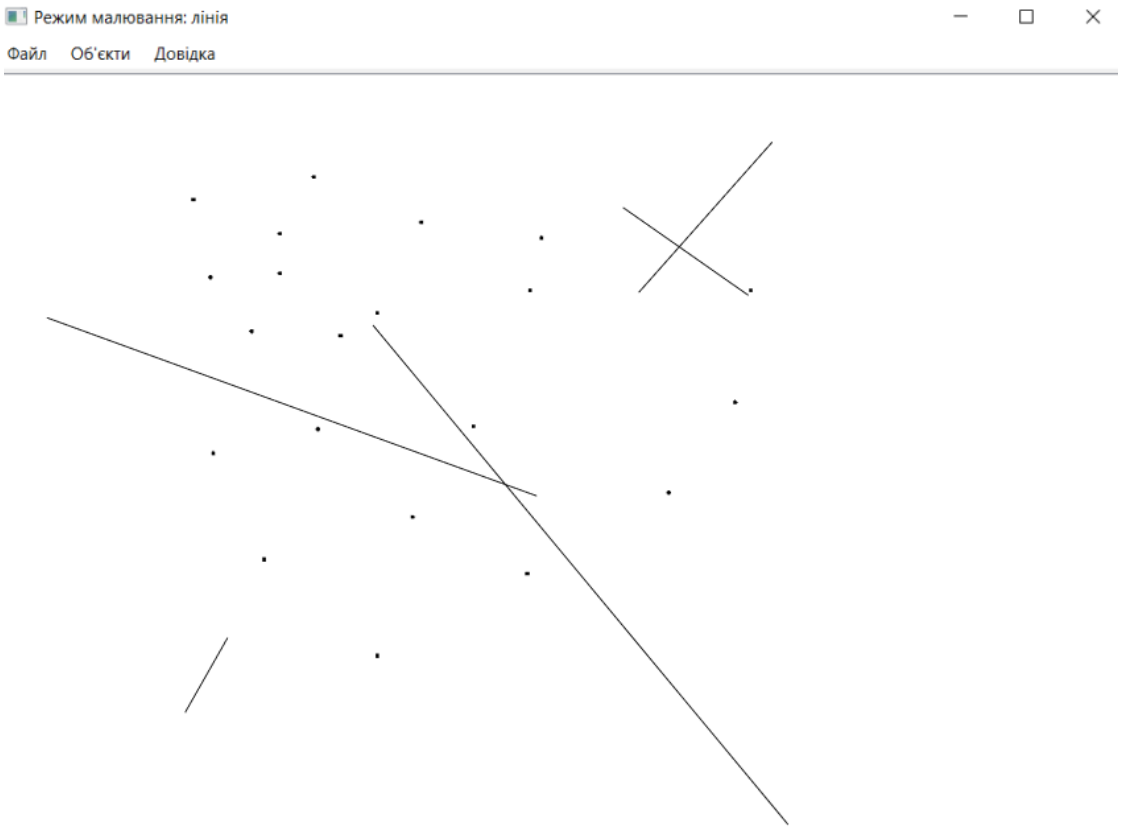


Рис 1.4: Демонстрація роботи режиму малювання лінії

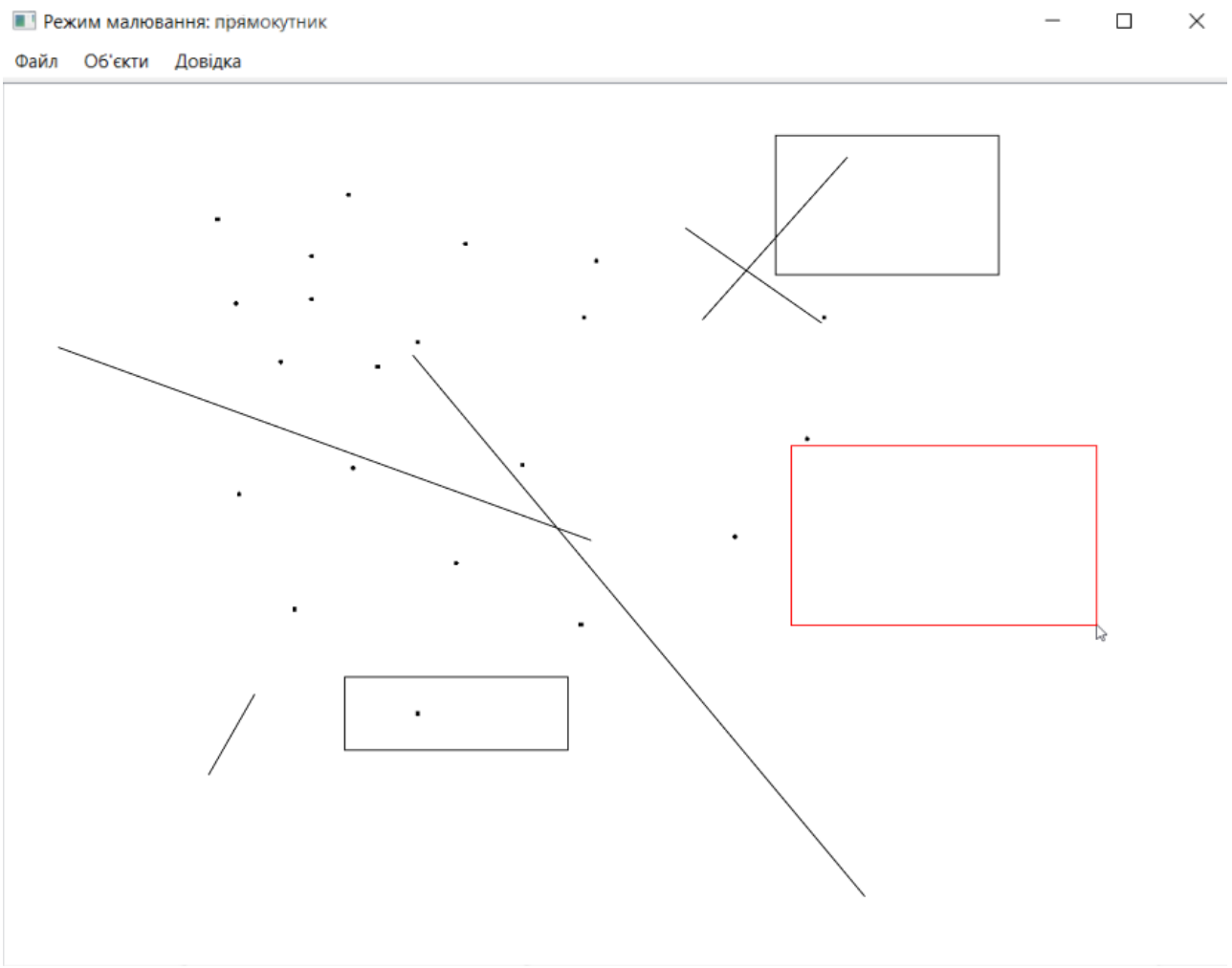


Рис 1.5: Демонстрація роботи режиму малювання прямокутника з чорним контуром без заповнення, гумовий слід - суцільна лінія червоного кольору

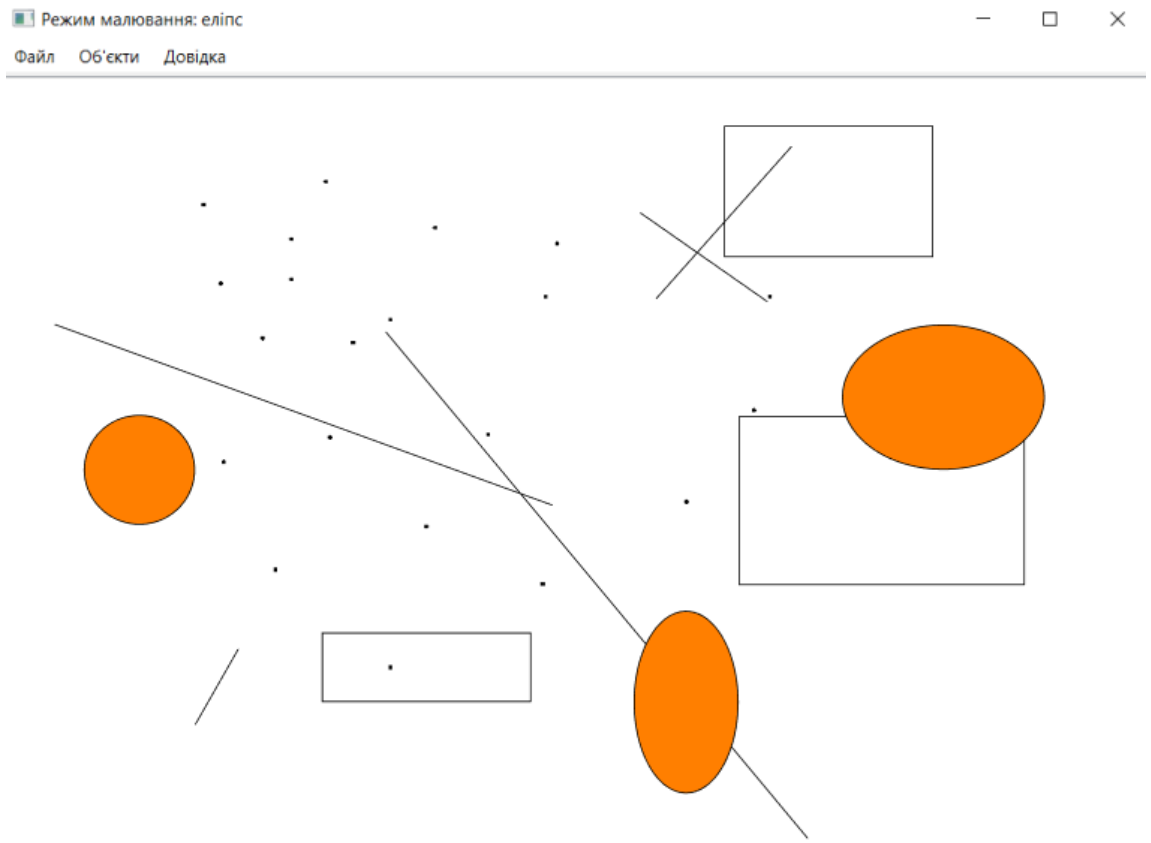
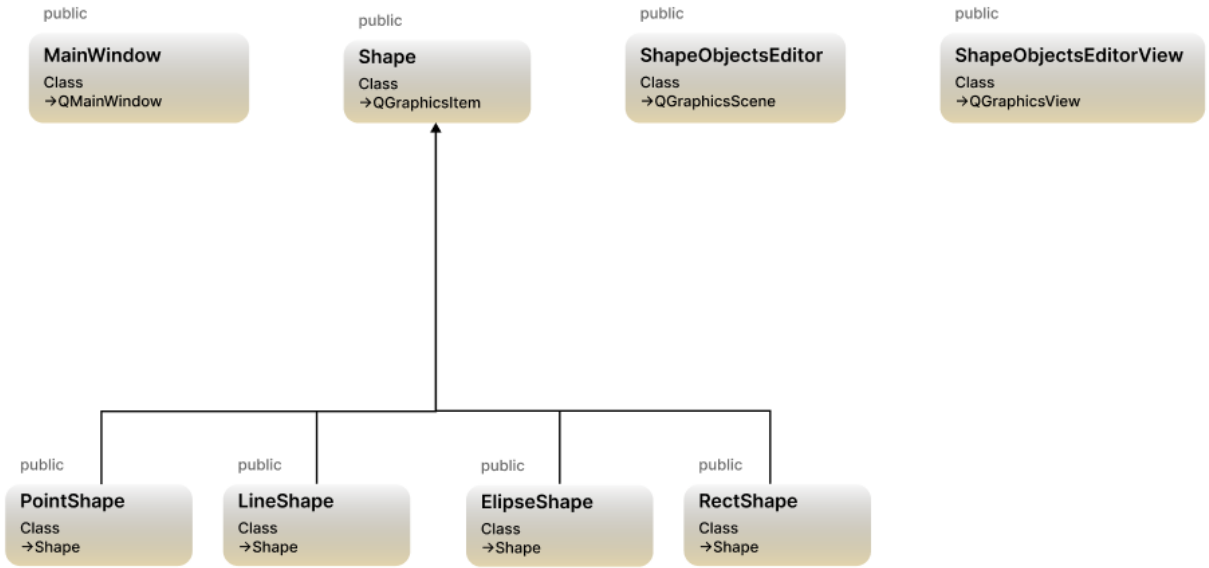


Рис 1.6: Демонстрація роботи режиму малювання еліпсу з чорним контуром та помаранчевим заповненням

Діаграма класів

*з урахуванням особливості розробки в середовищі Qt Creator



Висновок

У цій лабораторній роботі було створено програму для Windows на основі проєктів для Qt Creator з використанням інкапсуляції, абстракції типів, успадкування та поліморфізму на основі класів C++, запрограмовано простий графічний редактор в об’єктно-орієнтованому стилі, в якому є можливість намалювати такі фігури: крапка, лінія, прямокутник та еліпс.